ML Methods

Erwan Le Pennec

Erwan.Le-Pennec@polytechnique.edu

Winter 2023-2024

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

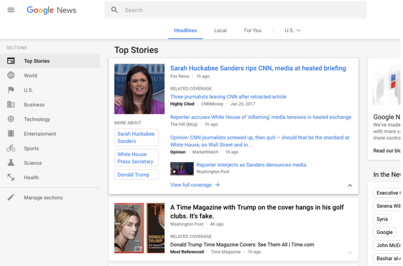
Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Machine Learning



Top Stories

Sarah Huckabee Sanders rips CNN, media at heated briefing
Fox News · 1h ago

RELATED COVERAGE

Three journalists leaving CNN after retracted article
Highly Cited · CNNMoney · Jun 26, 2017

Reporter accuses White House of 'inflaming' media tensions in heated exchange
The Hill (blog) · 1h ago

Opinion: CNN journalists screwed up, then quit — should that be the standard at White House, on Wall Street and in ...
Opinion · MarketWatch · 1h ago

Reporter interjects as Sanders denounces media
Washington Post

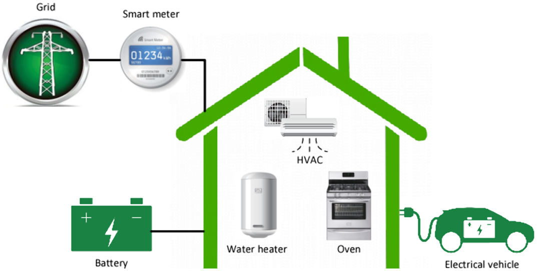
A Time Magazine with Trump on the cover hangs in his golf clubs. It's fake.
Washington Post · 4h ago

RELATED COVERAGE

Donald Trump Time Magazine Covers: See Them All | Time.com
Most Referenced · Time Magazine · 1h ago

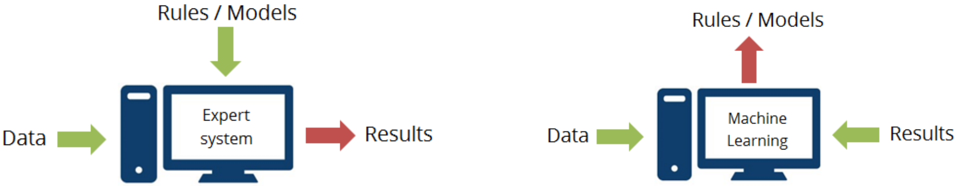
Google N
We've made
with more c
more contro
Read our bl

In the Ne



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

Machine Learning



Source: Council of Europe

The *classical* definition of 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.

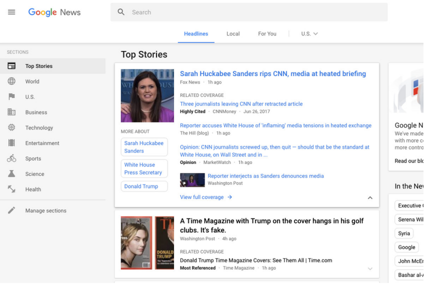
Object Detection

Source: MyCarDoesWhat.org

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



Top Stories

Sarah Huckabee Sanders rips CNN, media at heated briefing
Fox News · 1h ago

RELATED COVERAGE

Three journalists leaving CNN after retracted article
Highly Cited · CNNMoney · Jun 26, 2017

Reporter accuses White House of 'inflaming' media tensions in heated exchange
The Hill (blog) · 1h ago

Opinion: CNN journalists screwed up, then quit — should that be the standard at White House, on Wall Street and in ...
Opinion · MarketWatch · 1h ago

Reporter interjects as Sanders denounces media
Washington Post

View full coverage →

MORE ABOUT

Sarah Huckabee Sanders

White House Press Secretary

Donald Trump

A Time Magazine with Trump on the cover hangs in his golf clubs. It's fake.
Washington Post · 4h ago

RELATED COVERAGE

Donald Trump Time Magazine Covers: See Them All | Time.com
Most Referenced · Time Magazine · 1h ago

Google N
We've made
with more c
more contro

Read our bl

In the Ne

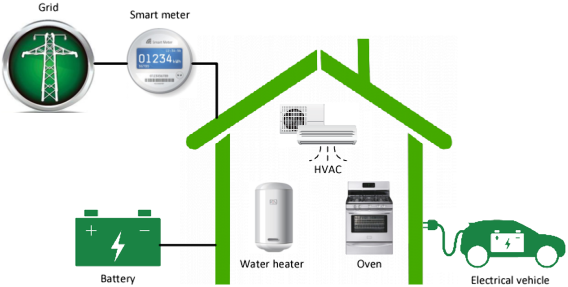
Executive
Serena Wil
Syria
Google
John McEr
Bashar al-

An article clustering algorithm:

- **Task**: group articles corresponding to the same news
- **Performance**: quality of the clusters
- **Experience**: set of articles

Source: theverge.com

Smart Grid Controler

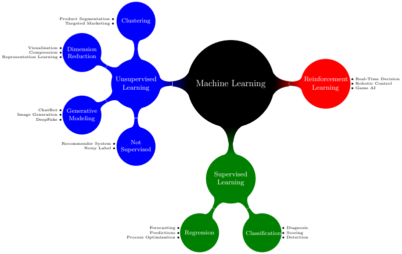


A controler in its sensors in a home smart grid:

- **Task**: control the devices in real-time
- **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 (unique) Ground Truth)

Supervised Learning

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

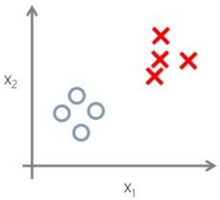
Reinforcement Learning

- **Task:** Actions
- **Performance:** Total reward
- **Experience:** Reward from env. (Interact. with env.)

