Foundations of Machine Learning

E. Le Pennec

MAP553 - Foundations of Machine Learning - Spring 2023
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
     - Machine Learning
     - Supervised Learning
     - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
     - Ensemble Methods
     - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9 References
Major Influences

Four major influences act today:

- The formal theories of statistics
- Accelerating developments in computers and display devices
- The challenge, in many fields, of more and ever larger bodies of data
- The emphasis on quantification in an ever wider variety of disciplines

Source: D. Donoho, 50 Years of Data Science
Major Influences - Tukey (1962)

Four major influences act today:

- The formal theories of statistics
- Accelerating developments in computers and display devices
- The challenge, in many fields, of more and ever larger bodies of data
- The emphasis on quantification in an ever wider variety of disciplines

- He was talking of Data Analysis.
- Data Mining, Machine Learning, Big Data, Data Science, Artificial Intelligence...
Statistical Learning: Introduction, Setting and Risk Estimation

Large Scale ML Is (Quite) Easy

Example of **off the shelves** solution

- Algorithm implementation + copy/paste + cloud computing.
- Machine learning on an arbitrary large dataset!
Data Science Is (Quite) Complex!
Data Ecosystem Is (Quite) Complex!
Data Ecosystem Is (Quite) Complex!

The Periodic Table of Data Science

An overview of key companies, resources and tools in data science (as of 4/12/2017)

Statistical Learning: Introduction, Setting and Risk Estimation

DataCamp

The Periodic Table of Data Science

An overview of key companies, resources and tools in data science (as of 4/12/2017)

Source: DataCamp
Monthly KPI Dashboard

KPI: Key Performance Index

\[ \text{Key Financial Controlling Indicators} \]

\[ \text{Cost Structure} \]

\[ \text{Cashflow and Working Capital over the next 3 years} \]

\[ \text{Planned Turnover and Net Income} \]
Realtime Log Dashboard

Collect & Transform

Search & Analyze

Visualize & Manage

IT: Information Technology
On-demand Legal Document Generation
AB Testing

Source: Optimizely
ER Waiting Time Prediction

ER: Emergency Room

Source: M. Evans
Weekly Churn Prediction

Customer Churn Prediction Accuracy

Correct prediction

Incorrect prediction

Source: Parthasarathy
Realtime Automatic Fruit Sorting
Realtime Chatbot
Customer Segmentation

Data Science Project - Customer Segmentation

Identifying the potential customer base for selling the product
Implementing Clustering Algorithms to group the customer base
Selling product to the identified customer group
Realtime Anomaly Detection
On-demand Fraud Detection
Prescriptive Maintenance

Statistical Learning: Introduction, Setting and Risk Estimation

THE EVOLUTION OF MAINTENANCE STRATEGIES

<table>
<thead>
<tr>
<th>REACTIVE</th>
<th>PREVENTIVE</th>
<th>PREDICTIVE</th>
<th>PRESCRIPTIVE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fix it when it breaks!</td>
<td>Maintain it at regular intervals so it doesn’t break!</td>
<td>Predict exactly when it will break and maintain it accordingly!</td>
<td>Let the machines help you decide how to avoid predicted failures!</td>
</tr>
</tbody>
</table>

Source: Limble
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9. References
Machine Learning

Statistical Learning: Introduction, Setting and Risk Estimation

Sources: MyCarDoesWhat.org/theverge.com/Zhiqiang Wan et al.

23
Machine Learning

A definition by Tom Mitchell

A computer program is said to learn from experience $E$ with respect to some class of tasks $T$ and performance measure $P$, if its performance at tasks in $T$, as measured by $P$, improves with experience $E$. 

Source: A. Barabasi

24
Object Detection

A detection algorithm:

- **Task**: say if an object is present or not in the image
- **Performance**: number of errors
- **Experience**: set of previously seen labeled images
Article Clustering

An article clustering algorithm:

- **Task**: group articles corresponding to the same news
- **Performance**: quality of the clusters
- **Experience**: set of articles
A controller in its sensors in a home smart grid:

- **Task**: control the devices
- **Performance**: energy costs
- **Experience**:
  - previous days
  - current environment and performed actions

Source: Zhiqiang Wan et al.
Three Kinds of Learning

**Unsupervised Learning**
- **Task:** Clustering/DR/Generative
- **Performance:** Quality
- **Experience:** Raw dataset (No Ground Truth)
- **Timing:** Offline/Batch (learning from past data) vs Online (continuous learning)

**Supervised Learning**
- **Task:** Prediction/Classification
- **Performance:** Average error
- **Experience:** Good Predictions (Ground Truth)

**Reinforcement Learning**
- **Task:** Actions
- **Performance:** Total reward
- **Experience:** Reward from env. (Interact. with env.)
Supervised and Unsupervised Learning

**Supervised Learning (Imitation)**
- **Goal:** Learn a function $f$ predicting a variable $Y$ from an individual $X$.
- **Data:** Learning set with labeled examples $(X_i, Y_i)$
- **Assumption:** Future data behaves as past data!
- **Predicting is not explaining!**
Supervised and Unsupervised Learning

Supervised Learning (Imitation)

- **Goal**: Learn a function $f$ predicting a variable $Y$ from an individual $X$.
- **Data**: Learning set with labeled examples $(X_i, Y_i)$
- **Assumption**: Future data behaves as past data!
- **Predicting is not explaining!**

Unsupervised Learning (Structure Discovery)

- **Goal**: Discover a structure within a set of individuals $(X_i)$.
- **Data**: Learning set with unlabeled examples $(X_i)$
- Unsupervised learning is not a well-posed setting...
Machine Can and Cannot

**Machine Can**
- Forecast (Prediction using the past)
- Detect expected changes
- Memorize/Reproduce
- Take a decision very quickly
- Learn from huge dataset
- Optimize a single task
- Replace/Help some humans

**Machine Cannot**
- Predict something never seen before
- Detect any new behaviour
- Create something brand new
- Understand the world
- Get smart really fast
- Go beyond their task
- Kill all humans

- Some progresses but still very far from the *singularity*...
Machine Learning

- Huge catalog of methods,
- Need to define the performance,
- Numerous tricks: feature design, hyperparameter selection...
Under and Over Fitting

Finding the Right Complexity

- What is best?
  - A simple model that is stable but false? *(oversimplification)*
  - A very complex model that could be correct but is unstable? *(conspiracy theory)*
- Neither of them: tradeoff that depends on the dataset.
Machine Learning Pipeline

Learning pipeline

- Test and compare models.
- Deployment pipeline is different!
Data Science ≠ Machine Learning

Main DS difficulties

- Figuring out the problem,
- Formalizing it,
- Storing and accessing the data,
- Deploying the solution,
- Not (always) the Machine Learning part!
MAP 553 - Foundation of Machine Learning - Goal

**Goal**

- Master the **statistical learning** framework and its challenges.
- Know the inner machinery of the most classical supervised and unsupervised **ML methods** in order to understand their strengths, limitations and connections.
- Understand some **optimization** tools used in ML as well as some **theoretical aspects** of ML.
- Not a course on practical tricks to use machine learning in a data product!

**Evaluation**

- A practical lab (5 pt)
- A final exam or a project (15 pt)
MAP 553 - Team

- Erwan Le Pennec
  - Erwan.Le-Pennec@polytechnique.edu

- Randal Douc
  - randal.douc@it-sudparis.eu

- El Mahdi El Mhamdi
  - elmahdi@elmhamdi.com

- Thierry Klein
  - thierry.klein@math.univ-toulouse.fr

- Edouard Oyallon
  - edouard.oyallon@lip6.fr
MAP 553 - Schedule

9 Lectures (8h30-10h30)

- Mon. 19/09: Statistical Learning: Introduction, Setting and Risk Estimation
- Mon. 26/09: ML Methods: Probabilistic Point of View
- Mon. 03/10: ML Methods: Optimization Point of View
- Mon. 10/10: Optimization: Gradient Descent Algorithms
- Mon. 17/10: ML Methods: Neural Networks and Deep Learning
- Mon. 24/10: ML Methods: Trees and Ensemble Methods
- Mon. 07/11: Unsupervised Learning: Dimension Reduction
- Mon. 14/11: Unsupervised Learning: Clustering
- Mon. 21/11: Statistical Learning: PAC-Bayesian Approach and Complexity Theory

- Mon. ??/12: Exam (?)
T. Hastie, R. Tibshirani, and J. Friedman.  
*The Elements of Statistical Learning.*  
Springer Series in Statistics, 2009

M. Mohri, A. Rostamizadeh, and A. Talwalkar.  
*Foundations of Machine Learning (2nd ed.)*  
MIT Press, 2018

A. Géron.  
*Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (3rd ed.)*  
O’Reilly, 2022

Ch. Giraud.  
*Introduction to High-Dimensional Statistics.*  
CRC Press, 2014

S. Bubeck.  
*Convex Optimization: Algorithms and Complexity.*  
Now Publisher, 2015
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
     - Risk Estimation and Model Selection
     - Cross Validation and Test
     - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
     - A Probabilistic Point of View
     - Parametric Conditional Density Modeling
     - Non Parametric Conditional Density Modeling
     - Generative Modeling
     - Model Selection
     - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
     - SVM
     - Penalization
     - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5. Gradient Descent Step
   - Non-Convex Setting
   - References

6. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

7. ML Methods: Trees and Ensemble Methods
   - Trees
     - Bagging and Random Forests
       - Bootstrap and Bagging
       - Randomized Rules and Random Forests
     - Boosting
       - AdaBoost as a Greedy Scheme
     - Ensemble Methods
     - References

8. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

10. References

39
Supervised Learning

Supervised Learning Framework

- Input measurement $X \in \mathcal{X}$
- Output measurement $Y \in \mathcal{Y}$.
- $(X, Y) \sim P$ with $P$ unknown.
- **Training data**: $D_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ (i.i.d. $\sim P$)

Often
- $X \in \mathbb{R}^d$ and $Y \in \{-1, 1\}$ (classification)
- or $X \in \mathbb{R}^d$ and $Y \in \mathbb{R}$ (regression).
- A **predictor** is a function in $\mathcal{F} = \{f : \mathcal{X} \rightarrow \mathcal{Y} \ \text{meas.}\}$

Goal

- **Construct a good predictor** $\hat{f}$ from the training data.

Need to specify the meaning of good.
- Classification and regression are almost the same problem!
Loss and Probabilistic Framework

**Loss function for a generic predictor**

- **Loss function**: \( \ell(Y, f(X)) \) measures the goodness of the prediction of \( Y \) by \( f(X) \)
- **Examples**:
  - 0/1 loss: \( \ell(Y, f(X)) = 1_{Y \neq f(X)} \)
  - Quadratic loss: \( \ell(Y, f(X)) = |Y - f(X)|^2 \)

**Risk function**

- Risk measured as the average loss for a new couple:
  \[
  R(f) = \mathbb{E}_{(X,Y) \sim \mathbb{P}}[\ell(Y, f(X))]
  \]
- **Examples**:
  - 0/1 loss: \( \mathbb{E}[\ell(Y, f(X))] = \mathbb{P}(Y \neq f(X)) \)
  - Quadratic loss: \( \mathbb{E}[\ell(Y, f(X))] = \mathbb{E}[|Y - f(X)|^2] \)

- **Beware**: As \( \hat{f} \) depends on \( D_n \), \( R(\hat{f}) \) is a random variable!
The best solution $f^*$ (which is independent of $D_n$) is

$$f^* = \arg \min_{f \in F} R(f) = \arg \min_{f \in F} \mathbb{E}[\ell(Y, f(X))] = \arg \min_{f \in F} \mathbb{E}_X \left[ \mathbb{E}_Y | X [\ell(Y, f(X))] \right]$$

**Bayes Predictor (explicit solution)**

- In binary classification with $0 - 1$ loss:
  $$f^*(X) = \begin{cases} +1 & \text{if } \mathbb{P}(Y = +1|X) \geq \mathbb{P}(Y = -1|X) \\ -1 & \text{if } \mathbb{P}(Y = -1|X) \geq 1/2 \end{cases}$$

- In regression with the quadratic loss
  $$f^*(X) = \mathbb{E}[Y|X]$$

**Issue:** Solution requires to know $\mathbb{E}[Y|X]$ for all values of $X$!
Goal

Machine Learning

- Learn a rule to construct a predictor \( \hat{f} \in \mathcal{F} \) from the training data \( \mathcal{D}_n \) s.t. the risk \( R(\hat{f}) \) is small on average or with high probability with respect to \( \mathcal{D}_n \).

- In practice, the rule should be an algorithm!

Canonical example: Empirical Risk Minimizer

- One restricts \( f \) to a subset of functions \( \mathcal{S} = \{ f_\theta, \theta \in \Theta \} \)
- One replaces the minimization of the average loss by the minimization of the empirical loss

\[
\hat{f} = f_\hat{\theta} = \arg\min_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(X_i))
\]

- Examples:
  - Linear regression
  - Linear classification with

\[
\mathcal{S} = \{ x \mapsto \text{sign}(x^T \beta + \beta^{(0)}) \} / \beta \in \mathbb{R}^d, \beta^{(0)} \in \mathbb{R} \}
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
     - Cross Validation and Test
     - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
     - A Probabilistic Point of View
     - Parametric Conditional Density Modeling
     - Non Parametric Conditional Density Modeling
     - Generative Modeling
     - Model Selection
     - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
     - SVM
     - Penalization
     - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
     - Empirical Risk Minimization
       - Empirical Risk Minimization
       - ERM and PAC Analysis
       - Hoeffding and Finite Class
       - McDiarmid and Rademacher Complexity
       - VC Dimension
       - Structural Risk Minimization
     - References

6 References
Example: TwoClass Dataset

Synthetic Dataset

- Two features/covariates.
- Two classes.

- Dataset from *Applied Predictive Modeling*, M. Kuhn and K. Johnson, Springer
- Numerical experiments with R and the \{caret\} package.
Example: Linear Discrimination

Logistic

Decision region

Decision boundary

classes

Class1

Class2

classes

Class1

Class2
Example: More Complex Model

Naive Bayes with kernel density estimates

Decision region

Decision boundary

classes

Class 1

Class 2
Example: KNN
Example: KNN

k-NN with k=5

Decision region

Decision boundary

classes
- Class1
- Class2

classes
- Class1
- Class2
Example: KNN

k-NN with \( k=9 \)

Decision region

Decision boundary

classes

- Class1
- Class2
Example: KNN

k-NN with $k=13$

Decision region

Decision boundary

classes

- Red: Class1
- Blue: Class2

PredictrB vs. PredictorA
Example: KNN

k-NN with $k=17$

Decision region

Decision boundary

classes
- Class1
- Class2
Example: KNN
Example: KNN

k-NN with $k=25$

Decision region

Decision boundary

classes
- Class1
- Class2

classes
- Class1
- Class2
Example: KNN

- k-NN with $k=29$
  - Decision region
  - Decision boundary
  - Classes:
    - Class1
    - Class2
Example: KNN

k-NN with k=33

Decision region

Decision boundary

classes

Class1

Class2
Example: KNN

k-NN with $k=37$

Decision region

Decision boundary

classes
- Class1
- Class2
Example: KNN
Notice: k-NN with k=53

Decision region

Decision boundary

classes
- Class1
- Class2
Example: KNN

k-NN with k=61

Decision region

PredictorA

PredictorB

0.2
0.4
0.6

0.2
0.4
0.6

classes

Class1

Class2

Decision boundary

PredictorA

PredictorB

0.2
0.4
0.6

0.2
0.4
0.6

classes

Class1

Class2
Example: KNN

k-NN with $k=69$

Decision region

Decision boundary

classes
- Class1
- Class2
Example: KNN
Example: KNN

k-NN with k=85

Decision region

Decision boundary

classes

Class1

Class2
Example: KNN

k-NN with k=101

Decision region

Decision boundary

classes
Class1
Class2

classes
Class1
Class2
Example: KNN

k-NN with $k=109$

Decision region

Decision boundary

classes
- Class1
- Class2

PredictorB

0.2 0.4 0.6

0.2 0.4 0.6

PredictorA

PredictorA
Example: KNN

k-NN with k=117

Decision region:
- PredictorA vs. PredictorB
- Classes: Class1 (red) and Class2 (blue)

Decision boundary:
- PredictorA vs. PredictorB
- Classes: Class1 (red) and Class2 (blue)
Example: KNN

k-NN with k=125

Decision region

Decision boundary

classes
- Class1
- Class2
Example: KNN

k-NN with k=133

Decision region

Decision boundary

classes
- Class1
- Class2
Example: KNN

k-NN with k=141

Decision region

Decision boundary

- Classes:
  - Class1
  - Class2
Example: KNN

k-NN with $k=149$

Decision region

classes
- Class 1: Red
- Class 2: Blue

Decision boundary

classes
- Class 1: Red circles
- Class 2: Blue circles
Example: KNN

k-NN with $k=157$

Decision region

Decision boundary
Example: KNN

k-NN with $k=165$

Decision region

```
PredictorB
0.6 -
0.4 -
0.2 -
0.2 0.4 0.6
PredictorA
```

Decision boundary

```
PredictorB
0.6 -
0.4 -
0.2 -
0.2 0.4 0.6
PredictorA
```

classes

- Class1
- Class2
Example: KNN

k-NN with k=173

Decision region

Decision boundary
Example: KNN

k-NN with $k=181$

- Decision region
- Decision boundary

Classes:
- Class1
- Class2
Example: KNN

k-NN with k=189

Decision region

Decision boundary

classes

Class1

Class2
Example: KNN

k-NN with k=197
Risk behaviour

- Learning/training risk (empirical risk on the learning/training set) decays when the complexity of the method increases.
- Quite different behavior when the risk is computed on new observations (generalization risk).
- Overfit for complex methods: parameters learned are too specific to the learning set!
- General situation! (Think of polynomial fit...)
- Need to use a different criterion than the training risk!
# Risk Estimation vs Method Selection

## Predictor Risk Estimation
- **Goal:** Given a predictor $f$ assess its quality.
- **Method:** Hold-out risk computation (Empirical risk correction).
- **Usage:** Compute an estimate of the risk of a selected $f$ using a test set to be used to monitor it in the future.

- Basic block very well understood.

## Method Selection
- **Goal:** Given a ML method assess its quality.
- **Method:** Cross Validation (Empirical risk correction)
- **Usage:** Compute risk estimates for several ML methods using training/validation sets to choose the most promising one.

- Estimates can be pointwise or better intervals.
- Multiple test issues in method selection.
Cross Validation and Empirical Risk Correction

Two Approaches

- **Cross validation**: Use empirical risk criterion but on independent data, very efficient (and almost always used in practice!) but slightly biased as its target uses only a fraction of the data.

- **Correction approach**: use empirical risk criterion but correct it with a term increasing with the complexity of $S$

\[ R_n(\hat{f}_S) \rightarrow R_n(\hat{f}_S) + \text{cor}(S) \]

and choose the method with the smallest corrected risk.

Which loss to use?

- The loss used in the risk: most natural!
- The loss used to estimate $\hat{\theta}$: penalized estimation!

- Other performance measure can be used.
Cross Validation

- **Very simple idea:** use a second learning/verification set to compute a verification risk.
- Sufficient to remove the dependency issue!
- Implicit random design setting...

Cross Validation

- Use $(1 - \epsilon) \times n$ observations to train and $\epsilon \times n$ to verify!
- Possible issues:
  - Validation for a learning set of size $(1 - \epsilon) \times n$ instead of $n$?
  - Unstable risk estimate if $\epsilon n$ is too small?

- Most classical variations:
  - Hold Out,
  - Leave One Out,
  - $V$-fold cross validation.
Hold Out

**Principle**
- Split the dataset $\mathcal{D}$ in 2 sets $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$ of size $n \times (1 - \epsilon)$ and $n \times \epsilon$.
- Learn $\hat{f}^{HO}$ from the subset $\mathcal{D}_{\text{train}}$.
- Compute the empirical risk on the subset $\mathcal{D}_{\text{test}}$:
  \[
  R^{HO}_n(\hat{f}^{HO}) = \frac{1}{n\epsilon} \sum_{(X_i, Y_i) \in \mathcal{D}_{\text{test}}} \ell(Y_i, \hat{f}^{HO}(X_i))
  \]

**Predictor Risk Estimation**
- Use $\hat{f}^{HO}$ as predictor.
- Use $R^{HO}_n(\hat{f}^{HO})$ as an estimate of the risk of this estimator.

**Method Selection by Cross Validation**
- Compute $R^{HO}_n(\hat{f}^{S,H0})$ for all the considered methods, where $\hat{f}^{S,H0}$ is the predictor from the selected method.
- Select the method with the smallest CV risk,
- Reestimate the $\hat{f}^{S}$ with all the data.
### Hold Out

#### Principle
- Split the dataset $\mathcal{D}$ in 2 sets $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$ of size $n \times (1 - \epsilon)$ and $n \times \epsilon$.
- Learn $\hat{f}^{\text{HO}}$ from the subset $\mathcal{D}_{\text{train}}$.
- Compute the empirical risk on the subset $\mathcal{D}_{\text{test}}$:
  \[
  R_{n}^{\text{HO}}(\hat{f}^{\text{HO}}) = \frac{1}{n\epsilon} \sum_{(X_i, Y_i) \in \mathcal{D}_{\text{test}}} \ell(Y_i, \hat{f}^{\text{HO}}(X_i))
  \]

- Only possible setting for risk estimation.

#### Hold Out Limitation for Method Selection
- Biased toward simpler method as the estimation does not use all the data initially.
- Learning variability of $R_{n}^{\text{HO}}(\hat{f}^{\text{HO}})$ not taken into account.
V-fold Cross Validation

Principle

- Split the dataset $D$ in $V$ sets $D_v$ of almost equals size.
- For $v \in \{1, \ldots, V\}$:
  - Learn $\hat{f}^{-v}$ from the dataset $D$ minus the set $D_v$.
  - Compute the empirical risk:
    \[ R_n^{-v}(\hat{f}^{-v}) = \frac{1}{n_v} \sum_{(X_i, Y_i) \in D_v} \ell(Y_i, \hat{f}^{-v}(X_i)) \]
- Compute the average empirical risk:
  \[ R_n^{CV}(\hat{f}) = \frac{1}{V} \sum_{v=1}^{V} R_n^{-v}(\hat{f}^{-v}) \]

- Estimation of the quality of a method not of a given predictor.
- Leave One Out : $V = n$. 
V-fold Cross Validation

Analysis (when \( n \) is a multiple of \( V \))

- The \( \mathcal{R}_n^{-V}(\hat{f}^{-V}) \) are identically distributed variable but are not independent!
- Consequence:

\[
\mathbb{E} \left[ \mathcal{R}_n^{CV}(\hat{f}) \right] = \mathbb{E} \left[ \mathcal{R}_n^{-V}(\hat{f}^{-V}) \right]
\]
\[
\text{Var} \left[ \mathcal{R}_n^{CV}(\hat{f}) \right] = \frac{1}{V} \text{Var} \left[ \mathcal{R}_n^{-V}(\hat{f}^{-V}) \right] + \left( 1 - \frac{1}{V} \right) \text{Cov} \left[ \mathcal{R}_n^{-V}(\hat{f}^{-V}), \mathcal{R}_n^{-V}(\hat{f}^{-V'}) \right]
\]

- Average risk for a sample of size \((1 - \frac{1}{V})n\).
- Variance term much more complex to analyze!
- Fine analysis shows that the larger \( V \) the better...

- Accuracy/Speed tradeoff: \( V = 5 \) or \( V = 10 \)...
Linear Regression and Leave One Out

- Leave One Out = $V$ fold for $V = n$: very expensive in general.

A fast LOO formula for the linear regression

- **Prop:** for the least squares linear regression,
  \[
  \hat{f}^{-i}(X_i) = \frac{\hat{f}(X_i) - h_{ii}Y_i}{1 - h_{ii}}
  \]
  with $h_{ii}$ the $i$th diagonal coefficient of the hat (projection) matrix.
- Proof based on linear algebra!
- Leads to a fast formula for LOO:
  \[
  \mathcal{R}_n^{LOO}(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} \frac{|Y_i - \hat{f}(X_i)|^2}{(1 - h_{ii})^2}
  \]
Example: KNN ($\hat{k} = 61$ using cross-validation)
**Train/Validation/Test**

- **Selection Bias Issue:**
  - After method selection, the cross validation is biased.
  - Furthermore, it qualifies the method and not the final predictor.
  - Need to (re)estimate the risk of the final predictor.

---

(Train/Validation)/Test strategy

- **Split** the dataset in two a (Train/Validation) and Test.
- Use **CV** with the (Train/Validation) to **select a method**.
- Train this method on (Train/Validation) to **obtain a single predictor**.
- Estimate the **performance of this predictor** on Test.

- Every choice made from the data is part of the method!
Empirical loss of an estimator computed on the dataset used to chose it is biased!
Empirical loss is an optimistic estimate of the true loss.

**Risk Correction Heuristic**
- Estimate an upper bound of this optimism for a given family.
- Correct the empirical loss by adding this upper bound.

**Rk:** Finding such an upper bound can be complicated!
Correction often called a **penalty**.
Penalization

Penalized Loss

- Minimization of
  \[ \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(X_i)) + \text{pen}(\theta) \]
  where \( \text{pen}(\theta) \) is a risk correction (penalty).

Penalties

- Upper bound of the optimism of the empirical loss
- Depends on the loss and the framework!

Instantiation

- Mallows Cp: Least Squares with \( \text{pen}(\theta) = 2 \frac{d}{n} \sigma^2 \).
- AIC Heuristics: Maximum Likelihood with \( \text{pen}(\theta) = \frac{d}{n} \).
- BIC Heuristics: Maximum Likelihood with \( \text{pen}(\theta) = \log(n) \frac{d}{n} \).
- Structural Risk Minimization: Pred. loss and clever penalty.
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Empirical Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
## Comparison of Two Means

### Means

- **Setting:** r.v. $e_i^{(l)}$ with $1 \leq i \leq n_l$ and $l \in \{1, 2\}$ and their means

$$
\bar{e^{(l)}} = \frac{1}{n_l} \sum_{i=1}^{n_l} e_i^{(l)}
$$

- **Question:** are the means $\bar{e^{(l)}}$ statistically different?

### Classical i.i.d setting

- **Assumption:** $e_i^{(l)}$ are i.i.d. for each $l$.

- **Test formulation:** Can we reject the null hypothesis that $\mathbb{E}[e^{(1)}] = \mathbb{E}[e^{(2)}]$?

- **Methods:**
  - Gaussian (Student) test using asymptotic normality of a mean.
  - Non-parametric permutation test.

- Gaussian approach is linked to confidence intervals.
- The larger $n_l$ the smaller the confidence intervals.
## Non i.i.d. case

- **Assumption:** $e^{(l)}_i$ are i.d. for each $l$ but not necessarily independent.

- **Test formulation:** Can we reject the null hypothesis that $\mathbb{E}[e^{(1)}] = \mathbb{E}[e^{(2)}]$?

- **Methods:**
  - Gaussian (Student) test using asymptotic normality of a mean but variance is hard to estimate.
  - Non-parametric permutation test but no confidence intervals.

- Setting for Cross Validation (other than holdout).
- Much more complicated than the i.i.d. case
Comparison of Several Means

Several means

- **Assumption:** $e_i^{(l)}$ are i.d. for each $l$ but not necessarily independent.
- **Tests formulation:**
  - Can we reject the null hypothesis that the $\mathbb{E}[e^{(l)}]$ are different?
  - Is the smaller mean statistically smaller than the second one?
- **Methods:**
  - Gaussian (Student) test using asymptotic normality of a mean with multiple tests correction.
  - Non-parametric permutation test but no confidence intervals.
- Setting for Cross Validation (other than holdout).
- The more models one compares:
  - the larger the confidence intervals
  - the most probable the best model is a lucky winner
- Justify the fallback to the simplest model that could be the best one.
PAC Approach

Cross-Validation risk: estimate of the average risk of a ML method.

No risk bound on the predictor obtained in practice.

Probably-Approximately-Correct (PAC) Approach

Replace the control on the average risk by a probabilistic bound

\[ \mathbb{P} \left( \mathbb{E} \left[ \ell(Y, \hat{f}(X)) \right] > R \right) \leq \epsilon \]

Requires estimating quantiles of the risk.
Cross Validation and Confidence Interval

- How to replace pointwise estimation by a confidence interval?
- Can we use the variability of the CV estimates?
- **Negative result:** No unbiased estimate of the variance!

**Gaussian Interval (Comparison of the means and \( \sim \) indep.)**

- Compute the empirical variance and divide it by the number of folds to construct an asymptotic Gaussian confidence interval,
- Select the simplest model whose value falls into the confidence interval of the model having the smallest CV risk.

**PAC approach (Quantile, \( \sim \) indep. and small risk estim. error)**

- Compute the raw medians (or a larger raw quantiles)
- Select the model having the smallest quantiles to ensure a small risk with high probability.

- Always reestimate the chosen model with all the data.
- To obtain an unbiased risk estimate of the final predictor: hold out risk on untouched test data.
Cross Validation
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     (Plain) Parametric Density Estimation
     Latent Variables
     Approximate Simulation
     Diffusion Model
     Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
References

T. Hastie, R. Tibshirani, and J. Friedman.  
*The Elements of Statistical Learning.*  
Springer Series in Statistics, 2009

M. Mohri, A. Rostamizadeh, and A. Talwalkar.  
*Foundations of Machine Learning (2nd ed.)*  
MIT Press, 2018

A. Géron.  
*Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (3rd ed.)*  
O’Reilly, 2022

Ch. Giraud.  
*Introduction to High-Dimensional Statistics.*  
CRC Press, 2014

S. Bubeck.  
*Convex Optimization: Algorithms and Complexity.*  
Now Publisher, 2015
Licence and Contributors

Creative Commons Attribution-ShareAlike (CC BY-SA 4.0)

- You are free to:
  - Share: copy and redistribute the material in any medium or format
  - Adapt: remix, transform, and build upon the material for any purpose, even commercially.
- Under the following terms:
  - Attribution: You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - ShareAlike: If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - No additional restrictions: You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Contributors

- Main contributor: E. Le Pennec
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9 References
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9 References
Motivation

Ozone pollution

- **Data:** Air Breizh, Summer 2001
- **Input:** Temperature at 12h00
- **Output:** max Ozone concentration

Source: Air Breizh

Data: Air Breizh, Summer 2001
Input: Temperature at 12h00
Output: max Ozone concentration

Source: Air Breizh
Motivation

Credit Default, Credit Score, Bank Risk, Market Risk Management

- **Data:** Client profile, Client credit history...
- **Input:** Client profile
- **Output:** Credit risk
Motivation

Marketing: advertisement, recommendation...

- **Data**: User profile, Web site history...
- **Input**: User profile, Current web page
- **Output**: Advertisement with price, recommendation...

Spam detection (Text classification)
A definition by Tom Mitchell

A computer program is said to learn from experience $E$ with respect to some class of tasks $T$ and performance measure $P$, if its performance at tasks in $T$, as measured by $P$, improves with experience $E$. 
## Supervised Learning

### Supervised Learning Framework

- Input measurement $X \in \mathcal{X}$
- Output measurement $Y \in \mathcal{Y}$.
- $(X, Y) \sim P$ with $P$ unknown.
- **Training data**: $D_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ (i.i.d. $\sim P$)

- Often
  - $X \in \mathbb{R}^d$ and $Y \in \{-1, 1\}$ (classification)
  - or $X \in \mathbb{R}^d$ and $Y \in \mathbb{R}$ (regression).
- A **predictor** is a function in $\mathcal{F} = \{f : \mathcal{X} \to \mathcal{Y} \text{ meas.}\}$

### Goal

- Construct a **good** predictor $\hat{f}$ from the training data.

- Need to specify the meaning of good.
- Classification and regression are almost the **same** problem!
Loss and Probabilistic Framework

Loss function for a generic predictor

- **Loss function**: $\ell(Y, f(X))$ measures the goodness of the prediction of $Y$ by $f(X)$
- **Examples**:
  - 0/1 loss: $\ell(Y, f(X)) = 1_{Y \neq f(X)}$
  - Quadratic loss: $\ell(Y, f(X)) = |Y - f(X)|^2$

Risk function

- Risk measured as the average loss for a new couple:
  \[
  \mathcal{R}(f) = \mathbb{E}_{(X, Y) \sim P}[\ell(Y, f(X))]
  \]
- **Examples**:
  - 0/1 loss: $\mathbb{E}[\ell(Y, f(X))] = P(Y \neq f(X))$
  - Quadratic loss: $\mathbb{E}[\ell(Y, f(X))] = \mathbb{E}[|Y - f(X)|^2]$
- **Beware**: As $\hat{f}$ depends on $D_n$, $\mathcal{R}(\hat{f})$ is a random variable!
The best solution $f^*$ (which is independent of $D_n$) is

$$f^* = \arg \min_{f \in F} \mathcal{R}(f) = \arg \min_{f \in F} \mathbb{E}[\ell(Y, f(X))] = \arg \min_{f \in F} \mathbb{E}_X \left[ \mathbb{E}_{Y|X}[\ell(Y, f(X))] \right]$$

**Bayes Predictor (explicit solution)**

- In binary classification with $0-1$ loss:

$$f^*(X) = \begin{cases} 
+1 & \text{if } \mathbb{P}(Y = +1|X) \geq \mathbb{P}(Y = -1|X) \\
\Leftrightarrow \mathbb{P}(Y = +1|X) \geq 1/2 \\
-1 & \text{otherwise}
\end{cases}$$

- In regression with the quadratic loss

$$f^*(X) = \mathbb{E}[Y|X]$$

**Issue:** Solution requires to know $\mathbb{E}[Y|X]$ for all values of $X$!
ML Methods: Probabilistic Point of View

**Goal**

Machine Learning

- Learn a rule to construct a **predictor** $\hat{f} \in \mathcal{F}$ from the training data $\mathcal{D}_n$ s.t. the risk $R(\hat{f})$ is **small on average** or with high probability with respect to $\mathcal{D}_n$.

- In practice, the rule should be an algorithm!

**Canonical example: Empirical Risk Minimizer**

- One restricts $f$ to a subset of functions $\mathcal{S} = \{f_\theta, \theta \in \Theta\}$
- One replaces the minimization of the average loss by the minimization of the empirical loss

$$\hat{f} = f_{\hat{\theta}} = \text{argmin}_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(X_i))$$

- Examples:
  - Linear regression
  - Linear classification with

$$\mathcal{S} = \{x \mapsto \text{sign}\{x^T \beta + \beta(0)\} / \beta \in \mathbb{R}^d, \beta(0) \in \mathbb{R}\}$$
Example: TwoClass Dataset

**Synthetic Dataset**

- Two features/covariates.
- Two classes.

- Dataset from *Applied Predictive Modeling*, M. Kuhn and K. Johnson, Springer
- Numerical experiments with R and the {caret} package.
Example: Linear Discrimination

Logistic

Decision region

Decision boundary

classes

Class1
Class2

classes

Class1
Class2
Example: More Complex Model

Naive Bayes with kernel density estimates

Decision region

Decision boundary

classes
Class1
Class2

classes
Class1
Class2
Bias-Variance Dilemma

- **General setting:**
  - $\mathcal{F} = \{\text{measurable functions } \mathcal{X} \rightarrow \mathcal{Y}\}$
  - Best solution: $f^* = \arg\min_{f \in \mathcal{F}} \mathcal{R}(f)$
  - Class $\mathcal{S} \subset \mathcal{F}$ of functions
  - Ideal target in $\mathcal{S}$: $f^*_\mathcal{S} = \arg\min_{f \in \mathcal{S}} \mathcal{R}(f)$
  - Estimate in $\mathcal{S}$: $\hat{f}_\mathcal{S}$ obtained with some procedure

**Approximation error and estimation error (Bias/Variance)**

$$\mathcal{R}(\hat{f}_\mathcal{S}) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(f^*_\mathcal{S}) - \mathcal{R}(f^*)}_{\text{Approximation error}} + \underbrace{\mathcal{R}(\hat{f}_\mathcal{S}) - \mathcal{R}(f^*_\mathcal{S})}_{\text{Estimation error}}$$

- Approx. error can be large if the model $\mathcal{S}$ is not suitable.
- Estimation error can be large if the model is complex.
Under-fitting / Over-fitting Issue

- Different behavior for different model complexity
- **Low complexity model** are easily learned but the approximation error (**bias**) may be large (**Under-fit**).
- **High complexity model** may contain a good ideal target but the estimation error (**variance**) can be large (**Over-fit**)

**Bias-variance trade-off** \(\iff\) avoid **overfitting** and **underfitting**

- **Rk:** Better to think in term of method (including feature engineering and specific algorithm) rather than only of model.
Probabilistic and Optimization Framework

How to find a good function $f$ with a small risk

$$ R(f) = \mathbb{E}[\ell(Y, f(X))] $$

**Canonical approach:**

$$ \hat{f}_S = \arg\min_{f \in S} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i)) $$

**Problems**

- How to choose $S$?
- How to compute the minimization?

**A Probabilistic Point of View**

**Solution:** For $X$, estimate $Y|X$ plug this estimate in the Bayes classifier:

(Generalized) Linear Models, Kernel methods, $k$-nn, Naive Bayes, Tree, Bagging...

**An Optimization Point of View**

**Solution:** If necessary replace the loss $\ell$ by an upper bound $\tilde{\ell}$ and minimize the empirical loss: SVR, SVM, Neural Network, Tree, Boosting...
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
     - A Probabilistic Point of View
     - Parametric Conditional Density Modeling
     - Non Parametric Conditional Density Modeling
     - Generative Modeling
     - Model Selection
     - Penalization
     - References

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights
   - Optimization: Gradient Descent Algorithms
     - Introduction
     - Gradient Descent
     - Proximal Descent
     - Coordinate Descent
     - Gradient Descent Acceleration
     - Stochastic Gradient Descent

4. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - Reference

5. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

6. Unsupervised Learning: Dimension Reduction and Clustering
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

7. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

8. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

9. References
## Three Classical Methods in a Nutshell

### Logistic Regression
- Let \( f_\theta(X) = X^\top \beta + \beta^{(0)} \) with \( \theta = (\beta, \beta^{(0)}) \).
- Let \( P_\theta(Y = 1|X) = \frac{e^{-f_\theta(X)}}{1 + e^{f_\theta(X)}} \).
- Estimate \( \theta \) by \( \hat{\theta} \) using a Maximum Likelihood.
- Classify using \( P_{\hat{\theta}}(Y = 1|X) > 1/2 \).

### \( k \) Nearest Neighbors
- For any \( X' \), define \( V_{X'} \) as the \( k \) closest samples \( X_i \) from the dataset.
- Compute a score \( g_k = \sum_{X_i \in V_{X'}} 1_{Y_i = k} \).
- Classify using \( \text{arg max } g_k \) (majority vote).
Three Classical Methods in a Nutshell

Quadratic Discriminant Analysis

- For each class, estimate the mean $\mu_k$ and the covariance matrix $\Sigma_k$.
- Estimate the proportion $\mathbb{P}(Y = k)$ of each class.
- Compute a score $\ln(\mathbb{P}(X|Y = k)) + \ln(\mathbb{P}(Y = k))$
  \[
  g_k(X) = -\frac{1}{2}(X - \mu_k)^\top \Sigma_k^{-1} (X - \mu_k)
  - \frac{d}{2} \ln(2\pi) - \frac{1}{2} \ln(|\Sigma_k|) + \ln(\mathbb{P}(Y = k))
  \]
- Classify using $\text{arg max } g_k$

Those three methods rely on a similar heuristic: the probabilistic point of view!
The best solution $f^*$ (which is independent of $D_n$) is

$$f^* = \arg \min_{f \in F} R(f) = \arg \min_{f \in F} \mathbb{E}[\ell(Y, f(X))] = \arg \min_{f \in F} \mathbb{E}_X \mathbb{E}_{Y|X}[\ell(Y, f(X))]$$

### Bayes Predictor (explicit solution)

- In binary classification with 0–1 loss:
  $$f^*(X) = \begin{cases} +1 & \text{if } \mathbb{P}(Y = +1|X) \geq \mathbb{P}(Y = -1|X) \\ -1 & \text{otherwise} \end{cases}$$
  $$\Leftrightarrow \mathbb{P}(Y = +1|X) \geq 1/2$$

- In regression with the quadratic loss
  $$f^*(X) = \mathbb{E}[Y|X]$$

**Issue:** Explicit solution requires to know $Y|X$ (or $\mathbb{E}[Y|X]$) for all values of $X$!
**Plugin Predictor**

- **Idea**: Estimate $Y | X$ by $\hat{Y} | X$ and plug it the Bayes classifier.

**Plugin Bayes Predictor**

- In binary classification with $0 - 1$ loss:
  \[
  \hat{f}(X) = \begin{cases} 
  +1 & \text{if } P(Y = +1 | X) \geq P(Y = -1 | X) \\
  -1 & \text{otherwise}
  \end{cases}
  \]

- In regression with the quadratic loss
  \[
  \hat{f}(X) = \mathbb{E}[Y | X]
  \]

- **Rk**: Direct estimation of $\mathbb{E}[Y | X]$ by $\mathbb{E}[\hat{Y} | X]$ also possible...

How to estimate $Y|X$?

**Three main heuristics**

- **Parametric Conditional modeling**: Estimate the law of $Y|X$ by a parametric law $L_{\theta}(X)$: (generalized) linear regression...

- **Non Parametric Conditional modeling**: Estimate the law of $Y|X$ by a non parametric estimate: kernel methods, loess, nearest neighbors...

- **Fully Generative modeling**: Estimate the law of $(X, Y)$ and use the Bayes formula to deduce an estimate of $Y|X$: LDA/QDA, Naive Bayes...