- **Timing:** Offline/Batch (learning from past data) vs Online (continuous learning)

DR: Dimension Reduction

Supervised and Unsupervised

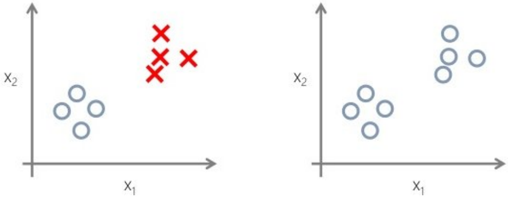


Source: KDnuggets

Supervised Learning (Imitation)

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

Supervised and Unsupervised



Source: KDnuggets

Supervised Learning (Imitation)

- **Goal:** Learn a function f predicting a variable Y from an individual $\underline{X}$.
- **Data:** Learning set with labeled examples $(\underline{X}_i, Y_i)$

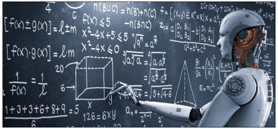
- **Assumption:** Future data behaves as past data!
- **Predicting is not explaining!**

Unsupervised Learning (Structure Discovery)

- **Goal:** Discover/use a structure of a set of individuals $(\underline{X}_i)$.
- **Data:** Learning set with unlabeled examples $(\underline{X}_i)$ (or variations. . .)

- Unsupervised learning is not a well-posed setting. . .

Machine Can and Cannot



Machine Can

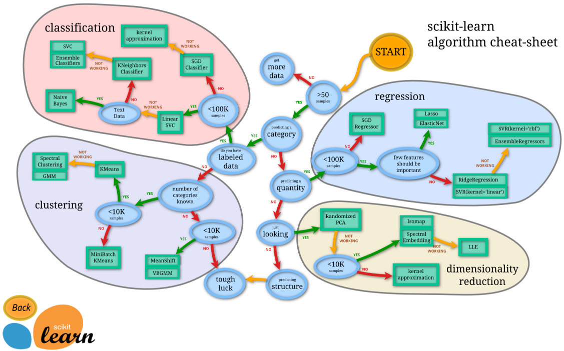
- Forecast (Prediction using the past)
- Detect expected changes
- Memorize/Reproduce/Imitate
- Take decisions very quickly
- Generate a lot of variations
- Learn from huge dataset
- Optimize a single task
- Help (or replace) some human beings

Machine Cannot

- Predict something never seen before
- Detect any new behaviour
- Create something brand new
- Understand the world
- Plan by reasoning
- Get smart really fast
- Go beyond their task
- Replace (or kill) all human beings

- A lot of progresses but still very far from the *singularity*...

Machine Learning

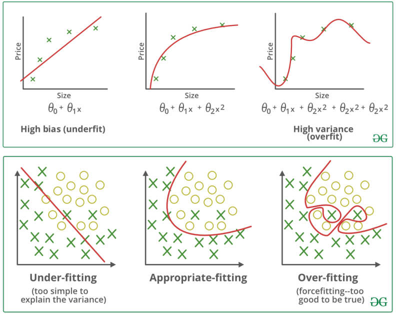


Machine Learning Methods

- Huge catalog of methods,
- Need to define the performance,
- Numerous tricks: feature design, performance estimation. . .

Source: scikit-learn.org

Under and Over Fitting

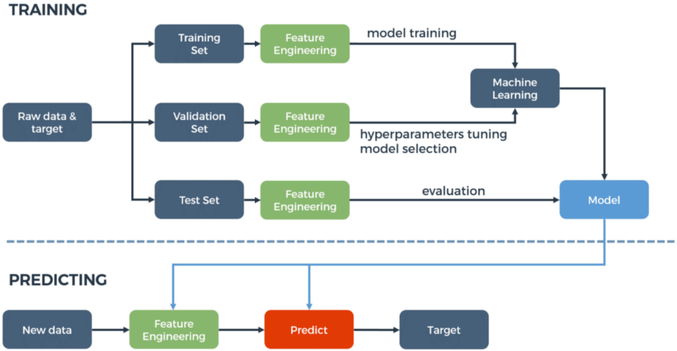

Price
Size
$\theta_0 + \theta_1 x$
High bias (underfit)
Price
Size
$\theta_0 + \theta_1 x + \theta_2 x^2$
Price
Size
$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_2 x^2 + \theta_2 x^2$
High variance (overfit)
Under-fitting
(too simple to explain the variance)
Appropriate-fitting
Over-fitting
(forcefitting--too good to be true)

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.

Source: geeksforgeeks.com

Machine Learning Pipeline



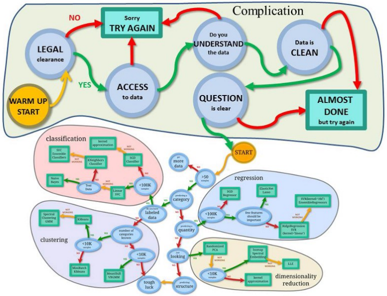
Learning pipeline

- Test and compare models.

- Deployment pipeline is different!

Source: CDiscount

Data Science ≠ Machine Learning



Main DS difficulties

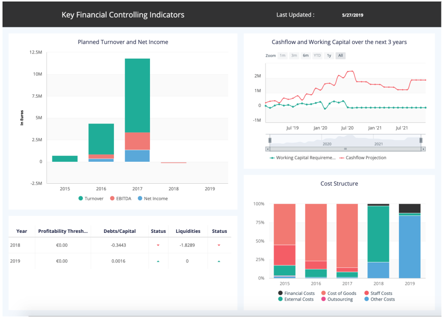
- Figuring out the problem,
- Formalizing it,
- Storing and accessing the data,
- Deploying the solution,
- Not (always) the Machine Learning part!

Source: Ch. Bourguignat

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Monthly KPI Dashboard



Key Financial Controlling Indicators

Last Updated : 5/27/2019

Year	Profitability Thresh...	Debts/Capital	Status	Liquidities	Status
2018	€0.00	-0.3443	▼	-1.8289	▼
2019	€0.00	0.0016	▲	0	▲

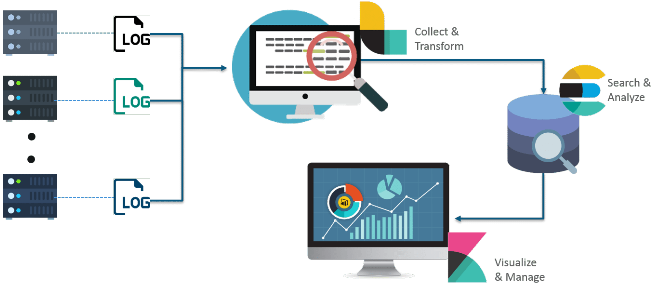
Monthly KPI Dashboard

- Using financial data to display important KPI for top managers every month in a slide
- Automation to guaranty the quality of the results.

Source: decisyon

KPI: Key Performance Index

Realtime Log Dashboard



Realtime Log Dashboard

- Use log data to show the state of a system to IT in real-time using on-premise tools.
- Automation to handle the huge volumetry.

Source: edureka!

IT: Information Technology

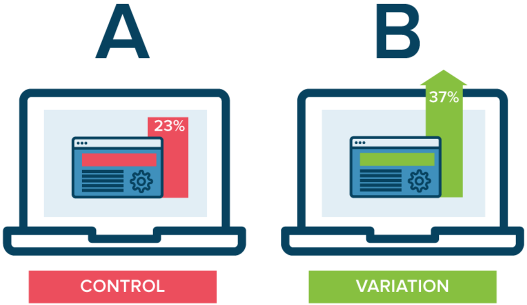
On-demand Legal Document Generation

On-demand Legal Document Generation

- Use raw data to legal document template for a lawyer on-demand using a local database.
- First draft to be edited by the lawyer.

Source: Amicus Attorney

AB Testing

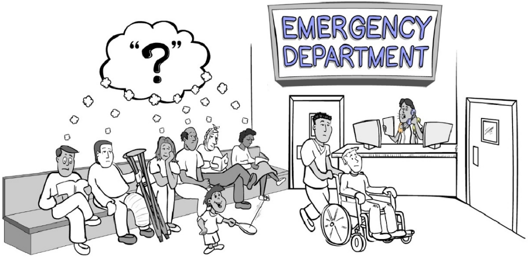


Source: Optimizely

AB Testing

- Using customer journet to help marketing decides between two versions of a website
- Automation to guaranty the accuracy of the results.

ER Waiting Time Prediction



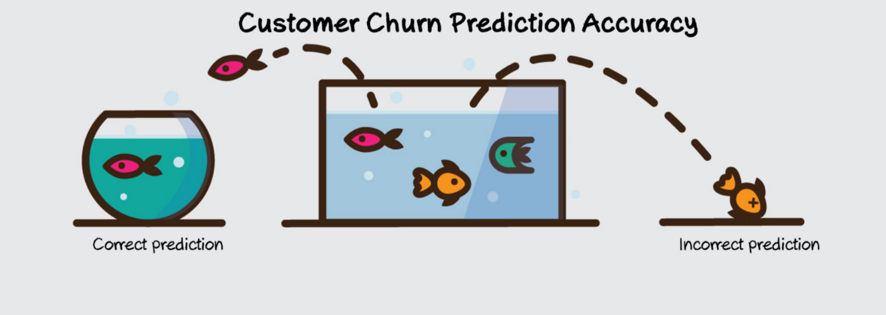
Source: M. Evans

Real-Time ER Waiting Time Prediction

- Use patient data to provide in real-time an estimate of the remaining waiting time to the ER patient.
- Tool helping to bear the wait.

ER: Emergency Room

Weekly Churn Prediction

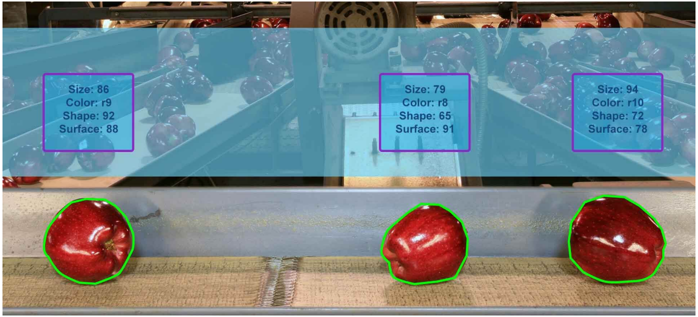


Weekly Churn Prediction

- Using consumer characteristics and history to give a churn score to the marketing every week using the cloud.
- Automation to scale to the volumetry but no strategy recommendation.