- **Rk**: Direct estimation of $\mathbb{E}[Y|X]$ by $\mathbb{E}[Y|X]$ also possible...
Plugin Classifier

- **Input**: a data set $\mathcal{D}_n$
  
  Learn $Y|X$ or equivalently $\mathbb{P}(Y = k|X)$ (using the data set) and plug this estimate in the Bayes classifier

- **Output**: a classifier $\hat{f} : \mathbb{R}^d \rightarrow \{-1, 1\}$

  $$\hat{f}(X) = \begin{cases} 
  +1 & \text{if } \mathbb{P}(Y = 1|X) \geq \mathbb{P}(Y = -1|X) \\
  -1 & \text{otherwise} 
  \end{cases}$$

- Can we guaranty that the classifier is good if $Y|X$ is well estimated?
Classification Risk Analysis

**Theorem**

If \( \hat{f} = \text{sign}(2\hat{p}_1 - 1) \) then

\[
E\left[ \ell^{0,1}(Y, \hat{f}(X)) \right] - E\left[ \ell^{0,1}(Y, f^*(X)) \right] \\
\leq E\left[ \| \hat{Y}|X - Y|X \|_1 \right] \\
\leq \left( E\left[ 2KL(Y|X, \hat{Y}|X) \right] \right)^{1/2}
\]

- If one estimates \( P(Y = 1|X) \) well then one estimates \( f^* \) well!
- Link between a *conditional density estimation* task and a *classification* one!
- **Rk:** In general, the conditional density estimation task is more complicated as one should be good for all values of \( P(Y = 1|X) \) while the classification task focus on values around 1/2 for the 0/1 loss!
- In *regression*, (often) direct control of the quadratic loss...
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   • Introduction
   • Machine Learning
   • Supervised Learning
   • Risk Estimation and Model Selection
   • Cross Validation and Test
   • References

2 ML Methods: Probabilistic Point of View
   • Motivation
   • Supervised Learning
   • A Probabilistic Point of View
     • Parametric Conditional Density Modeling
       • Non Parametric Conditional Density Modeling
       • Generative Modeling
       • Model Selection
       • Penalization

3 ML Methods: Optimization Point of View
   • Supervised Learning
   • Optimization Point of View
     • SVM
     • Penalization
     • Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   • Introduction
   • Gradient Descent
   • Proximal Descent
   • Coordinate Descent
   • Gradient Descent Acceleration
   • Stochastic Gradient Descent

5 Gradient Descent Step
   • Non-Convex Setting
   • References

6 ML Methods: Neural Networks and Deep Learning
   • Introduction
   • From Logistic Regression to NN
   • NN Optimization
   • NN Regularization
   • Image and CNN
   • Text, Recurrent Neural Networks and Transformers
   • NN Architecture
   • References

7 ML Methods: Trees and Ensemble Methods
   • Trees
   • Bagging and Random Forests
     • Bootstrap and Bagging
     • Randomized Rules and Random Forests
   • Boosting
     • AdaBoost as a Greedy Scheme
     • Boosting
   • Ensemble Methods
   • References

8 Unsupervised Learning: Dimension Reduction and Clustering
   • Unsupervised Learning?
   • A First Glimpse
     • Clustering
     • Dimensionality Curse
     • Dimension Reduction
     • Generative Modeling

9 Dimension Reduction
   • Simplification
   • Reconstruction Error
   • Relationship Preservation
   • Comparing Methods?

10 Clustering
    • Prototype Approaches
    • Contiguity Approaches
    • Agglomerative Approaches
    • Other Approaches
    • Scalability

11 Generative Modeling
    • (Plain) Parametric Density Estimation
    • Latent Variables
    • Approximate Simulation
    • Diffusion Model
    • Generative Adversarial Network

12 Applications to Text
    • References

13 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
    • Supervised Learning
    • Empirical Risk Minimization
      • Empirical Risk Minimization
      • ERM and PAC Analysis
      • Hoeffding and Finite Class
      • McDiarmid and Rademacher Complexity
      • VC Dimension
      • Structural Risk Minimization
    • References
**Parametric Conditional Density Models**

- **Idea:** Estimate directly $Y|X$ by a parametric conditional density $P_\theta(Y|X)$.

**Maximum Likelihood Approach**

- Classical choice for $\theta$:
  \[
  \hat{\theta} = \arg\min_{\theta} \sum_{i=1}^{n} \log P_\theta(Y_i|X_i)
  \]

- **Goal:** Minimize the Kullback-Leibler divergence between the conditional law of $Y|X$ and $P_\theta(Y|X)$
  \[
  E[KL(Y|X, P_\theta(Y|X))]
  \]

- **Rk:** This is often not (exactly) the learning task!
- Large choice for the family $\{P_\theta(Y|X)\}$ but depends on $Y$ (and $X$).
- **Regression:** One can also model directly $E[Y|X]$ by $f_\theta(X)$ and estimate it with a least-squares criterion...
Linear Conditional Density Models

Linear Models

- Classical choice: \( \theta = (\theta', \varphi) \)
  \[
  P_\theta(Y|X) = P_{X^T \beta, \varphi}(Y)
  \]
- Very strong assumption!

- Classical examples:
  - Binary variable: logistic, probit...
  - Discrete variable: multinomial logistic regression...
  - Integer variable: Poisson regression...
  - Continuous variable: Gaussian regression...
Binary Classifier

**Plugin Linear Classification**

- Model $\mathbb{P}(Y = +1|X)$ by $h(X^T \beta + \beta^{(0)})$ with $h$ non decreasing.
- $h(X^T \beta + \beta^{(0)}) > 1/2 \iff X^T \beta + \beta^{(0)} - h^{-1}(1/2) > 0$
- Linear Classifier: $\text{sign}(X^T \beta + \beta^{(0)} - h^{-1}(1/2))$

**Plugin Linear Classifier Estimation**

- Classical choice for $h$:
  
  $h(t) = \frac{e^t}{1 + e^t}$  
  logit or logistic
  
  $h(t) = \Phi_N(t)$  
  probit
  
  $h(t) = 1 - e^{-e^t}$  
  log-log

- Choice of the best $\beta$ from the data.
### Maximum Likelihood Estimate

#### Probabilistic Model

- By construction, \( Y|X \) follows \( B(\mathbb{P}(Y = +1|X)) \)
- Approximation of \( Y|X \) by \( B(h(x^\top \beta + \beta^{(0)})) \)
- *Natural* probabilistic choice for \( \beta \): maximum likelihood estimate.
- *Natural* probabilistic choice for \( \beta \): \( \beta \) approximately minimizing a distance between \( B(h(x^\top \beta)) \) and \( B(\mathbb{P}(Y = 1|X)) \).

#### Maximum Likelihood Approach

- Minimization of the negative log-likelihood:
  \[
  - \sum_{i=1}^{n} \log(\mathbb{P}(Y_i|X_i)) = - \sum_{i=1}^{n} \left( 1_{Y_i=1} \log(h(X_i^\top \beta)) + 1_{Y_i=-1} \log(1 - h(X_i^\top \beta)) \right)
  \]
- Minimization possible if \( h \) is regular...
Maximum Likelihood Estimate

KL Distance and negative log-likelihood

- **Natural distance**: Kullback-Leibler divergence
  \[
  \text{KL}(\mathcal{B}(\mathbb{P}(Y = 1|X)), \mathcal{B}(h(X^\top \beta)))
  = \mathbb{E}_X \left[ \mathbb{P}(Y = 1|X) \log \frac{\mathbb{P}(Y = 1|X)}{h(X^\top \beta)} 
  + \mathbb{P}(Y = -1|X) \log \frac{1 - \mathbb{P}(Y = 1|X)}{1 - h(X^\top \beta)} \right]
  = \mathbb{E}_X \left[ -\mathbb{P}(Y = 1|X) \log(h(X^\top \beta)) 
  - \mathbb{P}(Y = -1|X) \log(1 - h(X^\top \beta)) \right] + C_{X,Y}
  
- **Empirical counterpart** = negative log-likelihood (up to 1/n factor):
  \[
  - \frac{1}{n} \sum_{i=1}^{n} \left( 1_{Y_i = 1} \log(h(X_i^\top \beta)) + 1_{Y_i = -1} \log(1 - h(X_i^\top \beta)) \right)
  \]
Logistic Regression

Logistic Regression and Odd

- Logistic model: \( h(t) = \frac{e^t}{1+e^t} \) (most natural choice...)
- The Bernoulli law \( \mathcal{B}(h(t)) \) satisfies then
  \[
  \frac{P(Y = 1)}{P(Y = -1)} = e^t \iff \log \frac{P(Y = 1)}{P(Y = -1)} = t
  \]
- Interpretation in term of odd.
- Logistic model: linear model on the logarithm of the odd
  \[
  \log \frac{P(Y = 1|X)}{P(Y = -1|X)} = X^T \beta
  \]

Associated Classifier

- Plugin strategy:
  \[
  f_\beta(X) = \begin{cases} 
  1 & \text{if } \frac{e^{X^T \beta}}{1 + e^{X^T \beta}} > 1/2 \iff X^T \beta > 0 \\
  -1 & \text{otherwise}
  \end{cases}
  \]
Logistic Regression and Minimization

Likelihood Rewriting

- Negative log-likelihood:

\[- \frac{1}{n} \sum_{i=1}^{n} \left( \mathbf{1}_{Y_i=1} \log(h(X_i^T \beta)) + \mathbf{1}_{Y_i=-1} \log(1 - h(X_i^T \beta)) \right)\]

\[= -\frac{1}{n} \sum_{i=1}^{n} \left( \mathbf{1}_{Y_i=1} \log \frac{e^{X_i^T \beta}}{1 + e^{X_i^T \beta}} + \mathbf{1}_{Y_i=-1} \log \frac{1}{1 + e^{X_i^T \beta}} \right)\]

\[= \frac{1}{n} \sum_{i=1}^{n} \log \left( 1 + e^{-Y_i(X_i^T \beta)} \right)\]

- Convex and smooth function of $\beta$
- Easy optimization.
Example: Logistic
Feature Design

Transformed Representation

- From $X$ to $\Phi(X)$!
- New description of $X$ leads to a different **linear** model:
  $$f_\beta(X) = \Phi(X)^T \beta$$

Feature Design

- Art of choosing $\Phi$.
- Examples:
  - Renormalization, (domain specific) transform
  - Basis decomposition
  - Interaction between different variables...
Example: Quadratic Logistic

Quadratic Logistic

Decision region

Decision boundary

classes
- Class1
- Class2
Gaussian Linear Regression

Gaussian Linear Model

- **Model:** $Y|X \sim \mathcal{N}(X^\top \beta, \sigma^2)$ plus independence
- Probably the most classical model of all time!
- Maximum Likelihood with explicit formulas for the two parameters.

- In regression, estimation of $\mathbb{E}[Y|X]$ is sufficient: other/no model for the noise possible.
Extension of Gaussian Linear Regression

Generalized Linear Model

- Model entirely characterized by its mean (up to a scalar nuisance parameter) \( \nu(\mathbb{E}_\theta[Y]) = \theta \) with \( \nu \) invertible.

- Exponential family: Probability law family \( P_\theta \) such that the density can be written
  \[
  f(y, \theta, \varphi) = e^{\frac{y\theta - \nu(\theta)}{\varphi} + w(y, \varphi)}
  \]
  where \( \varphi \) is a nuisance parameter and \( w \) a function independent of \( \theta \).

- Examples:
  - Gaussian: \( f(y, \theta, \varphi) = e^{-\frac{y\theta - \theta^2/2}{\varphi} - \frac{y^2/2}{\varphi}} \)
  - Bernoulli: \( f(y, \theta) = e^{y\theta - \ln(1+e^\theta)} \) (\( \theta = \ln p/(1-p) \))
  - Poisson: \( f(y, \theta) = e^{(y\theta - e^\theta) + \ln(y!)} \) (\( \theta = \ln \lambda \))

- Linear Conditional model: \( Y|X \sim P_{x^\top \beta} \ldots \)

- ML fit of the parameters
Non Parametric Conditional Estimation

- **Idea:** Estimate $Y|X$ or $\mathbb{E}[Y|X]$ directly without resorting to an explicit parametric model.

Two heuristics:
- $Y|X$ (or $\mathbb{E}[Y|X]$) is almost constant (or simple) in a neighborhood of $X$. (Kernel methods)
- $Y|X$ (or $\mathbb{E}[Y|X]$) can be approximated by a model whose dimension depends on the complexity and the number of observation. (Quite similar to parametric model plus model selection...)

- Focus on **kernel methods**!
Kernel Methods

**Idea:** The behavior of $Y|X$ is locally *constant* or simple!

### Kernel

- Choose a kernel $K$ (think of a weighted neighborhood).
- For each $\tilde{X}$, compute a simple localized estimate of $Y|X$.
- Use this local estimate to take the decision.

- In regression, estimation of $\mathbb{E}[Y|X]$ is sufficient.
Example: $k$ Nearest-Neighbors (with $k = 3$)
Example: \( k \) Nearest-Neighbors (with \( k = 4 \))
**k Nearest-Neighbors**

- Neighborhood $\mathcal{V}_x$ of $x$: $k$ learning samples closest from $x$.

**k-NN as local conditional density estimate**

$$
\mathbb{P}(\hat{Y} = 1|X) = \frac{\sum_{x_i \in \mathcal{V}_x} \mathbf{1}_{\{y_i=+1\}}}{|\mathcal{V}_x|}
$$

- KNN Classifier:

$$
\hat{f}_{KNN}(X) = \begin{cases} 
+1 & \text{if } \mathbb{P}(\hat{Y} = 1|X) \geq \mathbb{P}(\hat{Y} = -1|X) \\
-1 & \text{otherwise}
\end{cases}
$$

- **Lazy learning**: all the computations have to be done at prediction time.

- **Remark**: You can also use your favorite kernel estimator...
Example: KNN

k-NN with k=1

Decision region

Decision boundary

classes
- Class1
- Class2
Example: KNN
Example: KNN

k-NN with $k=9$

Decision region

Decision boundary

classes
- Class1
- Class2
Example: KNN

k-NN with k=13

Decision region

PredictorB

0.6

0.4

0.2

PredictorA

0.2

0.4

0.6

Decision boundary

PredictorB

0.6

0.4

0.2

0.2

0.4

0.6

classes

Class1

Class2

classes

Class1

Class2
Example: KNN

k-NN with $k=17$

Decision region

**classes**
- Class 1: Red
- Class 2: Blue

Decision boundary

**classes**
- Class 1: Red
- Class 2: Blue
Example: KNN

k-NN with $k=21$

Decision region

Decision boundary

classes

- Class 1
- Class 2
Example: KNN

k-NN with k=25

Decision region

Decision boundary

classes

Class1
Class2

classes

Class1
Class2
Example: KNN

k-NN with $k=29$

Decision region

Decision boundary

classes

Class1

Class2
Example: KNN

k-NN with $k=33$

Decision region

Decision boundary

classes

- Class1
- Class2

- Class1
- Class2
Example: KNN

k-NN with k=37

Decision region

Decision boundary

classes
Class1
Class2

classes
Class1
Class2
Example: KNN

k-NN with $k=45$

Decision region

Decision boundary
Example: KNN

k-NN with k=53

Decision region

Decision boundary

classes
Class1
Class2

classes
Class1
Class2
Example: KNN

k-NN with k=61

Decision region

Decision boundary

classes
- Class1
- Class2
Example: KNN

**k-NN with k=69**

- **Decision region**
- **Decision boundary**

Classes:
- Class 1: Red
- Class 2: Blue
Example: KNN

k-NN with k=77

Decision region

Decision boundary

classes
- Class1
- Class2
Example: KNN

k-NN with k=85

Decision region

- Classes
  - Class 1
  - Class 2

Decision boundary

- Classes
  - Class 1
  - Class 2
Example: KNN
Example: KNN
Example: KNN

**k-NN with k=117**

**Decision region**

**Decision boundary**

**Classes:**
- Class 1 (Red)
- Class 2 (Blue)
Example: KNN

k-NN with k=125

Decision region

Decision boundary

classes
- Class1
- Class2
Example: KNN

k-NN with k=133

Decision region

Decision boundary

classes
- Class1
- Class2

classes
- Class1
- Class2
Example: KNN

k-NN with k=141

Decision region

Decision boundary

classes

Class1

Class2
Example: KNN

k-NN with k=149

Decision region

Decision boundary

classes

Class1

Class2

classes

Class1

Class2
Example: KNN

k-NN with k=157

Decision region

Decision boundary

classes

Class1
Class2

classes

Class1
Class2
Example: KNN

k-NN with k=165

Decision region

Decision boundary
Example: KNN

k-NN with $k=173$

Decision region

Decision boundary

classes

Class1

Class2

classes

Class1

Class2
Example: KNN

k-NN with k=181

Decision region

Decision boundary

classes
- Class1
- Class2

PredictorB

PredictorA
Example: KNN

k-NN with k=189

Decision region

Decision boundary

classes

Class1

Class2
Example: KNN

k-NN with $k=197$

Decision region

Decision boundary
A naive idea

- $E[Y|X]$ can be approximated by a local average:

$$\hat{f}(X) = \frac{1}{|\{X_i \in \mathcal{N}(X)\}|} \sum_{X_i \in \mathcal{N}(X)} Y_i$$

where $B(X)$ is a neighborhood of $X$.

**Heuristic:**

- If $X \to E[Y|X]$ is regular then

$$E[Y|X] \approx E[E[Y|X']|X' \in \mathcal{N}(X)] = E[Y|X' \in \mathcal{N}(X)]$$

- Replace an expectation by an empirical average:

$$E[Y|X' \in \mathcal{N}(X)] \approx \frac{1}{|\{X_i \in \mathcal{N}(X)\}|} \sum_{X_i \in \mathcal{N}(X)} Y_i$$
Regression and Local Averaging

**Neighborhood and Size**

- Most classical choice: \( \mathcal{N}(X) = \{X', \|X - X'\| \leq h\} \) where \( \|\cdot\| \) is a (pseudo) norm and \( h \) a size (bandwidth) parameter.
- In principle, the norm and \( h \) could vary with \( X \), and the norm can be replaced by a (pseudo) distance.
- Focus here on a fixed distance with a fixed bandwidth \( h \) case.

**Bandwidth Heuristic**

- A large bandwidth ensures that the average is taken on many samples and thus the variance is small...
- A small bandwidth is thus that the approximation \( \mathbb{E}[Y|X] \approx \mathbb{E}[Y|X' \in \mathcal{N}(X)] \) is more accurate (small bias).
**Weighted Local Averaging**

**Weighted Local Average**

- Replace the neighborhood $\mathcal{N}(X)$ by a decaying **window function** $w(X, X')$.
- $\mathbb{E}[Y|X]$ can be approximated by a **weighted local average**:
  \[
  \hat{f}(X) = \frac{\sum w(X, X') Y_i}{\sum w(X, X')}.\]

**Kernel**

- Most classical choice: $w(X, X') = K \left( \frac{X-X'}{h} \right)$ where $h$ the bandwidth is a scale parameter.
- Examples:
  - **Box kernel**: $K(t) = 1_{\|t\|\leq 1}$ (Neighborhood)
  - **Triangular kernel**: $K(t) = \max(1 - \|t\|, 0)$.
  - **Gaussian kernel**: $K(t) = e^{-t^2/2}$
- **Rk**: $K$ and $\lambda K$ yields the same estimate.
Density Estimation

- How to estimate the density $p$ of $X$ with respect to the Lebesgue measure from an i.i.d. sample $(X_1, \ldots, X_n)$.

- **Parametric approach**: density has a known parameterized shape and estimate those parameters.

- **Nonparametric approach**: density has a no known parameterized shape and
  - Approximate it by a parametric one, whose parameters can be estimated
  - Estimate directly the density

- Important **nonparametric statistic topic**!

- Used in generative modeling...
Link with Density Estimation

Kernel Density Estimation (Parzen)

- Choose a positive kernel $K$ such that $\int K(x)dx = 1$
- Use as an estimate

$$\hat{p}(X) = \frac{1}{n} \sum_{i=1}^{n} K(X - X_i)$$

- If $K = \frac{1}{Z_h} 1_{\|t\| \leq h}$, easy interpretation as a local empirical density of samples!
- General $K$ corresponds to a smoothed version.
- Often $K_h(t) = \frac{1}{h^d} K(t/h)$ and let

$$\hat{p}_h(X) = \frac{1}{n} \sum_{i=1}^{n} K_h(X - X_i)$$
Properties

- **Error decomposition:**
  \[
  \mathbb{E} \left[ |p(X) - \hat{p}_h(X)|^2 \right] = \mathbb{E} [p(X) - \hat{p}_h(X)]^2 + \text{Var} [p(X) - \hat{p}_h(X)]
  \]

- **Bias:**
  \[
  \mathbb{E} [p(X) - \hat{p}_h(X)] = p(X) - (K_h * p)(X)
  \]

- **Variance:** if \( p \) is upper bounded by \( p_{\text{max}} \) then
  \[
  \text{Var} [p(X) - \hat{p}_h(X)] \leq \frac{p_{\text{max}} \int K_h^2(x) dx}{n h^d}
  \]
Link with Density Estimation

Bandwidth choice

- A small $h$ leads to a small bias but a large variance...
- A large $h$ leads to a small variance but a large bias...
- Theoretical analysis possible!
From Density Estimation to Regression

Nadaraya-Watson Heuristic

- Provided all the **densities** exist

\[
\mathbb{E}[Y|X] = \frac{\int Y \hat{p}(X, Y) dY}{\int \hat{p}(Y, X) dY} = \frac{\int Y p(X, Y) dY}{p(X)}
\]

- Replace the unknown densities by their **estimates**:

\[
\hat{p}(X) = \frac{1}{n} \sum_{i=1}^{n} K(X - X_i)
\]

\[
\hat{p}(X, Y) = \frac{1}{n} \sum_{i=1}^{n} K(X - X_i) K'(Y - Y_i)
\]

- Now if \( K' \) is a kernel such that \( \int Y K'(Y) dY = 0 \) then

\[
\int Y \hat{p}(X, Y) dY = \frac{1}{n} \sum_{i=1}^{n} K(X - X_i) Y_i
\]
From Density Estimation to Regression

Nadaraya-Watson

- Resulting estimator of $\mathbb{E}[Y|X]$

$$\hat{f}(X) = \frac{\sum_{i=1}^{n} Y_i K_h(X - X_i)}{\sum_{i=1}^{n} K_h(X - X_i)}$$

- Same local weighted average estimator!

Bandwidth Choice

- Bandwidth $h$ of $K$ allows to balance between bias and variance.
- Theoretical analysis of the error is possible.
- The smoother the densities the easier the estimation but the optimal bandwidth depends on the unknown regularity!
Local Linear Estimation

Another Point of View on Kernel

- Nadaraya-Watson estimator:
  \[
  \hat{f}(X) = \frac{\sum_{i=1}^{n} Y_i K_h(X - X_i)}{\sum_{i=1}^{n} K_h(X - X_i)}
  \]

- Can be viewed as a minimizer of
  \[
  \sum_{i=1}^{n} \left| Y_i - \beta \right|^2 K_h(X - X_i)
  \]

- Local regression of order 0.

Local Linear Model

- Estimate \( \mathbb{E}[Y|X] \) by \( \hat{f}(X) = \phi(X)^T \hat{\beta}(X) \) where \( \phi \) is any function of \( X \) and \( \hat{\beta}(X) \) is the minimizer of
  \[
  \sum_{i=1}^{n} \left| Y_i - \phi(X_i)^T \beta \right|^2 K_h(X - X_i).
  \]
1D Nonparametric Regression

- Assume that $X \in \mathbb{R}$ and let $\phi(X) = (1, X, \ldots, X^d)$.
- **LOESS estimate:** $\hat{f}(X) = \sum_{j=0}^{d} \hat{\beta}(X^{(j)}) X^j$ with $\hat{\beta}(X)$ minimizing

  $$\sum_{i=1}^{n} |Y_i - \sum_{j=0}^{d} \beta^{(j)} X^j| \leq 2 \cdot K_h(X - X_i).$$

- Most classical kernel used: Tricubic kernel
  $$K(t) = \max(1 - |t|^3, 0)^3$$

- Most classical degree: $2 \ldots$

- Local bandwidth choice such that a proportion of points belongs to the window.
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
      - Model Selection
      - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
      - Bootstrap and Bagging
      - Randomized Rules and Random Forests
   - Boosting
      - AdaBoost as a Greedy Scheme
      - Boosting
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
      - Clustering
      - Dimensionality Curse
      - Dimension Reduction
      - Generative Modeling
   - Dimension Reduction
      - Simplification
      - Reconstruction Error
      - Relationship Preservation
      - Comparing Methods?
   - Clustering
      - Prototype Approaches
      - Contiguity Approaches
      - Agglomerative Approaches
      - Other Approaches
      - Scalability
   - Generative Modeling
      - (Plain) Parametric Density Estimation
      - Latent Variables
      - Approximate Simulation
      - Diffusion Model
      - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
      - Empirical Risk Minimization
      - ERM and PAC Analysis
      - Hoeffding and Finite Class
      - McDiarmid and Rademacher Complexity
      - VC Dimension
      - Structural Risk Minimization
   - References

9 References
### Fully Generative Modeling

**Idea:** If one knows the law of \((X, Y)\) everything is easy!

#### Bayes formula

- With a slight abuse of notation,

$$P(Y|X) = \frac{P((X, Y))}{P(X)} = \frac{P(X|Y)P(Y)}{P(X)}$$

#### Generative Modeling:

- Propose a model for \((X, Y)\) (or equivalently \(X|Y\) and \(Y\)),
- Estimate it as a density estimation problem,
- Plug the estimate in the Bayes formula,
- Plug the conditional estimate in the Bayes *classifier*.

**Rk:** Require to estimate \((X, Y)\) rather than only \(Y|X\)!
- Great flexibility in the model design but may lead to complex computation.
Simpler setting in classification!

Bayes formula

\[ P(Y = k | X) = \frac{P(X | Y = k) P(Y = k)}{P(X)} \]

Binary Bayes classifier (the best solution)

\[ f^*(X) = \begin{cases} 
+1 & \text{if } P(Y = 1 | X) \geq P(Y = -1 | X) \\
-1 & \text{otherwise}
\end{cases} \]

Heuristic: Estimate those quantities and plug the estimations.

By using different models/estimators for \( P(X | Y) \), we get different classifiers.

Rk: No need to renormalize by \( P(X) \) to take the decision!
Discriminant Analysis (Gaussian model)

- The densities are modeled as multivariate normal, i.e.,
  \[ \mathbb{P}(X \mid Y = k) \sim \mathcal{N}_{\mu_k, \Sigma_k} \]

- Discriminant functions:
  \[ g_k(X) = \ln(\mathbb{P}(X \mid Y = k)) + \ln(\mathbb{P}(Y = k)) \]
  \[ g_k(X) = -\frac{1}{2}(X - \mu_k)^\top \Sigma_k^{-1}(X - \mu_k) \]
  \[ \quad - \frac{d}{2} \ln(2\pi) - \frac{1}{2} \ln(|\Sigma_k|) + \ln(\mathbb{P}(Y = k)) \]

- QDA (different \( \Sigma_k \) in each class) and LDA (\( \Sigma_k = \Sigma \) for all \( k \))

- **Beware:** this model can be false but the methodology remains valid!
Discriminant Analysis

- The probability densities are Gaussian
- The effect of any decision rule is to divide the feature space into some decision regions $\mathcal{R}_1, \mathcal{R}_2$
- The regions are separated by decision boundaries
Quadratic Discriminant Analysis

- The probability densities are Gaussian
- The effect of any decision rule is to divide the feature space into some decision regions $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_c$
- The regions are separated by decision boundaries
Discriminant Analysis

**Estimation**

In practice, we will need to estimate $\mu_k$, $\Sigma_k$ and $P_k := P(Y = k)$

- The estimate proportion $\hat{P}(Y = k) = \frac{n_k}{n} = \frac{1}{n} \sum_{i=1}^{n} 1\{Y_i = k\}$
- Maximum likelihood estimate of $\hat{\mu}_k$ and $\hat{\Sigma}_k$ (explicit formulas)

- DA classifier

$$\hat{f}_G(X) = \begin{cases} +1 & \text{if } \hat{g}_+ (X) \geq \hat{g}_- (X) \\ -1 & \text{otherwise} \end{cases}$$

- Decision boundaries: quadratic = degree 2 polynomials.
- If one imposes $\Sigma_{-1} = \Sigma_1 = \Sigma$ then the decision boundaries is a linear hyperplane.
Linear Discriminant Analysis

- $\Sigma \omega_1 = \Sigma \omega_2 = \Sigma$
- The decision boundaries are linear hyperplanes
Discriminant Analysis

Quadratic Discriminant Analysis

- $\Sigma_{\omega_1} \neq \Sigma_{\omega_2}$
- Arbitrary Gaussian distributions lead to Bayes decision boundaries that are general quadratics.
Example: LDA

Linear Discriminant Analysis

Decision region

Decision boundary
Example: QDA

Quadratic Discriminant Analysis

Decision region

Decision boundary

classes

Class1

Class2
Naive Bayes

- Classical algorithm using a crude modeling for $\mathbb{P}(X|Y)$:
  - Feature independence assumption:
    $$\mathbb{P}(X|Y) = \prod_{l=1}^{d} \mathbb{P}(X^{(l)}|Y)$$
  - Simple featurewise model: binomial if binary, multinomial if finite and Gaussian if continuous
- If all features are continuous, similar to the previous Gaussian but with a diagonal covariance matrix!
- Very simple learning even in very high dimension!
Example: Naive Bayes

Naive Bayes with Gaussian model

Decision region

Decision boundary

classes

- Class1
- Class2
Naive Bayes with Density Estimation
Example: Naive Bayes

Naive Bayes with kernel density estimates

Decision region

Decision boundary
Other Models

- Other models of the world!

**Bayesian Approach**
- Generative Model plus prior on the parameters
- Inference thanks to the Bayes formula

**Graphical Models**
- Markov type models on Graphs

**Gaussian Processes**
- Multivariate Gaussian models

...
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5. Gradient Descent Step
   - Non-Convex Setting
   - References

6. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

7. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

8. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

References
Bias-Variance Dilemma

- General setting:
  - \( \mathcal{F} = \{ \text{measurable functions } \mathcal{X} \to \mathcal{Y} \} \)
  - Best solution: \( f^* = \arg\min_{f \in \mathcal{F}} \mathcal{R}(f) \)
  - Class \( S \subset \mathcal{F} \) of functions
  - Ideal target in \( S \): \( f^*_S = \arg\min_{f \in S} \mathcal{R}(f) \)
  - Estimate in \( S \): \( \hat{f}_S \) obtained with some procedure

Approximation error and estimation error (Bias/Variance)

\[
\mathcal{R}(\hat{f}_S) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(f^*_S) - \mathcal{R}(f^*)}_{\text{Approximation error}} + \underbrace{\mathcal{R}(\hat{f}_S) - \mathcal{R}(f^*_S)}_{\text{Estimation error}}
\]

- Approx. error can be large if the model \( S \) is not suitable.
- Estimation error can be large if the model is complex.
Different behavior for different model complexity

- **Low complexity model** are easily learned but the approximation error (bias) may be large (**Under-fit**).
- **High complexity model** may contain a good ideal target but the estimation error (variance) can be large (**Over-fit**)

**Bias-variance trade-off** ⟷ avoid **overfitting** and **underfitting**

- **Rk:** Better to think in term of method (including feature engineering and specific algorithm) rather than only of model.
Risk behaviour

- Learning/training risk (empirical risk on the learning/training set) decays when the complexity of the method increases.
- Quite different behavior when the risk is computed on new observations (generalization risk).
- Overfit for complex methods: parameters learned are too specific to the learning set!
- General situation! (Think of polynomial fit...)
- Need to use a different criterion than the training risk!
Risk Estimation vs Method Selection

**Predictor Risk Estimation**

- **Goal:** Given a predictor $f$ assess its quality.
- **Method:** Hold-out risk computation (/ Empirical risk correction).
- **Usage:** Compute an estimate of the risk of a selected $f$ using a test set to be used to monitor it in the future.

- Basic block very well understood.

**Method Selection**

- **Goal:** Given a ML method assess its quality.
- **Method:** Cross Validation (/ Empirical risk correction)
- **Usage:** Compute risk estimates for several ML methods using training/validation sets to choose the most promising one.

- Estimates can be pointwise or better intervals.
- Multiple test issues in method selection.
Cross Validation and Empirical Risk Correction

Two Approaches

- **Cross validation**: Use empirical risk criterion but on independent data, very efficient (and almost always used in practice!) but slightly biased as its target uses only a fraction of the data.

- **Correction approach**: use empirical risk criterion but *correct* it with a term increasing with the complexity of $S$

$$R_n(\hat{f}_S) \rightarrow R_n(\hat{f}_S) + \text{cor}(S)$$

and choose the method with the smallest corrected risk.

Which loss to use?

- The loss used in the risk: most natural!
- The loss used to estimate $\hat{\theta}$: penalized estimation!

- Other performance measure can be used.
**Cross Validation**

- **Very simple idea:** use a second learning/verification set to compute a verification risk.
- Sufficient to remove the dependency issue!
- Implicit random design setting...

Cross Validation

- Use \((1 - \epsilon) \times n\) observations to train and \(\epsilon \times n\) to verify!
- Possible issues:
  - Validation for a learning set of size \((1 - \epsilon) \times n\) instead of \(n\)?
  - Unstable risk estimate if \(\epsilon n\) is too small?

- Most classical variations:
  - Hold Out,
  - Leave One Out,
  - \(V\)-fold cross validation.
Hold Out

**Principle**
- Split the dataset $\mathcal{D}$ in 2 sets $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$ of size $n \times (1 - \epsilon)$ and $n \times \epsilon$.
- Learn $\widehat{f}^{\text{HO}}$ from the subset $\mathcal{D}_{\text{train}}$.
- Compute the empirical risk on the subset $\mathcal{D}_{\text{test}}$:
  \[
  R_n^{\text{HO}}(\widehat{f}^{\text{HO}}) = \frac{1}{n\epsilon} \sum_{(X_i, Y_i) \in \mathcal{D}_{\text{test}}} \ell(Y_i, \widehat{f}^{\text{HO}}(X_i))
  \]

**Predictor Risk Estimation**
- Use $\widehat{f}^{\text{HO}}$ as predictor.
- Use $R_n^{\text{HO}}(\widehat{f}^{\text{HO}})$ as an estimate of the risk of this estimator.

**Method Selection by Cross Validation**
- Compute $R_n^{\text{HO}}(\widehat{f}_S^{\text{HO}})$ for all the considered methods,
- Select the method with the smallest CV risk,
- Reestimate the $\widehat{f}_S$ with all the data.
Hold Out

**Principle**

- Split the dataset $\mathcal{D}$ in 2 sets $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$ of size $n \times (1 - \epsilon)$ and $n \times \epsilon$.
- Learn $\hat{f}^{\text{HO}}$ from the subset $\mathcal{D}_{\text{train}}$.
- Compute the empirical risk on the subset $\mathcal{D}_{\text{test}}$:
  \[
  \mathcal{R}_n^{\text{HO}}(\hat{f}^{\text{HO}}) = \frac{1}{n\epsilon} \sum_{(X_i, Y_i) \in \mathcal{D}_{\text{test}}} \ell(Y_i, \hat{f}^{\text{HO}}(X_i))
  \]

- Only possible setting for risk estimation.

**Hold Out Limitation for Method Selection**

- Biased toward simpler method as the estimation does not use all the data initially.
- Learning variability of $\mathcal{R}_n^{\text{HO}}(\hat{f}^{\text{HO}})$ not taken into account.
V-fold Cross Validation

**Principle**

- Split the dataset $D$ in $V$ sets $D_v$ of almost equals size.
- For $v \in \{1, \ldots, V\}$:
  - Learn $\hat{f}^{-v}$ from the dataset $D$ minus the set $D_v$.
  - Compute the empirical risk:
    \[
    R_{n}^{-v}(\hat{f}^{-v}) = \frac{1}{n_v} \sum_{(X_i, Y_i) \in D_v} \ell(Y_i, \hat{f}^{-v}(X_i))
    \]
  - Compute the average empirical risk:
    \[
    R_{n}^{CV}(\hat{f}) = \frac{1}{V} \sum_{v=1}^{V} R_{n}^{-v}(\hat{f}^{-v})
    \]
- Estimation of the quality of a method not of a given predictor.
- Leave One Out : $V = n$. 

Source: M. Kühn
V-fold Cross Validation

Analysis (when \( n \) is a multiple of \( V \))

- The \( \mathcal{R}_n^{-V}(\hat{f}^{-V}) \) are identically distributed variable but are not independent!
- Consequence:
  
  \[
  \mathbb{E} \left[ \mathcal{R}_n^{CV}(\hat{f}) \right] = \mathbb{E} \left[ \mathcal{R}_n^{-V}(\hat{f}^{-V}) \right] 
  \]
  
  \[
  \text{Var} \left[ \mathcal{R}_n^{CV}(\hat{f}) \right] = \frac{1}{V} \text{Var} \left[ \mathcal{R}_n^{-V}(\hat{f}^{-V}) \right] 
  \]
  
  \[
  + \left( 1 - \frac{1}{V} \right) \text{Cov} \left[ \mathcal{R}_n^{-V}(\hat{f}^{-V}), \mathcal{R}_n^{-V'}(\hat{f}^{-V'}) \right] 
  \]

- Average risk for a sample of size \((1 - \frac{1}{V})n\).
- Variance term much more complex to analyze!
- Fine analysis shows that the larger \( V \) the better...

- Accuracy/Speed tradeoff: \( V = 5 \) or \( V = 10 \)...
Linear Regression and Leave One Out

- Leave One Out = \( V \) fold for \( V = n \): very expensive in general.

A fast LOO formula for the linear regression

- **Prop:** for the least squares linear regression,

\[
\hat{f}^{-i}(X_i) = \frac{\hat{f}(X_i) - h_{ii} Y_i}{1 - h_{ii}}
\]

with \( h_{ii} \) the \( i \)th diagonal coefficient of the hat (projection) matrix.

- Proof based on linear algebra!

- Leads to a fast formula for LOO:

\[
R_n^{LOO}(f) = \frac{1}{n} \sum_{i=1}^{n} \frac{|Y_i - \hat{f}(X_i)|^2}{(1 - h_{ii})^2}
\]
Selection Bias Issue:
- After method selection, the cross validation is biased.
- Furthermore, it qualifies the method and not the final predictor.
- Need to (re)estimate the risk of the final predictor.

(Train/Validation)/Test strategy
- **Split** the dataset in two a (Train/Validation) and Test.
- Use **CV** with the (Train/Validation) to **select a method**.
- Train this method on (Train/Validation) to **obtain a single predictor**.
- Estimate the **performance of this predictor** on Test.

Every choice made from the data is part of the method!
Risk Correction

- Empirical loss of an estimator computed on the dataset used to chose it is biased!
- Empirical loss is an optimistic estimate of the true loss.

Risk Correction Heuristic

- Estimate an upper bound of this optimism for a given family.
- Correct the empirical loss by adding this upper bound.

- **Rk:** Finding such an upper bound can be complicated!
- Correction often called a **penalty**.
Penalization

Penalized Loss

- Minimization of

$$\arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(X_i)) + \text{pen}(\theta)$$

where $\text{pen}(\theta)$ is a risk correction (penalty).

Penalties

- Upper bound of the optimism of the empirical loss
- Depends on the loss and the framework!

Instantiation

- Mallows Cp: Least Squares with $\text{pen}(\theta) = 2\frac{d}{n}\sigma^2$.
- AIC Heuristics: Maximum Likelihood with $\text{pen}(\theta) = \frac{d}{n}$.
- BIC Heuristics: Maximum Likelihood with $\text{pen}(\theta) = \log(n)\frac{d}{n}$.
- Structural Risk Minimization: Pred. loss and clever penalty.
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
   - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9. References
Bias-Variance Dilemma

- General setting:
  - $\mathcal{F} = \{\text{measurable functions } \mathcal{X} \to \mathcal{Y}\}$
  - Best solution: $f^* = \arg\min_{f \in \mathcal{F}} R(f)$
  - Class $S \subset \mathcal{F}$ of functions
  - Ideal target in $S$: $f^*_S = \arg\min_{f \in S} R(f)$
  - Estimate in $S$: $\hat{f}_S$ obtained with some procedure

Approximation error and estimation error (Bias/ Variance)

$$R(\hat{f}_S) - R(f^*) = R(f^*_S) - R(f^*) + R(\hat{f}_S) - R(f^*_S)$$

- Approx. error can be large if the model $S$ is not suitable.
- Estimation error can be large if the model is complex.
Under-fitting / Over-fitting Issue

- Different behavior for different model complexity
- **Low complexity model** are easily learned but the approximation error (bias) may be large (**Under-fit**).
- **High complexity model** may contain a good ideal target but the estimation error (variance) can be large (**Over-fit**)

**Bias-variance trade-off** $\iff$ avoid overfitting and underfitting

- **Rk:** Better to think in term of method (including feature engineering and specific algorithm) rather than only of model.
Simplified Models

Bias-Variance Issue

- Most complex models may not be the best ones due to the variability of the estimate.

- Naive idea: can we simplify our model without losing too much?
  - by using only a subset of the variables?
  - by forcing the coefficients to be small?
- Can we do better than exploring all possibilities?
Linear Models

- **Setting**: Gen. linear model = prediction of $Y$ by $h(x^T \beta)$.

### Model coefficients
- Model entirely specified by $\beta$.
- Coefficientwise:
  - $\beta^{(i)} = 0$ means that the $i$th covariate is not used.
  - $\beta^{(i)} \sim 0$ means that the $i$th covariate as a low influence...

- If some covariates are useless, better use a simpler model...

### Submodels
- **Simplify** the model through a constraint on $\beta$!
- **Examples**:
  - Support: Impose that $\beta^{(i)} = 0$ for $i \notin I$.
  - Support size: Impose that $\|\beta\|_0 = \sum_{i=1}^{d} 1_{\beta^{(i)} \neq 0} < C$
  - Norm: Impose that $\|\beta\|_p < C$ with $1 \leq p$ (Often $p = 2$ or $p = 1$)
Norms and Sparsity

Sparsity
- $\beta$ is sparse if its number of non-zero coefficients ($\ell_0$) is small...
- Easy interpretation in terms of dimension/complexity.

Norm Constraint and Sparsity
- Sparsest solution obtained by definition with the $\ell_0$ norm.
- No induced sparsity with the $\ell_2$ norm...
- Sparsity with the $\ell_1$ norm (can even be proved to be the same as with the $\ell_0$ norm under some assumptions).
- Geometric explanation.
Constraint and Penalization

Constrained Optimization

- Choose a constant $C$.
- Compute $\beta$ as

$$\arg\min_{\beta \in \mathbb{R}^d, \|\beta\| p \leq C} \frac{1}{n} \sum_{i=1}^{n} \bar{\ell}(Y_i, h(x_i^T \beta))$$

Lagrangian Reformulation

- Choose $\lambda$ and compute $\beta$ as

$$\arg\min_{\beta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \bar{\ell}(Y_i, h(x_i^T \beta)) + \lambda \|\beta\|_{p'}$$

with $p' = p$ except if $p = 0$ where $p' = 1$.
- Easier calibration... but no explicit model $S$.

- **Rk:** $\|\beta\|_p$ is not scaling invariant if $p \neq 0$...
- Initial rescaling issue.
Penalization

Penalized Linear Model

- Minimization of

\[
\arg\min_{\beta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, h(x_i^T \beta)) + \text{pen}(\beta)
\]

where \(\text{pen}(\beta)\) is a (sparsity promoting) penalty

- Variable selection if \(\beta\) is sparse.

Classical Penalties

- AIC: \(\text{pen}(\beta) = \lambda \| \beta \|_0\) (non-convex / sparsity)
- Ridge: \(\text{pen}(\beta) = \lambda \| \beta \|_2^2\) (convex / no sparsity)
- Lasso: \(\text{pen}(\beta) = \lambda \| \beta \|_1\) (convex / sparsity)
- Elastic net: \(\text{pen}(\beta) = \lambda_1 \| \beta \|_1 + \lambda_2 \| \beta \|_2^2\) (convex / sparsity)

- Easy optimization if \(\text{pen}\) (and the loss) is convex...

- **Need to specify** \(\lambda\) **to define a ML method!**
**Penalization and Cross-Validation**

### Practical Selection Methodology
- Choose a penalty family $\text{pen}_{\lambda}$.
- Compute a CV risk for the penalty $\text{pen}_{\lambda}$ for all $\lambda \in \Lambda$.
- Determine $\hat{\lambda}$ the $\lambda$ minimizing the CV risk.
- Compute the final model with the penalty $\text{pen}_{\hat{\lambda}}$.

**CV allows to select a ML method, penalized estimation with a penalty $\text{pen}_{\hat{\lambda}}$, not a single predictor hence the need of a final reestimation.**

### Why not using CV on a grid?
- Grid size scales exponentially with the dimension!
- **If the penalized minimization is easy**, much cheaper to compute the CV risk for all $\lambda \in \Lambda$...
- CV performs best when the set of candidates is not too big (or is structured...)

---

ML Methods: Probabilistic Point of View
References


A. Géron. *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (2nd ed.)* O'Reilly, 2019

Ch. Giraud. *Introduction to High-Dimensional Statistics*. CRC Press, 2014

Creative Commons Attribution-ShareAlike (CC BY-SA 4.0)

- You are free to:
  - **Share**: copy and redistribute the material in any medium or format
  - **Adapt**: remix, transform, and build upon the material for any purpose, even commercially.

- Under the following terms:
  - **Attribution**: You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - **ShareAlike**: If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - **No additional restrictions**: You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Contributors

- Main contributor: E. Le Pennec
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
     - Optimization Point of View
     - SVM
     - Penalization
     - Cross Validation and Weights
   - Optimization: Gradient Descent Algorithms
     - Introduction
     - Gradient Descent
     - Proximal Descent
     - Coordinate Descent
     - Gradient Descent Acceleration
     - Stochastic Gradient Descent
     - Gradient Descent Step
     - Non-Convex Setting
     - References
   - ML Methods: Neural Networks and Deep Learning
     - Introduction
     - From Logistic Regression to NN
     - NN Optimization
     - NN Regularization
     - Image and CNN
     - Text, Recurrent Neural Networks and Transformers
     - NN Architecture
     - References
   - ML Methods: Trees and Ensemble Methods
     - Trees
     - Bagging and Random Forests
       - Bootstrap and Bagging
       - Randomized Rules and Random Forests
     - Boosting
       - AdaBoost as a Greedy Scheme
       - Boosting
       - Ensemble Methods
       - References
   - Unsupervised Learning: Dimension Reduction and Clustering
     - Unsupervised Learning?
     - A First Glimpse
       - Clustering
       - Dimensionality Curse
       - Dimension Reduction
       - Generative Modeling
     - Dimension Reduction
       - Simplification
       - Reconstruction Error
       - Relationship Preservation
       - Comparing Methods?
     - Clustering
       - Prototype Approaches
       - Contiguity Approaches
       - Agglomerative Approaches
       - Other Approaches
       - Scalability
     - Generative Modeling
       - (Plain) Parametric Density Estimation
       - Latent Variables
       - Approximate Simulation
       - Diffusion Model
       - Generative Adversarial Network
     - Applications to Text
     - References

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
     - Ensemble Methods
     - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
Supervised Learning

Supervised Learning Framework

- Input measurement $X \in \mathcal{X}$
- Output measurement $Y \in \mathcal{Y}$.
- $(X, Y) \sim \mathbb{P}$ with $\mathbb{P}$ unknown.
- **Training data** : $\mathcal{D}_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ (i.i.d. $\sim \mathbb{P}$)

- Often
  - $X \in \mathbb{R}^d$ and $Y \in \{-1, 1\}$ (classification)
  - or $X \in \mathbb{R}^d$ and $Y \in \mathbb{R}$ (regression).
- A predictor is a function in $\mathcal{F} = \{f : \mathcal{X} \to \mathcal{Y} \text{ meas.}\}$

Goal

- Construct a **good** predictor $\hat{f}$ from the training data.

- Need to specify the meaning of good.
- Classification and regression are almost the same problem!
**Loss function for a generic predictor**

- **Loss function**: \( \ell(Y, f(X)) \) measures the goodness of the prediction of \( Y \) by \( f(X) \)
- **Examples**:
  - 0/1 loss: \( \ell(Y, f(X)) = 1_{Y \neq f(X)} \)
  - Quadratic loss: \( \ell(Y, f(X)) = |Y - f(X)|^2 \)

**Risk function**

- Risk measured as the average loss for a new couple:
  \[ R(f) = \mathbb{E}_{(X, Y) \sim P}[\ell(Y, f(X))] \]
- **Examples**:
  - 0/1 loss: \( \mathbb{E}[\ell(Y, f(X))] = P(Y \neq f(X)) \)
  - Quadratic loss: \( \mathbb{E}[\ell(Y, f(X))] = \mathbb{E}[|Y - f(X)|^2] \)

- **Beware**: As \( \hat{f} \) depends on \( D_n \), \( R(\hat{f}) \) is a random variable!
Best Solution

- The best solution $f^*$ (which is independent of $D_n$) is

$$f^* = \arg\min_{f \in F} R(f) = \arg\min_{f \in F} \mathbb{E}[\ell(Y, f(X))] = \arg\min_{f \in F} \mathbb{E}_X[\mathbb{E}_Y|X[\ell(Y, f(X))]]$$

Bayes Predictor (explicit solution)

- In binary classification with $0-1$ loss:

$$f^*(X) = \begin{cases} +1 & \text{if } \mathbb{P}(Y = +1|X) \geq \mathbb{P}(Y = -1|X) \\ \Leftrightarrow \mathbb{P}(Y = +1|X) \geq 1/2 & \\ -1 & \text{otherwise} \end{cases}$$

- In regression with the quadratic loss

$$f^*(X) = \mathbb{E}[Y|X]$$

**Issue:** Solution requires to know $\mathbb{E}[Y|X]$ for all values of $X$!
Goal

Machine Learning

- Learn a rule to construct a **predictor** $\hat{f} \in \mathcal{F}$ from the training data $\mathcal{D}_n$ s.t. the risk $\mathcal{R}(\hat{f})$ is **small on average** or with high probability with respect to $\mathcal{D}_n$.

- In practice, the rule should be an algorithm!

Canonical example: **Empirical Risk Minimizer**

- One restricts $f$ to a subset of functions $\mathcal{S} = \{f_\theta, \theta \in \Theta\}$
- One replaces the minimization of the average loss by the minimization of the empirical loss

$$\hat{f} = f_\hat{\theta} = \arg\min_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(X_i))$$

- Examples:
  - Linear regression
  - Linear classification with

$$\mathcal{S} = \{x \mapsto \text{sign}\{x^T \beta + \beta^{(0)}\} / \beta \in \mathbb{R}^d, \beta^{(0)} \in \mathbb{R}\}$$
Example: TwoClass Dataset

Synthetic Dataset

- Two features/covariates.
- Two classes.

- Dataset from *Applied Predictive Modeling*, M. Kuhn and K. Johnson, Springer
- Numerical experiments with R and the `{caret}` package.
Example: Linear Discrimination

Logistic

Decision region

Decision boundary

classes
- Class1
- Class2
Example: More Complex Model

Naive Bayes with kernel density estimates

Decision region

Decision boundary

preceded by:

classes
Red = Class 1
Blue = Class 2
Bias-Variance Dilemma

- General setting:
  - $\mathcal{F} = \{\text{measurable functions } \mathcal{X} \to \mathcal{Y}\}$
  - Best solution: $f^* = \operatorname{argmin}_{f \in \mathcal{F}} \mathcal{R}(f)$
  - Class $S \subset \mathcal{F}$ of functions
  - Ideal target in $S$: $f_S^* = \operatorname{argmin}_{f \in S} \mathcal{R}(f)$
  - Estimate in $S$: $\hat{f}_S$ obtained with some procedure

Approximation error and estimation error (Bias/Variance)

$$\mathcal{R}(\hat{f}_S) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(f_S^*) - \mathcal{R}(f^*)}_{\text{Approximation error}} + \underbrace{\mathcal{R}(\hat{f}_S) - \mathcal{R}(f_S^*)}_{\text{Estimation error}}$$

- Approx. error can be large if the model $S$ is not suitable.
- Estimation error can be large if the model is complex.
Different behavior for different model complexity

- **Low complexity model** are easily learned but the approximation error (**bias**) may be large (**Under-fit**).

- **High complexity model** may contain a good ideal target but the estimation error (**variance**) can be large (**Over-fit**)

**Bias-variance trade-off** $\iff$ avoid overfitting and underfitting

- **Rk**: Better to think in term of method (including feature engineering and specific algorithm) rather than only of model.
**Binary Classification Loss Issue**

**Empirical Risk Minimizer**

\[ \hat{f} = \arg\min_{f \in S} \frac{1}{n} \sum_{i=1}^{n} \ell_{0/1}(Y_i, f(X_i)) \]

- Classification loss: \( \ell_{0/1}(y, f(x)) = 1_{y \neq f(x)} \)
- Not convex and not smooth!
The best solution $f^*$ (which is independent of $D_n$) is

$$f^* = \arg \min_{f \in \mathcal{F}} \mathcal{R}(f) = \arg \min_{f \in \mathcal{F}} \mathbb{E}[\ell(Y, f(X))] = \arg \min_{f \in \mathcal{F}} \mathbb{E}_X \left[ \mathbb{E}_{Y|X}[\ell(Y, f(X))] \right]$$

**Bayes Predictor (explicit solution)**

In binary classification with $0-1$ loss:

$$f^*(X) = \begin{cases} +1 & \text{if } \mathbb{P}(Y = +1|X) \geq \mathbb{P}(Y = -1|X) \\ -1 & \text{otherwise} \end{cases}$$

**Issue:** Solution requires to **know** $\mathbb{E}[Y|X]$ for all values of $X$!

**Solution:** Replace it by an estimate.
Minimizer of the risk

\[ \hat{f} = \arg\min_{f \in S} \frac{1}{n} \sum_{i=1}^{n} \ell^{0/1}(Y_i, f(X_i)) \]

- **Issue:** Classification loss is not convex or smooth.
- **Solution:** Replace it by a convex majorant.
Probabilistic and Optimization Framework

How to find a good function $f$ with a small risk

$$\mathcal{R}(f) = \mathbb{E}[\ell(Y, f(X))]$$

Canonical approach: $\hat{f}_S = \arg\min_{f \in S} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i))$

Problems

- How to choose $S$?
- How to compute the minimization?

A Probabilistic Point of View

Solution: For $X$, estimate $Y|X$ plug this estimate in the Bayes classifier:

(Generalized) Linear Models, Kernel methods, $k$-nn, Naive Bayes, Tree, Bagging...

An Optimization Point of View

Solution: If necessary replace the loss $\ell$ by an upper bound $\bar{\ell}$ and minimize the empirical loss: SVR, SVM, Neural Network, Tree, Boosting...
# Outline

1. **Statistical Learning: Introduction, Setting and Risk Estimation**
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. **ML Methods: Probabilistic Point of View**
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. **ML Methods: Optimization Point of View**
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. **Optimization: Gradient Descent Algorithms**
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. **ML Methods: Neural Networks and Deep Learning**
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. **ML Methods: Trees and Ensemble Methods**
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. **Unsupervised Learning: Dimension Reduction and Clustering**
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
   - Generative Modeling

8. **Statistical Learning: PAC-Bayesian Approach and Complexity Theory**
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References
Probabilistic and Optimization Framework

How to find a good function $f$ with a small risk

$$R(f) = \mathbb{E}[\ell(Y, f(X))]$$

**Canonical approach:**

$$\hat{f}_S = \arg\min_{f \in S} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i))$$

**Problems**

- How to choose $S$?
- How to compute the minimization?

**A Probabilistic Point of View**

**Solution:** For $X$, estimate $Y|X$ plug this estimate in the Bayes classifier:

(Generalized) Linear Models, Kernel methods, $k$-nn, Naive Bayes, Tree, Bagging...

**An Optimization Point of View**

**Solution:** If necessary replace the loss $\ell$ by an upper bound $\bar{\ell}$ and minimize the empirical loss: SVR, SVM, Neural Network, Tree, Boosting...
Three Classical Methods in a Nutshell

**Penalized Logistic Regression**
- Let $f_\theta(X) = X^\top \beta + \beta^{(0)}$ with $\theta = (\beta, \beta^{(0)})$.
- Find $\hat{\theta} = \arg \min_\theta \frac{1}{n} \sum_{i=1}^n \log \left( 1 + e^{-Y_i f_\theta(X_i)} \right) + \lambda \|\beta\|_1$
- Classify using $\text{sign}(f_\hat{\theta})$

**Deep Learning**
- Let $f_\theta(X)$ with $f$ a feed forward neural network outputing two values with a softmax layer as a last layer.
- Optimize by gradient descent the cross-entropy $-\frac{1}{n} \sum_{i=1}^n \log \left( f_\theta(X_i)^{(Y_i)} \right)$
- Classify using $\text{sign}(f_\hat{\theta})$
Three Classical Methods in a Nutshell

Support Vector Machine

- Let \( f_\theta(X) = X^T \beta + \beta^{(0)} \) with \( \theta = (\beta, \beta^{(0)}) \).
- Find \( \hat{\theta} = \arg \min \frac{1}{n} \sum_{i=1}^{n} \max (1 - Y_i f_\theta(X_i), 0) + \lambda \| \beta \|_2^2 \)
- Classify using \( \text{sign}(f_\hat{\theta}) \)

Those three methods rely on a similar heuristic: the optimization point of view!
The best solution $f^*$ is the one minimizing

$$f^* = \arg \min R(f) = \arg \min \mathbb{E}[\ell(Y, f(X))]$$

One restricts $f$ to a subset of functions $S = \{f_\theta, \theta \in \Theta\}$

One replaces the minimization of the average loss by the minimization of the average empirical loss

$$\hat{f} = f_{\hat{\theta}} = \arg \min_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(X_i))$$

Intractable for the $\ell^0/1$ loss!
Convexification Strategy

**Risk Convexification**

- Replace the loss $\ell(Y, f_\theta(X))$ by a convex upperbound $\bar{\ell}(Y, f_\theta(X))$ (surrogate loss).
- Minimize the average of the surrogate empirical loss
  \[
  \tilde{f} = f_\hat{\theta} = \arg\min_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \bar{\ell}(Y_i, f_\theta(X_i))
  \]
- Use $\hat{f} = \text{sign}(\tilde{f})$

- Much easier optimization.

**Instantiation**

- Logistic (Revisited)
- Support Vector Machine
- (Deep) Neural Network
- Boosting
Classification Loss and Convexification

Convexification

- Replace the loss $\ell^{0/1}(Y, f(X))$ by
  \[ \tilde{\ell}(Y, f(X)) = l(\mathcal{Y}f(X)) \]
  with $l$ a convex function.

- **Further mild assumption:** $l$ is decreasing, differentiable at 0 and $l'(0) < 0$. 
Classification Loss and Convexification

Classical convexification

- Logistic loss: \( \tilde{\ell}(Y, f(X)) = \log_2(1 + e^{-Yf(X)}) \) (Logistic / NN)
- Hinge loss: \( \tilde{\ell}(Y, f(X)) = (1 - Yf(X))_+ \) (SVM)
- Exponential loss: \( \tilde{\ell}(Y, f(X)) = e^{-Yf(X)} \) (Boosting... )
The Target is the Bayes Classifier

- The minimizer of
  \[ \mathbb{E} \left[ \bar{\ell}(Y, f(X)) \right] = \mathbb{E} [\ell(Y f(X))] \]
  is the Bayes classifier \( f^* = \text{sign}(2\eta(X) - 1) \)

Control of the Excess Risk

- It exists a convex function \( \Psi \) such that
  \[ \Psi \left( \mathbb{E} \left[ \ell^{0/1}(Y, \text{sign}(f(X))) \right] - \mathbb{E} \left[ \ell^{0/1}(Y, f^*(X)) \right] \right) \]
  \[ \leq \mathbb{E} \left[ \bar{\ell}(Y, f(X)) \right] - \mathbb{E} \left[ \bar{\ell}(Y, f^*(X)) \right] \]

- Theoretical guarantee!
Logistic Revisited

- Ideal solution:

\[ \hat{f} = \arg\min_{f \in S} \frac{1}{n} \sum_{i=1}^{n} \ell^{0/1}(Y_i, f(X_i)) \]

Logistic regression

- Use \( f(X) = X^T \beta + \beta^{(0)} \).
- Use the logistic loss \( \tilde{\ell}(y, f) = \log_2(1 + e^{-yf}) \), i.e. the negative log-likelihood.

- Different vision than the statistician but same algorithm!
Logistic Revisited

Logistic

Decision region

Decision boundary

classes

Class1

Class2
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
     - SVM
     - Penalization
     - Cross Validation and Weights
   - Optimization: Gradient Descent Algorithms
     - Introduction
     - Gradient Descent
     - Proximal Descent
     - Coordinate Descent
     - Gradient Descent Acceleration
     - Stochastic Gradient Descent

4. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

5. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

6. Optimization: Gradient Descent Algorithms
   - Gradient Descent Step
   - Non-Convex Setting
   - References

7. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

8. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

9. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling

   - Dimension Reduction
     - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

10. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
    - Supervised Learning
    - Empirical Risk Minimization
      - Empirical Risk Minimization
      - ERM and PAC Analysis
      - Hoeffding and Finite Class
      - McDiarmid and Rademacher Complexity
      - VC Dimension
      - Structural Risk Minimization
    - References

11. References
Support Vector Machine

\[ f_\theta(X) = X^T \beta + \beta(0) \quad \text{with} \quad \theta = (\beta, \beta(0)) \]

\[ \hat{\theta} = \arg \min \frac{1}{n} \sum_{i=1}^{n} \max (1 - Y_i f_\theta(X_i), 0) + \lambda \| \beta \|^2 \]

Support Vector Machine

- Convexification of the 0/1-loss with the hinge loss:
  \[ 1 \cdot Y_i f_\theta(X_i) < 0 \leq \max (1 - Y_i f_\theta(X_i), 0) \]
- Penalization by the quadratic norm (Ridge/Tikhonov).
- Solution can be approximated by gradient descent algorithms.

- **Revisit** of the original point of view.
- Original point of view leads to a different optimization algorithm and to some extensions.
Ideal Separable Case

- Linear classifier: \( \text{sign}(X^T \beta + \beta^{(0)}) \)
- Separable case: \( \exists (\beta, \beta^{(0)}), \forall i, Y_i(X_i^T \beta + \beta^{(0)}) > 0 \)

How to choose \((\beta, \beta^{(0)})\) so that the separation is maximal?

- Strict separation: \( \exists (\beta, \beta^{(0)}), \forall i, Y_i(X_i^T \beta + \beta^{(0)}) \geq 1 \)
- Distance between \( X^T \beta + \beta^{(0)} = 1 \) and \( X^T \beta + \beta^{(0)} = -1 \):
  \[
  \frac{2}{\|\beta\|}
  \]
- Maximizing this distance is equivalent to minimizing \( \frac{1}{2} \|\beta\|^2 \).
Ideal Separable Case

Separable SVM

- Constrained optimization formulation:
  \[
  \min \frac{1}{2} \|\beta\|^2 \quad \text{with} \quad \forall i, \ Y_i(X_i^T \beta + \beta^{(0)}) \geq 1
  \]

- Quadratic Programming setting.
- Efficient solver available...
Non Separable Case

What about the non separable case?

SVM relaxation

- Relax the assumptions
  \[ \forall i, Y_i(X_i^T \beta + \beta(0)) \geq 1 \quad \text{to} \quad \forall i, Y_i(X_i^T \beta + \beta(0)) \geq 1 - s_i \]
  with the \textbf{slack variables} \( s_i \geq 0 \)

- Keep those slack variables as small as possible by minimizing
  \[ \frac{1}{2} ||\beta||^2 + C \sum_{i=1}^{n} s_i \]

where \( C > 0 \) is the \textbf{goodness-of-fit strength}
Non Separable Case

SVM

- Constrained optimization formulation:
  \[
  \min \frac{1}{2} \| \beta \|^2 + C \sum_{i=1}^{n} s_i \quad \text{with} \quad \begin{cases} 
  \forall i, \ Y_i(X_i^T \beta + \beta^{(0)}) \geq 1 - s_i \\
  \forall i, \ s_i \geq 0 
  \end{cases}
  \]

- **Hinge Loss** reformulation:
  \[
  \min \frac{1}{2} \| \beta \|^2 + C \sum_{i=1}^{n} \max(0, 1 - Y_i(X_i^T \beta + \beta^{(0)}))
  \]

- Constrained convex optimization algorithms vs gradient descent algorithms.
SVM as a Penalized Convex Relaxation

- Convex relaxation:

\[
\begin{align*}
\text{argmin} & \; \frac{1}{2}\|\beta\|^2 + C \sum_{i=1}^{n} \max(1 - Y_i(X_i^T \beta + \beta^{(0)}), 0) \\
& = \text{argmin} \; \frac{1}{n} \sum_{i=1}^{n} \max(1 - Y_i(X_i^T \beta + \beta^{(0)}), 0) + \frac{1}{Cn} \frac{1}{2} \|\beta\|^2
\end{align*}
\]

- Prop: \( \ell^{0/1}(Y_i, \text{sign}(X_i^T \beta + \beta^{(0)})) \leq \max(1 - Y_i(X_i^T \beta + \beta^{(0)}), 0) \)

Penalized convex relaxation (Tikhonov!)

\[
\frac{1}{n} \sum_{i=1}^{n} \ell^{0/1}(Y_i, \text{sign}(X_i^T \beta + \beta^{(0)})) + \frac{1}{Cn} \frac{1}{2} \|\beta\|^2
\leq \frac{1}{n} \sum_{i=1}^{n} \max(1 - Y_i(X_i^T \beta + \beta^{(0)}), 0) + \frac{1}{Cn} \frac{1}{2} \|\beta\|^2
\]
Support Vector Machine

Decision region

Decision boundary

classes
Class1
Class2

classes
Class1
Class2
The Kernel Trick

- Non linear separation: just replace $X$ by a non linear $\Phi(X)$...
- Knowing $\phi(X_j)^\top \phi(X_j)$ is sufficient to compute the SVM solution.

Kernel trick

- **Computing** $k(X, X') = \phi(X)^\top \phi(X')$ may be easier than computing $\phi(X)$, $\phi(X')$ and then the scalar product!
- $\phi$ can be specified through its definite positive kernel $k$.
- Examples: Polynomial kernel $k(X, X') = (1 + X^\top X')^d$, Gaussian kernel $k(X, X') = e^{-\|X-X'\|^2/2}$, ...
- RKHS setting!
- Can be used in (logistic) regression and more...
Support Vector Machine with polynomial kernel

Decision region

Decision boundary

classes
- Class1
- Class2
Support Vector Machine with Gaussian kernel

Decision region

Decision boundary

classes
- Class1
- Class2
SVM and Lagrangian

**SVM**

- Constrained optimization formulation:

\[
\min \frac{1}{2} \| \beta \|^2 + C \sum_{i=1}^{n} s_i \quad \text{with} \quad \begin{cases} 
\forall i, \ Y_i(X_i^\top \beta + \beta^{(0)}) \geq 1 - s_i \\
\forall i, \ s_i \geq 0
\end{cases}
\]

**SVM Lagrangian**

- Lagrangian:

\[
\mathcal{L}(\beta, \beta^{(0)}, s, \alpha, \mu) = \frac{1}{2} \| \beta \|^2 + C \sum_{i=1}^{n} s_i \\
+ \sum_{i} \alpha_i (1 - s_i - Y_i(X_i^\top \beta + \beta^{(0)})) - \sum_{i} \mu_i s_i
\]
### KKT Optimality Conditions

- **Stationarity:**
  \[
  \nabla_\beta \mathcal{L}(\beta, \beta^{(0)}, s, \alpha, \mu) = \beta - \sum_i \alpha_i Y_i X_i = 0 \\
  \nabla_{\beta^{(0)}} \mathcal{L}(\beta, \beta^{(0)}, s, \alpha, \mu) = - \sum_i \alpha_i = 0 \\
  \nabla_s L(\beta, \beta^{(0)}, s, \alpha, \mu) = C - \alpha_i - \mu_i = 0
  \]

- **Primal and dual admissibility:**
  \[
  (1 - s_i - Y_i(\mathbf{X}_i^T \beta + \beta^{(0)})) \leq 0, \quad s_i \geq 0, \quad \alpha_i \geq 0, \quad \text{and} \quad \mu_i \geq 0
  \]

- **Complementary slackness:**
  \[
  \alpha_i(1 - s_i - Y_i(\mathbf{X}_i^T \beta + \beta^{(0)})) = 0 \quad \text{and} \quad \mu_i s_i = 0
  \]

### Consequence

- **\( \beta^* \):**
  \[
  \beta^* = \sum_i \alpha_i Y_i \mathbf{X}_i \quad \text{and} \quad 0 \leq \alpha_i \leq C.
  \]

- If \( \alpha_i \neq 0 \), \( \mathbf{X}_i \) is called a **support vector** and either
  - \( s_i = 0 \) and \( Y_i(\mathbf{X}_i^T \beta^* + \beta^{(0)*}) = 1 \) (margin hyperplane),
  - or \( \alpha_i = C \) (outliers).

- **\( \beta^{(0)*} \):**
  \[
  \beta^{(0)*} = Y_i - \mathbf{X}_i^T \beta^* \quad \text{for any support vector with} \quad 0 < \alpha_i < C.
  \]
SVM Dual

SVM Lagrangian Dual

- **Lagrangian Dual:**

\[ Q(\alpha, \mu) = \min_{\beta, \beta^{(0)}, s} \mathcal{L}(\beta, \beta^{(0)}, s, \alpha, \mu) \]

- **Prop:**
  - if \( \sum_i \alpha_i Y_i \neq 0 \) or \( \exists i, \alpha_i + \mu_i \neq C \),
    \[ Q(\alpha, \mu) = -\infty \]
  - if \( \sum_i \alpha_i Y_i = 0 \) and \( \forall i, \alpha_i + \mu_i = C \),
    \[ Q(\alpha, \mu) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j Y_i Y_j X_i^\top X_j \]

SVM Dual problem

- **Dual problem is a Quadratic Programming problem:**

\[ \max_{\alpha \geq 0, \mu \geq 0} Q(\alpha, \mu) \Leftrightarrow \max_{0 \leq \alpha \leq C} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j Y_i Y_j X_i^\top X_j \]

- Involves the \( X_i \) only through their scalar products.
Mercer Theorem

Mercer Representation Theorem

- For any loss $\bar{\ell}$ and any increasing function $\Phi$, the minimizer in $\beta$ of
  \[
  \sum_{i=1}^{n} \bar{\ell}(Y_i, X_i^\top \beta + \beta^{(0)}) + \Phi(\|\beta\|_2)
  \]
  is a linear combination of the input points $\beta^* = \sum_{i=1}^{n} \alpha'_i X_i$.

- Minimization problem in $\alpha'$:
  \[
  \sum_{i=1}^{n} \bar{\ell}(Y_i, \sum_{j} \alpha'_j X_i^\top X_j + \beta^{(0)}) + \Phi(\|\beta\|_2)
  \]
  involving only the scalar product of the data.

- Optimal predictor requires only to compute scalar products.
  \[
  \hat{f}^*(X) = X^\top \beta^* + \beta^{(0),*} = \sum_{i} \alpha'_i X_i^\top X
  \]

- Transform a problem in dimension $\text{dim}(\mathcal{X})$ in a problem in dimension $n$.
- Direct minimization in $\beta$ can be more efficient...
Feature Map

Feature Engineering
- Art of creating **new features** from the existing one $X$.
- Example: add monomials $(X^{(i)})^2$, $X^{(i)}X^{(i')}$...
- Adding feature increases the dimension.

Feature Map
- Application $\phi : \mathcal{X} \rightarrow \mathbb{H}$ with $\mathbb{H}$ an Hilbert space.
- Linear decision boundary in $\mathbb{H}$: $\phi(X)^\top \beta + \beta^{(0)} = 0$ is **not an hyperplane anymore** in $\mathcal{X}$.

**Heuristic:** Increasing dimension allows to make data almost linearly separable.
Polynomial Mapping

Polynomial Mapping of order 2

- \( \phi : \mathbb{R}^2 \rightarrow \mathbb{R}^6 \)

\[
\phi(X) = \left( (X^{(1)})^2, (X^{(2)})^2, \sqrt{2}X^{(1)}X^{(2)}, \sqrt{2}X^{(1)}, \sqrt{2}X^{(2)}, 1 \right)
\]

- Allow to solve the XOR classification problem with the hyperplane \( X^{(1)}X^{(2)} = 0 \).

Polynomial Mapping and Scalar Product

- Prop:

\[
\phi(X)^T \phi(X') = (1 + X^T X')^2
\]
SVM Primal and Dual

Primal, Lagrangian and Dual

- **Primal:**
  \[
  \min \|\beta\|^2 + C \sum_{i=1}^{n} s_i \quad \text{with} \quad \begin{cases} 
  \forall i, \ Y_i(\phi(X_i)^T \beta + \beta^{(0)}) \geq 1 - s_i \\
  \forall i, \ s_i \geq 0
  \end{cases}
  \]

- **Lagrangian:**
  \[
  L(\beta, \beta^{(0)}, s, \alpha, \mu) = \frac{1}{2} \|\beta\|^2 + C \sum_{i=1}^{n} s_i \\
  + \sum_{i} \alpha_i (1 - s_i - Y_i(\phi(X_i)^T \beta + \beta^{(0)})) - \sum_{i} \mu_i s_i
  \]

- **Dual:**
  \[
  \max_{\alpha \geq 0, \mu \geq 0} Q(\alpha, \mu) \Leftrightarrow \max_{0 \leq \alpha \leq C} \sum_{i} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j Y_i Y_j \phi(X_i)^T \phi(X_j)
  \]
  \[
  \text{Optimal } \phi(X)^T \beta^* + \beta^{(0),*} = \sum_{i} \alpha_i Y_i \phi(X)^T \phi(X_i)
  \]

- **Only need to know to compute** \(\phi(X)^T \phi(X')\) **to obtain the solution.**
From Map to Kernel

- Many algorithms (e.g. SVM) require only to be able to compute the scalar product $\phi(X)^\top \phi(X')$.

**Kernel**

- Any application

$$k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$$

is called a **kernel** over $\mathcal{X}$.

**Kernel Trick**

- Computing directly the **kernel** $k(X, X') = \phi(X)^\top \phi(X')$ may be easier than computing $\phi(X), \phi(X')$ and then the scalar product.

- Here $k$ is defined from $\phi$.
- Under some assumption on $k$, $\phi$ can be implicitly **defined** from $k$!
A kernel $k$ is PDS if and only if

- $k$ is symmetric, i.e.
  \[ k(X, X') = k(X', X) \]
- for any $N \in \mathbb{N}$ and any $(X_1, \ldots, X_N) \in \mathcal{X}^N$,
  \[ K = [k(X_i, X_j)]_{1 \leq i, j \leq N} \]
  is positive semi-definite, i.e.
  \[ \forall u \in \mathbb{R}^N \]
  \[ u^\top K u = \sum_{1 \leq i, j \leq N} u^{(i)} u^{(j)} k(X_i, X_j) \geq 0 \]
  or equivalently all the eigenvalues of $K$ are non-negative.

The matrix $K$ is called the **Gram matrix** associated to $(X_1, \ldots, X_N)$. 

Reproducing Kernel Hilbert Space

Moore-Aronsajn Theorem

- For any PDS kernel $k: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, it exists a Hilbert space $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ with a scalar product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ such that
  - it exists a mapping $\phi: \mathcal{X} \rightarrow \mathcal{H}$ satisfying
    $$k(X, X') = \langle \phi(X), \phi(X') \rangle_{\mathcal{H}}$$
  - the reproducing property holds, i.e. for any $h \in \mathcal{H}$ and any $X \in \mathcal{X}$
    $$h(X) = \langle h, k(X, \cdot) \rangle_{\mathcal{H}}.$$

- By def., $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS).
- $\mathcal{H}$ is called the feature space associated to $k$ and $\phi$ the feature mapping.
- No unicity in general.
- **Rk:** if $k(X, X') = \phi'(X)^{\top} \phi'(X')$ with $\phi': \mathcal{X} \rightarrow \mathbb{R}^{p}$ then
  - $\mathcal{H}$ can be chosen as $\{X \mapsto \phi'(X)^{\top} \beta, \beta \in \mathbb{R}^{p}\}$ and $\|X \mapsto \phi'(X)^{\top} \beta\|_{\mathcal{H}}^{2} = \|\beta\|_{2}^{2}$.
  - $\phi(X') : X \mapsto \phi'(X)^{\top} \phi'(X').$
## Kernel Construction Machinery

### Separable Kernel
- For any function $\Psi : \mathcal{X} \to \mathbb{R}$, $k(X, X') = \Psi(X)\Psi(X')$ is PDS.

### Kernel Stability
- For any PDS kernels $k_1$ and $k_2$, $k_1 + k_2$ and $k_1k_2$ are PDS kernels.
- For any sequence of PDS kernels $k_n$ converging pointwise to a kernel $k$, $k$ is a PDS kernel.
- For any PDS kernel $k$ such that $|k| \leq r$ and any power series $\sum_n a_nz^n$ with $a_n \geq 0$ and a convergence radius larger than $r$, $\sum_n a_nk^n$ is a PDS kernel.
- For any PDS kernel $k$, the renormalized kernel $k'(X, X') = \frac{k(X, X')}{\sqrt{k(X, X)k(X', X')}}$ is a PDS kernel.
- Cauchy-Schwartz for $k$ PDS: $k(X, X')^2 \leq k(X, X)k(X', X')$
## Classical Kernels

### PDS Kernels

- **Vanilla kernel:**
  \[
  k(X, X') = X^\top X'
  \]
- **Polynomial kernel:**
  \[
  k(X, X') = (1 + X^\top X')^k
  \]
- **Gaussian RBF kernel:**
  \[
  k(X, X') = \exp\left(-\gamma \|X - X'\|^2\right)
  \]
- **Tanh kernel:**
  \[
  k(X, X') = \tanh(aX^\top X' + b)
  \]

- Most classical is the Gaussian RBF kernel...
- Lots of freedom to construct kernel for non classical data.
Representer Theorem

Let $k$ be a PDS kernel and $H$ its corresponding RKHS, for any increasing function $\Phi$ and any function $L : \mathbb{R}^n \rightarrow \mathbb{R}$, the optimization problem

$$\arg\min_{h \in H} L(h(X_1), \ldots, h(X_n)) + \Phi(\|h\|)$$

admits only solutions of the form

$$\sum_{i=1}^{n} \alpha_i' k(X_i, \cdot).$$

Examples:
- (kernelized) SVM
- (kernelized) Penalized Logistic Regression (Ridge)
- (kernelized) Penalized Regression (Ridge)
Kernelized SVM

**Primal**

- Constrained Optimization:
  \[
  \min_{f \in \mathbb{H}, \beta(0), s} \|f\|^2_{\mathbb{H}} + C \sum_{i=1}^{n} s_i \quad \text{with} \quad \begin{cases} 
  \forall i, Y_i(f(X_i) + \beta(0)) \geq 1 - s_i \\
  \forall i, s_i \geq 0 
\end{cases}
  \]

- Hinge loss:
  \[
  \min_{f \in \mathbb{H}, \beta(0)} \|f\|^2_{\mathbb{H}} + C \sum_{i=1}^{n} \max(0, 1 - Y_i(f(X_i) + \beta(0)))
  \]

- Representer:
  \[
  \min_{\alpha', \beta(0)} \sum_{i,j} \alpha'_i \alpha'_j k(X_i, X_j) \\
  + C \sum_{i=1}^{n} \max(0, 1 - Y_i(\sum_{j} \alpha'_j k(X_j, X_i) + \beta(0)))
  \]

**Dual**

- Dual:
  \[
  \max_{\alpha \geq 0, \mu \geq 0} Q(\alpha, \mu) \Leftrightarrow \max_{0 \leq \alpha \leq C} \sum_{i} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j Y_i Y_j k(X_i, X_j)
  \]
Support Vector Machine with polynomial kernel

Decision region:
- PredictorB
- PredictorA
- Classes: Class1, Class2

Decision boundary:
- PredictorB
- PredictorA
- Classes: Class1, Class2
Support Vector Machine with Gaussian kernel

Decision region

Decision boundary

classes

Class1
Class2

classes

Class1
Class2
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
Bias-Variance Dilemma

- General setting:
  - \( \mathcal{F} = \{ \text{measurable functions } \mathbf{X} \to \mathbf{Y} \} \)
  - Best solution: \( f^* = \text{argmin}_{f \in \mathcal{F}} \mathcal{R}(f) \)
  - Class \( S \subset \mathcal{F} \) of functions
  - Ideal target in \( S \): \( f^*_S = \text{argmin}_{f \in S} \mathcal{R}(f) \)
  - Estimate in \( S \): \( \hat{f}_S \) obtained with some procedure

Approximation error and estimation error (Bias/Variance)

\[
\mathcal{R}(\hat{f}_S) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(f^*_S) - \mathcal{R}(f^*)}_{\text{Approximation error}} + \underbrace{\mathcal{R}(\hat{f}_S) - \mathcal{R}(f^*_S)}_{\text{Estimation error}}
\]

- Approx. error can be large if the model \( S \) is not suitable.
- Estimation error can be large if the model is complex.
Different behavior for different model complexity

- **Low complexity model** are easily learned but the approximation error (bias) may be large (**Under-fit**).

- **High complexity model** may contain a good ideal target but the estimation error (variance) can be large (**Over-fit**)

**Bias-variance trade-off** $\iff$ avoid overfitting and underfitting

- **Rk**: Better to think in term of method (including feature engineering and specific algorithm) rather than only of model.
Simplified Models

Bias-Variance Issue

- Most complex models may not be the best ones due to the variability of the estimate.

- Naive idea: can we *simplify* our model without loosing too much?
  - by using only a subset of the variables?
  - by forcing the coefficients to be small?
- Can we do better than exploring all possibilities?

Source: Tibshirani et al.
Linear Models

- **Setting**: Gen. linear model = prediction of $Y$ by $h(x^T \beta)$.

**Model coefficients**

- Model entirely specified by $\beta$.
- Coefficientwise:
  - $\beta^{(i)} = 0$ means that the $i$th covariate is not used.
  - $\beta^{(i)} \sim 0$ means that the $i$th covariate as a low influence...

- If some covariates are useless, better use a simpler model...

**Submodels**

- *Simplify* the model through a constraint on $\beta$!
- Examples:
  - Support: Impose that $\beta^{(i)} = 0$ for $i \not\in I$.
  - Support size: Impose that $\|\beta\|_0 = \sum_{i=1}^d 1_{\beta^{(i)} \neq 0} < C$
  - Norm: Impose that $\|\beta\|_p < C$ with $1 \leq p$ (Often $p = 2$ or $p = 1$)
Norms and Sparsity

Sparsity
- \( \beta \) is sparse if its number of non-zero coefficients (\( \ell_0 \)) is small...
- Easy interpretation in terms of dimension/complexity.

Norm Constraint and Sparsity
- Sparsest solution obtained by definition with the \( \ell_0 \) norm.
- No induced sparsity with the \( \ell_2 \) norm...
- Sparsity with the \( \ell_1 \) norm (can even be proved to be the same as with the \( \ell_0 \) norm under some assumptions).
- Geometric explanation.
**Constraint and Penalization**

### Constrained Optimization
- Choose a constant $C$.
- Compute $\beta$ as

$$
\arg\min_{\beta \in \mathbb{R}^d, \|\beta\|_p \leq C} \frac{1}{n} \sum_{i=1}^{n} \tilde{\ell}(Y_i, h(x_i^\top \beta))
$$

### Lagrangian Reformulation
- Choose $\lambda$ and compute $\beta$ as

$$
\arg\min_{\beta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \tilde{\ell}(Y_i, h(x_i^\top \beta)) + \lambda \|\beta\|_{p'}
$$

with $p' = p$ except if $p = 0$ where $p' = 1$.
- Easier calibration... but no explicit model $S$.

- **Rk:** $\|\beta\|_p$ is not scaling invariant if $p \neq 0$.
- Initial rescaling issue.
Penalization

Penalized Linear Model

- Minimization of
  \[
  \arg\min_{\beta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \bar{\ell}(Y_i, h(x_i^T \beta)) + \text{pen}(\beta)
  \]
  where \(\text{pen}(\beta)\) is a (sparsity promoting) penalty
- Variable selection if \(\beta\) is sparse.