Source: Parthasarathy

Realtime Automatic Fruit Sorting



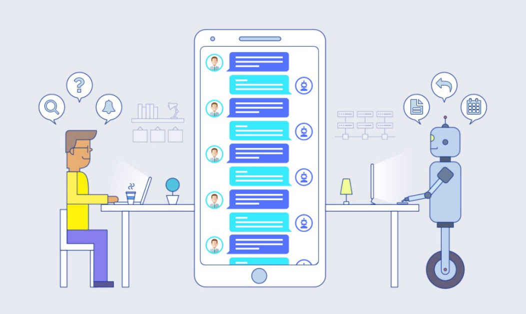
Source: BitRefine

Realtime Automatic Fruit Sorting

- Using camera to sort fruits in a plant in realtime using local computers with GPU.
- Automation to reduce cost.

GPU: Graphical Processing Unit

Realtime Chatbot

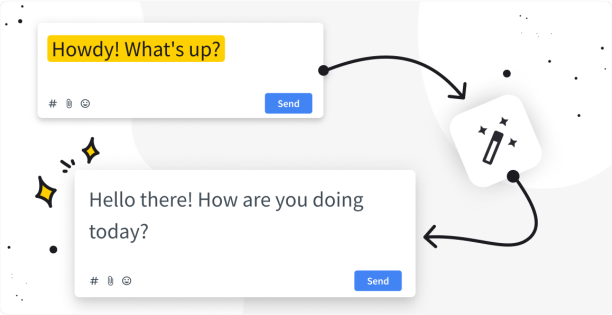


Realtime Chatbot

- Use previous interactions to predict answer to a consumer question in real-time using the cloud.
- Reduce human interaction cost.

Source: M'Bufung

Writing Assistant



Howdy! What's up?

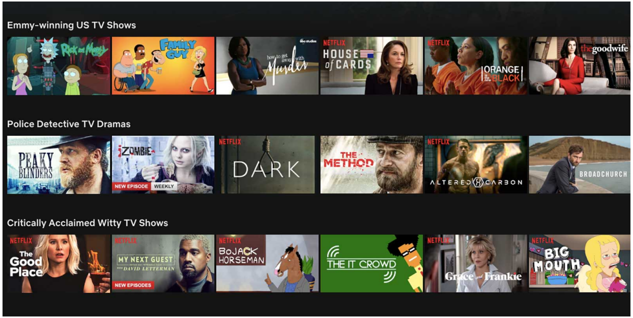
Hello there! How are you doing today?

Writing Assistant

- Enhance a text using AI in a communication system.
- Ease writing steps.

Source: LiveChat

Recommender System

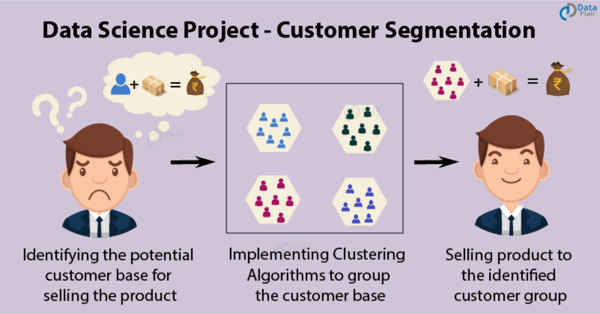


Video Recommender System

- Use client history to suggest in real-time interesting videos for the current user.
- Keep its users.

Source: topbots.com

Customer Segmentation

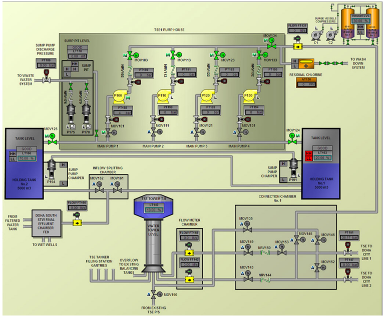

Data Science Project - Customer Segmentation
Data Flair
+
=
Identifying the potential customer base for selling the product
Implementing Clustering Algorithms to group the customer base
Selling product to the identified customer group

Customer Segmentation

- Use customer data to suggest homogeneous groups to the marketing each year.
- Easier to think in term of groups than individuals

Source: Data Flai

Realtime Anomaly Detection



Source: Wikipedia

Realtime Anomaly Detection

- Use production data to detect anomalies in a plant in real-time on a Scada system.
- Reduce failure cost.

Scada: Supervisory Control And Data Acquisition

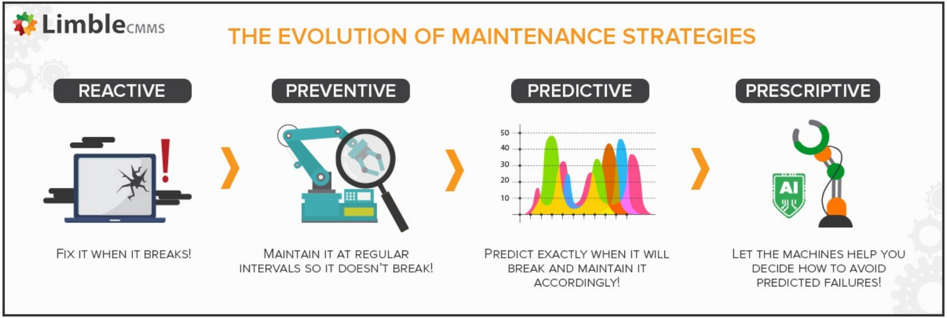
On-demand Fraud Detection

Source: MoneyKama

On-demand Fraud Detection

- Use claim and client data to detect fraud for an insurer on-demand using on-premise resources
- First automated pass on the claims.