Classical Penalties

- AIC: \(\text{pen}(\beta) = \lambda \| \beta \|_0\) (non-convex / sparsity)
- Ridge: \(\text{pen}(\beta) = \lambda \| \beta \|_2^2\) (convex / no sparsity)
- Lasso: \(\text{pen}(\beta) = \lambda \| \beta \|_1\) (convex / sparsity)
- Elastic net: \(\text{pen}(\beta) = \lambda_1 \| \beta \|_1 + \lambda_2 \| \beta \|_2^2\) (convex / sparsity)

- Easy optimization if \(\text{pen}\) (and the loss) is convex...
- **Need to specify \(\lambda\) to define a ML method!**
Penalization and Cross-Validation

Practical Selection Methodology

- Choose a penalty family \( \text{pen}_\lambda \).
- Compute a CV risk for the penalty \( \text{pen}_\lambda \) for all \( \lambda \in \Lambda \).
- Determine \( \hat{\lambda} \) the \( \lambda \) minimizing the CV risk.
- Compute the final model with the penalty \( \text{pen}_{\hat{\lambda}} \).

- CV allows to select a ML method, penalized estimation with a penalty \( \text{pen}_{\hat{\lambda}} \), not a single predictor hence the need of a final reestimation.

Why not using CV on a grid?

- Grid size scales exponentially with the dimension!
- **If the penalized minimization is easy**, much cheaper to compute the CV risk for all \( \lambda \in \Lambda \)...
- CV performs best when the set of candidates is not too big (or is structured...)

226
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights
    - Optimization: Gradient Descent Algorithms
      - Introduction
      - Gradient Descent
      - Proximal Descent
      - Coordinate Descent
      - Gradient Descent Acceleration
      - Stochastic Gradient Descent

4. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

5. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
     - Ensemble Methods
   - References

6. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

7. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

8. References

9. ML Methods: Optimization
   - Point of View
Unbalanced and Rebalanced Dataset

**Unbalanced Class**
- **Setting**: One of the class is much more present than the other.
- **Issue**: Classifier *too attracted* by the majority class!

**Rebalanced Dataset**
- **Setting**: Class proportions are different in the training and testing set (stratified sampling)
- **Issue**: Training risks are not estimate of testing risks.
Resampling Strategies

Resampling

- Modify the training dataset so that the classes are more balanced.
- Two flavors:
  - Sub-sampling which spoils data,
  - Over-sampling which needs to create new examples.
- Issues: Training data is not anymore representative of testing data
- Hard to do it right!
Resampling Effect

Testing
- Testing class prob.: $\pi_t(k)$
- Testing risk target:
\[
E_{\pi_t}[\ell(Y, f(X))] = \sum_k \pi_t(k)E[\ell(Y, f(X)) | Y = k]
\]

Training
- Training class prob.: $\pi_{tr}(k)$
- Training risk target:
\[
E_{\pi_{tr}}[\ell(Y, f(X))] = \sum_k \pi_{tr}(k)E[\ell(Y, f(X)) | Y = k]
\]

Implicit Testing Risk Using the Training One
- Amounts to use a weighted loss:
\[
E_{\pi_{tr}}[\ell(Y, f(X))] = \sum_k \pi_{tr}(k)E[\ell(Y, f(X)) | Y = k]
\]
\[
= \sum_k \pi_t(k)E\left[\frac{\pi_{tr}(k)}{\pi_t(k)}\ell(Y, f(X)) | Y = k\right]
\]
\[
= E_{\pi_t}\left[\frac{\pi_{tr}(Y)}{\pi_t(Y)}\ell(Y, f(X))\right]
\]
- Put more weight on less probable classes!
In unbalanced situation, often the cost of misprediction is not the same for all classes (e.g. medical diagnosis, credit lending...) 

Much better to use this explicitly than to do blind resampling!

**Weighted Loss**

- **Weighted loss:**
  \[ \ell(Y, f(X)) \rightarrow C(Y) \ell(Y, f(X)) \]

- Weighted risk target:
  \[ \mathbb{E}[C(Y) \ell(Y, f(X))] \]

- **Rk:** Strong link with \( \ell \) as \( C \) is independent of \( f \).
- Often allow reusing algorithm constructed for \( \ell \).
- \( C \) may also depend on \( X \). . .
Weighted Loss, $\ell^{0/1}$ loss and Bayes Classifier

- The Bayes classifier is now:
  $$f^\star = \arg\min \mathbb{E}[C(Y)\ell(Y, f(X))] = \arg\min \mathbb{E}_X[\mathbb{E}_{Y|X}[C(Y)\ell(Y, f(X))]]$$

Bayes Predictor

- For $\ell^{0/1}$ loss,
  $$f^\star(X) = \arg\max_k C(k)\mathbb{P}(Y = k|X)$$
  - Same effect than a threshold modification for the binary setting!
- Allow putting more emphasis on some classes than others.
Cost and Proportions

- Testing risk target:
  \[ \mathbb{E}_{\pi_t}[C_t(Y)\ell(Y, f(X))] = \sum_k \pi_t(k)C_t(k)\mathbb{E}[\ell(Y, f(X))|Y = k] \]

- Training risk target
  \[ \mathbb{E}_{\pi_{tr}}[C_{tr}(Y)\ell(Y, f(X))] = \sum_k \pi_{tr}(k)C_{tr}(k)\mathbb{E}[\ell(Y, f(X))|Y = k] \]

- **Coincide if**
  \[ \pi_t(k)C_t(k) = \pi_{tr}(k)C_{tr}(k) \]

- Lots of flexibility in the choice of \( C_t, C_{tr} \) or \( \pi_{tr} \).
Combining Weights and Resampling

Weighted Loss and Resampling

- **Weighted loss**: choice of a weight $C_t \neq 1$.
- **Resampling**: use a $\pi_{tr} \neq \pi_t$.

- Stratified sampling may be used to reduce the size of a dataset without losing a low probability class!

Combining Weights and Resampling

- **Weighted loss**: use $C_{tr} = C_t$ as $\pi_{tr} = \pi_t$.
- **Resampling**: use an implicit $C_{t}(k) = \pi_{tr}(k)/\pi_{t}(k)$.
- **Combined**: use $C_{tr}(k) = C_t(k)\pi_t(k)/\pi_{tr}(k)$.

- Most ML methods allow such weights!
References

T. Hastie, R. Tibshirani, and J. Friedman.  
*The Elements of Statistical Learning.*  
Springer Series in Statistics, 2009

M. Mohri, A. Rostamizadeh, and A. Talwalkar.  
*Foundations of Machine Learning (2nd ed.)*  
MIT Press, 2018

A. Géron.  
*Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (3rd ed.)*  
O’Reilly, 2022

Ch. Giraud.  
*Introduction to High-Dimensional Statistics.*  
CRC Press, 2014

S. Bubeck.  
*Convex Optimization: Algorithms and Complexity.*  
Now Publisher, 2015
Creative Commons Attribution-ShareAlike (CC BY-SA 4.0)

- You are free to:
  - **Share:** copy and redistribute the material in any medium or format
  - **Adapt:** remix, transform, and build upon the material for any purpose, even commercially.
- Under the following terms:
  - **Attribution:** You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - **ShareAlike:** If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - **No additional restrictions:** You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Contributors

- **Main contributor:** E. Le Pennec
Optimization: Gradient Descent Algorithms

1. Gradient Descent Step
2. Non-Convex Setting
3. References
ML Optimization Problem

Typical Optimization Problem in ML

- ML Opt.: \( \text{argmin } G(w) \) with \( G(w) = F(w) + R(w) \) where \( w \in \mathbb{R}^d \)
  - \( F \) a goodness-of-fit function: \( F(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_w(X_i)) \)
    - where \( \ell \) is some loss and \( f_w \) a parametric predictor,
  - \( R \) a regularizer: \( R(w) = \lambda \text{pen}(w) \)
    - where \( \text{pen}(\cdot) \) is some penalization function.

Examples

- Predictor:
  - Linear \( f_w(X_i) = X_i^T w \)
  - Neural Nets...

- Regularizer:
  - \( \text{pen}(w) = \|w\|_2^2 \) (ridge)
  - \( \text{pen}(w) = \|w\|_1 \) (Lasso)
## Losses

### Classification
- Logistic loss, $\ell(y, f) = \log(1 + e^{-yf})$
- Hinge loss, $\ell(y, f) = (1 - yf)_+$
- Quadratic hinge loss, $\ell(y, f) = \frac{1}{2}(1 - yf)_+^2$
- Huber loss $\ell(y, f) = -4yf \mathbf{1}_{yf < -1} + (1 - yf)_+^2 \mathbf{1}_{yf \geq -1}$

> These losses are convex upper bound of the 0/1 loss $\ell(y, y') = \mathbf{1}_{yy' \leq 0}$.

### Regression
- Quadratic loss, $\ell(y, f) = \frac{1}{2}(y - f)^2$
- Absolute loss, $\ell(y, f) = \frac{1}{2}|y - f|$
Linear Predictor

ML Problem

- Minimization of

\[ G(w) = F(w) + R(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, \langle X_i, w \rangle) + \lambda \text{pen}(w) \]

Gradient and Hessian of \( F \)

- Gradient:

\[ \nabla F(w) = \frac{1}{n} \sum_{i=1}^{n} \bar{\ell}(Y_i, \langle X_i, w \rangle) X_i \]

with \( \bar{\ell}(y, f) = \frac{\partial \ell(y, f)}{\partial f} \)

- Hessian matrix:

\[ \nabla^2 F(w) = \frac{1}{n} \sum_{i=1}^{n} \bar{\ell}'(Y_i, \langle X_i, w \rangle) X_i X_i^T \]

with \( \bar{\ell}'(y, f) = \frac{\partial^2 \ell(y, f)}{\partial f^2} \)
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Gradient Descent
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9 References
Zero Order Method

Zero Order Heuristic

- Zero order approximation:
  \[ G(w) = G(w') + O(\|w - w'\|) \]

Optimization in a Compact Set

- Optimization problem:
  \[ w^* \in \text{argmin } G(w). \quad w \in [0,1]^d \]

Exhaustive Search

- **Algorithm:**
  - Evaluate \( G \) on a grid,
  - Approximate the minimizer by the minimizer in the grid.
  - If \( G \) is \( C \)-Lipschitz, evaluating \( G \) on a grid of precision \( \epsilon/(\sqrt{d}C) \) is sufficient to find a \( \epsilon \)-minimizer of \( G \).
  - Required number of evaluation: \( N_\epsilon = O\left((C\sqrt{d}/\epsilon)^d\right) \)

Hardly usable even in moderate dimension! Rk: Better zero order methods exist...
First Order Method

First Order Heuristic
- First order approximation:
  \[ G(w) = G(w') + \nabla G(w')^\top (w - w') + o(\|w - w'\|) \]
- Best descent direction: \(-\nabla G(w')\)

Gradient Descent Algorithm
- Start from a point \(w^{[0]}\) and let \(k = 0\).
- Repeat until convergence:
  \[ w^{[k+1]} = w^{[k]} - \alpha^{[k]} \nabla G(w^{[k]}) \]
  \[ k \rightarrow k + 1 \]
  with \(\alpha^{[k]}\) a sequence of (small) steps.
Do The Gradient Descent Converge?

- **Not always!**

**Assumptions**
- Convexity assumption on $G$.
- Regularity assumption on $G$.
- Size of the steps $\alpha[k]$. 

Source: Lijia Yu
**Convexity and Regularity**

### Convex function
- A function $G : \mathbb{R}^d \rightarrow \mathbb{R}$ is **convex** on $\mathbb{R}^d$ if, for all $x, y \in \mathbb{R}^d$, for all $\lambda \in [0, 1]$, $G(\lambda x + (1 - \lambda)y) \leq \lambda G(x) + (1 - \lambda)f(y)$.
- If $f_w$ is linear, $F(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_w(X_i))$ is convex iff $f \mapsto \ell(y_i, f)$ is convex for any $i = 1, \ldots, n$.

### Regularity
- $G$ is **$L$-smooth** if it is continuously differentiable and if $\|\nabla G(w) - \nabla G(w')\|_2 \leq L \|w - w'\|_2$ for any $w, w' \in \mathbb{R}^d$.
- If $G$ is twice differentiable, this is equivalent to assuming $\lambda_{\text{max}}(\nabla^2 G(w)) \leq L$ for any $w \in \mathbb{R}^d$ (largest eigenvalue of the Hessian matrix of $G$ smaller than $L$).
## Least-Squares Regression

### Formulation

- **Minimization of**

\[
G(w) = F(w) + R(w) = \frac{1}{2n} \sum_{i=1}^{n} (Y_i - X_i^\top w)^2 + \lambda \text{pen}(w)
\]

### Gradient and Hessian of \( F \)

- **Gradient:**

\[
\nabla F(w) = \frac{1}{n} \sum_{i=1}^{n} (X_i^\top w - Y_i)X_i
\]

- **Hessian:**

\[
\nabla^2 F(w) = \frac{1}{n} \sum_{i=1}^{n} X_iX_i^\top
\]

so that

\[
L = \lambda_{\text{max}} \left( \frac{1}{n} \sum_{i=1}^{n} X_iX_i^\top \right)
\]
## Logistic Regression

### Formulation

- Minimization of
  \[
  G(w) = F(w) + R(w) = \frac{1}{n} \sum_{i=1}^{n} \log \left( 1 + e^{(-Y_i(X_i^T w))} \right) + \lambda \text{pen}(w)
  \]

### Gradient and Hessian of $F$

- **Gradient:**
  \[
  \nabla F(w) = \frac{1}{n} \sum_{i=1}^{n} Y_i (\sigma(Y_i X_i^T w) - 1) X_i
  \]

- **Hessian:**
  \[
  \nabla^2 F(w) = \frac{1}{n} \sum_{i=1}^{n} \sigma(Y_i X_i^T w) (1 - \sigma(Y_i X_i^T w)) X_i X_i^T
  \]

So that $F$ is $L$-smooth with
\[
L \leq \frac{1}{4} \lambda_{\max} \left( \frac{1}{n} \sum_{i=1}^{n} X_i X_i^T \right)
\]

- $\sigma(t) = e^{-t}/(1 + e^{-t})$. 
Convergence of GD

Theorem

Let $G : \mathbb{R}^d \rightarrow \mathbb{R}$ be a $L$-smooth convex function. Let $\mathbf{w}^*$ be the minimum of $f$ on $\mathbb{R}^d$. Then, Gradient Descent with step size $\alpha \leq 1/L$ satisfies

$$G(\mathbf{w}^{[k]}) - G(\mathbf{w}^*) \leq \frac{||\mathbf{w}^{[0]} - \mathbf{w}^*||_2^2}{2\alpha k}.$$

In particular, for $\alpha = 1/L$,

$$N_\epsilon = O(L||\mathbf{w}^{[0]} - \mathbf{w}^*||_2^2/(2\epsilon))$$

iterations are sufficient to get an $\epsilon$-approximation of the minimal value of $G$.

- Bound is independent of the dimension $d$.
- **Weak dependency** hidden in the constant $L$...
Descent Lemma

A **key** Property: the Descent Lemma.

- If $G$ is convex and $L$-smooth, then for any $w, w' \in \mathbb{R}^d$

\[
G(w) \leq G(w') + \nabla G(w')^\top (w - w') + \frac{L}{2} \|w - w'\|^2.
\]

**Link with the Gradient Descent Algorithm**

- At step $k$, for any $w$,

\[
G(w) \leq G(w^{[k]}) + \nabla G(w^{[k]})^\top (w - w^{[k]}) + \frac{L}{2} \|w - w^{[k]}\|^2
\]

\[
\leq G(w^{[k]}) + \frac{L}{2} \|w - (w^{[k]} - \frac{1}{L} \nabla G(w^{[k]}))\|^2 - \frac{1}{2L} \|\nabla G(w^{[k]})\|^2
\]

- Optimizing the upper bound in $w$ leads to

\[
w^{[k+1]} = w^{[k]} - \frac{1}{L} \nabla G(w^{[k]})
\]

- **Rk:** Newton iteration comes from the approximation

\[
G(w) = G(w') + \nabla G(w')^\top (w - w') + \frac{1}{2} (w - w')^\top \nabla^2 G(w')(w - w') + o(\|w - w'\|^2)
\]
Faster Rate - Strongly Convex Function

Strong convexity

- A function $G : \mathbb{R}^d \to \mathbb{R}$ is $\mu$-strongly convex if $x \mapsto G(x) - \frac{\mu}{2} \|x\|_2^2$ is convex.
- If $G$ is differentiable, this is equivalent to, for all $x, y \in \mathbb{R}^d$,
  $$G(y) \geq G(x) + \nabla G(x)^\top (y - x) + \frac{\mu}{2} \|y - x\|_2^2.$$ 
- If $G$ is twice differentiable, this is equivalent to
  $$\lambda_{\text{min}}(\nabla^2 G(x)) \geq \mu.$$

Theorem

- Let $G : \mathbb{R}^d \to \mathbb{R}$ be a $L$-smooth, $\mu$ strongly convex function. Let $w^*$ be the minimum of $G$ on $\mathbb{R}^d$. Then, Gradient Descent with step size $\alpha \leq 1/L$ satisfies
  $$G(w^{[k]}) - G(w^*) \leq \frac{1}{2\alpha} \left(1 - \alpha \mu\right)^k \|G(w^{[0]}) - G(w^*)\|_2^2.$$  
- $\textbf{Rk:}$ $N_\epsilon = O(- \log \epsilon/(\alpha \mu))$ iterations are sufficient to obtain an $\epsilon$-minimizer.
## Convexity

- If $G$ is a convex function then for any $\mathbf{w} \in \mathbb{R}^d$, there exists a (not necessarily unique) subgradient $\delta G(\mathbf{w})$ such that
  \[ G(\mathbf{w}') \geq G(\mathbf{w}) + \delta G(\mathbf{w})^\top (\mathbf{w}' - \mathbf{w}) \quad \text{for any } \mathbf{w}' \in \mathbb{R}^d \]

- If $G$ is differentiable then it is unique and equal to the gradient.

## Subgradient Descent Algorithm

Subgradient Descent Algorithm: Gradient Descent with subgradient instead of gradient.

## Regularity

- $G$ is **C-Lipschitz** if
  \[ |G(\mathbf{w}) - G(\mathbf{w}')| \leq C \| \mathbf{w} - \mathbf{w}' \|_2 \quad \text{for any } \mathbf{w}, \mathbf{w}' \in \mathbb{R}^d \]

- If $G$ is differentiable, this is equivalent to assuming
  \[ \| \nabla G(\mathbf{w}) \| \leq C \quad \text{for any } \mathbf{w} \in \mathbb{R}^d \]
Lipschitz Convex Function - Slower Rate

**Theorem**

- Let $G : \mathbb{R}^d \rightarrow \mathbb{R}$ be a convex function, $C$-Lipschitz in $B(w^*, R)$ where $w^*$ be the minimizer of $f$ on $\mathbb{R}^d$. Assume that 
  $$\alpha[k] > 0, \quad \alpha[k] \rightarrow 0, \quad \sum_k \alpha[k] = +\infty$$

  and  
  $$\|w[0] - w^*\| \leq R$$

  Then, Subgradient Descent with step size $\alpha[k]$ satisfies

  $$\min_{k' \leq k} G(w[k']) - G(w^*) \leq C \frac{R^2 + \sum_{k'=0}^{k} (\alpha[k'])^2}{2 \sum_{k'=0}^{k} \alpha[k']}$$

- In particular, for $\alpha[k] = \frac{r}{\sqrt{k+1}}$

  $$\min_{k' \leq k} G(w[k']) - G(w^*) \leq C \frac{R^2 + r^2 \log (k+1)}{4r\sqrt{k+1}}$$

  and

  $$N_\epsilon = O \left( \frac{(C(-\log \epsilon)/\epsilon)^2}{\epsilon} \right)$$
In practice, how to choose $\alpha$?

- **Theoretical setting of $\alpha = 1/L$ is a worst case scenario.**

**Exact line search**
- At each step, choose the best $\alpha$ by optimizing
  \[
  \alpha^{[k]} = \arg\min_{\alpha > 0} G(w - \alpha \nabla G(w)).
  \]
- **Too costly!**

**Backtracking line search**
- Fix a parameter $0 < \beta < 1$ and set $\alpha = \alpha_{\text{init}}$
- At each iteration,
  - while $G(w - \alpha \nabla G(w)) > G(w) - \frac{\alpha}{2} \|\nabla G(w)\|^2$,
    - modify $\alpha \leftarrow \beta \alpha$.
  - Use the final $\alpha$ as the stepsize for this iteration.
- **Simple and work pretty well in practice.**
- Theoretical guarantees available.
Outline

1. **Statistical Learning: Introduction, Setting and Risk Estimation**
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. **ML Methods: Probabilistic Point of View**
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. **ML Methods: Optimization Point of View**
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. **Optimization: Gradient Descent Algorithms**
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. **ML Methods: Neural Networks and Deep Learning**
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. **ML Methods: Trees and Ensemble Methods**
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. **Unsupervised Learning: Dimension Reduction and Clustering**
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?

   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability

   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network

   - Applications to Text
   - References

8. **Statistical Learning: PAC-Bayesian Approach and Complexity Theory**
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9. **References**
Function Decomposition and Descent

**Function Decomposition:** \( G = F + R \)
- \( F \) convex and \( L \)-smooth
- \( R \) convex and \( \text{simple} \)

**Another Descent Algorithm**
- At step \( k \), for any \( w \),
  \[
  G(w) \leq F(w) + R(w)
  \]

  \[
  \leq F(w^k) + \nabla F(w^k)^\top (w - w^k) + \frac{L}{2} \| w - w^k \|^2 + R(w)
  \]

  \[
  \leq F(w^k) + \frac{L}{2} \| w - (w^k - \frac{1}{L} \nabla F(w^k)) \|^2 - \frac{1}{2L} \| \nabla F(w^k) \|^2 + R(w)
  \]

- Optimizing the upper bound in \( w \) leads to
  \[
  w^{k+1} = \arg\min_w \frac{1}{2} \| w - (w^k - \frac{1}{L} \nabla F(w^k)) \|^2 + \frac{1}{L} R(w)
  \]

- \( R \text{ simple} \) means that this minimization is easy.
Proximal Operator and Proximal Gradient Descent

**Proximal Operator**

- For any convex function $R$:
  $$\text{prox}_\gamma R(w') = \arg\min_w \frac{1}{2} \| w - w' \|^2 + \gamma R(w)$$

- Generalization of the projection operator:
  - $R(w) = 1_\Omega$: $\text{prox}_\gamma R(w') = P_\Omega w'$
  - $R(w) = \frac{1}{2} \| w \|^2$: $\text{prox}_\gamma R(w') = \frac{1}{1+\gamma} w$.
  - $R(w) = \| w \|_1$: $\text{prox}_\gamma R(w') = T_\gamma(w')$ with $T_\gamma(w)_i = \text{sign}(w_i) \max(0, |w_i| - \gamma)$ (soft thresholding).

**Proximal Gradient Descent Algorithm for simple $R$**

- Start from a point $w^{[0]}$ and let $k = 0$.
- Repeat until convergence:
  - $w^{[k+1]} = \text{prox}_{1/L} R (w^{[k]} - \alpha^{[k]} \nabla F(w^{[k]}))$
  - $k \rightarrow k + 1$

with $\alpha^{[k]}$ a sequence of (small) steps.
Theoretical Guarantees

- **Same as smooth case!**

## Rates

- **$F$ $L$-smooth and $R$ simple:**
  \[ G(w^{[k]}) - G(w^*) \leq \frac{\|w^0 - w^*\|^2}{2\alpha k} \]
  and \( N_\epsilon = O(L\|w^0 - w^*\|^2/2\epsilon) \).

- **$F$ $L$-smooth and $\mu$-convex and $R$ simple:**
  \[ G(w^{[k]}) - G(w^*) \leq \frac{1}{2\alpha} \left(1 - \alpha\mu\right)^k \|G(w^0) - G(w^*)\|^2 \]
  and \( N_\epsilon = O(-\log \epsilon/(\alpha\mu)) \).

- **$F$ $C$-Lipschitz and $R$ is the characteristic function of a convex set:**
  \[ G(w^{[k]}) - G(w^*) \leq C \frac{R^2 + r^2 \log(k + 1)}{4r\sqrt{k + 1}} \]
  and \( N_\epsilon = O \left((C(-\log \epsilon)/\epsilon)^2\right) \).
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
     - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
## Coordinate Descent

### Heuristic
- Is $\mathbf{w}$ a minimizer of a $G$ when
  \[ G(\mathbf{w} + t \mathbf{e}_i) \geq G(\mathbf{w}) \quad \forall t \in \mathbb{R} \text{ and } 1 \leq i \leq d, \]
  with $G$ convex?
- **YES** if $G$ is smooth ($C^1$), $G$ is separable or $G$ is a sum of a smooth ($C^1$) and a separable function ($G(\mathbf{w}) = \sum_{i=1}^{d} g_i(w^{(i)})$).
- **NO** otherwise.

### Coordinate Descent Algorithm
- Start from an arbitrary $\mathbf{w}^{[0]}$
- Repeat until convergence:
  - Pick a coordinate $i$
  - Set $\mathbf{w}^{[k+1],(j)} = \mathbf{w}^{[k],(j)}$ for $j \neq i$.
  - **Optimize** only the $i$th coordinate to obtain $\mathbf{w}^{[k+1],(i)}$
  - $k \rightarrow k + 1$
- State-of-the-art is several ML problems!
Exact Coordinate Descent

**Exact Coordinate Descent (CD)**

- Start from an arbitrary $w^0$
- Repeat until convergence:
  - Pick a coordinate $i$
  - Set $w^{k+1}(j) = w^k(j)$ for $j \neq i$.
  - Compute
    $$w^{k+1}(i) = \arg\min_{z \in \mathbb{R}} G(w^k(1), \ldots, w^k(i-1), z, w^k(i+1), \ldots, w^k(d))$$
  - $k \rightarrow k + 1$
- Several way to choose the coordinate $i$: uniform sampling, deterministic cycle...
- Only 1D optimization problems to solve...but probably a lot of them.

**Theorem - Warga (1963)**

- If $G$ is continuously differentiable and strictly convex, then exact coordinate descent converges to a minimum.
Coordinate Gradient Descent

Coordinate Gradient Descent (CGD)

- Start from an arbitrary \( w^0 \)
- Repeat until convergence:
  - Pick a coordinate \( i \)
  - Set \( w^{k+1}(j) = w^k(j) \) for \( j \neq i \).
  - Compute
    \[
    w^{k+1}(i) = w^k(i) - \alpha^k \nabla(i)G(w^k)
    \]
  - \( k \rightarrow k + 1 \)

- For smooth function, step-size \( \alpha^k \) can be taken as \( \alpha^k = 1/L_i \) where \( i \) is the coordinate chosen and \( L_i \) the Lipschitz constant of
  \[
  G'(z) = G(w + ze_i) = G(w^{(1)}, \ldots, w^{(i-1)}, w^{(i)} + z, w^{(i+1)}, \ldots, w^{(d)})
  \]
- If \( G \) is \( L \)-smooth, \( L_i \leq L \).
Theorem - Nesterov (2012)

- Assume that $G$ is convex and smooth and that each $G^i$ is $L_i$-smooth.
- Consider a sequence $\{w^{[k]}\}$ given by CGD with $\alpha^{[k]} = 1/L_i$ and coordinates $i_1, i_2, \ldots$ chosen at random: i.i.d and uniform distribution in $\{1, \ldots, d\}$. Then

$$E \left[ G(w^{[k+1]}) - G(w^*) \right] \leq \frac{d}{d + k} \left( (1 - \frac{1}{d}) (G(w^{[0]}) - G(w^*)) + \frac{1}{2} \|w^{[0]} - w^*\|_L^2 \right),$$

with $\|w\|_L^2 = \sum_{j=1}^d L_j w^2_j$.

- Bound in expectation, since coordinates are taken at random.
- For cycling coordinates $i_k = (k \mod d) + 1$ the bound is much worse.
- Similar result when $G = F + R$ with $F$ smooth and $R$ simple and separable.
Comparison of Gradient Descent and Coordinate Gradient Descent

<table>
<thead>
<tr>
<th>Gradient Descent</th>
<th>Coordinate Gradient Descent</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cost of an iteration</strong> $O(d)$.</td>
<td><strong>Cost of an iteration</strong> $O(1)$.</td>
</tr>
<tr>
<td><strong>Number of iteration to obtain an $\epsilon$-minimizer:</strong></td>
<td><strong>Number of iteration to obtain an $\epsilon$-minimizer:</strong></td>
</tr>
<tr>
<td>$N_\epsilon = \frac{L \left| w^{[0]} - w^* \right|_2^2}{2\epsilon}$</td>
<td>$N_\epsilon = \frac{d}{\epsilon} \left( 1 - \frac{1}{d} \right) \left( G(w^{[0]}) - G(w^<em>) \right) + \frac{1}{2} \left| w^{[0]} - w^</em> \right|_L^2$</td>
</tr>
</tbody>
</table>

- Same order of complexity but smaller constant for CGD as soon as $\left( G(w^{[0]}) - G(w^*) \right) \ll L \left\| w^{[0]} - w^* \right\|_2^2$.
- In practice, often much faster (especially when dealing with anisotropic regularity).
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

9. References
Gradient Descent Acceleration

- Can we improve the number of iterations $O(L/\epsilon)$ (L-smooth) and $O(L/\mu \log(1/\epsilon))$ (L-smooth and $\mu$ strongly-convex) ?
- **Heuristic:** add some momentum to propagate gradients.

### Polyak’s momentum algorithm - Heavy ball method

- **Input:** starting point $w^{[0]}$, step size $\alpha^{[k]} > 0$, momentum $\beta \in [0,1]$
- While not converge do
  - $w^{[k+1/2]} = w^{[k]} - \alpha^{[k]} \nabla G(w^{[k]})$
  - $w^{[k+1]} = w^{[k+1/2]} + \beta (w^{[k]} - w^{[k-1]})$
  - $k \leftarrow k + 1$
- **Return** last $w^{[k+1]}$.

- Most classical value for $\beta$: 0.9
- When $\alpha^{[k]} = \alpha$, the update equation can be written

$$w^{[k+1]} = w^{[k]} - \alpha \sum_{t=1}^{k} \beta^{k-t} \nabla G(w^{(t)}).$$
Polyak’s Momentum Failure

Counter Example

- Polyak’s momentum algorithm fails to converge in some specific cases, for instance:

\[ \nabla G(x) = \begin{cases} 
25x & \text{if } x < 1 \\
x + 24 & \text{if } 1 \leq x < 2 \\
25x - 24 & \text{if } x \geq 2
\end{cases} \]

- In that case, \( G \) is \( \mu \) strongly convex and \( L \)-smooth with \( (\mu, L) = (1, 25) \). However, iterations given by Polyak’s algorithm cycles.
Nesterov Accelerated Gradient Descent

Nesterov Method

- **Input**: starting point \( w^0 = w^{-1/2} \), step size \( \alpha^k > 0 \), momentum \( \beta^k \in [0, 1] \)
- **While not converge** do
  - \( w^{k+1/2} = w^k - \alpha^k \nabla G(w^k) \)
  - \( w^{k+1} = w^{k+1/2} + \beta^k (w^{k+1/2} - w^{k-1/2}) \)
  - \( k \leftarrow k + 1 \)
- **Return** last \( w^{k+1} \).

**Subtle difference with momentum method.**

**Not much freedom for** \( \beta^k \):

- **Nesterov original choice**:
  - \( \beta^k = \frac{t^k - 1}{t[k+1]} \)
  - \( t^1 = 1 \)
  - \( t[k+1] = \frac{1+\sqrt{1+4(t^k)^2}}{2} \)

- **Simpler equivalent choice** \( \beta^k = \frac{k}{k+3} \)

- **Other choice possible for** \( \mu \) convex function.

- **Proximal version possible.**
Convergence Rate of Nesterov Accelerated Gradient

Theorem for $L$-smooth function

- Assume that $G$ is a $L$-smooth, convex function whose minimum is reached at $w^*$. Then, if $\beta^k = k/(k + 3)$,

$$G(w^k) - G(w^*) \leq \frac{2\|w^0 - w^*\|^2}{\alpha(k + 1)^2}.$$

Theorem for $\mu$ strongly convex function

- Assume that $G$ is a $L$-smooth, $\mu$ strongly convex function whose minimum is reached at $w^*$. Then, if $\beta^k = \frac{1 - \sqrt{\mu/L}}{1 + \sqrt{\mu/L}}$,

$$G(w^k) - G(w^*) \leq \frac{\|w^0 - w^*\|^2}{\alpha} (1 - \sqrt{\frac{\mu}{L}})^k.$$

- Faster rates than respectively $O(1/k)$ and $O((1 - \mu/L)^k)$.
- Theorems holds for the proximal algorithm with $G = F + R$ with $F$ smooth and $R$ simple.
### Optimal Bounds

#### General First Order Method

- Any iterative method that generates a sequence \( \{w^{[k]}\} \) s.t.
  \[
  w^{[k]} \in w^{[0]} + \text{Span}(\nabla f(w^{[0]}), \ldots, \nabla f(w^{[k-1]})).
  \]

#### Lower Bounds

- For any \( w^{[0]} \in \mathbb{R}^d \) and any \( k \) satisfying \( 1 \leq k \leq (d - 1)/2 \), there exists a \( L \)-smooth convex function \( f \) such that for any general first order method
  \[
  G(w^{[k]}) - G(w^*) \geq \frac{3L\|w^{[0]} - w^*\|^2_2}{32(k + 1)^2}.
  \]

- For any \( w^{[0]} \in \mathbb{R}^d \) and any \( k \leq (d - 1)/2 \), there exists a \( L \)-smooth, \( \mu \) strongly convex function \( f \) such that for any general first order method
  \[
  G(w^{[k]}) - G(w^*) \geq \frac{\mu}{2} \left( \frac{1 - \sqrt{\mu/L}}{1 + \sqrt{\mu/L}} \right)^{2k} \|w^{[0]} - w^*\|^2_2.
  \]

- Accelerated rates: best possible without further assumptions.
### Outline

1. **Statistical Learning: Introduction, Setting and Risk Estimation**
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. **ML Methods: Probabilistic Point of View**
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. **ML Methods: Optimization Point of View**
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. **Optimization: Gradient Descent Algorithms**
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. **ML Methods: Neural Networks and Deep Learning**
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. **ML Methods: Trees and Ensemble Methods**
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. **Unsupervised Learning: Dimension Reduction and Clustering**
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

8. **Statistical Learning: PAC-Bayesian Approach and Complexity Theory**
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9. **References**
Goodness-of-Fit and Stochastic Gradient Heuristic

Goodness-of-Fit Optimization

- Minimizer of $G = F + R$ with
  - $F$ smooth and $R$ simple (as before).
  - Empirical average structure for $F$:
    \[
    F(w) = \frac{1}{n} \sum_{i=1}^{n} F_i(w) = \mathbb{E}_n[F_i(w)]
    \]
    where each $F_i$ is smooth.

- Cost of evaluating $\nabla F$ is proportional to the dataset cardinality $n$ (and the dimension $d$).
- If the dataset is large, this can be an issue.
- **Stochastic Gradient Heuristic**: replace $\nabla F(w) = \mathbb{E}_n[\nabla F_i(w)]$ by a Monte Carlo approximation.
Stochastic Gradient

**Empirical average structure**

\[
\nabla F(w) = \frac{1}{n} \sum_{i=1}^{n} \nabla F_i(w) = \mathbb{E}_n[\nabla F_i(w)]
\]

**Monte Carlo approximation of an average:**

- draw uniformly \(m \ll n\) indices \(i_1, \ldots, i_m\)
- Replace \(\nabla F(w)\) by

\[
\hat{\nabla} F(w) = \frac{1}{m} \sum_{j=1}^{m} \nabla F_{i_j}(w)
\]

**Prop:** \(\mathbb{E}[\hat{\nabla} F(w)] = \nabla F(w)\)

- Extreme case \(m = 1\): unbiased but quite noisy estimate of the true gradient.
- Evaluation cost of \(\hat{\nabla} F(w)\) independent of \(n\).
- **Question:** Can we use this approximation of the true gradient in a descent algorithm?
Stochastic Gradient Descent Algorithm

- Start from a point $w^{[0]}$ and let $k = 0$.
- Repeat until convergence:
  - Draw $m$ indices $1 \leq i_j \leq n$ uniformly at random
  - Set $\hat{\nabla}F(w^{[k]}) = \frac{1}{m} \sum_{j=1}^{m} \nabla F_i(w^{[k]})$
  - $w^{[k+1]} = w^{[k]} - \alpha^{[k]} \left( \hat{\nabla}F(w^{[k]}) + \nabla R(w^{[k]}) \right)$
  - $k \rightarrow k + 1$

  with $\alpha^{[k]}$ a sequence of (small) steps.

- Each iteration has complexity $O(md)$ instead of $O(nd)$ for full gradient methods
- $m$ is called the batch size.
- **Rk:** no gain when using a batch size larger than 1, except if the sum can be parallelized.
- Proximal variant if $R$ is simple instead of smooth.
Convergence Rate of SGD

- Theoretical analysis requires some modifications:
  - at each step $w^{[k+1]}$ should be projected into a ball $B(c, R)$ with $R > 0$.
  - $G$ should be convex such that $\sup_i |\nabla F_i(w) + \nabla R(w)| \leq b$, $\forall w \in B(c, r)$
  - **Polyak-Ruppert** averaging: use SGD iterates $w^t$ but return an average.

### SGD Rate

- With $\alpha^{[k]} = 2R/(b\sqrt{k})$

\[
\mathbb{E} \left[ G\left( \frac{1}{k} \sum_{j=1}^{k} w^{[j]} \right) \right] - G(w^*) \leq \frac{3rb}{\sqrt{k}}
\]

- If $G$ is $\mu$-strictly convex then with $\alpha^{[k]} = 2/(\mu(k + 1))$,

\[
\mathbb{E} \left[ G\left( \frac{2}{k(k + 1)} \sum_{j=1}^{k} jw^{[j]} \right) \right] - G(w^*) \leq \frac{2b^2}{\mu(k + 1)}.
\]

- Without averaging, logarithmic loss.
- Same rate for proximal algorithm if $F$ is at least $L$-smooth and $R$ simple.
Convergence Rate of SGD

Simplified Result for the (Averaged) SGD

- If $G$ is convex and gradients are bounded ($\|\nabla F_i(w)\|_2 \leq b$) then the convergence rate is
  \[ O\left(\frac{1}{\sqrt{k}}\right) \text{ with } \alpha[k] = O\left(\frac{1}{\sqrt{k}}\right) \]
- $G$ is $\mu$-strongly convex, the rate is
  \[ O\left(\frac{1}{\mu k}\right) \text{ with } \alpha[k] = O\left(\frac{1}{\mu k}\right) \]

Comparison with GD

- Much slower rate:
  - for $L$ smooth function: $O(1/\sqrt{k})$ vs $O(1/k)$
  - For $\mu$ convex function: $O(1/k)$ vs $O((1 - \mu/L)^k)$
  - Hidden factor $\sigma$ in SGD measuring the variance of $\hat{\nabla} F$ which decreases when $m$ increases.
- Much cheaper cost per iteration: $O(md)$ vs $O(nd)$
SGD with Fixed Step Size

**Stochastic Gradient Descent Algorithm**

- Start from a point $\mathbf{w}^0$ and let $k = 0$.
- Repeat until convergence:
  - Draw $m$ indices $1 \leq i_j \leq n$ uniformly at random
  - Set $\widehat{\nabla} F(\mathbf{w}[k]) = \frac{1}{m} \sum_{k=1}^{m} \nabla F_{i_j}(\mathbf{w}[k])$
  - $\mathbf{w}[k+1] = \mathbf{w}[k] - \alpha \left( \widehat{\nabla} F(\mathbf{w}[k]) + \nabla R(\mathbf{w}[k]) \right)$
  - $k \rightarrow k + 1$

with $\alpha$ constant.

- No convergence but for $\mu$-convex function:
  $$\mathbb{E} \left[ \| \mathbf{w}[k] - \mathbf{w}^* \|^2 \right] \rightarrow O(\alpha/\mu)$$

- SGD behavior in practice: very fast initially but much slower afterward.
- Justify the use of decreasing step size when the error decay becomes slow.
Stabilization

- Difference due to the variance of the stochastic gradient.

Variance reduction of the gradient

- In the iterations of SGD, replace \( \nabla F_{i_k}(w^{[k-1]}) \) by
  \[
  \nabla F_{i_k}(w^{[k-1]}) - \nabla F_{i_k}(\tilde{w}) + \nabla F(\tilde{w})
  \]
  where \( \tilde{w} \) is an old value of the iterate, namely use
  \[
  w^{[k]} \leftarrow w^{[k-1]} - \alpha^{[k]}((\nabla F_{i_k}(w^{[k-1]}) - \nabla F_{i_k}(\tilde{w})) + \nabla F(\tilde{w}))
  \]
- Several instantiations
  - SVRG where \( \tilde{w} \) and \( \nabla F(\tilde{w}) \) is computed every few steps.
  - SAGA where, instead of \( \nabla F_i(\tilde{w}) \), one use \( F_i(w^{[k_i]}) \), where \( k_i \) is the last time the \( i \)th coordinate has been updated, even to compute the equivalent of \( \nabla F(\tilde{w}) \).
- Lead to faster rate but requires more memory (3d for SVRG and nd for SAGA).
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9 References
Adaptive Step Size

Stochastic Gradient Descent Algorithm

- Start from a point $\mathbf{w}^{(0)}$ and let $k = 0$.
- Repeat until convergence:
  - Draw $m$ indices $1 \leq i_j \leq n$ uniformly at random
  - Set $\hat{\nabla} F(\mathbf{w}[k]) = \frac{1}{m} \sum_{k=1}^{m} \nabla F_{i_j}(\mathbf{w}[k])$
  - $\mathbf{w}[k+1] = \mathbf{w}[k] - \alpha[k] \left( \hat{\nabla} F(\mathbf{w}[k]) + \nabla R(\mathbf{w}[k]) \right)$
  - $k \to k + 1$

with $\alpha[k]$ a sequence of (small) steps.

- Step sizes $\alpha[k]$ are crucial for convergence.
- How to choose them? Decaying? Fixed? Adaptative?
- Several algorithms available with no clear winner.

Determining a good learning rate becomes more of an art than science for many problems.