Prescriptive Maintenance



THE EVOLUTION OF MAINTENANCE STRATEGIES

REACTIVE › PREVENTIVE › PREDICTIVE › PRESCRIPTIVE

FIX IT WHEN IT BREAKS!

MAINTAIN IT AT REGULAR INTERVALS SO IT DOESN'T BREAK!

PREDICT EXACTLY WHEN IT WILL BREAK AND MAINTAIN IT ACCORDINGLY!

LET THE MACHINES HELP YOU DECIDE HOW TO AVOID PREDICTED FAILURES!

Source: Limble

Prescriptive Maintenance (Not yet available. . .)

- Use data to devise and apply the best maintenance plan in a plant using IOT.
- Reduce maintenance cost.

IOT: Internet of Things

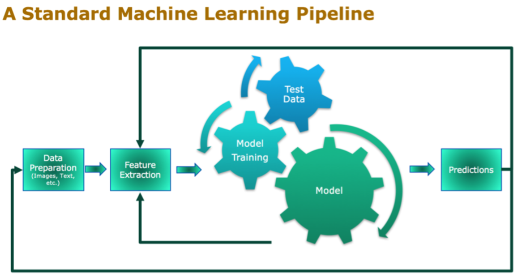
Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

What is a Method?

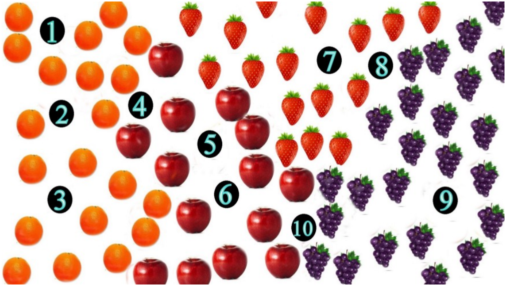

A Standard Machine Learning Pipeline
Data Preparation (Images, Text, etc.)
Feature Extraction
Test Data
Model Training
Model
Predictions

A Learning Method

- Formula/Algorithm allowing to make predictions
- Algorithm allowing to chose this formula/algorithm
- Data preprocessing (cleansing, coding. . .)
- Optimization criterion for the choice!

Source: Microsoft

Simple Approach: Similarity

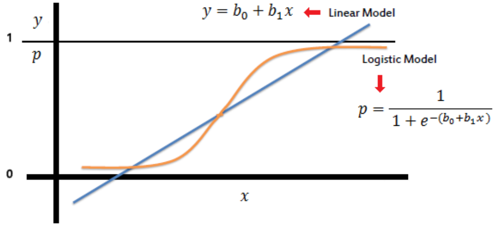


Similarity

- Imitate the answer to give by mixing answers to similar questions (**k nearest neighbors**)
- Require to search for those similar questions for each request
- Not always very efficient but fast to build (less to use. . .)
- Easy to understand and rather stable

Source: Analytics Vidhya

Simple Formula: Linear Method

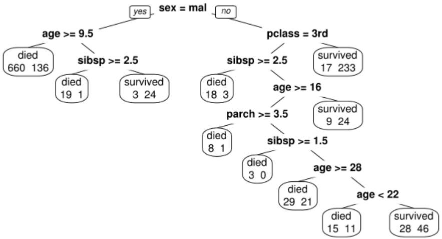


Linear Method

- Simple formula: $a_0 + a_1 X^{(1)} + \cdots + a_d X^{(d)}$
- Imitate the answer to give (**linear regression**) or a transformation of the conditional probability of the category (**logistic regression**)
- Numerous variations on the parameter optimization (**regularization**, **SVM**, ...)
- Pretty efficient and fast to build
- Easy to understand and rather stable

Source: J. Gomila

Simple Algorithm: Tree

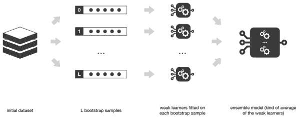


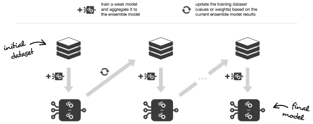
Tree

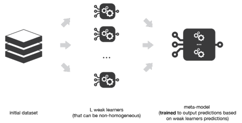
- Construction of a **decision tree**
- Impossible to really optimize but good tree can be obtained
- Not always very efficient but very quick to build
- Very easy to understand but not really stable

Source: M. Cliff

Combing Simple Things: Ensemble


0
1
L
...
...
initial dataset
L bootstrap samples
weak learners fitted on each bootstrap sample
ensemble model (kind of average of the weak learners)


train a weak model and aggregate it to the ensemble model
update the training dataset (values or weights) based on the current ensemble model results
initial dataset
...
final model

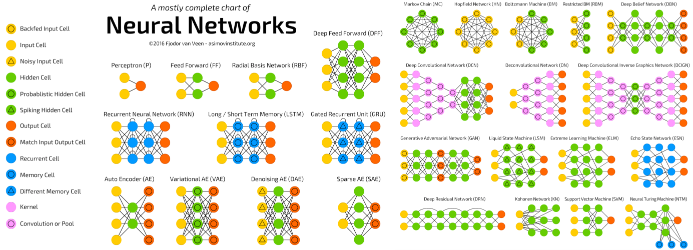

...
initial dataset
L weak learners (that can be non-homogeneous)
meta-model (**trained** to output predictions based on weak learners predictions)