M.D. Zeiler
**Gradient Descent and Reparameterization**

- **Gradient Descent:** \( \mathbf{w}^{[k+1]} = \mathbf{w}^{[k]} - \alpha \nabla G(\mathbf{w}^{[k]}) \).
- **Reparameterization:** \( \mathbf{w} = \mathbf{Wz} \).
- **Gradient Descent in** \( \mathbf{z} \): \( \mathbf{z}^{[k+1]} = \mathbf{z}^{[k]} - \alpha \mathbf{W}^\top \nabla G(\mathbf{Wz}^{[k]}) \).
- **Implied Gradient Descent in** \( \mathbf{w} \): \( \mathbf{w}^{[k+1]} = \mathbf{w}^{[k]} - \alpha \mathbf{WW}^\top \nabla G(\mathbf{w}^{[k]}) \).
- **Different dynamics if** \( \mathbf{WW}^\top \neq \mathbf{Id} \!**

**Gradient Descent and Norm**

- **Descent lemma upperbound:** \( f(\mathbf{w}') \leq f(\mathbf{w}) + \nabla G(\mathbf{w})^\top (\mathbf{w}' - \mathbf{w}) + \frac{1}{2\alpha} \| \mathbf{w}' - \mathbf{w} \|^2 \).
- **Uses the classical euclidean norm.**

**Using** \( f(\mathbf{w}') \leq f(\mathbf{w}) + \nabla G(\mathbf{w})^\top (\mathbf{w}' - \mathbf{w}) + \frac{1}{2\alpha} \| \mathbf{w}' - \mathbf{w} \|^2 \Sigma \) **leads to**

\[
\mathbf{w}^{[k+1]} = \mathbf{w}^{[k]} - \alpha \Sigma^{-1} \nabla G(\mathbf{w}^{[k]}).
\]

- **Different dynamics if** \( \Sigma \neq \mathbf{Id} \!**
Gradient Descent

Modified Gradient Descent Algorithm

- \( \mathbf{w}^{[k+1]} = \mathbf{w}^{[k]} - \alpha P \nabla G(\mathbf{w}^{[k]}) \) where \( P \) can be interpreted as \( WW^T \) or \( \Sigma \) of the previous slide.
- Convergence holds if corresponding descent lemma holds.
- \( P \) can even changes from one iteration to another.

Choices for \( P \)

- Newton method:
  - \( P = (\nabla^2 G)^{-1} \)
  - Hard to compute and no descent lemma property.
- Diagonal approximation:
  - \( P = D \)
  - Easy computation but no descent lemma in general.
  - Several strategies for the choice of \( D \ldots \)

- Coordinate Descent could be reframed in this framework!
Self Normalized Gradient Descent Algorithm

\[ w_j^{[k+1]} = w_j^{[k]} + \frac{\alpha^{[k]}_b}{d_j^{[k]}} m_j^{[k]} \]

**Self Normalized Gradient Descent**

- Gradient: \( m^{[k]} \simeq \nabla G(w^{[k]}) \)
- Renormalization factor: \( d_j^{[k]} \simeq |\nabla G(w^{[k]})|_j \)
- Base stepsize: \( \alpha^{[k]}_b \) could be constant or decaying...

- Makes the algorithm invariant to a diagonal rescaling.
- Amounts to use

\[ \| w' - w \|^2 \simeq \sum_j \frac{|w'_j - w_j|^2}{|\nabla G(w)|^2} \]
Optimization: Gradient Descent Algorithms

**ADAdaptive GRADient**

\[ w_j^{[k+1]} = w_j^{[k]} + \frac{\alpha_b^{[k]}}{d_j^{[k]}} m_j^{[k]} \]

**ADAdaptive GRADient update**

- \( m^{[k]} = \nabla G(w^{[k]}) \)
- \( d_j^{[k]} = \sqrt{\sum_{k'=1}^{k} (\nabla G(w^{[k']})_j)^2} \)
- \( \alpha_b^{[k]} = \alpha \)

- Step sizes grow as the inverse of the gradient magnitude.
- Accumulation of the gradients acts as a decreasing learning rate.
- Sensitive to initial condition and may require large initial parameter or restarts.
Optimization: Gradient Descent Algorithms

ADAptive GRADient

\[ w_j^{[k+1]} = w_j^{[k]} + \frac{\alpha_b^{[k]}}{d_j^{[k]}} m_j^{[k]} \]

**ADAptive GRADient update reformulation**

- \( m^{[k]} = \nabla G(w^{[k]}) \)
- \( d_j^{[k]} = \sqrt{v_j^{[k]}} \) with \( v_j^{[k]} = \frac{1}{k} \sum_{k'=1}^{k} (\nabla G(w^{[k']}))_j^2 \)
- \( \alpha_b^{[k]} = \alpha / \sqrt{k} \)

- Explicit averaging and decay.
RMSprop

\[ w_j^{[k+1]} = w_j^{[k]} + \frac{\alpha_b^{[k]}}{d_j^{[k]} m_j^{[k]}} \]

**RMSprop update**

- \( m^{[k]} = \nabla G(w^{[k]}) \)
- \( d_j^{[k]} = \sqrt{v_j^{[k]}} \) with \( v_j^{[k]} = \rho v_j^{[k-1]} + (1 - \rho)(\nabla G(w^{[k]})_j)^2 \)
- \( \alpha_b^{[k]} = \alpha \)

- Unpublished method!
- Exponential average instead of classical one.
- No step size decay.
- Often very efficient (especially in a non stationary setting).
ADAM: ADAdptive Moment estimation

\[ w_j^{[k+1]} = w_j^{[k]} + \frac{\alpha[k]}{d_j^{[k]}} m_j^{[k]} \]

**ADAM**

- Exponential averaging estimation: let \( \tilde{m}_0 = \tilde{v}_0 = 0 \),
  \[ \tilde{m}^{[k]} = \beta_m \tilde{m}^{[k-1]} + (1 - \beta_m) \nabla G(w^{[k]}) \]
  and
  \[ \tilde{v}^{[k]} = \beta_v \tilde{v}^{[k-1]} + (1 - \beta_v) (\nabla G(w^{[k]}))^2 \]

- Bias correction: \( m^{[k]} = \frac{\tilde{m}^{[k]}}{1 - \beta_m^k} \)
  and \( v^{[k]} = \frac{\tilde{v}^{[k]}}{1 - \beta_v^k} \)

- \( \alpha[k] = \alpha \).

- Use a smoothed estimation of the first two moments of the gradients.
- Often efficient.
- ADAMax: \( v^{[k]} = \tilde{v}^{[k]} = \max(\beta_v v^{[k-1]}, |\nabla G(w^{[k]})|) \).
Optimization: Gradient Descent Algorithms

AdaDelta

\[ \mathbf{w}_j^{[k+1]} = \mathbf{w}_j^{[k]} + \frac{\alpha_b^{[k]}}{d_j^{[k]}} m_j^{[k]} \]

- \( m^{[k]} = \nabla G(\mathbf{w}^{[k]}) \)
- \( d_j^{[k]} = \frac{\sqrt{v_j^{[k]}}}{\sqrt{\tilde{v}^{[k-1]}}} \) with \( v_j^{[k]} = \rho v_j^{[k-1]} + (1 - \rho)(\nabla G(\mathbf{w}^{[k]})^2 \) and \( \tilde{v}_j^{[k]} = \rho \tilde{v}_j^{[k-1]} + (1 - \rho) \left( \frac{\alpha_j^{[k]}}{d_k^{[k]}} \nabla G(\mathbf{w}^{[k]})_j \right)^2 \)
- \( \alpha_b^{[k]} = 1 \)

- Only one parameter!
- Based on a dimensional analysis... with a strange looking renormalization!
Optimization: Gradient Descent Algorithms

\[ w_j^{[k+1]} = w_j^{[k]} + \frac{\alpha_j^{[k]}}{d_j^{[k]}} m_j^{[k]} \]

AdaDelta

- \( m^{[k]} = \nabla G(w^{[k]}) \) and \( d_j^{[k]} = 1 \)
- \( \alpha_j^{[0]} = \alpha \) and \( \alpha_j^{[k]} = \frac{\alpha_j^{[k-1]}}{\sqrt{\rho + (1 - \rho)(\nabla G(w^{[k]}))_j^2 / v_j^{[k-1]}}} \) with
  \[
  v_j^{[k]} = \rho v_j^{[k-1]} + (1 - \rho)(\nabla G(w^{[k]}))_j^2 , \text{ and} \]
  \[
  v_j^{[k]} = \rho \tilde{v}_j^{[k-1]} + (1 - \rho) \left( \alpha_j^{[k]} \right)^2 \left( \nabla G(w^{[k]})_j^2 \right) , \text{ and} \]
  \[
  \alpha_j^{[k-1]} = \sqrt{\tilde{v}_j^{[k]} / v[k]_j} \]

- Better insight (and more flexible initial step)!
- Quite different from the previous schemes.
Cyclic Learning Rate

- Cycle between small and large learning rate with stopping when the learning rate is small.
- One cycle strategy seems possible!
- Plain SGD with a constant step is also making a comeback!
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

9. References
Optimization: Gradient Descent Algorithms

(S)GD and Non Convex Function

- No global convergence result for non convex functions optimization (NP Hard problem)
- Nevertheless (Stochastic) Gradient Descent algorithm can be used.

**Theoretical results?**

- Typical weak convergence results:
  - convergence to a stationary point
  - convergence to a local minimum...
- Often in specific cases.

- In practice: convergence depends a lot on the step size $\alpha$ choice strategy.
- In general, more exploration than in the convex setting.
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9. References
References

T. Hastie, R. Tibshirani, and J. Friedman.  
*The Elements of Statistical Learning.*  
Springer Series in Statistics, 2009

M. Mohri, A. Rostamizadeh, and A. Talwalkar.  
*Foundations of Machine Learning (2nd ed.)*  
MIT Press, 2018

A. Géron.  
*Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (3rd ed.)*  
O’Reilly, 2022

Ch. Giraud.  
*Introduction to High-Dimensional Statistics.*  
CRC Press, 2014

S. Bubeck.  
*Convex Optimization: Algorithms and Complexity.*  
Now Publisher, 2015
Creative Commons Attribution-ShareAlike (CC BY-SA 4.0)

- You are free to:
  - **Share:** copy and redistribute the material in any medium or format
  - **Adapt:** remix, transform, and build upon the material for any purpose, even commercially.

- Under the following terms:
  - **Attribution:** You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - **ShareAlike:** If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - **No additional restrictions:** You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Contributors

- **Main contributor:** E. Le Pennec
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
     - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?

8. Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability

9. Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

10. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
    - Supervised Learning
    - Empirical Risk Minimization
      - Empirical Risk Minimization
      - ERM and PAC Analysis
      - Hoeffding and Finite Class
      - McDiarmid and Rademacher Complexity
      - VC Dimension
      - Structural Risk Minimization
    - References
Perceptron

Perceptron (Rosenblatt 1957)

- Inspired from biology.
- Very simple (linear) model!
- Physical implementation and proof of concept.
Perceptron

- Inspired from biology.
- Very simple (linear) model!
- Physical implementation and proof of concept.

Perceptron (Rosenblatt 1957)
Perceptron

Perceptron (Rosenblatt 1957)

- Inspired from biology.
- Very simple (linear) model!
- Physical implementation and proof of concept.
Perceptron

Perceptron (Rosenblatt 1957)

- Inspired from biology.
- Very simple (linear) model!
- Physical implementation and proof of concept.
Artificial Neuron and Logistic Regression

**Artificial neuron**
- Structure:
  - Mix inputs with a **weighted sum**, 
  - Apply a (non linear) **activation function** to this sum, 
  - Possibly threshold the result to make a decision. 
- Weights learned by minimizing a loss function.
- Equivalent to linear regression when using a linear activation function!

**Logistic unit**
- Structure:
  - Mix inputs with a **weighted sum**, 
  - Apply the **logistic function** \( \sigma(t) = \frac{e^t}{1 + e^t} \), 
  - Threshold at 1/2 to make a decision! 
- Logistic weights learned by minimizing the -log-likelihood.
Multilayer Perceptron

MLP (Rumelhart, McClelland, Hinton - 1986)

- Multilayer Perceptron: cascade of layers of artificial neuron units.
- Optimization through a gradient descent algorithm with a clever implementation (Backprop).
- Construction of a function by composing simple units.
- MLP corresponds to a specific direct acyclic graph structure.
- Non convex optimization problem!
Multilayer Perceptron

Neural Network

Decision region

Decision boundary

classes

- Class1
- Class2
Deep Neural Network structure

- Deep cascade of layers!
- No conceptual novelty…
- But a lot of tricks allowing to obtain a good solution: clever initialization, better activation function, weight regularization, accelerated stochastic gradient descent, early stopping…
- Use of GPU and a lot of data…
- Very impressive results!
Deep Neural Network

H2O NN

Decision region

Decision boundary

PredictorB

PredictorA

classes

Class1

Class2

classes

Class1

Class2
Deep Learning

Family of Machine Learning algorithm combining:

- a (deep) multilayered structure,
- a clever optimization including initialization and regularization.

- Examples: Deep NN, AutoEncoder, Recursive NN, GAN, Transformer...
- Interpretation as a **Representation Learning**.
- **Transfer learning**: use as initialization a pretrained net.
- Very efficient and still evolving!
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
   - Generative Modeling
   - Dimension Reduction
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?

8. Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

10. References
Logistic Regression as NN

Binary Logistic Regression

- Conditional probability model:
  \[ P(Y = 1 | X) = \sigma(w^\top X + b) = \frac{e^{w^\top X + b}}{1 + e^{w^\top X + b}} \]

- Model weights \( w \in \mathbb{R}^d \) and intercept (or bias) \( b \in \mathbb{R} \).

Artificial Neuron or Unit

- \( X \) is the input (single feature vector).
- \( z(X) = w^\top X + b \) is called pre-activation.
- \( y(X) = \sigma(z(X)) \) is the output (in \([0, 1]\) in this case)
- \( w \) is weights and \( b \) is bias
- \( \sigma \) is an activation function
Softmax Regression

Multinomial Logistic Model

- Cond. prob. model: \( P(Y = k | X) = \frac{e^{W_k^T X + b_k}}{\sum_{k'=1}^{K} e^{W_{k'}^T X + b_{k'}}} \)

- Model weights: \( k \) vector \( W_k \in \mathbb{R}^d \) or equivalently a model weight matrix \( W \) with \( W_{*,k} = W_k \)

Softmax Regression (Softmax Layer)

- \( z_k(X) = W_{*,k}^T X + b_k \) pre-activations or logits

- \( y_k(X) = \frac{e^{z_k(X)}}{\sum_{k'=1}^{K} e^{z_{k'}(X)}} \) coming out of the softmax activation

- Matricially: \( y(X) = \text{softmax}(z(X)) = \text{softmax}(W^T X + b) \)
One-Hidden Layer Neural Network

Recursive definition:

\[ y(X) = \text{softmax}(z^{(O)}) = \text{softmax}(W^{(O)\top} h + b^{(O)}) \]
\[ = \text{softmax}(W^{(O)\top} g(z^{(H)}) + b^{(O)}) \]
\[ = \text{softmax}(W^{(O)\top} g(W^{(H)\top} x + b^{(H)}) + b^{(O)}) \]

- \( g \) is an activation function applied entrywise on \( z_1^H, \ldots, z_H^H \)

- This neural network has a width-\( H \) hidden layer
One-Hidden Layer Neural Network

**Vector representation:**

\[ y(X) = \text{softmax}(z^{(O)}) = \text{softmax}(\mathbf{W}^{(O)^T} h + b^{(O)}) \]
\[ = \text{softmax}(\mathbf{W}^{(O)^T} g(z^{(H)}) + b^{(O)}) \]
\[ = \text{softmax}(\mathbf{W}^{(O)^T} g(\mathbf{W}^{(H)^T} x + b^{(H)}) + b^{(O)}) \]

- \( g \) is an **activation function** applied entrywise.

- Equivalent representation focusing on layers.
Entrywise Activation Functions

- Most classical: **sigmoid**, **tanh** and **ReLU** (rectified linear unit)

\[
\sigma(z) = \frac{1}{1 + e^{-z}}, \quad \tanh(z) = \frac{e^{2z} - 1}{e^{2z} + 1}, \quad \text{ReLU}(z) = \max(0, z)
\]

- Their **derivatives** are given by

\[
\sigma'(z) = \sigma(z)(1 - \sigma(z))
\]
\[
\tanh(z) = 1 - \tanh(z)^2,
\]
\[
\text{ReLU}'(z) = 1_{z>0}
\]

- Note that \(\tanh(z) = 2\sigma(2z) - 1\)
Activation Functions

ReLU Activation

- ReLU: \( z \mapsto \max(0, z) \)
- Often the best choice for deep neural networks
- Easier to optimize, since their behavior is closer to linear
- Its gradient is not defined at \( z = 0 \): not a problem since, during training, it’s unlikely that any input equals 0

In contrast to ReLu, Sigmoid activations \( \sigma \) and tanh (not recommended anymore) saturate for large positive (or large negative values).

Identity Activation

- Hidden layer:
  - Amount to use \( h = g(VU^T z + b) \) instead of \( h = g(W^T z + b) \)
  - \( W \) is factorized as \( UV^T \)
  - Reduces the dimension of the model
- Output layer: linear model on the last layer
Softmax

- Softmax is not an element-wise activation:

\[
\text{softmax}(z) = \frac{1}{\sum_{k=1}^{K} e^{z_k}} \begin{bmatrix} e^{z_1} \\ \vdots \\ e^{z_K} \end{bmatrix}
\]

- Gradient:

\[
\frac{\partial \text{softmax}(z)_k}{\partial x_{k'}} = \begin{cases} 
\text{softmax}(z)_k \times (1 - \text{softmax}(z)_k) & \text{if } k = k' \\
-\text{softmax}(z)_k \times \text{softmax}(z)_{k'} & \text{otherwise}
\end{cases}
\]

- Maps \( z \in \mathbb{R}^K \) to the space of vectors in \([0, 1]^K\) with entries that sum to 1 (probabilities)

- The inputs of the softmax are called the logits in deep learning
Feed-Forward Neural Network

Heuristic: It’s easy to construct a complex function by composing simple elements in some order.

Layerwise structure:

\[
\begin{align*}
    h^{(1)} &= g^{(1)}(W^{(1)\top}x + b^{(1)}) \\
    h^{(2)} &= g^{(2)}(W^{(2)\top}h^{(1)} + b^{(2)}) \\
    & \quad \vdots \\
    h^{(L)} &= g^{(L)}(W^{(L)\top}h^{(L-1)} + b^{(L)}) \\
    y &= \text{softmax}(W^{(O)\top}h^{(L)} + b^{(O)})
\end{align*}
\]

Need to choose the width of each layer and the depth of the network.

The name comes from the fact that information flows through the layers.

For regression, replace the softmax by identity.
Universal Approximation Theorem (Hornik, 1991)

- A single hidden layer neural network with a linear output unit can approximate any continuous function arbitrarily well, given enough hidden units.

- Valid for most activation functions.
- No bounds on the number of required units... (Asymptotic flavor)
- A single hidden layer is sufficient but more may require less units?
Network Architecture

What is best?
- One hidden layer with large width?
- ...or several layers with smaller width?

A Recipe
- For the same number of parameters, several layers with a smaller width lead to better generalization. (More abstract layers)
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
     - NN Regularization
     - Image and CNN
     - Text, Recurrent Neural Networks and Transformers
     - NN Architecture
     - References
   - ML Methods: Trees and Ensemble Methods
     - Trees
     - Bagging and Random Forests
       - Bootstrap and Bagging
       - Randomized Rules and Random Forests
     - Boosting
       - AdaBoost as a Greedy Scheme
       - Boosting
       - Ensemble Methods
       - References
   - Unsupervised Learning: Dimension Reduction and Clustering
     - Unsupervised Learning?
     - A First Glimpse
       - Clustering
       - Dimensionality Curse
       - Dimension Reduction
       - Generative Modeling
     - Dimension Reduction
       - Simplification
       - Reconstruction Error
       - Relationship Preservation
       - Comparing Methods?
     - Clustering
       - Prototype Approaches
       - Contiguity Approaches
       - Agglomerative Approaches
       - Other Approaches
       - Scalability
     - Generative Modeling
       - (Plain) Parametric Density Estimation
       - Latent Variables
       - Approximate Simulation
       - Diffusion Model
       - Generative Adversarial Network
     - Applications to Text
     - References

6 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

7 Supervised Learning
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

8 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
     - NN Regularization
     - Image and CNN
     - Text, Recurrent Neural Networks and Transformers
     - NN Architecture
     - References
   - ML Methods: Trees and Ensemble Methods
     - Trees
     - Bagging and Random Forests
       - Bootstrap and Bagging
       - Randomized Rules and Random Forests
     - Boosting
       - AdaBoost as a Greedy Scheme
       - Boosting
       - Ensemble Methods
       - References
   - Unsupervised Learning: Dimension Reduction and Clustering
     - Unsupervised Learning?
     - A First Glimpse
       - Clustering
       - Dimensionality Curse
       - Dimension Reduction
       - Generative Modeling
     - Dimension Reduction
       - Simplification
       - Reconstruction Error
       - Relationship Preservation
       - Comparing Methods?
     - Clustering
       - Prototype Approaches
       - Contiguity Approaches
       - Agglomerative Approaches
       - Other Approaches
       - Scalability
     - Generative Modeling
       - (Plain) Parametric Density Estimation
       - Latent Variables
       - Approximate Simulation
       - Diffusion Model
       - Generative Adversarial Network
     - Applications to Text
     - References

9 References
How to Train a DNN

- Use a differentiable loss.

**Loss Minimization**

- Compute the prediction and the loss *(forward-propagation)*
- Compute the gradients *(back-propagation)*
- Use a Stochastic Gradient Descent algorithm

- In practice, use
  - A nice open source library *(tensorflow, keras, pyTorch)*
  - A GPU...since training a DNN can benefit from massive parallel computations
Neural Network Optimization Formulation

Feed-Forward Neural Network

- Network structure:
  \[ h^{(1)} = g^{(1)}(W^{(1)\top} x + b^{(1)}) \]
  \[ h^{(2)} = g^{(2)}(W^{(2)\top} h^{(1)} + b^{(2)}) \]
  \[ \ldots \]
  \[ h^{(L)} = g^{(L)}(W^{(L)\top} h^{(L-1)} + b^{(L)}) \]

- Output function:
  \[ f(x, w) = \text{softmax}(W^{(O)\top} h^{(L)} + b^{(O)}) \]
  with \( w = (W^{(1)}, b^{(1)}, \ldots, W^{(L)}, b^{(L)}, W^{(o)}, b^{(o)}) \)

Optimization Formulation

- Minimization of
  \[ G(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i, w)) + \underbrace{\text{pen}(w)}_{R(w)} \]
  with \( \ell \) a loss function and \( \text{pen} \) a penalization.

- Most classical choices:
  - \( \ell \) is the cross entropy (-log-likelihood)
  - \( \text{pen} \) is a ridge penalty on the \( W \).
Optimization Algorithm

Optimization Formulation

- Minimization of

\[ G(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i, w)) + \text{pen}(w) \]

\( F(w) \)

with \( \ell \) a loss function and \( \text{pen} \) a penalization.

Optimization Algorithm

- Non convex optimization problem.
- Stochastic Gradient Descent algorithm.
- Key issue: computing

\[ \nabla F_i(w) = \nabla \ell(Y_i, f(X_i, w)) \]
Clever algorithm to compute $\nabla F_i(w) = \nabla \ell(Y_i, f(X_i, w))$. 

Key observation: $f$ is obtained through a direct acyclic graph of composition.

Key lemma (Chain Rule): $\frac{\partial}{\partial w} f(g(x, w)) = \frac{\partial}{\partial g}(g(x, w)) \frac{\partial g}{\partial w}(x, w)$.

More generally: $\frac{\partial z}{\partial x_i} = \sum_j \frac{\partial z}{\partial y_j} \frac{\partial y_j}{\partial x_i}$.

Back-propagation: use of this chain rule to compute the derivatives in $\nabla F_i(w)$ starting from the parameters of the last layer and going backward to the ones of the first one.
Partial Derivative and Back Propagation

Set of Partial Derivative Linked $\nabla F_i(\mathbf{w}) = \nabla \ell(Y_i, f(\mathbf{X}_i, \mathbf{w}))$

- **Loss:** $\frac{\partial F_i}{\partial f_j}(\mathbf{w}) = \frac{\partial \ell}{\partial f_j}(Y_i, f(\mathbf{X}_i, \mathbf{w}))$.

- **Output layer:**
  - Softmax: $\frac{\partial f_i}{\partial z_k^{(o)}}(\mathbf{w})$ is known.
  - Pre-activation at the output layer:
    - Parameters: $\frac{\partial z_j^{(o)}}{\partial W_{j,k}^{(o)}} = h_k^{(L)}(\mathbf{w})$ and $\frac{\partial z_j^{(o)}}{\partial b^{(o)}} = 1$
    - Previous Layer: $\frac{\partial z_j^{(o)}}{\partial h_j^{(L)}} = \mathbf{W}_{j,k}^{(0)}$

- **Layer $l$**:
  - Activation: $\frac{\partial h_j^{(l)}}{\partial z_j^{(l)}} = g_j^{(l)'}(z_j^{(l)})$
  - Pre-activation:
    - Parameters: $\frac{\partial z_j^{(l)}}{\partial W_{j,k}^{(l)}} = h_{k}^{l-1}(\mathbf{w})$ and $\frac{\partial z_j^{(l)}}{\partial b^{(l)}} = 1$
    - Previous Layer: $\frac{\partial z_j^{(l)}}{\partial h_j^{(l-1)}} = \mathbf{W}_{j,k}^{(l)}$

- Recurs. comput. with the chain rule $\Rightarrow$ efficient comput. of $\frac{\partial F_i}{\partial W_{j,k}}$ and $\frac{\partial F_i}{\partial b}$.
Back-Propagation Algorithm

- This set is sufficient to compute $\nabla F_i(w) = \nabla \ell(Y_i, f(X_i, w))$.
- Those partial derivatives can be computed in a backward-pass...
- provided all the intermediate values of $h$ and $z$ have been computed in a forward-pass
SGD and DNN

- Nearly all of deep learning is powered by **SGD**.

### Stochastic Gradient Descent for DNN

- Regular Stochastic Gradient Algorithm with
  - adaptive step size (ADAM, RMSProp...) or fixed size with or without restarts,
  - batch size larger than 1 to use GPU parallelism but not too large.

### Initialization

- SGD does not work without a good initialization scheme.

- **Key properties:**
  - Break symmetry by random initialization around 0.
  - Order of magnitude at layer $l$ given by $1 / \sqrt{H_l + H_{l-1}}$ provided the input is normalized.

- **Examples:**
  - Normal with $\mathcal{N}(0, 2/(H_l + H_{l-1}))$
  - Uniform with $\mathcal{U}(-\sqrt{6}/(H_l + H_{l-1}), \sqrt{6}/(H_l + H_{l-1}))$
A Challenging Optimization Problem

Challenges

- Non convex optimization in very high dimension.
- SGD can be stuck around critical point and the gradient may not diminishes.
- Several local minima due to invariance by permutation and scaling.
- Some local minima may be bad.

- **Natural concerns but it seems that for sufficiently large network most of the local minima have a similar low cost value.**
- Larger networks may be easier to train than smaller ones...
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
Regularization by Penalization or Projection

Penalization or Projection

- Regularization on weights and not on bias.
- Mostly $\ell^1$ or $\ell^2$.
- Two possibilities:
  - penalization: $\text{pen}(W) = \lambda \| W \|
  - projection: $\| W \| \leq r$

- Projection seems to work better as it only affects large weights.
- Avoid dead units.
Regularization by Early Stopping

- SGD optimizes Train error but true goal is Test error.
- **Early Stopping**: stop SGD when test error rises.
- In practice, store the parameters each time the test error improves and use the best set of parameters in the end.

- Most widely used of regularization in DL.
- Number of iterations becomes a method parameter.
- For least-squares regression and gradient descent, early stopping \(\approx\) ridge penalization on the weights.

---

Source: Goodfellow, Bengio and Courville
Regularization by Dropout

**Dropout**

- Idea: At each step, disable temporarily some connections by modifying temporarily the weights.
- Importance sampling idea:
  \[
  \tilde{W}_{j,k}^{(\cdot)} = \begin{cases} 
  0 & \text{with probability } p \\
  \frac{1}{(1-p)} W_{j,k}^{(\cdot)} & \text{with probability } 1 - p
  \end{cases}
  \]
- The average value of the weights are preserved.
- Interpretation in term of ensemble methods (bagging)
- No dropout after training!
Regularization by Batch Normalization

- Idea: Maintain the scales of the neuron outputs during training.

### Batch Normalization Layer

- During training, at each step and for each neuron:
  - Compute the mean value on the current batch
    \[
    \mu_{k}^{(l)} = \frac{1}{m} \sum_{i=1}^{m} h_{k}^{(l)}(X_i, w)
    \]
  - Compute the variance on the current batch
    \[
    \sigma_{k}^{2,(l)} = \frac{1}{m} \sum_{i=1}^{m} (h_{k}^{(l)}(X_i, w) - \mu_{k}^{(l)})^2
    \]
  - Renormalize the entries
    \[
    \tilde{h}_{k}^{(l)}(X_i, w) = \frac{h_{k}^{(l)}(X_i, w) - \mu_{k}^{(l)}}{\sigma_{k}^{(l)}}
    \]
  - Train (update) the scale and shift parameters:
    \[
    y_{k}^{(l+1)} = \gamma_{k}^{(l+1)} \tilde{h}_{k}^{(l)}(X_i, w) + \beta_{k}^{(l+1)}
    \]

- Differentiability in \( \gamma \) and \( \beta \): SGD to train them.

- Final freeze of the mean and the variance yield the final NN.

- Helps a lot!
Regularization by Pooling

- Idea: replace some outputs by a statistic.

### Pooling

- Replace values by a local summary statistic.
- Most classical summary choice: maximum.
- (Sub)Gradient can still be computed.
- Often combine with a subsampling to reduce the network size.

- Most useful when output position matters as in images.
- Very useful to deal with pictures of various sizes.
Regularization by Data Augmentation

- Idea: Augment artificially the number of samples

Data Augmentation

- Augment the dataset by generating new samples from original ones.
- Often mild variations of existing one: shift, rotation, zoom, noise...
- Often very efficient if the variations used correspond to specific problem invariance.
- Not so easy to generate completely new samples (GAN?)
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9 References

ML Methods: Neural Networks and Deep Learning
Convolutional Neural Networks (CNN)

- Reduce number of network weights:
  - Use only **local** computation (locality of information),
  - Use the **same weights** everywhere (translation invariance)

- Keep the hierarchical structure.
Convolutional Layer

**Convolution**

- **Filter structure:**
  - Output is computed using only neighbors (locality of information)
  - Same computation with respect to the neighbors position everywhere (translation invariance)

- **Translation invariance ↔ convolution**

- **Subsampling of resulting output possible (stride) but issues with high frequency part of filters.**
- **Better subsampling stability after pooling.**
Maps and Layers

- Each layer contains a **stack** of outputs (3D tensor).
- Filter: convolution in 2D but free along the stack direction.
- Further reduction of the weight number using 2D spatial convolution followed by arbitrary 1D filter along stack.
LeNet (LeCun, 89)

AlexNet (Krishevsky et al., 12)

- Stack of convolutional layers followed by a fully connected one.
- Key is to be able to learn in less than a few days!
Text as Sequences

A recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed cycle. This creates an internal state of the network which allows it to exhibit dynamic temporal behavior. Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs. This makes them applicable to tasks such as unsegmented connected handwriting recognition or speech recognition.

Sequences

- Word = sequence of letters.
- Text = sequence of letters/words.
- Capitalize on this structure.
Recurrent Neural Networks

Recurrent Neural Network Unit

- Input seen as a sequence.
- **Simple** computational units with shared weights.
- Information transfer through a context!
- Several architectures!
Automatic Translation

Encoder/Decoder structure

- Word vectors, RNN, stacked structure.
Automatic Translation

Encoder/Decoder structure
- Much more complex structure: asymmetric, attention order...
Automatic Captioning

Encoder/Decoder structure

- Much more complex structure: asymmetric, attention order...
Text as Graph

- More than just sequential dependency.
- Each word is related to (all the) other words.
- Graph structure with words and directed relations between words.
Attention between words

- Words encoded by $h_i$ at layer $l$.
- Compute individual value for each word: $v_i = V^l h_i$
- Compute combined value for each word: $h'_i = \sum_j w_{i,j} v_j$
- **(Self) Attention:** weight $w_{i,j}$ defined by
  $$w_{i,j} = \text{SoftMax} \left( \langle Q^l h_i, K^l h_j \rangle \right)$$
  
  - $Q^l h_i$ is called a query and $K^l h_j$ a key.
Transformer

- Block combining several attention heads and a classical MLP.

Encoder/Decoder Architecture

- Combine several transformers and more MLP in a task-adapted architecture.
- End-to-end training is not easy (initialization, optimization...).
- Initial embedding at token level rather than word level to cope with new words!
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Dimension Reduction
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
Basic CNN Architecture

- Nb of layers, size of layers,
- Activation function, pooling...
- Optimization algorithm!

Source: Goodfellow, Bengio and Courville
CNN Architecture

- Lot of freedom!
ML Methods: Neural Networks and Deep Learning

Other Architectures

- Several structures:
  - Different tasks...
  - Different inputs.

- **Representation Learning is everywhere**

- Key: Differentiability by composition of basic differentiable units.
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9 References
References

T. Hastie, R. Tibshirani, and J. Friedman.  
*The Elements of Statistical Learning.*  
Springer Series in Statistics, 2009

M. Mohri, A. Rostamizadeh, and A. Talwalkar.  
*Foundations of Machine Learning (2nd ed.)*  
MIT Press, 2018

A. Géron.  
*Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (3rd ed.)*  
O’Reilly, 2022

Ch. Giraud.  
*Introduction to High-Dimensional Statistics.*  
CRC Press, 2014

S. Bubeck.  
*Convex Optimization: Algorithms and Complexity.*  
Now Publisher, 2015
References

I. Goodfellow, Y. Bengio, and A. Courville.  
*Deep Learning.*  
MIT Press, 2016

F. Chollet.  
*Deep Learning with Python (2nd ed.)*  
Manning, 2021

F. Chollet, T. Kalinowski, and J. J. Allaire.  
*Deep Learning with R (2nd ed.)*  
Manning, 2022

A. Géron.  
*Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (3rd ed.)*  
O'Reilly, 2022
Creative Commons Attribution-ShareAlike (CC BY-SA 4.0)

- You are free to:
  - **Share:** copy and redistribute the material in any medium or format
  - **Adapt:** remix, transform, and build upon the material for any purpose, even commercially.

- Under the following terms:
  - **Attribution:** You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - **ShareAlike:** If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - **No additional restrictions:** You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Contributors

- **Main contributor:** E. Le Pennec
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5 Gradient Descent Step
   - Non-Convex Setting
   - References

6 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

7 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

8 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
     - Applications to Text
   - References

9 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

10 References

11 ML Methods: Trees and Ensemble Methods
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neuronal Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?

8. Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability

9. Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

10. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
    - Supervised Learning
    - Empirical Risk Minimization
      - Empirical Risk Minimization
      - ERM and PAC Analysis
      - Hoeffding and Finite Class
      - McDiarmid and Rademacher Complexity
      - VC Dimension
      - Structural Risk Minimization
    - References

11. References
Guess Who?

A game of questions

- Game invented in 1979 in the UK.
- **Goal**: discover the character chosen by your opponent before he discovers yours.
- **Optimal strategy**: choose at each step the question that splits the remaining characters in two groups with the least possible difference in size.
- **Information Theory!**

- Adaptive construction of a tree of questions!
- Optimal tree of questions can be constructed without knowing the answers... but during a game only a path of the tree is used...
Classification And Regression Trees

Tree principle (CART by Breiman (85) / ID3 by Quinlan (86))

- Construction of a recursive partition through a tree structured set of questions (splits around a given value of a variable)
- For a given partition, probabilistic approach and optimization approach yield the same predictor!
- A simple majority vote/averaging in each leaf

- Quality of the prediction depends on the tree (the partition).
- **Intuitively:**
  - small leaves lead to low bias, but large variance
  - large leaves lead to large bias, but low variance...
- **Issue:** Minim. of the (penalized) empirical risk is NP hard!
- Practical tree construction are all based on two steps:
  - a top-down step in which branches are created (branching)
  - a bottom-up in which branches are removed (pruning)
CART
Branching

Greedy top-bottom approach

- Start from a single region containing all the data
- Recursively split those regions along a certain variable and a certain value
- **No regret strategy** on the choice of the splits!
- **Heuristic:** choose a split so that the two new regions are as *homogeneous* possible...
Branching

\[ X_1 < .5? \]
- Yes
- No

Greedy top-bottom approach

- Start from a single region containing all the data
- Recursively split those regions along a certain variable and a certain value

- **No regret strategy** on the choice of the splits!
- **Heuristic**: choose a split so that the two new regions are as *homogeneous* possible...
Branching

Greedy top-bottom approach

- Start from a single region containing all the data
- Recursively split those regions along a certain variable and a certain value

- **No regret strategy** on the choice of the splits!
- **Heuristic:** choose a split so that the two new regions are as \textit{homogeneous} possible...
Branching

Greedy top-bottom approach

- Start from a single region containing all the data
- Recursively split those regions along a certain variable and a certain value

- **No regret strategy** on the choice of the splits!
- **Heuristic**: choose a split so that the two new regions are as *homogeneous* possible...
**Branching**

Various definition of *inhomogeneous*

- **CART:** empirical loss based criterion (least squares/prediction error)
  \[
  C(R, \overline{R}) = \sum_{x_i \in R} \ell(y_i, y(R)) + \sum_{x_i \in \overline{R}} \ell(y_i, y(\overline{R}))
  \]

- **CART:** Gini index (Classification)
  \[
  C(R, \overline{R}) = \sum_{x_i \in R} p(R)(1 - p(R)) + \sum_{x_i \in \overline{R}} p(\overline{R})(1 - p(\overline{R}))
  \]

- **C4.5:** entropy based criterion (Information Theory)
  \[
  C(R, \overline{R}) = \sum_{x_i \in R} H(R) + \sum_{x_i \in \overline{R}} H(\overline{R})
  \]

- CART with Gini is probably the most used technique...

- Other criterion based on \(\chi^2\) homogeneity or based on different local predictors (generalized linear models...)
Branching

Choice of the split in a given region

- Compute the criterion for **all features and all possible splitting points** (necessarily among the data values in the region)
- Choose the split **minimizing** the criterion

**Variations:** split at all categories of a categorical variable using a clever category ordering (ID3), split at a restricted set of points (quantiles or fixed grid)

**Stopping rules:**
- when a leaf/region contains less than a prescribed number of observations
- when the region is sufficiently homogeneous...

May lead to a quite complex tree: over-fitting possible!

Additional pruning often use.
Pruning

- **Model selection** within the (rooted) subtrees of previous tree!
- Number of subtrees can be quite large, but the tree structure allows to find the best model efficiently.

### Key idea

- The predictor in a leaf depends only on the values in this leaf.
- **Efficient bottom-up (dynamic programming) algorithm** if the criterion used satisfies an additive property

\[
C(T) = \sum_{\mathcal{L} \in T} c(\mathcal{L})
\]

- Example: AIC / CV.
Examples of criterion satisfying this assumptions

- **AIC type criterion:**
  \[\sum_{i=1}^{n} \bar{\ell}(y_i, f_{\mathcal{L}}(x_i)) + \lambda |\mathcal{T}| = \sum_{\mathcal{L} \in \mathcal{T}} \left( \sum_{x_i \in \mathcal{L}} \bar{\ell}(y_i, f_{\mathcal{L}}(x_i)) + \lambda \right)\]

- **Simple cross-Validation (with \((x'_i, y'_i)\) a different dataset):**
  \[\sum_{i=1}^{n'} \bar{\ell}(y'_i, f_{\mathcal{L}}(x'_i)) = \sum_{\mathcal{L} \in \mathcal{T}} \left( \sum_{x'_i \in \mathcal{L}} \bar{\ell}(y'_i, f_{\mathcal{L}}(x'_i)) \right)\]

- Limit over-fitting for a single tree.

- **Rk:** almost never used when combining several trees...
Key observation: at a given node, the best subtree is either the current node or the union of the best subtrees of its child.

Dynamic programming algorithm

- Compute the individual cost $c(\mathcal{L})$ of each node (including the leaves)
- Scan all the nodes in reverse order of depth:
  - If the node $\mathcal{L}$ has no child, set its best subtree $T(\mathcal{L})$ to $\{\mathcal{L}\}$ and its current best cost $c'(\mathcal{L})$ to $c(\mathcal{L})$
  - If the children $\mathcal{L}_1$ and $\mathcal{L}_2$ are such that $c'(\mathcal{L}_1) + c'(\mathcal{L}_2) \geq c(\mathcal{L})$, then prune the child by setting $T(\mathcal{L}) = \{\mathcal{L}\}$ and $c'(\mathcal{L}) = c(\mathcal{L})$
  - Otherwise, set $T(\mathcal{L}) = T(\mathcal{L}_1) \cup T(\mathcal{L}_2)$ and $c'(\mathcal{L}) = c'(\mathcal{L}_1) + c'(\mathcal{L}_2)$
- The best subtree is the best subtree $T(\mathcal{R})$ of the root $\mathcal{R}$.

Optimization cost proportional to the number of nodes and not the number of subtrees!
Extensions

- **Local estimation** of the proportions or of the conditional mean.
- **Recursive Partitioning methods:**
  - Recursive construction of a partition
  - Use of simple local model on each part of the partition
- **Examples:**
  - CART, ID3, C4.5, C5
  - MARS (local linear regression models)
  - Piecewise polynomial model with a dyadic partition...
- **Book:** *Recursive Partitioning and Applications* by Zhang and Singer
CART

Decision region

Classes:
- Class 1 (Red)
- Class 2 (Blue)

Decision boundary

Classes:
- Class 1 (Red)
- Class 2 (Blue)
CART: Pros and Cons

Pros
- Leads to an easily interpretable model
- Fast computation of the prediction
- Easily deals with categorical features (and missing values)

Cons
- Greedy optimization
- Hard decision boundaries
- Lack of stability
Ensemble methods

- Lack of robustness for single trees.
- How to combine trees?