Ensemble Methods

- Strategy:
  - **Bagging:** construction of variations in parallel and averaging (**random forest**)
  - **Boosting:** construction of sequential improvements (**XGBoost**, **Lightgbm**)
  - **Stacking:** Use of a first set of predictors as features
- Very good performance for structured data but quite slow to build
- Stable but hard to understand

Source: J. Rocca

Chain Simple Things: Deep Learning


A mostly complete chart of
Neural Networks
©2016 Fjodor van Veen - asimovinstitute.org
Backfed Input Cell
Input Cell
Noisy Input Cell
Hidden Cell
Probablistic Hidden Cell
Spiking Hidden Cell
Output Cell
Match Input Output Cell
Recurrent Cell
Memory Cell
Different Memory Cell
Kernel
Convolution or Pool
Perceptron (P)
Feed Forward (FF)
Radial Basis Network (RBF)
Deep Feed Forward (DFF)
Recurrent Neural Network (RNN)
Long / Short Term Memory (LSTM)
Gated Recurrent Unit (GRU)
Auto Encoder (AE)
Variational AE (VAE)
Denoising AE (DAE)
Sparse AE (SAE)
Markov Chain (MC)
Hopfield Network (HN)
Boltzmann Machine (BM)
Restricted BM (RBM)
Deep Belief Network (DBN)
Deep Convolutional Network (DCN)
Deconvolutional Network (DN)
Deep Convolutional Inverse Graphics Network (DCIGN)
Generative Adversarial Network (GAN)
Liquid State Machine (LSM)
Extreme Learning Machine (ELM)
Echo State Network (ESN)
Deep Residual Network (DRN)
Kohonen Network (KN)
Support Vector Machine (SVM)
Neural Turing Machine (NTM)

Deep Learning

- Chain of simple formulae (**Neural Network**)
- Joint optimization
- Very good performance for unstructured data but slow to build
- Mildly stable and very hard to understand

Source: Asimov Institute

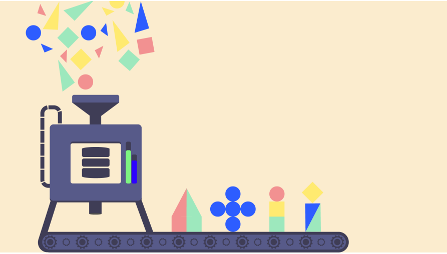
Methods: Pros and Cons

Method	Performance	Training Speed	Inf. Speed	Stability	Interpretability
Similarity	-	$\emptyset$	−	+	+
Linear	+	++	++	++	+
Tree	-	++	++	-	++
Ensemble	++	-	+	++	-
Deep	++	−	-	-	−

Take Away Message

- No unanimously best solution
- Impossible to guess which method is going to be the best!
- A good practice is to always try a linear method as well as an ensemble one for structured data or deep one for unstructured data

Preprocessing



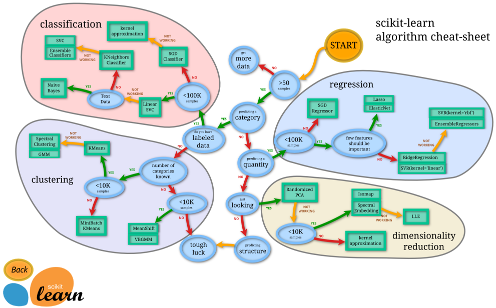
Source: inovex

Preprocessing

- Art of creating sophisticated representations of initial data
- Key for good performances
- Examples: individual transformation, variable combination, category (and text) coding. . .

- **Important part of the learning method**

Methods/Models in Machine Learning

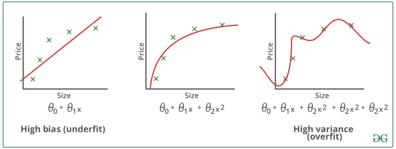


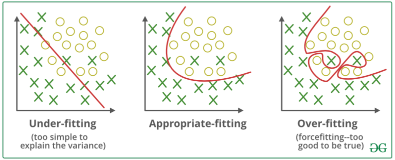
ML Methods

- Huge catalog of methods,
- Need to define the performance,
- Need to represent well the data
- Need to choose the **best** method yielding a good model

Source: scikit-learn.org

Under and Over Fitting



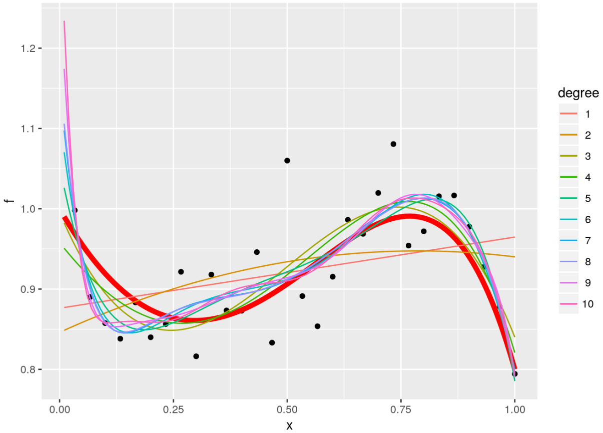


Source: geeksforgeeks.com

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.