### Parallel construction

- Construct several trees from bootstrapped samples and average the responses *(Bagging)*
- Add more randomness in the tree construction *(Random Forests)*

### Sequential construction

- Construct a sequence of trees by reweighting sequentially the samples according to their difficulties *(AdaBoost)*
- Reinterpretation as a stagewise additive model *(Boosting)*
Ensemble methods

Bagging

Decision region

Decision boundary

classes

PredictorB

PredictorA

Class1
Class2

classes

PredictorB

PredictorA

Class1
Class2
Ensemble methods

Random Forest

Decision region

Decision boundary

classes

Class 1
Class 2

classes

Class 1
Class 2
Ensemble methods

AdaBoost

Decision region

Decision boundary
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
     - Ensemble Methods
   - References

6 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9 References
Independent Average

Stability through averaging

- Very simple idea to obtain a more stable estimator.
- **Vote/average** of $B$ predictors $f_1, \ldots, f_B$ obtained with independent datasets of size $n$!

$$f_{agr} = \text{sign} \left( \frac{1}{B} \sum_{b=1}^{B} f_b \right) \quad \text{or} \quad f_{agr} = \frac{1}{B} \sum_{i=1}^{B} f_b$$

- **Regression**: $\mathbb{E}[f_{agr}(x)] = \mathbb{E}[f_b(x)]$ and $\text{Var} \left[ f_{agr}(x) \right] = \frac{\text{Var}[f_b(x)]}{B}$
- **Prediction**: slightly more complex analysis
- Averaging leads to **variance reduction**, i.e. stability!

- **Issue**: cost of obtaining $B$ independent datasets of size $n$!
Bagging and Bootstrap


**Stability through bootstrapping**

- Instead of using $B$ independent datasets of size $n$, draw $B$ datasets from a single one using a **uniform with replacement** scheme (Bootstrap).
- **Rk:** On average, a fraction of $(1 - 1/e) \approx 0.63$ examples are unique among each drawn dataset...
- The $f_b$ are still identically distributed but **not independent** anymore.
- Price for the non independence: $\mathbb{E}[f_{agr}(x)] = \mathbb{E}[f_b(x)]$ and
  \[
  \mathbb{V}ar[f_{agr}(x)] = \mathbb{V}ar[f_b(x)] + \left(1 - \frac{1}{B}\right) \rho(x)
  \]
  with $\rho(x) = \text{Cov}[f_b(x), f_{b'}(x)] \leq \mathbb{V}ar[f_b(x)]$ with $b \neq b'$.
- **Bagging:** Bootstrap Aggregation

- Better aggregation scheme exists...
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   • Introduction
   • Machine Learning
   • Supervised Learning
   • Risk Estimation and Model Selection
   • Cross Validation and Test
   • References

2 ML Methods: Probabilistic Point of View
   • Motivation
   • Supervised Learning
   • A Probabilistic Point of View
   • Parametric Conditional Density Modeling
   • Non Parametric Conditional Density Modeling
   • Generative Modeling
   • Model Selection
   • Penalization

3 ML Methods: Optimization Point of View
   • Supervised Learning
   • Optimization Point of View
   • SVM
   • Penalization
   • Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   • Introduction
   • Gradient Descent
   • Proximal Descent
   • Coordinate Descent
   • Gradient Descent Acceleration
   • Stochastic Gradient Descent
   • Gradient Descent Step
   • Non-Convex Setting
   • References

5 ML Methods: Neural Networks and Deep Learning
   • Introduction
   • From Logistic Regression to NN
   • NN Optimization
   • NN Regularization
   • Image and CNN
   • Text, Recurrent Neural Networks and Transformers

6 ML Methods: Trees and Ensemble Methods
   • Trees
   • Bagging and Random Forests
   • Bootstrap and Bagging
   • Randomized Rules and Random Forests
   • Boosting
   • AdaBoost as a Greedy Scheme
   • Ensemble Methods
   • References

7 Unsupervised Learning: Dimension Reduction and Clustering
   • Unsupervised Learning?
   • A First Glimpse
   • Clustering
   • Dimensionality Curse
   • Dimension Reduction
   • Generative Modeling
   • Dimension Reduction
   • Simplification
   • Reconstruction Error
   • Relationship Preservation
   • Comparing Methods?
   • Clustering
   • Prototype Approaches
   • Contiguity Approaches
   • Agglomerative Approaches
   • Other Approaches
   • Scalability
   • Generative Modeling
   • (Plain) Parametric Density Estimation
   • Latent Variables
   • Approximate Simulation
   • Diffusion Model
   • Generative Adversarial Network
   • Applications to Text
   • References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   • Supervised Learning
   • Empirical Risk Minimization
   • Empirical Risk Minimization
   • ERM and PAC Analysis
   • Hoeffding and Finite Class
   • McDiarmid and Rademacher Complexity
   • VC Dimension
   • Structural Risk Minimization
   • References

9 References

376
Randomized Predictors

- Correlation leads to less variance reduction:
  \[
  \text{Var}[f_{\text{agr}}(x)] = \frac{\text{Var}[f_b(x)]}{B} + \left(1 - \frac{1}{B}\right) \rho(x)
  \]
  with \( \rho(x) = \text{Cov}[f_b(x), f_{b'}(x)] \) with \( b \neq b' \).

- **Idea**: Reduce the correlation by adding more randomness in the predictor.

Randomized Predictors

- Construct predictors that depend on a **randomness source** \( R \) that may be chosen independently for all bootstrap samples.

- This **reduces** the correlation between the estimates and thus the **variance**…

- But may **modify heavily the estimates** themselves!

- **Performance gain** not obvious from theory…
Random Forest

- Example of randomized predictors based on trees proposed by Breiman in 2001...

**Random Forest**

- Draw $B$ resampled datasets from a single one using a uniform with replacement scheme (Bootstrap)
- For each resampled dataset, construct a tree using a different randomly drawn subset of variables at each split.

- Most important parameter is the **subset size**:
  - if it is too large then we are back to bagging
  - if it is too small the mean of the predictors is probably not a good predictor...

- **Recommendation:**
  - Classification: use a proportion of $1/\sqrt{p}$
  - Regression: use a proportion of $1/3$

- **Sloppier stopping rules** and pruning than in CART...
Extra Trees

- Extremely randomized trees!

**Extra Trees**

- Variation of random forests.
- Instead of trying all possible cuts, try only $K$ cuts at random for each variable.
- No bootstrap in the original article.

- Cuts are defined by a threshold drawn uniformly in the feature range.
- Much faster than the original forest and similar performance.
- Theoretical performance analysis very challenging!
Error Estimate and Variable Ranking

Out Of the Box Estimate

- For each sample \( x_i \), a prediction can be made using only the resampled datasets not containing \( x_i \)...
- The corresponding empirical prediction error is **not prone to overfitting** but does not correspond to the final estimate...
- Good proxy nevertheless.

Forests and Variable Ranking

- **Importance**: Number of use or criterion gain at each split can be used to rank the variables.
- **Permutation tests**: Difference between OOB estimate using the true value of the \( j \)th feature and a value drawn a random from the list of possible values.

- Up to OOB error, the permutation technique is not specific to trees.
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

6 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9 References
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension

9. References
AdaBoost

- **Idea:** learn a sequence of predictors trained on weighted dataset with weights depending on the loss so far.

**Iterative scheme proposed by Schapire and Freund**

- Set $w_{1,i} = 1/n$; $t = 0$ and $f = 0$
- For $t = 1$ to $t = T$
  - $h_t = \arg\min_{h \in H} \sum_{i=1}^{n} w_{t,i} \ell(0.5)(y_i, h(x_i))$
  - Set $\epsilon_t = \sum_{i=1}^{n} w_{t,i} \ell(0.5)(y_i, h_t(x_i))$ and $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$
  - let $w_{t+1,i} = \frac{w_{t,i} e^{-\alpha_t y_i h_t(x_i)}}{Z_{t+1}}$ where $Z_{t+1}$ is a renormalization constant such that $\sum_{i=1}^{n} w_{t+1,i} = 1$
  - $f = f + \alpha_t h_t$
- Use $f = \sum_{i=1}^{T} \alpha_t h_t$

- **Intuition:** $w_{t,i}$ measures the difficulty of learning the sample $i$ at step $t$...
AdaBoost

AdaBoost Intuition

- $h_t$ obtained by minimizing a weighted loss

$$h_t = \arg\min_{h \in \mathcal{H}} \sum_{i=1}^{n} w_{t,i} \ell^0/1(y_i, h(x_i))$$

- Update the current estimate with

$$f_t = f_{t-1} + \alpha_t h_t$$
AdaBoost

AdaBoost Intuition

- Weight $w_{t,i}$ should be large if $x_i$ is not well-fitted at step $t - 1$ and small otherwise.
- Use a weight proportional to $e^{-y_if_{t-1}(x_i)}$ so that it can be recursively updated by

$$w_{t+1,i} = w_{t,i} \times \frac{e^{-\alpha ty_i h_t(x_i)}}{Z_t}$$
AdaBoost Intuition

- Set $\alpha_t$ such that

$$
\sum_{y_i h_t(x_i) = 1} w_{t+1,i} = \sum_{y_i h_t(x_i) = -1} w_{t+1,i}
$$

or equivalently

$$
\left( \sum_{y_i h_t(x_i) = 1} w_{t,i} \right) e^{-\alpha_t} = \left( \sum_{y_i h_t(x_i) = -1} w_{t,i} \right) e^{\alpha_t}
$$
AdaBoost

AdaBoost Intuition

Using

\[ \epsilon_t = \sum_{y_i h_t(x_i) = -1} w_{t,i} \]

leads to

\[ \alpha_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t} \quad \text{and} \quad Z_t = 2 \sqrt{\epsilon_t (1 - \epsilon_t)} \]
AdaBoost

Exponential Stagewise Additive Modeling

- Set $t = 0$ and $f = 0$.
- For $t = 1$ to $T$,
  - $(h_t, \alpha_t) = \arg\min_{h, \alpha} \sum_{i=1}^{n} e^{-y_i(f(x_i) + \alpha h(x_i))}$
  - $f = f + \alpha_t h_t$
- Use $f = \sum_{t=1}^{T} \alpha_t h_t$

- **Greedy optimization** of a classifier as a linear combination of $T$ classifier for the exponential loss.
- Those two algorithms are exactly the same!
Revisited AdaBoost

AdaBoost

- Set $t = 0$ and $f = 0$.
- For $t = 1$ to $T$,
  - $(h_t, \alpha_t) = \arg\min_{h, \alpha} \sum_{i=1}^{n} e^{-y_i(f(x_i)+\alpha h(x_i))}$
  - $f = f + \alpha_t h_t$
- Use $f = \sum_{t=1}^{T} \alpha_t h_t$

- **Greedy iterative scheme** with only two parameters: the class $\mathcal{H}$ of weak classifiers and the number of step $T$.
- In the literature, one can read that Adaboost does not overfit! This is not true and $T$ should be chosen with care...
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
   - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
   - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
     - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9 References
## Weak Learners

### Weak Learner
- Simple predictor belonging to a set $\mathcal{H}$.
- Easy to learn.
- Need to be only slightly better than a constant predictor.

### Weak Learner Examples
- **Decision Tree** with few splits.
- **Stump** decision tree with one split.
- **(Generalized) Linear Regression** with few variables.

### Boosting
- Sequential Linear Combination of Weak Learner
- Attempt to minimize a loss.
- Example of ensemble method.
- Link with Generalized Additive Modeling.
**Generic Boosting**

- **Greedy optim.** yielding a linear combination of weak learners.

**Generic Boosting**

**Algorithm:**
- Set \( t = 0 \) and \( f = 0 \).
- For \( t = 1 \) to \( T \),
  - \((h_t, \alpha_t) = \arg\min_{h, \alpha} \sum_{i=1}^{n} \tilde{\ell}(y_i, f(x_i) + \alpha h(x_i))\)
  - \( f = f + \alpha_t h_t \)
- Use \( f = \sum_{t=1}^{T} \alpha_t h_t \)

**AKA as Forward Stagewise Additive Modeling**
- AdaBoost with \( \tilde{\ell}(y, h) = e^{-yh} \)
- LogitBoost with \( \tilde{\ell}(y, h) = \log_2(1 + e^{-yh}) \)
- \( L_2 \)Boost with \( \tilde{\ell}(y, h) = (y - h)^2 \) (Matching pursuit)
- \( L_1 \)Boost with \( \tilde{\ell}(y, h) = |y - h| \)
- HuberBoost with \( \tilde{\ell}(y, h) = |y - h|^2 1_{|y - h| < \epsilon} + (2\epsilon |y - h| - \epsilon^2) 1_{|y - h| \geq \epsilon} \)

- Simple principle but **no easy numerical scheme** except for AdaBoost and \( L_2 \)Boost...
Gradient Boosting

- **Issue:** At each boosting step, one need to solve

  \[
  (h_t, \alpha_t) = \arg\min_{h, \alpha} \sum_{i=1}^{n} \tilde{\ell}(y_i, f(x_i) + \alpha h(x_i)) = L(y, f + \alpha h)
  \]

- **Idea:** Replace the function by a **first order approximation**

  \[
  L(y, f + \alpha h) \sim L(y, f) + \alpha \langle \nabla L(y, f), h \rangle
  \]

**Gradient Boosting**

- Replace the minimization step by a **gradient descent** step:
  - Choose \( h_t \) as the best possible descent direction in \( \mathcal{H} \) according to the approximation
  - Choose \( \alpha_t \) that minimizes \( L(y, f + \alpha h_t) \) (line search)

- **Rk:** Exact gradient direction often not possible!

- Need to find efficiently this best possible direction...
Best Direction

- Gradient direction:

\[ \nabla L(y, f) \quad \text{with} \quad \nabla_i L(y, f) = \frac{\partial}{\partial f(x_i)} \left( \sum_{i' = 1}^{n} \bar{\ell}(y_{i'}, f(x_{i'})) \right) \]

\[ = \frac{\partial}{\partial f(x_i)} \bar{\ell}(y_i, f(x_i)) \]

Best Direction within \( \mathcal{H} \)

- Direct formulation:

\[ h_t \in \arg\min_{h \in \mathcal{H}} \frac{\sum_{i=1}^{n} \nabla_i L(y, f) h(x_i)}{\sqrt{\sum_{i=1}^{n} |h(x_i)|^2}} \left( = \frac{\langle \nabla L(y, f), h \rangle}{\| h \|} \right) \]

- Equivalent (least-squares) formulation: \( h_t = -\beta_t h_t' \) with

\[ (\beta_t, h_t') \in \arg\min_{(\beta, h) \in \mathbb{R} \times \mathcal{H}} \sum_{i=1}^{n} \left| \nabla_i L(y, f) - \beta h(x_i) \right|^2 \left( = \| \nabla L - \beta h \|^2 \right) \]

- Choice of the formulation will depend on \( \mathcal{H} \)...
Gradient Boosting of Classifiers

- **Assumptions:**
  - $h$ is a binary classifier, $h(x) = \pm 1$ and thus $\|h\|^2 = n$.
  - $\ell(y, f(x)) = l(yf(x))$ so that $\nabla_i L(y, f) = y_i l'(y_i f(x_i))$.

- Best direction $h_t$ in $\mathcal{H}$ using the first formulation
  
  $$h_t = \arg\min_{h \in \mathcal{H}} \sum_i \nabla_i L(y, f) h(x_i)$$

**AdaBoost Type Minimization**

- Best direction rewriting
  
  $$h_t = \arg\min_{h \in \mathcal{H}} \sum_i l'(y_i f(x_i)) y_i h(x_i)$$

  $$= \arg\min_{h \in \mathcal{H}} \sum_i (-l')(y_i f(x_i))(2\ell^{0/1}(y_i, h(x_i)) - 1)$$

- **AdaBoost type weighted loss minimization** as soon as $(-l')(y_i f(x_i)) \geq 0$:
  
  $$h_t = \arg\min \sum_i (-l')(y_i f(x_i))\ell^{0/1}(y_i, h(x_i))$$
Gradient Boosting of Classifiers

**Gradient Boosting**

- **(Gradient) AdaBoost:** \( \ell(y, f) = \exp(-yf) \)
  - \( l(x) = \exp(-x) \) and thus \((-l')(y_if(x_i)) = e^{-y_if(x_i)} \geq 0\)
  - \( h_t \) is the same as in AdaBoost
  - \( \alpha_t \) also... (explicit computation)

- **LogitBoost:** \( \ell(y, f) = \log_2(1 + e^{-yf}) \)
  - \( l(x) = \log_2(1 + e^{-x}) \) and thus \((-l')(y_if(x_i)) = \frac{e^{-y_if(x_i)}}{\log(2)(1+e^{-y_if(x_i)})} \geq 0\)
  - Less weight on misclassified samples than in AdaBoost...
  - No explicit formula for \( \alpha_t \) (line search)
  - Different path than with the (non-computable) classical boosting!

- **SoftBoost:** \( \ell(y, f) = \max(1 - yf, 0) \)
  - \( l(x) = \max(1 - x, 0) \) and \((-l')(y_if(x_i)) = 1_{y_if(x_i) \leq 1} \geq 0\)
  - Do not use the samples that are sufficiently well classified!
Gradient Boosting and Least Squares

- Least squares formulation is often preferred when $|h| \neq 1$.

**Least Squares Gradient Boosting**

- Find $h_t = -\beta_t h'_t$ with

$$
(\beta_t, h'_t) \in \arg\min_{(\beta, h) \in \mathbb{R} \times \mathcal{H}} \sum_{i=1}^{n} |\nabla_i L(y, f) - \beta h(x_i)|^2
$$

- Classical least squares if $\mathcal{H}$ is a finite dimensional vector space!
- Not a usual least squares in general but a classical regression problem!

- Numerical scheme depends on the loss...
Gradient Boosting and Least Squares

Examples

- **Gradient $L_2$Boost:**
  - $\ell(y, f) = |y - f|^2$ and $\nabla_i L(y_i, f(x_i)) = -2(y_i - f(x_i))$:
    $$(\beta_t, h'_t) \in \arg\min_{(\beta, h) \in \mathbb{R} \times \mathcal{H}} \sum_{i=1}^{n} |2y_i - 2(f(x_i) - \beta/2h(x_i))|^2$$
  - $\alpha_t = -\beta_t/2$
  - Equivalent to classical $L_2$-Boosting

- **Gradient $L_1$Boost:**
  - $\ell(y, f) = |y - f|$ and $\nabla_i L(y_i, f(x_i)) = -\text{sign}(y_i - f(x_i))$:
    $$(\beta_t, h'_t) \in \arg\min_{(\beta, h) \in \mathbb{R} \times \mathcal{H}} \sum_{i=1}^{n} |-\text{sign}(y_i - f(x_i)) - \beta h(x_i)|^2$$
  - Robust to outliers...

- Classical choice for $\mathcal{H}$: Linear Model in which each $h$ depends on a small subset of variables.
Gradient Boosting and Least Squares

- Least squares formulation can also be used in classification!
- Assumption:
  - \( \ell(y, f(x)) = l(yf(x)) \) so that \( \nabla_i L(y_i, f(x_i)) = y_i l'(y_i f(x_i)) \)

**Least Squares Gradient Boosting for Classifiers**

- Least Squares formulation:
  \[
  (\beta_t, h'_t) \in \arg\min_{(\beta, h) \in \mathbb{R} \times \mathcal{H}} \sum_{i=1}^{n} |y_i l'(y_i f(x_i)) - \beta h(x_i)|^2
  \]

- Equivalent formulation:
  \[
  (\beta_t, h'_t) \in \arg\min_{(\beta, h) \in \mathbb{R} \times \mathcal{H}} \sum_{i=1}^{n} |(-l')(y_i f(x_i)) - (-\beta) y_i h(x_i)|^2
  \]

- **Intuition**: Modify misclassified examples without modifying too much the well-classified ones...
Boosting Variations

Stochastic Boosting

- **Idea:** change the learning set at each step.
- Two possible reasons:
  - Optimization over all examples too costly
  - Add variability to use an averaged solution
- Two different samplings:
  - Use sub-sampling, if you need to reduce the complexity
  - Use re-sampling, if you add variability...
- Stochastic Gradient name mainly used for the first case...

Second Order Boosting

- Replace the first order approximation by a second order one and avoid the line search...
XGBoost

- Very efficient boosting algorithm proposed by Chen and Guestrin in 2014.

**eXtreme Gradient Boosting**

- Gradient boosting for a (penalized) smooth loss using a second order approximation and the least squares approximation.
- Reduced stepsize with a shrinkage of the *optimal* parameter.
- Feature subsampling.
- Weak learners:
  - Trees: limited depth, penalized size and parameters, fast approximate best split.
  - Linear model: elastic-net penalization.

- Excellent baseline!
- Lightgbm and CatBoost are also excellent similar choices!
**Outline**

1. **Statistical Learning: Introduction, Setting and Risk Estimation**
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. **ML Methods: Probabilistic Point of View**
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. **ML Methods: Optimization Point of View**
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. **Optimization: Gradient Descent Algorithms**
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5. **ML Methods: Trees and Ensemble Methods**
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - **Ensemble Methods**
     - References

6. **ML Methods: Neural Networks and Deep Learning**
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

7. **Unsupervised Learning: Dimension Reduction and Clustering**
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8. **Statistical Learning: PAC-Bayesian Approach and Complexity Theory**
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. **References**
Ensemble Methods

- **Averaging**: combine several models by averaging (bagging, random forests, ...)  
- **Boosting**: construct a sequence of (weak) classifiers (XGBoost, LightGBM, CatBoost)  
- **Stacking**: use the outputs of several models as features (tpot, ...)  

- Loss of interpretability but gain in performance  
- Beware of overfitting with stacking: the second learning step should be done with fresh data.  
- No end to end optimization as in deep learning!
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
References

T. Hastie, R. Tibshirani, and J. Friedman.  
*The Elements of Statistical Learning.*  
Springer Series in Statistics, 2009

M. Mohri, A. Rostamizadeh, and A. Talwalkar.  
*Foundations of Machine Learning (2nd ed.)*  
MIT Press, 2018

A. Géron.  
*Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (3rd ed.)*  
O’Reilly, 2022

Ch. Giraud.  
*Introduction to High-Dimensional Statistics.*  
CRC Press, 2014

S. Bubeck.  
*Convex Optimization: Algorithms and Complexity.*  
Now Publisher, 2015


A. Géron. *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (2nd ed.)* O’Reilly, 2019
Creative Commons Attribution-ShareAlike (CC BY-SA 4.0)

- You are free to:
  - **Share**: copy and redistribute the material in any medium or format
  - **Adapt**: remix, transform, and build upon the material for any purpose, even commercially.
- Under the following terms:
  - **Attribution**: You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - **ShareAlike**: If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - **No additional restrictions**: You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Contributors

- **Main contributor**: E. Le Pennec
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   • Introduction
   • Machine Learning
   • Supervised Learning
   • Risk Estimation and Model Selection
   • Cross Validation and Test
   • References

2. ML Methods: Probabilistic Point of View
   • Motivation
   • Supervised Learning
   • A Probabilistic Point of View
   • Parametric Conditional Density Modeling
   • Non Parametric Conditional Density Modeling
   • Generative Modeling
   • Model Selection
   • Penalization

3. ML Methods: Optimization Point of View
   • Supervised Learning
   • Optimization Point of View
   • SVM
   • Penalization
   • Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   • Introduction
   • Gradient Descent
   • Proximal Descent
   • Coordinate Descent
   • Gradient Descent Acceleration
   • Stochastic Gradient Descent
   • Gradient Descent Step
   • Non-Convex Setting
   • References

5. ML Methods: Neural Networks and Deep Learning
   • Introduction
   • From Logistic Regression to NN
   • NN Optimization
   • NN Regularization
   • Image and CNN
   • Text, Recurrent Neural Networks and Transformers
   • NN Architecture
   • References

6. ML Methods: Trees and Ensemble Methods
   • Trees
   • Bagging and Random Forests
   • Bootstrap and Bagging
   • Randomized Rules and Random Forests
   • Boosting
   • AdaBoost as a Greedy Scheme
   • Boosting
   • Ensemble Methods
   • References

7. Unsupervised Learning: Dimension Reduction and Clustering
   • Unsupervised Learning?
   • A First Glimpse
   • Clustering
   • Dimensionality Curse
   • Dimension Reduction
   • Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   • Statistical Learning
   • PAC-Bayesian Approach
   • Complexity Theory
   • Supervised Learning
   • Empirical Risk Minimization
   • Empirical Risk Minimization
   • ERM and PAC Analysis
   • Hoeffding and Finite Class
   • McDiarmid and Rademacher Complexity
   • VC Dimension
   • Structural Risk Minimization

9. References
Unsupervised Learning:
Dimension Reduction and Clustering

Unsupervised Learning?
A First Glimpse
Clustering
Dimensionality Curse
Dimension Reduction
Generative Modeling

Dimension Reduction
Simplification
Reconstruction Error
Relationship Preservation
Comparing Methods?

Clustering
Prototype Approaches
Contiguity Approaches
Agglomerative Approaches
Other Approaches
Scalability

Generative Modeling
(Plain) Parametric Density Estimation
Latent Variables
Approximate Simulation
Diffusion Model
Generative Adversarial Network

Applications to Text
References

Statistical Learning: PAC-Bayesian Approach and Complexity Theory
Supervised Learning
Empirical Risk Minimization
Empirical Risk Minimization
ERM and PAC Analysis
Hoeffding and Finite Class
McDiarmid and Rademacher Complexity
VC Dimension
Structural Risk Minimization

References
Motivation

What is possible with data without labels?

- To group them?
- To visualize them in a 2 dimensional space?
- To generate more data?
To group them?

- **Data:** Base of customer data containing their properties and past buying records
- **Goal:** Use the customers *similarities* to find groups.
- **Clustering:** propose an explicit *grouping* of the customers
- **Visualization:** propose a representation of the customers so that the groups are *visible*. (Bonus)
To visualize them?

- **Data**: Images of a single object
- **Goal**: Visualize the *similarities* between images.
- **Visualization**: propose a representation of the images so that similar images are *close*.
- **Clustering**: use this representation to cluster the images. (Bonus)
To generate more data?

- **Data:** Images.
- **Goal:** Generate images similar to the ones in the dataset.
- **Generative Modeling:** propose (and train) a generator.
A definition by Tom Mitchell

A computer program is said to learn from **experience E** with respect to some **class of tasks T** and **performance measure P**, if its performance at tasks in T, as measured by P, improves with experience E.
Supervised Learning

Experience, Task and Performance measure

- **Training data**: \( \mathcal{D} = \{(X_1, Y_1), \ldots, (X_n, Y_n)\} \) (i.i.d. \( \sim \mathbb{P} \))
- **Predictor**: \( f: \mathcal{X} \rightarrow \mathcal{Y} \) measurable
- **Cost/Loss function**: \( \ell(f(X), Y) \) measure how well \( f(X) \) predicts \( Y \)
- **Risk**: 
  \[
  R(f) = \mathbb{E}[\ell(Y, f(X))] = \mathbb{E}_X[\mathbb{E}_{Y|X}[\ell(Y, f(X))]]
  \]

  Often \( \ell(f(X), Y) = \|f(X) - Y\|^2 \) or \( \ell(f(X), Y) = 1_{Y \neq f(X)} \)

Goal

- Learn a rule to construct a **predictor** \( \hat{f} \in \mathcal{F} \) from the training data \( \mathcal{D}_n \) s.t. **the risk** \( R(\hat{f}) \) is **small on average** or with high probability with respect to \( \mathcal{D}_n \).
### Experience, Task and Performance measure

- **Training data**: \( D = \{X_1, \ldots, X_n\} \) (i.i.d. \( \sim \mathbb{P} \))
- **Task**: ???
- **Performance measure**: ???

No obvious task definition!

### Tasks for this lecture

- **Dimension reduction**: construct a map of the data in a low dimensional space without distorting it too much.
- **Clustering (or unsupervised classification)**: construct a grouping of the data in homogeneous classes.
Unsupervised Learning: Dimension Reduction and Clustering

Dimension Reduction

- **Training data**: \( \mathcal{D} = \{X_1, \ldots, X_n\} \in \mathcal{X}^n \) (i.i.d. \( \sim \) \( \mathbb{P} \))
- Space \( \mathcal{X} \) of possibly high dimension.

**Dimension Reduction Map**

- Construct a map \( \Phi \) from the space \( \mathcal{X} \) into a space \( \mathcal{X}' \) of smaller dimension:

\[
\Phi : \mathcal{X} \rightarrow \mathcal{X}'
\]

\[
X \mapsto \Phi(X)
\]

- Map can be defined only on the dataset.

**Motivations**

- Visualization of the data
- Dimension reduction (or embedding) before further processing
Dimension Reduction

- Need to control the **distortion** between \( \mathcal{D} \) and \( \Phi(\mathcal{D}) = \{\Phi(X_1), \ldots, \Phi(X_n)\} \)

**Distortion(s)**

- **Reconstruction error:**
  - Construct \( \tilde{\Phi} \) from \( \mathcal{X}' \) to \( \mathcal{X} \)
  - Control the error between \( \mathcal{X} \) and its reconstruction \( \tilde{\Phi}(\Phi(\mathcal{X})) \)

- **Relationship preservation:**
  - Compute a *relation* \( X_i \) and \( X_j \) and a *relation* between \( \Phi(X_i) \) and \( \Phi(X_j) \)
  - Control the difference between those two *relations*.

- Lead to different constructions....
Clustering

- **Training data**: $\mathcal{D} = \{X_1, \ldots, X_n\} \in \mathcal{X}^n$ (i.i.d. $\sim \mathbb{P}$)
- Latent groups?

Clustering

- Construct a map $f$ from $\mathcal{D}$ to $\{1, \ldots, K\}$ where $K$ is a number of classes to be fixed:

  $$f : X_i \mapsto k_i$$

- Similar to classification except:
  - no ground truth (no given labels)
  - label only elements of the dataset!

Motivations

- Interpretation of the groups
- Use of the groups in further processing
Clustering

- Need to define the **quality** of the cluster.
- No obvious measure!

### Clustering quality

- Inner homogeneity: samples in the same group should be similar.
- Outer inhomogeneity: samples in two different groups should be different.

- Several possible definitions of similar and different.
- Often based on the distance between the samples.
- Example based on the Euclidean distance:
  - Inner homogeneity $= \text{intra-class variance}$,
  - Outer inhomogeneity $= \text{inter-class variance}$.

**Beware:** choice of the number of clusters $K$ often complex!
Generative Modeling

- **Training data**: \( D = \{(X_1, Y_1), \ldots, (X_n, Y_n)\} \in (X \times Y)^n \) (i.i.d. \( \sim P \))
- Same kind of data than for supervised learning if \( Y \neq \emptyset \).

<table>
<thead>
<tr>
<th>Generative Modeling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Construct a map ( G ) from the product of ( Y ) and a randomness source ( \Omega ) to ( X )</td>
</tr>
<tr>
<td>( G : Y \times \Omega \rightarrow X )</td>
</tr>
<tr>
<td>( (Y, \omega) \mapsto X )</td>
</tr>
</tbody>
</table>

- Unconditional model if \( Y = \emptyset \)...

**Motivation**
- Generate plausible novel conditional samples based on a given dataset.

**Sample Quality**
- Related to the proximity between the law of \( G(Y, \omega) \) and the law of \( X|Y \).
- Most classical choice is the Kullback-Leibler divergence.
Generative Modeling

Ingredients

- Generator $F_\theta(Y, \omega)$ and cond. density prob. $p_\theta(X|Y)$ (Explicit vs implicit link)
- Simple / Complex / Approximate estimation...

Some Possible Choices

<table>
<thead>
<tr>
<th>Probabilistic model</th>
<th>Generator</th>
<th>Estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>Simple (parametric)</td>
<td>Explicit</td>
</tr>
<tr>
<td>Flow</td>
<td>Image of simple model</td>
<td>Explicit</td>
</tr>
<tr>
<td>Factorization</td>
<td>Factorization of simple model</td>
<td>Explicit</td>
</tr>
<tr>
<td>VAE</td>
<td>Simple model with latent var.</td>
<td>Explicit</td>
</tr>
<tr>
<td>EBM</td>
<td>Arbitrary</td>
<td>Implicit (MCMC)</td>
</tr>
<tr>
<td>Diffusion</td>
<td>Continuous noise</td>
<td>Implicit (MCMC)</td>
</tr>
<tr>
<td>GAN</td>
<td>Discrete Noise with latent var.</td>
<td>Explicit</td>
</tr>
<tr>
<td></td>
<td>Implicit</td>
<td>Explicit</td>
</tr>
</tbody>
</table>

- SOTA: Diffusion based approach!

Bonus Task: Representation Learning

- **General observation:** most data do not have a label!
- **Example:** The number of images on which someone has described the content of the image is a tiny fraction of the images online.
- Labeling is very expensive and time consuming
- A lot of information can be extracted from the structure of the data, before seeing any label.

How can we leverage the large quantity of un-labeled data?

- Learn relevant features ( = representations) in an unsupervised fashion
- Use those features to solve a supervised task with a fraction of labeled data.

- **Semi-supervised framework**
- ↔ Very useful in practice, for images, time series, text.
Semi-supervised Framework

- With representation learned in an unsupervised fashion + a simple linear model, one can achieve the same performance with 10% of data labeled than with a fully annotated dataset.

- Complementary regularization based approaches also exist.
Unsupervised Learning is a Versatile Approach!

Except for generative modeling, the learner is always right

- A subjective measure of performance
- Subjective choices for the algorithmic constraints (e.g., the type of transformation of the data we allow for low-dimensional representation, type of groups in clustering)
- \(\Rightarrow\) Very difficult or impossible to tell which is the best method.

Yet:
- Extremely important in practice:
  - 90-99% of the data is un-labeled!
  - the tasks themselves are fundamental
- Huge success in various fields (NLP, images...)

Unsupervised Learning: Dimension Reduction and Clustering
Unsupervised Learning is a Versatile Approach!

**Lecture goals** for the three main tasks

- Discussing possible choices of measures of performance and algorithmic constraints
- Understand the correspondences between those choices and a variety of classical algorithms
- For the simplest algorithms (PCA, k-means), get a precise mathematical understanding of the learning process.
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9. References
What’s a group?

- No simple or unanimous definition!
- Require a notion of similarity/difference...

**Three main approaches**

- A group is a set of samples similar to a prototype.
- A group is a set of samples that can be linked by contiguity.
- A group can be obtained by fusing some smaller groups...
Prototype Approach

- A group is a set of samples similar to a prototype.
- Most classical instance: $k$-means algorithm.
- Principle: alternate prototype choice for the current groups and group update based on those prototypes.

- Number of groups fixed at the beginning
- No need to compare the samples between them!
Contiguity Approach

- A group is the set of samples that can be linked by contiguity.
- Most classical instance: DBScan
- Principle: group samples by contiguity if possible (proximity and density)

- Some samples may remain isolated.
- Number of groups controlled by the scale parameter.

DBSCAN: Density-Based Spatial Clustering of Applications with Noise
Agglomerative Approach

- A group can be obtained by fusing some smaller groups...
- Hierarchical clustering principle: sequential merging of groups according to a best merge criterion

- Numerous variations on the merging criterion...
- Number of groups chosen afterward.
Choice of the method and of the number of groups

No method or number of groups is better than the others...

- Criterion not necessarily explicit!
- No cross validation possible
- Choice of the number of groups: a priori, heuristic, based on the final usage...
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
     - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9 References

431
DISCLAIMER: Even if they are used everywhere beware of the usual distances in high dimension!

Dimensionality Curse

- Previous approaches based on distances.
- Surprising behavior in high dimension: everything is ((often) as) far away.
- Beware of categories...
Dimensionality Curse

- DISCLAIMER: Even if they are used everywhere beware of the usual distances in high dimension!

High Dimensional Geometry Curse

- Folks theorem: In high dimension, everyone is alone.
- Theorem: If $X_1, \ldots, X_n$ in the hypercube of dimension $d$ such that their coordinates are i.i.d then

$$d^{-1/p} \left( \max \|X_i - X_j\|_p - \min \|X_i - X_j\|_p \right) = 0 + O_p \left( \sqrt{\frac{\log n}{d}} \right)$$

$$\frac{\min \|X_i - X_j\|_p}{\max \|X_i - X_j\|_p} = 1 + O_p \left( \sqrt{\frac{\log n}{d}} \right).$$

- When $d$ is large, all the points are almost equidistant...
- Nearest neighbors are meaningless!
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Optimization Point of View
   - SVM
   - Penalties
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
     - Empirical Risk Minimization
   - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9 References

434
Visualization and Dimension Reduction

- How to view a dataset in high dimension!
- High dimension: dimension larger than 2!
- **Projection** onto a 2D space.
How to view a dataset in high dimension!

High dimension: dimension larger than 2!

Projection onto a 2D space.
Visualization and Dimension Reduction

- How to view a dataset in high dimension!
- High dimension: dimension larger than 2!
- **Projection** onto a 2D space.
Visualization and Dimension Reduction

- How to view a dataset in high dimension!
- High dimension: dimension larger than 2!
- **Projection** onto a 2D space.
Principal Component Analysis

Simple formula: \( \tilde{X} = P(X - m) \)

How to chose \( P \)?

- Maximising the dispersion of the points?
- Allowing to well reconstruct \( X \) from \( \tilde{X} \)?
- Preserving the relationship between the \( X \) through those between the \( \tilde{X} \)?
Principal Component Analysis

Simple formula: $\tilde{X} = P(X - m)$

How to choose $P$?

- Maximising the dispersion of the points?
- Allowing to well reconstruct $X$ from $\tilde{X}$?
- Preserving the relationship between the $X$ through those between the $\tilde{X}$?

- The 3 approaches yield the same solution!
Reconstruction Approaches

- Learn a formula to encode and one formula to decode.
- Auto-encoder structure

- Yields a formula for new points.
**Reconstruction Approaches**

- Learn a formula to encode and one formula to decode.
- Auto-encoder structure

Yields a formula for new points.
Reconstruction Approaches

- Learn a formula to encode and one formula to decode.
- Auto-encoder structure

Yields a formula for new points.
Relationship Preservation Approaches

- Based on the definition of the relationship notion (in both worlds).
- Huge flexibility

- Not always yields a formula for new points.
Choices of Methods and Dimension

No Better Choice?

- Different criterion for different methods: impossible to use cross-validation.
- The larger the dimension the easier is to be faithful!
- In visualization, dimension 2 is the only choice.
- Heuristic criterion for the dimension choice: elbow criterion (no more gain), stability... 

- Dimension Reduction is rarely used standalone but rather as a step in a predictive/prescriptive method.
- The dimension becomes an hyper-parameter of this method.
Representation Learning

- How to transform arbitrary objects into numerical vectors?
- Objects: Categorical variables, Words, Images/Sounds...

- The two previous dimension reduction approaches can be used (given possibly a first simple high dimensional representation)
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
     - A First Glimpse
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
   - Dimensionality Curse
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
   - Comparing Methods?
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9. References
Generative Modeling

- Generate new samples similar to the ones in an original dataset.
- Generation may be conditioned by a input.
- Key for image generation...and chatbot!
Heuristic: If we can estimate the (conditional) law $\mathbb{P}$ of the data and can simulate it, we can obtain new samples similar to the input ones.

**Estimation and Simulation**

- How to estimate the density?
- How to simulate the estimate density?
- Other possibilities?
Simple Estimation and Simple Simulation

Parametric Model, Image and Factorization

- **Use**
  - a simple parametric model, . . .
  - or the image of a parametric model (flow), . . .
  - or a factorization of a parametric model (recurrent model)

as they are *simple* to estimate and to simulate.

- Estimation by Maximum Likelihood principle.
- Recurrent models are used in Large Language Models!
Unsupervised Learning: Dimension Reduction and Clustering

Complex Estimation and Simple Simulation

Introduce a latent variable $Z$ from which $X$ is easy to sample.

Estimation based on approximate Maximum Likelihood (VAE/ELBO)

The latent variable $Z$ often lives in a smaller dimensional space.

It can be generated by a simple method (or a more complex one...).
Complex Estimation and Complex Simulation

Monte Carlo Markov Chain

- Rely on much more complex probability model...
- which can only be simulated numerically.
- Often combined with noise injection to stabilizes the numerical scheme (Diffusion).

- Much more expensive to simulate than with Latent Variable approaches.
Generative Adversarial Network

- Bypass the density estimation problem, by transforming the problem into a competition between the generator and a discriminator.
- The better the generator, the harder it is for the generator to distinguish true samples from synthetic ones.
- No explicit density!

- Fast simulator but unstable training...
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5. Dimension Reduction
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?

6. Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability

7. Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network

8. Applications to Text
   - References

   - Supervised Learning
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization

10. Unsupervised Learning: Dimension Reduction and Clustering
    - Unsupervised Learning?
    - A First Glimpse
      - Clustering
      - Dimensionality Curse
      - Dimension Reduction
      - Generative Modeling

References
Dimension Reduction

- **Training data**: $\mathcal{D} = \{X_1, \ldots, X_n\} \in \mathcal{X}^n$ (i.i.d. $\sim \mathbb{P}$)
- Space $\mathcal{X}$ of possibly high dimension.

Dimension Reduction Map

- Construct a map $\Phi$ from the space $\mathcal{X}$ into a space $\mathcal{X}'$ of **smaller dimension**:
  $\Phi : \mathcal{X} \rightarrow \mathcal{X}'$
  $X \mapsto \Phi(X)$

Criterion

- Reconstruction error
- Relationship preservation
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

References
### How to Simplify?

**A Projection Based Approach**

- **Observations:** $X_1, \ldots, X_n \in \mathbb{R}^d$
- **Simplified version:** $\Phi(X_1), \ldots, \Phi(X_n) \in \mathbb{R}^{d'}$ with $\Phi$ an affine projection preserving the mean $\Phi(X) = P(X - m) + m$ with $P^\top = P = P^2$ and $m = \frac{1}{n} \sum_i X_i$.

### How to choose $P$?

- **Inertia criterion:**
  $$\max_P \sum_{i,j} \|\Phi(X_i) - \Phi(X_j)\|^2?$$

- **Reconstruction criterion:**
  $$\min_P \sum_i \|X_i - \Phi(X_i)\|^2?$$

- **Relationship criterion:**
  $$\min_P \sum_{i,j} |(X_i - m)^\top (X_j - m) - (\Phi(X_i) - m)^\top (\Phi(X_j) - m)|^2?$$

- **Rk:** Best solution is $P = I$! Need to reduce the rank of the projection to $d' < d$...
**Inertia criterion**

- **Heuristic:** a good representation is such that the projected points are far apart.