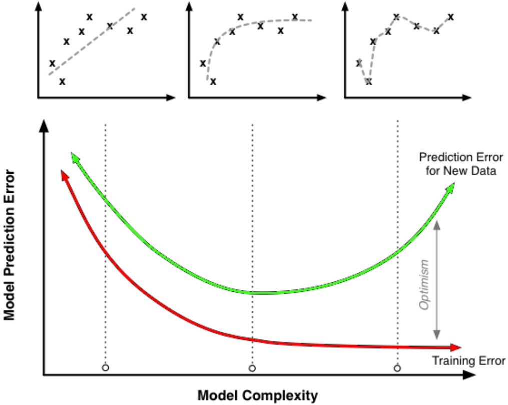
Which Method to Use?



Competition between several polynomial models.

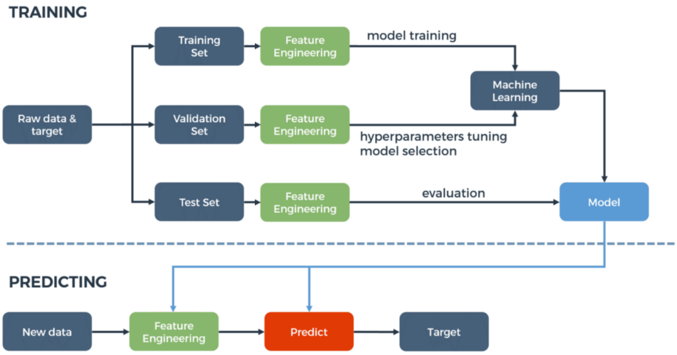
- Toy model where everything is known.

Over-fitting, Under-fitting and Complexity



Source: A. Ng

ML Pipeline



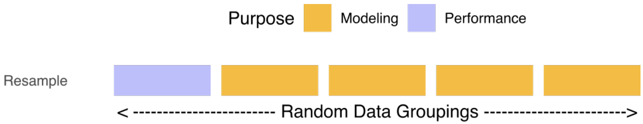
Source: CDiscount

Learning pipeline

- Test and compare models.

- Deployment pipeline is different!

Cross Validation Principle

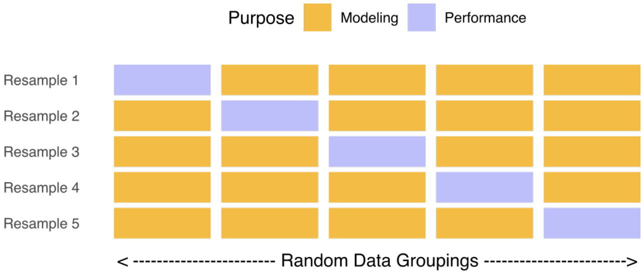


Purpose 🟨 Modeling 🟪 Performance

Resample	🟪	🟨	🟨	🟨	🟨

< ----------------------- Random Data Groupings ----------------------->

- Train a model and check its quality on diffent pieces of the data.



Purpose 🟨 Modeling 🟪 Performance

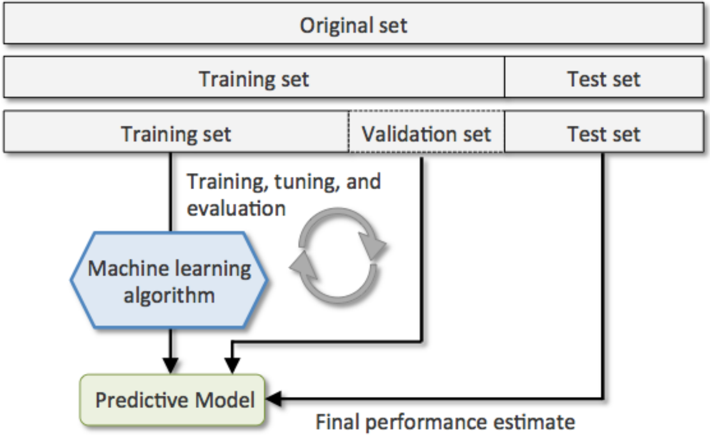
Resample 1	🟪	🟨	🟨	🟨	🟨
Resample 2	🟨	🟪	🟨	🟨	🟨
Resample 3	🟨	🟨	🟪	🟨	🟨
Resample 4	🟨	🟨	🟨	🟪	🟨
Resample 5	🟨	🟨	🟨	🟨	🟪

< ----------------------- Random Data Groupings ----------------------->

- Check the quality of a method by repeating the previous approach.
- **Beware:** a different predictor is learnt for each split.

Source: M. Kühn

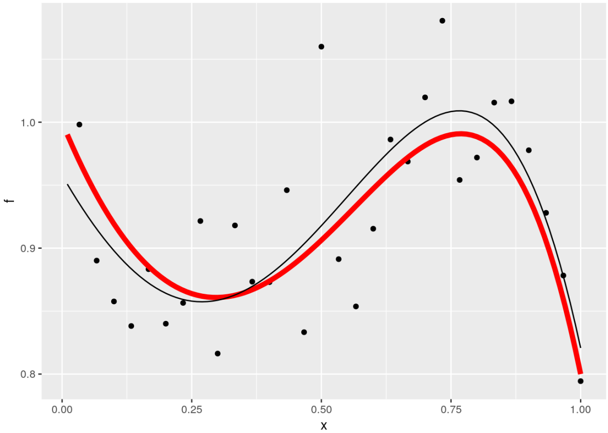
The Full Cross Validation Scheme


Original set
Training set
Test set
Training set
Validation set
Test set
Training, tuning, and evaluation
Machine learning algorithm
Predictive Model
Final performance estimate

- Most important part of machine learning.
- Automatic choice of model possible by (intelligent ?) exploration. . .