### Two views on inertia

- **Inertia:**

  \[
  I = \frac{1}{2n^2} \sum_{i,j} \|X_i - X_j\|^2 = \frac{1}{n} \sum_{i=1}^{n} \|X_i - m\|^2
  \]

  2 times the mean squared distance to the mean = Mean squared distance between individual

### Inertia criterion (Principal Component Analysis)

- **Criterion:**

  \[
  \max_P \sum_{i,j} \frac{1}{2n^2} \|PX_i - PX_j\|^2 = \max_P \frac{1}{n} \sum_i \|PX_i - m\|^2
  \]

- **Solution:** Choose \( P \) as a projection matrix on the space spanned by the \( d' \) first eigenvectors of \( \Sigma = \frac{1}{n} \sum_i (X_i - m)(X_i - m)^\top \)
First Component of the PCA

\[ \tilde{X} = m + a^\top (X - m) a \text{ with } \|a\| = 1 \]

- Inertia: \( \frac{1}{n} \sum_{i=1}^{n} a^\top (X_i - m)(X_i - m)^\top a \)

**Principal Component Analysis: optimization of the projection**

- Maximization of \( \tilde{l} = \frac{1}{n} \sum_{i=1}^{n} a^\top (X_i - m)(X_i - m)^\top a = a^\top \Sigma a \) with

\[ \Sigma = \frac{1}{n} \sum_{i=1}^{n} (X_i - m)(X_i - m)^\top \text{ the empirical covariance matrix}. \]

- Explicit optimal choice given by the eigenvector of the largest eigenvalue of \( \Sigma \).
Principal Component Analysis: sequential optimization of the projection

- Explicit optimal solution obtained by the projection on the eigenvectors of the largest eigenvalues of $\Sigma$.
- Projected inertia given by the sum of those eigenvalues.
- Often fast decay of the eigenvalues: some dimensions are much more important than others.
- Not exactly the curse of dimensionality setting...
- Yet a lot of small dimension can drive the distance!
Reconstruction Criterion

- **Heuristic:** A good representation is such that the projected points are close to the original ones.

Reconstruction Criterion

- Criterion: \( \min_P \sum_i \frac{1}{n} \|X_i - (P(X_i - m) + m)\|^2 = \min_P \frac{1}{n} \sum_i \|(I - P)(X_i - m)\|^2 \)

- **Solution:** Choose \( P \) as a projection matrix on the space spanned by the \( d' \) first eigenvectors of \( \Sigma = \frac{1}{n} \sum_i (X_i - m)(X_i - m)^\top \)

- Same solution with a different heuristic!

- Proof (Pythagora):
  \[
  \sum_i \|X_i - m\|^2 = \sum_i \left(\|P(X_i - m)\|^2 + \|(I - P)(X_i - m)\|^2\right)
  \]
PCA, Reconstruction and Distances

Close projection doesn’t mean close individuals!

- Same projections but different situations.
- Quality of the reconstruction measured by the angle with the projection space!
Relationship Criterion

- **Heuristic:** a good representation is such that the projected points scalar products are similar to the original ones.

**Relationship Criterion (Multi Dimensional Scaling)**

- **Criterion:** \( \min_P \sum_{i,j} |(X_i - m)^\top (X_j - m) - (\Phi(X_i) - m)^\top (\Phi(X_j) - m)|^2 \)

- **Solution:** Choose \( P \) as a projection matrix on the space spanned by the \( d' \) first eigenvectors of \( \Sigma = \frac{1}{n} \sum_i (X_i - m)(X_i - m)^\top \)

- Same solution with a different heuristic!
- Much more involved justification!
Unsupervised Learning: Dimension Reduction and Clustering

**Link with SVD**

- **PCA model:** $X - m \simeq P(X - m)$
- **Prop:** $P = V V^T$ with $V$ an orthormal family in dimension $d$ of size $d'$.
- **PCA model with $V$:** $X - m \simeq V V^T (X - m)$ where $\tilde{X} = V^T (X - m) \in \mathbb{R}^{d'}$
- **Row vector rewriting:** $X^T - m^T \simeq \tilde{X}^T V^T$

**Matrix Rewriting and Low Rank Factorization**

- **Matrix rewriting**

$$
\begin{pmatrix}
X_1^T - m^T \\
\vdots \\
X_n^T - m^T
\end{pmatrix}
\sim
\begin{pmatrix}
\tilde{X}_1^T \\
\vdots \\
\tilde{X}_n^T
\end{pmatrix}
\sim
V^T
$$

- **Low rank matrix factorization! (Truncated SVD solution...)**
Any matrix $n \times d$ matrix $A$ can be decomposed as

$$
A = U D W^T
$$

with $U$ and $W$ two orthonormal matrices and $D$ a \textit{diagonal} matrix with decreasing values.
Unsupervised Learning: Dimension Reduction and Clustering

SVD

Low Rank Approximation

- The best low rank approximation or rank $r$ is obtained by restriction of the matrices to the first $r$ dimensions:

$$
\begin{pmatrix}
A \\
(n \times d)
\end{pmatrix} \simeq
\begin{pmatrix}
U_r \\
(n \times r)
\end{pmatrix}
\begin{pmatrix}
D_{r,r} \\
(r \times r)
\end{pmatrix}
\begin{pmatrix}
W_r^T \\
(r \times d)
\end{pmatrix}
$$

for both the operator norm and the Frobenius norm!

- PCA: Low rank approximation with Frobenius norm, $d' = r$ and

$$
\begin{pmatrix}
X_1^T - m^T \\
\vdots \\
X_n^T - m^T
\end{pmatrix} \leftrightarrow A,
\begin{pmatrix}
\tilde{X}_1^T \\
\vdots \\
\tilde{X}_n^T
\end{pmatrix} \leftrightarrow U_r D_{r,r},
V^T \leftrightarrow W_r^T
$$
### SVD Decompositions

- **Recentered data:**

  \[
  R = \begin{pmatrix}
  X_1^T - m^T \\
  \vdots \\
  X_n^T - m^T
  \end{pmatrix} = UDW^T
  \]

- **Covariance matrix:**

  \[
  \Sigma = R^T R = WD^T DW
  \]
  with \( D^T D \) diagonal.

- **Gram matrix (matrix of scalar products):**

  \[
  G = RR^T = UDD^T U
  \]
  with \( DD^T \) diagonal.

- Those are the same \( U, W \) and \( D \), hence the link between all the approaches.
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Relationship Preservation
   - Comparing Methods?
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

9 References
Reconstruction Error Approach

Goal

- Construct a map \( \Phi \) from the space \( \mathcal{X} \) into a space \( \mathcal{X}' \) of smaller dimension:
  \[
  \Phi : \mathcal{X} \rightarrow \mathcal{X}'
  \]
  \[
  \mathcal{X} \mapsto \Phi(\mathcal{X})
  \]

- Construct \( \tilde{\Phi} \) from \( \mathcal{X}' \) to \( \mathcal{X} \)

- Control the error between \( \mathcal{X} \) and its reconstruction \( \tilde{\Phi}(\Phi(\mathcal{X})) \)

- Canonical example for \( \mathcal{X} \in \mathbb{R}^d \): find \( \Phi \) and \( \tilde{\Phi} \) in a parametric family that minimize
  \[
  \frac{1}{n} \sum_{i=1}^{n} \| \mathcal{X}_i - \tilde{\Phi}(\Phi(\mathcal{X}_i)) \|^2
  \]
Principal Component Analysis

- $\mathcal{X} \in \mathbb{R}^d$ and $\mathcal{X}' = \mathbb{R}^{d'}$
- Affine model $\mathcal{X} \sim m + \sum_{i=1}^{d'} X^{(i)} V^{(i)}$ with $(V^{(i)})$ an orthonormal family.
- Equivalent to:
  \[ \Phi(\mathcal{X}) = V^\top (\mathcal{X} - m) \quad \text{and} \quad \tilde{\Phi}(\mathcal{X}') = m + VX' \]
- Reconstruction error criterion:
  \[ \frac{1}{n} \sum_{i=1}^{n} \| \mathcal{X}_i - (m + VV^\top (\mathcal{X}_i - m)) \|^2 \]
- **Explicit solution**: $m$ is the empirical mean and $V$ is any orthonormal basis of the space spanned by the $d'$ first eigenvectors (the one with largest eigenvalues) of the empirical covariance matrix $\frac{1}{n} \sum_{i=1}^{n} (\mathcal{X}_i - m)(\mathcal{X}_i - m)^\top$. 
Principal Component Analysis

### PCA Algorithm

- Compute the empirical mean $m = \frac{1}{n} \sum_{i=1}^{n} X_i$
- Compute the empirical covariance matrix $\frac{1}{n} \sum_{i=1}^{n} (X_i - m)(X_i - m)^\top$.
- Compute the $d'$ first eigenvectors of this matrix: $V^{(1)}, \ldots, V^{(d')}$
- Set $\Phi(X) = V^\top (X - m)$

- **Complexity:** $O(n(d + d^2) + d'd^2)$
- **Interpretation:**
  - $\Phi(X) = V^\top (X - m)$: coordinates in the restricted space.
  - $V^{(i)}$: influence of each original coordinates in the ith new one.

- **Scaling:** This method is not invariant to a scaling of the variables! It is custom to normalize the variables (at least within groups) before applying PCA.
Decathlon
Swiss Roll
Principal Component Analysis

Decathlon

Decathlon Renormalized

Swiss Roll
Multiple Factor Analysis

- PCA assumes $\mathcal{X} = \mathbb{R}^d$!
- How to deal with categorical values?
- MFA = PCA with clever coding strategy for categorical values.

### Categorical value code for a single variable

- Classical redundant dummy coding:
  \[
  X \in \{1, \ldots, V\} \mapsto P(X) = (1_{X=1}, \ldots, 1_{X=V})^T
  \]
- Compute the mean (i.e. the empirical proportions):
  \[
  \bar{P} = \frac{1}{n} \sum_{i=1}^{n} P(X_i)
  \]
- Renormalize $P(X)$ by $1/\sqrt{(V - 1)\bar{P}}$:
  \[
  P(X) = (1_{X=1}, \ldots, 1_{X=V}) \mapsto \left( \frac{1_{X=1}}{\sqrt{(V - 1)\bar{P}_1}}, \ldots, \frac{1_{X=V}}{\sqrt{(V - 1)\bar{P}_V}} \right) = P^r(X)
  \]
- $\chi^2$ type distance!
Multiple Factor Analysis

- PCA becomes the minimization of
  \[
  \frac{1}{n} \sum_{i=1}^{n} \| P^r(X_i) - (m + VV^T (P^r(X_i) - m)) \|^2
  \]
  \[
  = \frac{1}{n} \sum_{i=1}^{n} \sum_{v=1}^{V} \left| 1_{X_i=v} - (m' + \sum_{l=1}^{d'} V^{(l)}^T (P(X_i) - m') V^{(l,v)}) \right|^2
  \]

- Interpretation:
  - \( m' = \bar{P} \)
  - \( \Phi(X) = V^T (P^r(X) - m) \): coordinates in the restricted space.
  - \( V^{(l)} \) can be interpreted as a probability profile.

- Complexity: \( O(n(V + V^2) + d' V^2) \)

- Link with Correspondence Analysis (CA)
Multiple Factor Analysis

MFA Algorithm

- Redundant dummy coding of each categorical variable.
- Renormalization of each block of dummy variable.
- Classical PCA algorithm on the resulting variables

- Interpretation as a reconstruction error with a rescaled/\(\chi^2\) metric.
- Interpretation:
  - \(\Phi(X) = V^\top (P^r(X) - m)\): coordinates in the restricted space.
  - \(V^{(i)}\): influence of each modality/variable in the ith new coordinates.

- **Scaling**: This method is not invariant to a scaling of the continuous variables! It is custom to normalize the variables (at least within groups) before applying PCA.
Multiple Factor Analysis
Non Linear PCA

PCA Model

- PCA: Linear model assumption
  \[ \mathbf{X} \simeq \mathbf{m} + \sum_{l=1}^{d'} \mathbf{X}_{l}^{(l)} \mathbf{V}^{(l)} = \mathbf{m} + \mathbf{VX}' \]
  - with
    - \( \mathbf{V}^{(l)} \) orthonormal
    - \( \mathbf{X}_{l}^{(l)} \) without constraints.

- Two directions of extension:
  - Other constraints on \( \mathbf{V} \) (or the coordinates in the restricted space): ICA, NMF, Dictionary approach
  - PCA on a non-linear image of \( \mathbf{X} \): kernel-PCA

- Much more complex algorithm!
ICA (Independent Component Analysis)

- Linear model assumption

\[ X \simeq m + \sum_{l=1}^{d'} X_{l,()}^{(l)} V^{(l)} = m + VX' \]

- with
  - \( V^{(l)} \) without constraints.
  - \( X_{l,()}^{(l)} \) independent

NMF (Non Negative Matrix Factorization)

- (Linear) Model assumption

\[ X \simeq \sum_{l=1}^{d'} X_{l,()}^{(l)} V^{(l)} = VX' \]

- with
  - \( V^{(l)} \) non-negative
  - \( X_{l,()}^{(l)} \) non-negative.
### Non Linear PCA

**Dictionary**

- (Linear) Model assumption
  \[ X \simeq m + \sum_{l=1}^{d'} X',(l) V^{(l)} = m + VX' \]
  - with
    - \( V^{(l)} \) without constraints
    - \( X' \) sparse (with a lot of 0)

**Kernel PCA**

- Linear model assumption
  \[ \Psi(X - m) \simeq \sum_{l=1}^{d'} X',(l) V^{(l)} = VX' \]
  - with
    - \( V^{(l)} \) orthonormal
    - \( X' \) without constraints.
Non Linear PCA

Decathlon

Swiss Roll

ICA

NMF

Kernel PCA
Auto Encoder

Deep Auto Encoder

- Construct a map $\Phi$ with a **NN** from the space $\mathcal{X}$ into a space $\mathcal{X}'$ of smaller dimension:

  $$\Phi : \mathcal{X} \rightarrow \mathcal{X}'$$

  $$\mathcal{X} \mapsto \Phi(\mathcal{X})$$

- Construct $\tilde{\Phi}$ with a **NN** from $\mathcal{X}'$ to $\mathcal{X}$

- Control the error between $\mathcal{X}$ and its reconstruction $\tilde{\Phi}(\Phi(\mathcal{X}))$:

  $$\frac{1}{n} \sum_{i=1}^{n} \| \mathcal{X}_i - \tilde{\Phi}(\Phi(\mathcal{X}_i)) \|^2$$

  - Optimization by gradient descent.
  - **NN** can be replaced by another parametric function...
Deep Auto Encoder

Shallow Auto Encoder (PCA)

Deep Auto Encoder

Unsupervised Learning: Dimension Reduction and Clustering
Unsupervised Learning: Dimension Reduction and Clustering

Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

9. References
Pairwise Relation

- Different point of view!
- Focus on pairwise relation \( \mathcal{R}(X_i, X_j) \).

Distance Preservation

- Construct a map \( \Phi \) from the space \( \mathcal{X} \) into a space \( \mathcal{X}' \) of smaller dimension:
  \[
  \Phi : \mathcal{X} \rightarrow \mathcal{X}' \\
  X \mapsto \Phi(X) = X'
  \]
- such that
  \[
  \mathcal{R}(X_i, X_j) \sim \mathcal{R}'(X'_i, X'_j)
  \]

- Most classical version (MDS):
  - Scalar product relation: \( \mathcal{R}(X_i, X_j) = (X_i - m)^\top (X_j - m) \)
  - Linear mapping \( X' = \Phi(X) = V^\top (X - m) \).
  - Euclidean scalar product matching:
    \[
    \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \left| (X_i - m)^\top (X_j - m) - (X'_i)^\top X'_j \right|^2
    \]
- \( \Phi \) often defined only on \( \mathcal{D} \). . .
MultiDimensional Scaling

MDS Heuristic

- Match the scalar products:
  \[
  \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \left| (X_i - m)^\top (X_j - m) - X_i'\top X_j' \right|^2
  \]

- Linear method: \( X' = U^\top (X - m) \) with \( U \) orthonormal

- **Beware:** \( X \) can be unknown, only the scalar products are required!

- Resulting criterion: minimization in \( U^\top (X_i - m) \) of
  \[
  \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \left| (X_i - m)^\top (X_j - m) - (X_i - m)^\top UU^\top (X_j - m) \right|^2
  \]
  without using explicitly \( X \) in the algorithm...

- Explicit solution obtained through the eigendecomposition of the know Gram matrix \( (X_i - m)^\top (X_j - m) \) by keeping only the \( d' \) largest eigenvalues.
In this case, MDS yields the same result as the PCA (but with different inputs, distance between observation vs correlations)!

**Explanation:** Same SVD problem up to a transposition:
- MDS
  \[
  \overline{X}_n^\top \overline{X}_n \sim \overline{X}_n^\top UU^\top \overline{X}_n
  \]
- PCA
  \[
  \overline{X}_n^\top \overline{X}_n^\top \sim U^\top \overline{X}_n \overline{X}_n^\top U
  \]

**Complexity:** PCA \(O((n + d')d^2)\) vs MDS \(O((d + d')n^2)\)…
MultiDimensional Scaling

Decathlon

Swiss Roll

PCA

MDS
Generalized MDS

- Preserving the scalar products amounts to preserve the Euclidean distance.
- Easier **generalization** if we work in terms of distance!

Generalized MDS:

- Generalized MDS:
  - Distance relation: \( \mathcal{R}(X_i, X_j) = d(X_i, X_j) \)
  - Linear mapping \( X' = \Phi(X) = V^\top (X - m) \).
  - Euclidean matching:
    \[
    \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} |d(X_i, X_j) - d'(X'_i, X'_j)|^2
    \]

- Strong connection (but no equivalence) with MDS when \( d(x, y) = \|x - y\|^2 \)
- **Minimization**: Simple gradient descent can be used (can be stuck in local minima).
MDS: equivalent to PCA (but more expensive) if $d(x, y) = \|x - y\|^2$

ISOMAP: use a localized distance instead to limit the influence of very far point.

**ISOMAP**

- For each point $X_i$, define a neighborhood $\mathcal{N}_i$ (either by a distance or a number of points) and let
  
  $$d_0(X_i, X_j) = \begin{cases} 
  +\infty & \text{if } X_j \notin \mathcal{N}_i \\
  \|X_i - X_j\|^2 & \text{otherwise}
  \end{cases}$$

- Compute the shortest path distance for each pair.
- Use the MDS algorithm with this distance.
ISOMAP

Decathlon

Swiss Roll
Random Projection

Random Projection Heuristic

- Draw at random $d'$ unit vector (direction) $U_i$.
- Use $X' = U^\top (X - m)$ with $m = \frac{1}{n} \sum_{i=1}^{n} X_i$

- **Property:** If $X$ lives in a space of dimension $d''$, then, as soon as, $d' \sim d'' \log(d'')$,
  $$\|X_i - X_j\|^2 \sim \frac{d'}{d''} \|X'_i - X'_j\|^2$$

- Do not really use the data!
Random Projection

Decathlon

Swiss Roll
t-Stochastic Neighbor Embedding

**SNE heuristic**

- From $X_i \in X$, construct a set of conditional probability:
  \[
P_{j|i} = \frac{e^{-\|X_i - X_j\|^2/2\sigma_i^2}}{\sum_{k \neq i} e^{-\|X_i - X_k\|^2/2\sigma_i^2}} \quad P_{i|i} = 0
  \]

- Find $X'_i$ in $\mathbb{R}^{d''}$ such that the set of conditional probability:
  \[
  Q_{j|i} = \frac{e^{-\|X'_i - X'_j\|^2/2\sigma_i^2}}{\sum_{k \neq i} e^{-\|X'_i - X'_k\|^2/2\sigma_i^2}} \quad Q_{i|i} = 0
  \]
  is close from $P$.

- **t-SNE**: use a Student-t term $(1 + \|X'_i - X'_j\|^2)^{-1}$ for $X'_i$

- Minimize the Kullback-Leibler divergence $\left( \sum_{i,j} P_{j|i} \log \frac{P_{j|i}}{Q_{j|i}} \right)$ by a simple gradient descent (can be stuck in local minima).

- Parameters $\sigma_i$ such that $H(P_i) = - \sum_{j=1}^n P_{j|i} \log P_{j|i} = \text{cst.}$
t-Stochastic Neighbor Embedding

- Very successful/ powerful technique in practice
- Convergence may be long, unstable, or strongly depending on parameters.
- See this distill post for many impressive examples

Representation depending on t-SNE parameters
t-SNE

Decathlon

Swiss Roll
UMAP

- Topological Data Analysis inspired.

**Uniform Manifold Approximation and Projection**

- Define a notion of asymmetric scaled local proximity between neighbors:
  - Compute the $k$-neighborhood of $X_j$, its diameter $\sigma_i$ and the distance $\rho_i$ between $X_j$ and its nearest neighbor.
  - Define
    
    $$w_i(X_i, X_j) = \begin{cases} 
    e^{-(d(X_i, X_j) - \rho_i)/\sigma_i} & \text{for } X_j \text{ in the } k\text{-neighborhood} \\
    0 & \text{otherwise}
    \end{cases}$$

- Symmetrize into a *fuzzy* nearest neighbor criterion
  $$w(X_i, X_j) = w_i(X_i, X_j) + w_j(X_j, X_i) - w_i(X_i, X_j)w_j(X_j, X_i)$$

- Determine the points $X'_i$ in a low dimensional space such that
  $$\sum_{i \neq j} w(X_i, X_j) \log \left( \frac{w(X_i, X_j)}{w'(X'_i, X'_j)} \right) + (1 - w(X_i, X_j)) \log \left( \frac{1 - w(X_i, X_j))}{(1 - w'(X'_i, X'_j))} \right)$$

- Can be performed by local gradient descent.
UMAP

Unsupervised Learning: Dimension Reduction and Clustering

Decathlon

Swiss Roll
Graph based

Graph heuristic

- Construct a graph with weighted edges $w_{i,j}$ measuring the proximity of $X_i$ and $X_j$ ($w_{i,j}$ large if close and 0 if there is no information).
- Find the points $X'_i \in \mathbb{R}^{d'}$ minimizing
  \[
  \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{i,j} \|X'_i - X'_j\|^2
  \]

- Need of a constraint on the size of $X'_i$...
- Explicit solution through linear algebra: $d'$ eigenvectors with smallest eigenvalues of the Laplacian of the graph $D - W$, where $D$ is a diagonal matrix with $D_{i,i} = \sum_j w_{i,j}$.
- Variation on the definition of the Laplacian...
Unsupervised Learning: Dimension Reduction and Clustering

Graph

Decathlon

Swiss Roll
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Ensemble Methods

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension

9. References

Unsupervised Learning: Dimension Reduction and Clustering
How to Compare Different Dimensionality Reduction Methods?

- **Difficult!** Once again, the metric is very subjective.

**However, a few possible attempts**

- Did we preserve a lot of inertia with only a few directions?
- Do those directions *make sense* from an expert point of view?
- Do the low dimension representation *preserve* some important information?
- Are we better on *subsequent task*?
A Challenging Example: MNIST

MNIST Dataset
- Images of 28 × 28 pixels.
- No label used!
- 4 different embeddings.
A Challenging Example: MNIST

MNIST Dataset
- Images of $28 \times 28$ pixels.
- No label used!
- 4 different embeddings.
A Challenging Example: MNIST

MNIST Dataset
- Images of $28 \times 28$ pixels.
- No label used!
- 4 different embeddings.

- Quality evaluated by visualizing the true labels not used to obtain the embeddings.
- Only a few labels could have been used.
A Simpler Example: A 2D Set

Cluster Dataset
- Set of points in 2D.
- No label used!
- 3 different embeddings.
A Simpler Example: A 2D Set

Cluster Dataset
- Set of points in 2D.
- No label used!
- 3 different embeddings.
A Simpler Example: A 2D Set

Cluster Dataset

- Set of points in 2D.
- No label used!
- 3 different embeddings.

- Quality evaluated by stability...
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References
2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization
3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights
4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References
5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References
6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References
7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
9. References
Unsupervised Learning: Dimension Reduction and Clustering

Clustering

- **Training data**: \( \mathcal{D} = \{X_1, \ldots, X_n\} \in \mathcal{X}^n \) (i.i.d. \( \sim \mathcal{P} \))
- Latent groups?

**Clustering**

- Construct a map \( f \) from \( \mathcal{D} \) to \( \{1, \ldots, K\} \) where \( K \) is a number of classes to be fixed:

\[
f : \ X_i \mapsto k_i
\]

**Motivations**

- Interpretation of the groups
- Use of the groups in further processing

- Several strategies possible!
- Can use dimension reduction as a preprocessing.
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Introduction
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9. References
### Partition Based

#### Partition Heuristic
- Clustering is defined by a partition in $K$ classes...
- that minimizes a homogeneity criterion.

#### K- Means
- Cluster $k$ defined by a center $\mu_k$.
- Each sample is associated to the closest center.
- Centers defined as the minimizer of $\sum_{i=1}^{n} \min_k \| X_i - \mu_k \|^2$

- Iterative scheme (Loyd):
  - Start by a (pseudo) random choice for the centers $\mu_k$
  - Assign each samples to its nearby center
  - Replace the center of a cluster by the mean of its assigned samples.
  - Repeat the last two steps until convergence.
Partition Based

K-means, step 0 - 4
Partition based

- Other schemes:
  - McQueen: modify the mean each time a sample is assigned to a new cluster.
  - Hartigan: modify the mean by removing the considered sample, assign it to the nearby center and recompute the new mean after assignment.

A good initialization is crucial!

- Initialize by samples.
- k-Mean++: try to take them as separated as possible.
- No guarantee to converge to a global optimum: repeat and keep the best result!

- Complexity: $O(n \times K \times T)$ where $T$ is the number of steps in the algorithm.
Partition based

- k-Medoid: use a sample as a center
  - PAM: for a given cluster, use the sample that minimizes the intra distance (sum of the squared distance to the other points)
  - Approximate medoid: for a given cluster, assign the point that is the closest to the mean.

**Complexity**

- PAM: \( O(n^2 \times T) \) in the worst case!
- Approximate medoid: \( O(n \times K \times T) \) where \( T \) is the number of steps in the algorithm.

**Remark:** Any distance can be used... but the complexity of computing the centers can be very different.
K-Means

$k = 4$

$k = 10$

$k = 10$
### Model Heuristic

- Use a generative model of the data:
  \[
P(X) = \sum_{k=1}^{K} \pi_k P_{\theta_k}(X|k)
\]
  where \(\pi_k\) are proportions and \(P_{\theta_k}(X|k)\) are parametric probability models.

- Estimate those parameters (often by a ML principle).

- Assign each observation to the class maximizing the a posteriori probability (obtained by Bayes formula)
  \[
  \frac{\hat{\pi}_k \hat{P}_{\hat{\theta}_k}(X|k)}{\sum_{k'=1}^{K} \hat{\pi}_{k'} \hat{P}_{\hat{\theta}_{k'}}(X|k')}
  \]

- Link with Generative model in supervised classification!
Large choice of parametric models.

**Gaussian Mixture Model**

- Use

\[ P_{\theta_k}(\tilde{X}|k) \sim \mathcal{N}(\mu_k, \Sigma_k) \]

with \( \mathcal{N}(\mu, \Sigma) \) the Gaussian law of mean \( \mu \) and covariance matrix \( \Sigma \).

- Efficient optimization algorithm available (EM)
- Often some constraint on the covariance matrices: identical, with a similar structure...
- Strong connection with \( K \)-means when the covariance matrices are assumed to be the same multiple of the identity.
Probabilistic latent semantic analysis (PLSA)

- Documents described by their word counts $w$
- Model:

$$P(w) = \sum_{k=1}^{K} \pi_k P_{\theta_k}(w|k)$$

with $k$ the (hidden) topic, $\pi_k$ a topic probability and $P_{\theta_k}(w|k)$ a multinomial law for a given topic.
- Clustering according to

$$P(k|w) = \frac{\hat{\pi}_k \hat{P}_{\hat{\theta}_k}(w|k)}{\sum_{k'} \hat{\pi}_{k'} \hat{P}_{\hat{\theta}_{k'}}(w|k')}$$

- Same idea than GMM!
- Bayesian variant called LDA.
Model Based

### Parametric Density Estimation Principle

- Assign a probability of membership.
- Lots of theoretical studies...
- Model selection principle can be used to select $K$ the number of classes (or rather to avoid using a nonsensical $K$...):
  - AIC / BIC / MDL penalization
  - Cross Validation is also possible!

- Complexity: $O(n \times K \times T)$
Gaussian Mixture Models

$k = 4$

$k = 10$

$k = 10$
<table>
<thead>
<tr>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>9</td>
</tr>
</tbody>
</table>
Density Heuristic

- Cluster are connected dense zone separated by low density zone.
- Not all points belong to a cluster.

Basic bricks:
- Estimate the density.
- Find points with high densities.
- Gather those points according to the density.

Density estimation:
- Classical kernel density estimators...

Gathering:
- Link points of high density and use the resulted component.
- Move them toward top of density hill by following the gradient and gather all the points arriving at the same summit.
Examples

- **DBSCAN**: link point of high densities using a very simple kernel.
- **PdfCLuster**: find connected zone of high density.
- **Mean-shift**: move points toward top of density **hill** following an evolving kernel density estimate.

- **Complexity**: $O(n^2 \times T)$ in the worst case.
- Can be reduced to $O(n \log(n) T)$ if samples can be encoded in a tree structure (n-body problem type approximation).
Unsupervised Learning: Dimension Reduction and Clustering

DBSCAN

$\epsilon = .45$  $\epsilon = .2$  $\epsilon = .1$
Agglomerative Clustering

**Agglomerative Clustering Heuristic**

- Start with very small clusters (a sample by cluster?)
- Sequential merging of the most similar clusters...
- according to some *greedy* criterion $\Delta$.

- Generates a hierarchy of clustering instead of a single one.
- Need to select the number of cluster afterwards.
- Several choices for the merging criterion...
- Examples:
  - Minimum Linkage: merge the closest cluster in term of the usual distance
  - Ward’s criterion: merge the two clusters yielding the less inner inertia loss (k-means criterion)
Agglomerative Clustering

Algorithm

- Start with $(C_i^{(0)}) = (\{X_i\})$ the collection of all singletons.
- At step $s$, we have $n - s$ clusters $(C_i^{(s)})$:
  - Find the two most similar clusters according to a criterion $\Delta$:
    $$(i, i') = \arg\min_{(j, j')} \Delta(C_j^{(s)}, C_{j'}^{(s)})$$
  - Merge $C_i^{(s)}$ and $C_{i'}^{(s)}$ into $C_i^{(s+1)}$
  - Keep the $n - s - 2$ other clusters $C_{i''}^{(s+1)} = C_{i''}^{(s)}$
- Repeat until there is only one cluster.

- Complexity: $O(n^3)$ in general.
- Can be reduced to $O(n^2)$
  - if only a bounded number of merging is possible for a given cluster,
  - for the most classical distances by maintaining a nearest neighbors list.
Agglomerative Clustering

Merging criterion based on the distance between points

- **Minimum linkage:**
  \[\Delta(C_i, C_j) = \min_{X_i \in C_i} \min_{X_j \in C_j} d(X_i, X_j)\]

- **Maximum linkage:**
  \[\Delta(C_i, C_j) = \max_{X_i \in C_i} \max_{X_j \in C_j} d(X_i, X_j)\]

- **Average linkage:**
  \[\Delta(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{X_i \in C_i} \sum_{X_j \in C_j} d(X_i, X_j)\]

- Clustering based on the proximity...
Merging criterion based on the inertia (distance to the mean)

- **Ward’s criterion:**
  \[
  \Delta(C_i, C_j) = \sum_{X_i \in C_i} \left( d^2(X_i, \mu_{C_i \cup C_j}) - d^2(X_i, \mu_{C_i}) \right) \\
  + \sum_{X_j \in C_j} \left( d^2(X_j, \mu_{C_i \cup C_j}) - d^2(X_j, \mu_{C_j}) \right)
  \]

- **If** \(d\) **is the Euclidean distance:**
  \[
  \Delta(C_i, C_j) = \frac{2|C_i||C_j|}{|C_i| + |C_j|} d^2(\mu_{C_i}, \mu_{C_j})
  \]

- **Same criterion than in the k-means algorithm but greedy optimization.**
Unsupervised Learning: Dimension Reduction and Clustering

Agglomerative Clustering

Single

Complete

Ward

Dendogram

$k = 4$

$k = 10$

$k = 20$
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

9. References

523
Grid based

Grid heuristic

- Split the space in pieces
- Group those of high density according to their proximity

- Similar to density based estimate (with partition based initial clustering)
- Space splitting can be fixed or adaptive to the data.
- Examples:
  - STING (Statistical Information Grid): Hierarchical tree construction plus DBSCAN type algorithm
  - AMR (Adaptive Mesh Refinement): Adaptive tree refinement plus $k$-means type assignment from high density leaves.
  - CLIQUE: Tensorial grid and 1D detection.
- Linked to Divisive clustering (DIANA)
Graph based

- Spectral clustering: dimension reduction + k-means.
- Message passing: iterative local algorithm.
- Graph cut: min/max flow.

- Kohonen Map,

- ...
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

9. References
Scalability

Large dataset issue

- When $n$ is large, a $O(n^\alpha \log n)$ with $\alpha > 1$ is not acceptable!
- How to deal with such a situation?

**Beware:** Computing all the pairwise distance requires $O(n^2)$ operations!

Ideas

- Sampling
- Online processing
- Simplification
- Parallelization
Sampling

Sampling heuristic

- Use only a subsample to construct the clustering.
- Assign the other points to the constructed clusters afterwards.

- Requires a clustering method that can assign new points (partition, model...)
- Often repetition and choice of the best clustering
- Example:
  - CLARA: K-medoid with sampling and repetition
- Two-steps algorithm:
  - Generate a large number $n'$ of clusters using a fast algorithm (with $n' \ll n$)
  - Cluster the clusters with a more accurate algorithm.
Online heuristic

- Modify the current clusters according to the value of a single observation.

- Requires compactly described clusters.
- Examples:
  - Add to an existing cluster (and modify it) if it is close enough and create a new cluster otherwise (*k*-means without reassignment)
  - Stochastic descent gradient (GMM)
- May lead to far from optimal clustering.
Simplification

Simplification heuristic

- Simplify the algorithm to be more efficient at the cost of some precision.

- Algorithm dependent!

- Examples:
  - Replace groups of observation (preliminary cluster) by the (approximate) statistics.
  - Approximate the distances by cheaper ones.
  - Use n-body type techniques.
Parallelization

Parallelization heuristic

- Split the computation on several computers.
- Algorithm dependent!
- Examples:
  - Distance computation in $k$-means, parameter gradient in model based clustering
  - Grid density estimation, Space splitting strategies
- Classical batch sampling not easy to perform as partitions are not easily merged...
# Outline

<table>
<thead>
<tr>
<th>1</th>
<th>Statistical Learning: Introduction, Setting and Risk Estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>1.2</td>
<td>Machine Learning</td>
</tr>
<tr>
<td>1.3</td>
<td>Supervised Learning</td>
</tr>
<tr>
<td>1.4</td>
<td>Risk Estimation and Model Selection</td>
</tr>
<tr>
<td>1.5</td>
<td>Cross Validation and Test</td>
</tr>
<tr>
<td>1.6</td>
<td>References</td>
</tr>
<tr>
<td>2</td>
<td>ML Methods: Probabilistic Point of View</td>
</tr>
<tr>
<td>2.1</td>
<td>Motivation</td>
</tr>
<tr>
<td>2.2</td>
<td>Supervised Learning</td>
</tr>
<tr>
<td>2.3</td>
<td>A Probabilistic Point of View</td>
</tr>
<tr>
<td>2.4</td>
<td>Parametric Conditional Density Modeling</td>
</tr>
<tr>
<td>2.5</td>
<td>Non Parametric Conditional Density Modeling</td>
</tr>
<tr>
<td>2.6</td>
<td>Generative Modeling</td>
</tr>
<tr>
<td>2.7</td>
<td>Model Selection</td>
</tr>
<tr>
<td>2.8</td>
<td>Penalization</td>
</tr>
<tr>
<td>3</td>
<td>ML Methods: Optimization Point of View</td>
</tr>
<tr>
<td>3.1</td>
<td>Supervised Learning</td>
</tr>
<tr>
<td>3.2</td>
<td>Optimization Point of View</td>
</tr>
<tr>
<td>3.3</td>
<td>SVM</td>
</tr>
<tr>
<td>3.4</td>
<td>Penalization</td>
</tr>
<tr>
<td>3.5</td>
<td>Cross Validation and Weights</td>
</tr>
<tr>
<td>4</td>
<td>Optimization: Gradient Descent Algorithms</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>4.2</td>
<td>Gradient Descent</td>
</tr>
<tr>
<td>4.3</td>
<td>Proximal Descent</td>
</tr>
<tr>
<td>4.4</td>
<td>Coordinate Descent</td>
</tr>
<tr>
<td>4.5</td>
<td>Gradient Descent Acceleration</td>
</tr>
<tr>
<td>4.6</td>
<td>Stochastic Gradient Descent</td>
</tr>
<tr>
<td>4.7</td>
<td>Gradient Descent Step</td>
</tr>
<tr>
<td>4.8</td>
<td>Non-Convex Setting</td>
</tr>
<tr>
<td>4.9</td>
<td>References</td>
</tr>
<tr>
<td>5</td>
<td>ML Methods: Neural Networks and Deep Learning</td>
</tr>
<tr>
<td>5.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>5.2</td>
<td>From Logistic Regression to NN</td>
</tr>
<tr>
<td>5.3</td>
<td>NN Optimization</td>
</tr>
<tr>
<td>5.4</td>
<td>NN Regularization</td>
</tr>
<tr>
<td>5.5</td>
<td>Image and CNN</td>
</tr>
<tr>
<td>5.6</td>
<td>Text, Recurrent Neural Networks and Transformers</td>
</tr>
<tr>
<td>5.7</td>
<td>NN Architecture</td>
</tr>
<tr>
<td>5.8</td>
<td>References</td>
</tr>
<tr>
<td>6</td>
<td>ML Methods: Trees and Ensemble Methods</td>
</tr>
<tr>
<td>6.1</td>
<td>Trees</td>
</tr>
<tr>
<td>6.2</td>
<td>Bagging and Random Forests</td>
</tr>
<tr>
<td>6.3</td>
<td>Bootstrap and Bagging</td>
</tr>
<tr>
<td>6.4</td>
<td>Randomized Rules and Random Forests</td>
</tr>
<tr>
<td>6.5</td>
<td>Boosting</td>
</tr>
<tr>
<td>6.6</td>
<td>AdaBoost as a Greedy Scheme</td>
</tr>
<tr>
<td>6.7</td>
<td>Boosting</td>
</tr>
<tr>
<td>6.8</td>
<td>Ensemble Methods</td>
</tr>
<tr>
<td>6.9</td>
<td>References</td>
</tr>
<tr>
<td>7</td>
<td>Unsupervised Learning: Dimension Reduction and Clustering</td>
</tr>
<tr>
<td>7.1</td>
<td>Unsupervised Learning?</td>
</tr>
<tr>
<td>7.2</td>
<td>A First Glimpse</td>
</tr>
<tr>
<td>7.3</td>
<td>Clustering</td>
</tr>
<tr>
<td>7.4</td>
<td>Dimensionality Curse</td>
</tr>
<tr>
<td>7.5</td>
<td>Dimension Reduction</td>
</tr>
<tr>
<td>7.6</td>
<td>Generative Modeling</td>
</tr>
<tr>
<td>7.7</td>
<td>Dimension Reduction</td>
</tr>
<tr>
<td>7.8</td>
<td>Simplification</td>
</tr>
<tr>
<td>7.9</td>
<td>Reconstruction Error</td>
</tr>
<tr>
<td>7.10</td>
<td>Relationship Preservation</td>
</tr>
<tr>
<td>7.11</td>
<td>Comparing Methods?</td>
</tr>
<tr>
<td>7.12</td>
<td>Clustering</td>
</tr>
<tr>
<td>7.13</td>
<td>Prototype Approaches</td>
</tr>
<tr>
<td>7.14</td>
<td>Contiguity Approaches</td>
</tr>
<tr>
<td>7.15</td>
<td>Agglomerative Approaches</td>
</tr>
<tr>
<td>7.16</td>
<td>Other Approaches</td>
</tr>
<tr>
<td>7.17</td>
<td>Scalability</td>
</tr>
<tr>
<td>7.18</td>
<td>Generative Modeling</td>
</tr>
<tr>
<td>7.19</td>
<td>(Plain) Parametric Density Estimation</td>
</tr>
<tr>
<td>7.20</td>
<td>Latent Variables</td>
</tr>
<tr>
<td>7.21</td>
<td>Approximate Simulation</td>
</tr>
<tr>
<td>7.22</td>
<td>Diffusion Model</td>
</tr>
<tr>
<td>7.23</td>
<td>Generative Adversarial Network</td>
</tr>
<tr>
<td>7.24</td>
<td>Applications to Text</td>
</tr>
<tr>
<td>7.25</td>
<td>References</td>
</tr>
<tr>
<td>8</td>
<td>Statistical Learning: PAC-Bayesian Approach and Complexity Theory</td>
</tr>
<tr>
<td>8.1</td>
<td>Supervised Learning</td>
</tr>
<tr>
<td>8.2</td>
<td>Empirical Risk Minimization</td>
</tr>
<tr>
<td>8.3</td>
<td>Empirical Risk Minimization</td>
</tr>
<tr>
<td>8.4</td>
<td>ERM and PAC Analysis</td>
</tr>
<tr>
<td>8.5</td>
<td>Hoeffding and Finite Class</td>
</tr>
<tr>
<td>8.6</td>
<td>McDiarmid and Rademacher Complexity</td>
</tr>
<tr>
<td>8.7</td>
<td>VC Dimension</td>
</tr>
<tr>
<td>8.8</td>
<td>Structural Risk Minimization</td>
</tr>
<tr>
<td>8.9</td>
<td>References</td>
</tr>
<tr>
<td>9</td>
<td>References</td>
</tr>
</tbody>
</table>
Generative Modeling

- **Training data**: \( \mathcal{D} = \{(X_1, Y_1), \ldots, (X_n, Y_n)\} \in (\mathcal{X} \times \mathcal{Y})^n \) (i.i.d. \( \sim \mathbb{P} \))

- Same kind of data than for supervised learning if \( \mathcal{Y} \neq \emptyset \).

Generative Modeling

- Construct a map \( G \) from the product of \( \mathcal{Y} \) and a randomness source \( \Omega \) to \( \mathcal{X} \):
  
  \[
  G : \mathcal{Y} \times \Omega \to \mathcal{X}
  \]
  
  \[ (Y, \omega) \mapsto X \]

- Unconditional model if \( \mathcal{Y} = \emptyset \ldots \)

Motivation

- Generate plausible novel conditional samples based on a given dataset.

Sample Quality

- Related to the proximity between the law of \( G(Y, \omega) \) and the law of \( X|Y \).

- Most classical choice is the Kullback-Leibler divergence.
Generative Modeling

Ingredients

- Generator $F_\theta(Y, \omega)$ and cond. density prob. $p_\theta(X|Y)$ (Explicit vs implicit link)
- Simple / Complex / Approximate estimation...

Some Possible Choices

<table>
<thead>
<tr>
<th>Probabilistic model</th>
<th>Generator</th>
<th>Estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>Simple (parametric)</td>
<td>Explicit</td>
</tr>
<tr>
<td>Flow</td>
<td>Image of simple model</td>
<td>Explicit</td>
</tr>
<tr>
<td>Factorization</td>
<td>Factorization of simple model</td>
<td>Explicit</td>
</tr>
<tr>
<td>VAE</td>
<td>Simple model with latent var.</td>
<td>Explicit</td>
</tr>
<tr>
<td>EBM</td>
<td>Arbitrary</td>
<td>Implicit (MCMC)</td>
</tr>
<tr>
<td>Diffusion</td>
<td>Continuous noise</td>
<td>Implicit (MCMC)</td>
</tr>
<tr>
<td>GAN</td>
<td>Discrete Noise with latent var.</td>
<td>Implicit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- SOTA: Diffusion based approach!

Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

9. References
Density Estimation
Flow
Factorization
Latent
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Ensemble Methods

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension

9. References
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

9. References
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

9. References

545
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

9. References
Text and Representation

- Need to transform a text into a numerical vector to reuse the previous algorithms!
- Art still in progress.

Important steps:
- Token extraction
- Token vectorization
- Learning algorithm
Token Extraction

From a text to a sequence of *tokens* (words, characters, subwords...).

- Need of cleaning or pre-processing: spelling checker, stemming, lemmatization...
- Often with a further reduction of the number of possible tokens.
- Beware to not oversimplify!
Bag of Words

- Most simple approach to transform a text into a vector.
- Simple count of the words belonging to a predefined vocabulary.

- Counts preferably replaced by frequencies (or tf-idf...)
- Often combined with dimension reduction:
  - restriction to an interesting vocabulary
  - use of principal component analysis (latent semantic analysis)

*td-idf: text frequency - inverse document frequency*
Article Clustering

- Clustering algorithms directly on the bag-of-words representation
- Most used algorithm is a variation around the $k$-means algorithm.
Word Representation

- More accuracy by working at the token (word) scale.
- Two approaches:
  - Associate to a word the frequency of the other words in its neighborhood and performing dimension reduction on this first representation.
  - Learn for each word a vector allowing to predict by a simple formula (scalar product) whether one word appears in the neighborhood of the other one.
- Similar results but the second approach is more flexible.
Deep Learning

- Propose a formula allowing to do computations on the word starting by associating vectors to each word.
- Learning the best possible vectors for a given task: auto-prediction (self-supervised) or prediction (supervised).
- Tremendous progress in the last years thanks to deep neural net architectures (RNN, Transformer...).
Large Language Models

- Huge neural networks relying on transformers and (pre)trained on huge corpus with self-supervised tasks.
- Three architectures:
  - Decoder: prediction of next word (online).
  - Encoder: prediction of inner word(s) (offline).
  - Encoder/Decoder: prediction of a sentence from another (offline).
- Can be used as a basis for further specialized training or directly.

Source: Xia et al.
How to associate a sentiment to a text?

- Four possible approaches:
  - Simple approach (without learning) that averages the *sentiments* of the words used in a text using a fixed table.
  - Simple approach (supervised and linear) where this table is learned from examples.
  - Direct approach (supervised) where one predicts directly the sentiment from examples.
  - Zero-Shot approach (without learning?) where one uses directly a Large Language Model trained on a huge corpus (unrelated to the application).

- Direct approach more efficient provided one has sufficient data and one starts from a pretrained model.
Unsupervised Learning: Dimension Reduction and Clustering

Unsupervised Learning?
A First Glimpse
Clustering
Dimensionality Curse
Dimension Reduction
Generative Modeling

Gradient Descent Step
Non-Convex Setting
References

ML Methods: Neural Networks and Deep Learning
Introduction
From Logistic Regression to NN
NN Optimization
NN Regularization
Image and CNN
Text, Recurrent Neural Networks and Transformers
NN Architecture
References

ML Methods: Trees and Ensemble Methods
Trees
Bagging and Random Forests
Bootstrap and Bagging
Randomized Rules and Random Forests
Boosting
AdaBoost as a Greedy Scheme
Boosting
Ensemble Methods
References

Optimization: Gradient Descent Algorithms
Introduction
Gradient Descent
Proximal Descent
Coordinate Descent
Gradient Descent Acceleration
Stochastic Gradient Descent

Dimension Reduction
Simplification
Reconstruction Error
Relationship Preservation
Comparing Methods?
Clustering
Prototype Approaches
Contiguity Approaches
Agglomerative Approaches
Other Approaches
Scalability
Generative Modeling
(Plain) Parametric Density Estimation
Latent Variables
Approximate Simulation
Diffusion Model
Generative Adversarial Network
Applications to Text
References

Statistical Learning: PAC-Bayesian Approach and Complexity Theory
Supervised Learning
Empirical Risk Minimization
Empirical Risk Minimization
ERM and PAC Analysis
Hoeffding and Finite Class
McDiarmid and Rademacher Complexity
VC Dimension
Structural Risk Minimization
References

References
References


M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning (2nd ed.*)* MIT Press, 2018


Ch. Giraud. *Introduction to High-Dimensional Statistics*. CRC Press, 2014

References

F. Husson, S. Le, and J. Pagès.  
*Exploratory Multivariate Analysis by Example Using R (2nd ed.)*  
Chapman and Hall/CRC, 2017

B. Ghojogh, M. Crowley, F. Karray, and A. Ghodsi.  
*Elements of Dimensionality Reduction and Manifold Learning.*  
Springer, 2023

Ch. Aggarwal and Ch. Reddy.  
*Data Clustering: Algorithms and Applications.*  
Chapman and Hall/CRC, 2013

Ch. Hennig, M. Meila, F. Murtagh, and R. Rocci.  
*Handbook of Cluster Analysis.*  
Chapman and Hall/CRC, 2015

Ch. Bouveyron, G. Celeux, B. Murphy, and A. Raftery.  
*Model-Based Clustering and Classification for Data Science.*  
Cambridge University Press, 2019

J. Tomczak.  
*Deep Generative Modeling.*  
Springer, 2021

D. Foster.  
*Generative Deep Learning (2nd ed.)*  
O’Reilly, 2023

A. Géron.  
*Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (3rd ed.)*  
O’Reilly, 2022
Licence and Contributors

Creative Commons Attribution-ShareAlike (CC BY-SA 4.0)

- You are free to:
  - Share: copy and redistribute the material in any medium or format
  - Adapt: remix, transform, and build upon the material for any purpose, even commercially.

- Under the following terms:
  - Attribution: You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - ShareAlike: If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - No additional restrictions: You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Contributors

- Main contributor: E. Le Pennec
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
Supervised Learning

Supervised Learning Framework

- Input measurement $X \in \mathcal{X}$
- Output measurement $Y \in \mathcal{Y}$.
- $(X, Y) \sim P$ with $P$ unknown.
- **Training data**: $D_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ (i.i.d. $\sim P$)

- Often
  - $X \in \mathbb{R}^d$ and $Y \in \{-1, 1\}$ (classification)
  - or $X \in \mathbb{R}^d$ and $Y \in \mathbb{R}$ (regression).
- A **predictor** is a function in $\mathcal{F} = \{f : \mathcal{X} \to \mathcal{Y} \text{ meas.}\}$

Goal

- Construct a **good** predictor $\hat{f}$ from the training data.

- Need to specify the meaning of good.
- Classification and regression are almost the **same** problem!
## Loss and Probabilistic Framework

### Loss function for a generic predictor

- **Loss function**: $\ell(Y, f(X))$ measures the goodness of the prediction of $Y$ by $f(X)$

- **Examples**:
  - 0/1 loss: $\ell(Y, f(X)) = 1_{Y \neq f(X)}$
  - Quadratic loss: $\ell(Y, f(X)) = |Y - f(X)|^2$

### Risk function

- Risk measured as the average loss for a new couple:
  \[
  \mathcal{R}(f) = \mathbb{E}_{(X, Y) \sim P}[\ell(Y, f(X))]
  \]

- **Examples**:
  - 0/1 loss: $\mathbb{E}[\ell(Y, f(X))] = P(Y \neq f(X))$
  - Quadratic loss: $\mathbb{E}[\ell(Y, f(X))] = \mathbb{E}[|Y - f(X)|^2]$

- **Beware**: As $\hat{f}$ depends on $D_n$, $\mathcal{R}(\hat{f})$ is a random variable!
Best Solution

- The best solution $f^*$ (which is independent of $D_n$) is
  
  $$f^* = \arg\min_{f \in \mathcal{F}} \mathcal{R}(f) = \arg\min_{f \in \mathcal{F}} \mathbb{E}[\ell(Y, f(X))] = \arg\min_{f \in \mathcal{F}} \mathbb{E}_X \left[ \mathbb{E}_{Y|X}[\ell(Y, f(X))] \right]$$

Bayes Predictor (explicit solution)

- In binary classification with $0-1$ loss:
  
  $$f^*(X) = \begin{cases} 
  +1 & \text{if } \mathbb{P}(Y = +1|X) \geq \mathbb{P}(Y = -1|X) \\
  \Leftrightarrow \mathbb{P}(Y = +1|X) \geq 1/2 \\
  -1 & \text{otherwise}
  \end{cases}$$

- In regression with the quadratic loss
  
  $$f^*(X) = \mathbb{E}[Y|X]$$

Issue: Solution requires to know $\mathbb{E}[Y|X]$ for all values of $X$!
Goal

Machine Learning

- Learn a rule to construct a predictor $\hat{f} \in \mathcal{F}$ from the training data $\mathcal{D}_n$ s.t. the risk $\mathcal{R}(\hat{f})$ is small on average or with high probability with respect to $\mathcal{D}_n$.

- In practice, the rule should be an algorithm!

Canonical example: Empirical Risk Minimizer

- One restricts $f$ to a subset of functions $S = \{f_\theta, \theta \in \Theta\}$
- One replaces the minimization of the average loss by the minimization of the empirical loss

$$\hat{f} = f_\hat{\theta} = \arg\min_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(X_i))$$

- Examples:
  - Linear regression
  - Linear classification with

$$S = \{x \mapsto \text{sign}\{x^T \beta + \beta^{(0)}\} / \beta \in \mathbb{R}^d, \beta^{(0)} \in \mathbb{R}\}$$
Bias-Variance Dilemma

- General setting:
  - $\mathcal{F} = \{\text{measurable functions } \mathcal{X} \to \mathcal{Y}\}$
  - Best solution: $f^* = \arg\min_{f \in \mathcal{F}} \mathcal{R}(f)$
  - Class $\mathcal{S} \subset \mathcal{F}$ of functions
  - Ideal target in $\mathcal{S}$: $f^*_\mathcal{S} = \arg\min_{f \in \mathcal{S}} \mathcal{R}(f)$
  - Estimate in $\mathcal{S}$: $\hat{f}_\mathcal{S}$ obtained with some procedure

Approximation error and estimation error (Bias/Variance)

$$\mathcal{R}(\hat{f}_\mathcal{S}) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(f^*_\mathcal{S}) - \mathcal{R}(f^*)}_{\text{Approximation error}} + \underbrace{\mathcal{R}(\hat{f}_\mathcal{S}) - \mathcal{R}(f^*_\mathcal{S})}_{\text{Estimation error}}$$

- Approx. error can be large if the model $\mathcal{S}$ is not suitable.
- Estimation error can be large if the model is complex.
Different behavior for different model complexity

- **Low complexity model** are easily learned but the approximation error (bias) may be large (**Under-fit**).
- **High complexity model** may contain a good ideal target but the estimation error (variance) can be large (**Over-fit**)

**Bias-variance trade-off** $\iff$ avoid **overfitting** and **underfitting**

- **Rk:** Better to think in term of method (including feature engineering and specific algorithm) rather than only of model.
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling
   - Dimension Reduction
     - Simplification
     - Reconstruction Error
     - Relationship Preservation
     - Comparing Methods?
   - Clustering
     - Prototype Approaches
     - Contiguity Approaches
     - Agglomerative Approaches
     - Other Approaches
     - Scalability
   - Generative Modeling
     - (Plain) Parametric Density Estimation
     - Latent Variables
     - Approximate Simulation
     - Diffusion Model
     - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
     - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
     - Structural Risk Minimization
   - References

9. References
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Introduction
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Dimension Reduction
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References
Empirical Risk Minimization

Empirical Risk Minimizer (ERM)

- For any loss $\ell$ and function class $S$,

$$\hat{f} = \arg\min_{f \in S} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i)) = \arg\min_{f \in S} \mathcal{R}_n(f)$$

- Key property:

$$\mathcal{R}_n(\hat{f}) \leq \mathcal{R}_n(f), \forall f \in S$$

- Minimization not always tractable in practice!

- Focus on the $\ell^{0/1}$ case:
  - only algorithm is to try all the functions,
  - not feasible is there are many functions
  - but interesting hindsight!
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

9. References
ERM and PAC Analysis

- Theoretical control of the random (error estimation) term:
  \[ \mathcal{R}(\hat{f}) - \mathcal{R}(f^*_S) \]

**Probably Almost Correct Analysis**

- **Theoretical guarantee** that
  \[ \mathbb{P}\left( \mathcal{R}(\hat{f}) - \mathcal{R}(f^*_S) \leq \epsilon_S(\delta) \right) \geq 1 - \delta \]
  for a suitable \( \epsilon_S(\delta) \geq 0 \).
- Imply:
  - \( \mathbb{P}\left( \mathcal{R}(\hat{f}) - \mathcal{R}(f^*) \leq \mathcal{R}(f^*_S) - \mathcal{R}(f^*) + \epsilon_S(\delta) \right) \geq 1 - \delta \)
  - \( \mathbb{E}\left[ \mathcal{R}(\hat{f}) - \mathcal{R}(f^*_S) \right] \leq \int_0^{+\infty} \delta_S(\epsilon)d\epsilon \)

- The result should hold without any assumption on the law of the error estimation.
A General Decomposition

- By construction:

\[ R(\hat{f}) - R(f^*_S) = R(\hat{f}) - R_n(\hat{f}) + R_n(\hat{f}) - R_n(f^*_S) + R_n(f^*_S) - R(f^*_S) \]
\[ \leq R(\hat{f}) - R_n(\hat{f}) + R_n(f^*_S) - R(f^*_S) \]
\[ \leq \left( R(\hat{f}) - R(f^*_S) \right) - \left( R_n(\hat{f}) - R_n(f^*_S) \right) \]

Four possible upperbounds

- \[ R(\hat{f}) - R(f^*_S) \leq \sup_{f \in S} \left( (R(f) - R(f^*_S)) - (R_n(f) - R_n(f^*_S)) \right) \]
- \[ R(\hat{f}) - R(f^*_S) \leq \sup_{f \in S} \left( R(f) - R_n(f) \right) + \left( R_n(f^*_S) - R(f^*_S) \right) \]
- \[ R(\hat{f}) - R(f^*_S) \leq \sup_{f \in S} \left( R(f) - R_n(f) \right) + \sup_{f \in S} \left( R_n(f) - R(f) \right) \]
- \[ R(\hat{f}) - R(f^*_S) \leq 2 \sup_{f \in S} |R(f) - R_n(f)| \]

- Supremum of centered random variables!
- **Key:** Concentration of each variable...
Risk Bounds

- By construction, for any $f' \in S$,
  $$\mathcal{R}(f') = \mathcal{R}_n(f') + (\mathcal{R}(f') - \mathcal{R}_n(f'))$$

A uniform upper bound for the risk

- Simultaneously $\forall f' \in S$,
  $$\mathcal{R}(f') \leq \mathcal{R}_n(f') + \sup_{f \in S} (\mathcal{R}(f) - \mathcal{R}_n(f))$$

- Supremum of centered random variables!
- **Key:** Concentration of each variable . . .
- Can be interpreted as a justification of the ERM!
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension

9. References
Concentration of the Empirical Loss

- **Empirical loss:**

\[ \mathcal{R}_n(f) = \frac{1}{n} \sum_{i=1}^{n} \ell^{0/1}(Y_i, f(X_i)) \]

**Properties**

- \( \ell^{0/1}(Y_i, f(X_i)) \) are i.i.d. random variables in [0, 1].

**Concentration**

\[
\begin{align*}
\mathbb{P}(\mathcal{R}(f) - \mathcal{R}_n(f) \leq \epsilon) & \geq 1 - e^{-2ne^2} \\
\mathbb{P}(\mathcal{R}_n(f) - \mathcal{R}(f) \leq \epsilon) & \geq 1 - e^{-2ne^2} \\
\mathbb{P}(|\mathcal{R}_n(f) - \mathcal{R}(f)| \leq \epsilon) & \geq 1 - 2e^{-2ne^2}
\end{align*}
\]

- Concentration of sum of bounded independent variables!
- Hoeffding theorem.
- Equiv. to \( \mathbb{P}\left(\mathcal{R}(f) - \mathcal{R}_n(f) \leq \sqrt{\log(1/\delta)/(2n)}\right) \geq 1 - \delta \)
Theorem

Let $Z_i$ be a sequence of ind. centered r.v. supported in $[a_i, b_i]$ then

$$
P \left( \sum_{i=1}^{n} Z_i \geq \epsilon \right) \leq e^{-\sum_{i=1}^{n} \frac{2\epsilon^2}{(b_i - a_i)^2}}$$

Proof ingredients:
- Chernov bounds:
  $$
P \left( \sum_{i=1}^{n} Z_i \geq \epsilon \right) \leq \frac{\mathbb{E} \left[ e^{\lambda \sum_{i=1}^{n} Z_i} \right]}{e^{\lambda \epsilon}} \leq \prod_{i=1}^{n} \frac{\mathbb{E} \left[ e^{\lambda Z_i} \right]}{e^{\lambda \epsilon}}$$

- Exponential moment bounds: $\mathbb{E} \left[ e^{\lambda Z_i} \right] \leq e^{\frac{\lambda^2 (b_i - a_i)^2}{8}}$

- Optimization in $\lambda$

Prop:

$$
\mathbb{E} \left[ e^{\lambda \sum_{i=1}^{n} Z_i} \right] \leq e^{\frac{\lambda^2 \sum_{i=1}^{n} (b_i - a_i)^2}{8}}.
$$
Hoeffding Inequality

Theorem

- Let $Z_i$ be a sequence of independent centered random variables supported in $[a_i, b_i]$ then

$$\mathbb{P}\left(\sum_{i=1}^{n} Z_i \geq \epsilon\right) \leq e^{-\frac{2\epsilon^2}{\sum_{i=1}^{n} (b_i-a_i)^2}}$$

- $Z_i = \frac{1}{n} \left( \mathbb{E}\left[\ell^{0/1}(Y, f(X))\right] - \ell^{0/1}(Y_i, f(X_i)) \right)$

- $\mathbb{E}[Z_i] = 0$ and $Z_i \in \left[\frac{1}{n} \left( \mathbb{E}\left[\ell^{0/1}(Y, f(X))\right] - 1 \right), \frac{1}{n} \mathbb{E}\left[\ell^{0/1}(Y, f(X))\right] \right]$

- Concentration:

$$\mathbb{P}(\mathcal{R}(f) - \mathcal{R}_n(f) \geq \epsilon) \leq e^{-2n\epsilon^2}$$

- By symmetry,

$$\mathbb{P}(\mathcal{R}_n(f) - \mathcal{R}(f) \geq \epsilon) \leq e^{-2n\epsilon^2}$$

- Combining the two yields

$$\mathbb{P}(|\mathcal{R}_n(f) - \mathcal{R}(f)| \geq \epsilon) \leq 2e^{-2n\epsilon^2}$$
Finite Class Case

Concentration

- If $S$ is finite of cardinality $|S|$, 
  \[
  \mathbb{P}
  \left( \sup_{f} (\mathcal{R}(f) - \mathcal{R}_n(f)) \leq \sqrt{\frac{\log |S| + \log(1/\delta)}{2n}} \right) \geq 1 - \delta
  \]
  \[
  \mathbb{P}
  \left( \sup_{f} |\mathcal{R}_n(f) - \mathcal{R}(f)| \leq \sqrt{\frac{\log |S| + \log(1/\delta)}{2n}} \right) \geq 1 - 2\delta
  \]

- Control of the supremum by a quantity depending on the cardinality and the probability parameter $\delta$.
- Simple combination of Hoeffding and a union bound.
Finite Class Case

PAC Bounds

- If $S$ is finite of cardinality $|S|$, with proba greater than $1 - 2\delta$

$$R(\hat{f}) - R(f^*_S) \leq \sqrt{\frac{\log |S| + \log(1/\delta)}{2n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$$

$$\leq 2\sqrt{\frac{\log |S| + \log(1/\delta)}{2n}}$$

- If $S$ is finite of cardinality $|S|$, with proba greater than $1 - \delta$, simultaneously $\forall f' \in S$,

$$R(f') \leq R_n(f') + \sqrt{\frac{\log |S| + \log(1/\delta)}{2n}}$$

$$\leq R_n(f') + \sqrt{\frac{\log |S|}{2n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$$
Finite Class Case

PAC Bounds

- If $S$ is finite of cardinality $|S|$, with proba greater than $1 - 2\delta$

\[
\mathcal{R}(\hat{f}) - \mathcal{R}(f^*_S) \leq \sqrt{\frac{\log |S|}{2n}} + \sqrt{\frac{2\log(1/\delta)}{n}}
\]

- If $S$ is finite of cardinality $|S|$, with proba greater than $1 - \delta$, simultaneously $\forall f' \in S$,

\[
\mathcal{R}(f') \leq \mathcal{R}_n(f') + \sqrt{\frac{\log |S|}{2n}} + \sqrt{\frac{\log(1/\delta)}{2n}}
\]

- Risk increases with the cardinality of $S$.
- Similar issue in cross-validation!
- No direct extension for an infinite $S$...
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supplementary Information

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting
   - References

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

9. References

582
Concentration of the Supremum of Empirical Losses

- Supremum of Empirical losses:
  \[
  \Delta_n(S)(X_1, \ldots, X_n) = \sup_{f \in S} \mathcal{R}(f) - \mathcal{R}_n(f)
  \]
  \[
  = \sup_{f \in S} \left( \mathbb{E} \left[ \ell^{0/1}(Y, f(X)) \right] - \frac{1}{n} \sum_{i=1}^{n} \ell^{0/1}(Y_i, f(X_i)) \right)
  \]

**Properties**

- Bounded difference:
  \[
  |\Delta_n(S)(X_1, \ldots, X_i, \ldots X_n) - \Delta_n(S)(X_1, \ldots X'_i, \ldots, X_n)| \leq 1/n
  \]

**Concentration**

\[
P(\Delta_n(S) - \mathbb{E}[\Delta_n(S)] \leq \epsilon) \geq 1 - e^{-2n\epsilon^2}
\]

- Concentration of bounded difference function.
- Generalization of Hoeffding theorem: McDiarmid Theorem.
**McDiarmid Inequality**

**Bounded difference function**
- \( g : \mathcal{X}^n \to \mathbb{R} \) is a bounded difference function if it exist \( c_i \) such that
  \[
  \forall (X_i)_{i=1}^n, (X'_i)_{i=1}^n \in \mathbb{R}, \quad \left| g(X_1, \ldots, X_i, \ldots, X_n) - g(X_1, \ldots, X'_i, \ldots, X_n) \right| \leq c_i
  \]

**Theorem**
- If \( g \) is a bounded difference function and \( X_i \) are independent random variables then
  \[
  \mathbb{P}(g(X_1, \ldots, X_n) - \mathbb{E}[g(X_1, \ldots, X_n)] \geq \epsilon) \leq e^{-\frac{-2\epsilon^2}{\sum_{i=1}^n c_i^2}}
  \]
  \[
  \mathbb{P}(\mathbb{E}[g(X_1, \ldots, X_n)] - g(X_1, \ldots, X_n) \geq \epsilon) \leq e^{-\frac{-2\epsilon^2}{\sum_{i=1}^n c_i^2}}
  \]

**Proof ingredients:**
- Chernov bounds
- Martingale decomposition...
McDiarmid Inequality

**Theorem**

- If $g$ is a bounded difference function and $X_i$ are independent random variables then

  $$\mathbb{P}(g(X_1, \ldots, X_n) - \mathbb{E}[g(X_1, \ldots, X_n)] \geq \epsilon) \leq e^{\frac{-2\epsilon^2}{\sum_{i=1}^{n} c_i^2}}$$

- Using $g = \Delta_n(S)$ for which $c_i = \frac{1}{n}$ yields immediately

  $$\mathbb{P}(\Delta_n(S) - \mathbb{E}[\Delta_n(S)] \geq \epsilon) \leq e^{\frac{-2\epsilon^2}{\sum_{i=1}^{n} c_i^2}} = e^{-2n\epsilon^2}$$

- We derive then

  $$\mathbb{P}(\Delta_n(S) \geq \mathbb{E}[\Delta_n(S)] + \epsilon) \leq e^{\frac{-2\epsilon^2}{\sum_{i=1}^{n} c_i^2}} = e^{-2n\epsilon^2}$$

- It remains to upperbound

  $$\mathbb{E}[\Delta_n] = \mathbb{E} \left[ \sup_{f \in S} \mathcal{R}(f) - \mathcal{R}_n(f) \right]$$
Rademacher Complexity

**Theorem**
- Let $\sigma_i$ be a sequence of i.i.d. random symmetric Bernoulli variables (Rademacher variables):

$$\mathbb{E} \left[ \sup_{f \in S} (\mathcal{R}(f) - \mathcal{R}_n(f)) \right] \leq 2\mathbb{E} \left[ \sup_{f \in S} \frac{1}{n} \sum_{i=1}^{n} \sigma_i \ell^{0/1}(Y_i, f(X_i)) \right]$$

**Rademacher complexity**
- Let $B \subset \mathbb{R}^n$, the Rademacher complexity of $B$ is defined as

$$R_n(B) = \mathbb{E} \left[ \sup_{b \in B} \frac{1}{n} \sum_{i=1}^{n} \sigma_i b_i \right]$$

- Theorem gives an upper bound of the expectation in terms of the average Rademacher complexity of the random set $B_n(S) = \{(\ell^{0/1}(Y_i, f(X_i)))_{i=1}^{n}, f \in S\}$.

- **Back to finite setting:** This set is at most of cardinality $2^n$. 
Finite Set Rademacher Complexity Bound

**Theorem**

- If $B$ is finite and such that $\forall b \in B, \frac{1}{n} \|b\|_2^2 \leq M^2$, then
  
  $$R_n(B) = \mathbb{E} \left[ \sup_{b \in B} \frac{1}{n} \sum_{i=1}^{n} \sigma_i b_i \right] \leq \sqrt{\frac{2M^2 \log |B|}{n}}$$

- If $B = B_n(S) = \{(\ell^0/1(Y_i, f(X_i)))_{i=1}^n, f \in S\}$, we have $M = 1$ and thus
  
  $$R_n(B) \leq \sqrt{\frac{2 \log |B_n(S)|}{n}}$$

- We obtain immediately
  
  $$\mathbb{E} \left[ \sup_{f \in S} (\mathcal{R}(f) - \mathcal{R}_n(f)) \right] \leq \mathbb{E} \left[ \sqrt{8 \log |B_n(S)|} \right].$$
Finite Set Rademacher Complexity Bound

**Theorem**

- With probability greater than $1 - 2\delta$,
  
  $$\mathcal{R}(\hat{f}) - \mathcal{R}(f^*_S) \leq \mathbb{E} \left[ \sqrt{\frac{8 \log |B_n(S)|}{n}} \right] + \sqrt{\frac{2 \log(1/\delta)}{n}}$$

- With probability greater than $1 - \delta$, simultaneously $\forall f' \in S$
  
  $$\mathcal{R}(f') \leq \mathcal{R}_n(f') + \mathbb{E} \left[ \sqrt{\frac{8 \log |B_n(S)|}{n}} \right] + \sqrt{\frac{\log(1/\delta)}{2n}}$$

- This is a direct consequence of the previous bound.
Finite Set Rademacher Complexity Bound

**Corollary**

- If \( S \) is finite then with probability greater than \( 1 - 2\delta \)

\[
\mathcal{R}(\hat{f}) - \mathcal{R}(f^*_S) \leq \sqrt{\frac{8 \log |S|}{n}} + \sqrt{\frac{2 \log(1/\delta)}{n}}
\]

- If \( S \) is finite then with probability greater than \( 1 - \delta \), simultaneously \( \forall f' \in S \)

\[
\mathcal{R}(f') \leq \mathcal{R}_n(f') + \sqrt{\frac{8 \log |S|}{n}} + \sqrt{\frac{\log(1/\delta)}{2n}}
\]

- It suffices to notice that

\[
|B_n(S)| = |\{(\ell^0/1(Y_i, f(X_i)))_{i=1}^n, f \in S\}| \leq |S|
\]
Finite Set Rademacher Complexity Bound

- Same result with Hoeffding but with **better** constants!

\[
\mathcal{R}(\hat{f}) - \mathcal{R}(f_S^*) \leq \sqrt{\frac{\log |S|}{2n}} + \sqrt{\frac{2 \log(1/\delta)}{n}}
\]

\[
\mathcal{R}(f') \leq \mathcal{R}_n(f') + \sqrt{\frac{\log |S|}{2n}} + \sqrt{\frac{\log(1/\delta)}{2n}}
\]

- Difference due to the **crude** upperbound of

\[
\mathbb{E} \left[ \sup_{f \in S} (\mathcal{R}(f) - \mathcal{R}_n(f)) \right]
\]

- **Why bother?:** We do not have to assume that \( S \) is finite!

\[
|B_n(S)| \leq 2^n
\]
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Gradient Descent Step
   - Non-Convex Setting

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization

9. References
## Back to the Bound

### Theorem

\[ \mathbb{E} \left[ \sup_{f \in \mathcal{S}} (\mathcal{R}(f) - \mathcal{R}_n(f)) \right] \leq \mathbb{E} \left[ \sqrt{\frac{8 \log |B_n(S)|}{n}} \right] \]

- **Key quantity:** \( \mathbb{E} \left[ \sqrt{\frac{8 \log |B_n(S)|}{n}} \right] \)
- **Hard to control due to its structure!**

### A first data dependent upperbound

\[ \mathbb{E} \left[ \sqrt{\frac{8 \log |B_n(S)|}{n}} \right] \leq \sqrt{\frac{8 \log \mathbb{E}[|B_n(S)|]}{n}} \]  

**(Jensen)**

- **Depends on the unknown** \( P \)!
Shattering Coefficient

Shattering Coefficient (or Growth Function)

- The shattering coefficient of the class $S$, $s(S,n)$, is defined as
  \[
s(S, n) = \sup \{ |\{ (\ell(Y_i, f(X_i)))_{i=1}^n, f \in S \} | \}
  \]
  \[
  \text{subject to } ((X_1, Y_1), \ldots, (X_n, Y_n)) \in \mathcal{X} \times \{-1,1\}^n
  \]

- By construction, $|B_n(S)| \leq s(S, n) \leq \min(2^n, |S|)$.

A data independent upperbound

\[
\mathbb{E} \left[ \sqrt{\frac{8 \log |B_n(S)|}{n}} \right] \leq \sqrt{\frac{8 \log s(S, n)}{n}}
\]
Theorem

- With probability greater than $1 - 2\delta$, 
  
  $$\mathcal{R}(\hat{f}) - \mathcal{R}(f^*_S) \leq \sqrt{\frac{8 \log s(S, n)}{n}} + \sqrt{\frac{2 \log(1/\delta)}{n}}$$

- With probability greater than $1 - \delta$, simultaneously $\forall f' \in S$, 
  
  $$\mathcal{R}(f') \leq \mathcal{R}_n(f') + \sqrt{\frac{8 \log s(S, n)}{n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$$

- Depends only on the class $S$!
Vapnik-Chervonenkis Dimension

**VC Dimension**
- The VC dimension $d_{VC}$ of $S$ is defined as the largest integer $d$ such that
  
  $$s(S, d) = 2^d$$

- The VC dimension can be infinite!

**VC Dimension and Dimension**
- **Prop:** If $\text{span}(S)$ corresponds to the sign of functions in a linear space of dimension $d$ then $d_{VC} \leq d$.

- VC dimension similar to the usual dimension.
VC Dimension and Sauer’s Lemma

**Sauer’s Lemma**

- If the VC dimension $d_{VC}$ of $S$ is finite

$$s(S, n) \leq \begin{cases} 2^n & \text{if } n \leq d_{VC} \\ \left( \frac{en}{d_{VC}} \right)^{d_{VC}} & \text{if } n > d_{VC} \end{cases}$$

- **Cor.:** $\log s(S, n) \leq d_{VC} \log \left( \frac{en}{d_{VC}} \right)$ if $n > d_{VC}$. 
### VC Dimension and PAC Bounds

<table>
<thead>
<tr>
<th>PAC Bounds</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>If</strong> $S$ is of VC dimension $d_{VC}$ then if $n &gt; d_{VC}$</td>
</tr>
<tr>
<td><strong>With probability greater than</strong> $1 - 2\delta$,</td>
</tr>
<tr>
<td>$R(\hat{f}) - R(f^*<em>S) \leq \sqrt{\frac{8d</em>{VC} \log \left( \frac{en}{d_{VC}} \right)}{n}} + \sqrt{\frac{2 \log(1/\delta)}{n}}$</td>
</tr>
<tr>
<td><strong>With probability greater than</strong> $1 - \delta$, simultaneously $\forall f' \in S$,</td>
</tr>
<tr>
<td>$R(f') \leq R_n(f') + \sqrt{\frac{8d_{VC} \log \left( \frac{en}{d_{VC}} \right)}{n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$</td>
</tr>
</tbody>
</table>

**Rk:** If $d_{VC} = +\infty$ no uniform PAC bounds exists!
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent
   - Non-Convex Setting
   - References

5 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
   - Bootstrap and Bagging
   - Randomized Rules and Random Forests
   - Boosting
   - AdaBoost as a Greedy Scheme
   - Boosting
   - Ensemble Methods
   - References

7 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
   - Clustering
   - Dimensionality Curse
   - Dimension Reduction
   - Generative Modeling
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?
   - Clustering
   - Prototype Approaches
   - Contiguity Approaches
   - Agglomerative Approaches
   - Other Approaches
   - Scalability
   - Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network
   - Applications to Text
   - References

8 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
   - References

9 References

598
**PAC Bounds**

- Let $\pi_f > 0$ such that $\sum_{f \in S} \pi_f = 1$
- With proba greater than $1 - 2\delta$,
  \[ R(\hat{f}) - R(f^*_{S}) \leq \sqrt{\frac{\log(1/\pi_f)}{2n}} + \sqrt{\frac{2\log(1/\delta)}{n}} \]
- With proba greater than $1 - \delta$, simultaneously $\forall f' \in S$,
  \[ R(f') \leq R_n(f') + \sqrt{\frac{\log(1/\pi_f)}{2n}} + \sqrt{\frac{\log(1/\delta)}{2n}} \]

- Very similar proof than the uniform one!
- Much more interesting idea when combined with several models...
Models, Non Uniform Risk Bounds and SRM

- Assume we have a countable collection of set \((S_m)_{m \in \mathcal{M}}\) and let \(\pi_m\) be such that \(\sum_{m \in \mathcal{M}} \pi_m = 1\).

**Non Uniform Risk Bound**

- With probability \(1 - \delta\), simultaneously for all \(m \in \mathcal{M}\) and all \(f \in S_m\),
  
  \[
  R(f) \leq R_n(f) + \mathbb{E} \left[ \sqrt{\frac{8 \log |B_n(S_m)|}{n}} \right] + \sqrt{\frac{\log(1/\pi_m)}{2n}} + \sqrt{\frac{\log(1/\delta)}{2n}}
  \]

**Structural Risk Minimization**

- Choose \(\hat{f}\) as the minimizer over \(m \in \mathcal{M}\) and \(f \in S_m\) of
  
  \[
  R_n(f) + \mathbb{E} \left[ \sqrt{\frac{8 \log |B_n(S_m)|}{n}} \right] + \sqrt{\frac{\log(1/\pi_m)}{2n}}
  \]

- Mimics the minimization of the integrated risk!
SRM and PAC Bound

PAC Bound

- If \( \hat{f} \) is the SRM minimizer then with probability \( 1 - 2\delta \),
  \[
  \mathcal{R}(\hat{f}) \leq \inf_{m \in \mathcal{M}} \inf_{f \in \mathcal{S}_m} \left( \mathcal{R}(f) + \mathbb{E} \left[ \sqrt{\frac{8 \log |B_n(S_m)|}{n}} \right] + \sqrt{\frac{\log(1/\pi_m)}{2n}} \right) + \sqrt{\frac{2 \log(1/\delta)}{n}}
  \]

- The SRM minimizer balances the risk \( \mathcal{R}(f) \) and the upper bound on the estimation error \( \mathbb{E} \left[ \sqrt{\frac{8 \log |B_n(S_m)|}{n}} \right] + \sqrt{\frac{\log(1/\pi_m)}{2n}} \).

- \( \mathbb{E} \left[ \sqrt{\frac{8 \log |B_n(S_m)|}{n}} \right] \) can be replaced by an upper bound (for instance a VC based one)…
Outline

1 Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2 ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3 ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4 Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5 Gradient Descent Step
   - Non-Convex Setting
   - References

6 ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Regularization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

7 ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
   - Ensemble Methods
   - References

8 Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

9 Dimension Reduction
   - Simplification
   - Reconstruction Error
   - Relationship Preservation
   - Comparing Methods?

10 Clustering
    - Prototype Approaches
    - Contiguity Approaches
    - Agglomerative Approaches
    - Other Approaches
    - Scalability

11 Generative Modeling
   - (Plain) Parametric Density Estimation
   - Latent Variables
   - Approximate Simulation
   - Diffusion Model
   - Generative Adversarial Network

12 Applications to Text
   - References

13 Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - Supervised Learning
   - Empirical Risk Minimization
     - Empirical Risk Minimization
     - ERM and PAC Analysis
     - Hoeffding and Finite Class
     - McDiarmid and Rademacher Complexity
     - VC Dimension
   - Structural Risk Minimization
   - References

14 References
Ch. Giraud.  
*Introduction to High-Dimensional Statistics.*  
CRC Press, 2014

M. Mohri, A. Rostamizadeh, and A. Talwalkar.  
*Foundations of Machine Learning (2nd ed.)*  
MIT Press, 2018

S. Shalev-Shwartz and S. Ben-David.  
*Understanding Machine Learning.*  
Cambridge University Press, 2014
Creative Commons Attribution-ShareAlike (CC BY-SA 4.0)

- You are free to:
  - **Share**: copy and redistribute the material in any medium or format
  - **Adapt**: remix, transform, and build upon the material for any purpose, even commercially.

- Under the following terms:
  - **Attribution**: You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - **ShareAlike**: If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - **No additional restrictions**: You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Contributors

- **Main contributor**: E. Le Pennec
Outline

1. Statistical Learning: Introduction, Setting and Risk Estimation
   - Introduction
   - Machine Learning
   - Supervised Learning
   - Risk Estimation and Model Selection
   - Cross Validation and Test
   - References

2. ML Methods: Probabilistic Point of View
   - Motivation
   - Supervised Learning
   - A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
   - Model Selection
   - Penalization

3. ML Methods: Optimization Point of View
   - Supervised Learning
   - Optimization Point of View
   - SVM
   - Penalization
   - Cross Validation and Weights

4. Optimization: Gradient Descent Algorithms
   - Introduction
   - Gradient Descent
   - Proximal Descent
   - Coordinate Descent
   - Gradient Descent Acceleration
   - Stochastic Gradient Descent

5. ML Methods: Neural Networks and Deep Learning
   - Introduction
   - From Logistic Regression to NN
   - NN Optimization
   - NN Initialization
   - Image and CNN
   - Text, Recurrent Neural Networks and Transformers
   - NN Architecture
   - References

6. ML Methods: Trees and Ensemble Methods
   - Trees
   - Bagging and Random Forests
     - Bootstrap and Bagging
     - Randomized Rules and Random Forests
   - Boosting
     - AdaBoost as a Greedy Scheme
     - Boosting
     - Ensemble Methods
     - References

7. Unsupervised Learning: Dimension Reduction and Clustering
   - Unsupervised Learning?
   - A First Glimpse
     - Clustering
     - Dimensionality Curse
     - Dimension Reduction
     - Generative Modeling

8. Statistical Learning: PAC-Bayesian Approach and Complexity Theory
   - References

9. References

M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning (2nd ed.)* MIT Press, 2018


Ch. Giraud. *Introduction to High-Dimensional Statistics.* CRC Press, 2014

Creative Commons Attribution-ShareAlike (CC BY-SA 4.0)

- You are free to:
  - **Share**: copy and redistribute the material in any medium or format
  - **Adapt**: remix, transform, and build upon the material for any purpose, even commercially.

- Under the following terms:
  - **Attribution**: You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - **ShareAlike**: If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - **No additional restrictions**: You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Contributors

- **Main contributor**: E. Le Pennec