Source: Shan-Hung Wu & DataLab

Best Polynomial



Competition results

- The true model is not the winner!

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Interpretation?

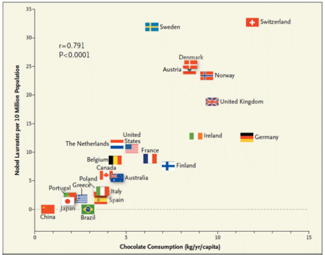


Figure 1. Correlation between Countries' Annual Per Capita Chocolate Consumption and the Number of Nobel Laureates per 10 Million Population.

NEW ENGLAND JOURNAL OF MEDICINE

Nombre de prix Nobel par dix millions d'habitants en fonction de la consommation nationale de chocolat en kilogrammes par personne et par an.

Image : Franz H. Messerli, *The New England Journal of Medicine* **367**(16) (2012), p. 1562-1564

Is this that easy?

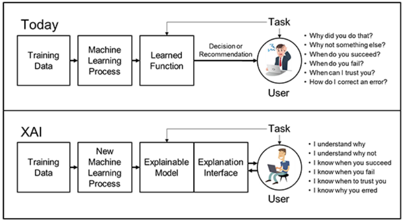
- Simple formula setting:

$$Y \simeq f(X) = a_0 + a_1 X^{(1)} + a_2 X^{(2)} + \cdots + a_d X^{(d)}$$

- Beware of the interpretation!
- Everything being equal... Correlation is not causality...

Source: F. Messerli

Interpretability



Today

Training Data → Machine Learning Process → Learned Function → Decision or Recommendation → User

Task

- Why did you do that?
- Why not something else?
- When do you succeed?
- When do you fail?
- When can I trust you?
- How do I correct an error?

XAI

Training Data → New Machine Learning Process → Explainable Model | Explanation Interface ⇄ User

Task

- I understand why
- I understand why not
- I know when you succeed
- I know when you fail
- I know when to trust you
- I know why you erred

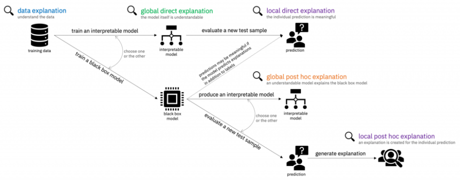
Intepretability or Explainability

- Interpretability: possibility to give a causal aspect to the formula.
- Explainability: possibility to find the variables having an effect on the decision and their effect.

- Explainability is much easier than interpretability.
- Additional constraints that may limit performances.
- Transparency (on the datasets, the criterion optimized and the algorithms) yields already a lot of information.

Source: Darpa

eXplainable AI (XAI)


data explanation
understand the data
global direct explanation
the model itself is understandable
local direct explanation
the individual prediction is meaningful
training data
train an interpretable model
interpretable model
evaluate a new test sample
prediction
choose one or the other
train a black box model
predictions may be meaningful if the model predicts explanations in addition to labels
global post hoc explanation
an understandable model explains the black box model
black box model
produce an interpretable model
interpretable model
choose one or the other
evaluate a new test sample
local post hoc explanation
an explanation is created for the individual prediction
prediction
generate explanation

Source: IBM

A few directions

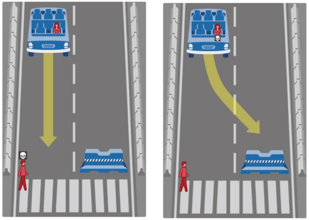
- Data Explanation.
- Use of explainable methods (linear?).
- Use of black box methods:
  - Global explanation (variable importance)
  - Local explanation (linear approximationn, alternative scenario. . .)

- Causality very hard to access without a real experimental plan with interventions!

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Metric and Solution

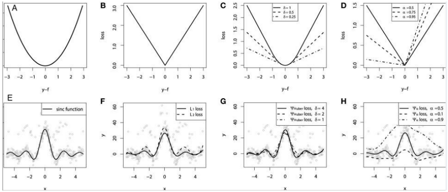


Source: J. Marshall

Quality metric has a strong impact on the solution.

- Implicite encoding rather than an explicit one!
- Often simplified criterion in the optimization part.
- More involved criterion can be used in evaluation.

Supervised Performance Metrics

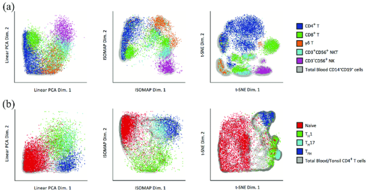


Measure of the cost of not being perfect!

- Criterion used to *optimize* the predictor and/or *evaluate* its interest.
- Classical metrics: quadratic error, zero/one error.
- Many other possible choices, idealy encoding domain expertise (asymmetry. . .)
- The criterion can be different between optimization and evaluation because of computation requirements.
- Very important factor (too) often neglected.

Source: P. Grover

Unsupervised Performance Metrics

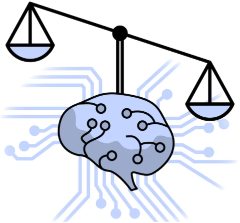


Measure the quality of the result!

- Dimension Reduction / Representation: reconstruction quality, relationship preservation. . .
- Clustering: measure of intra-group proximity and inter-group difference?
- Very subjective criterion!
- Hard to define the right distances especially for discrete variables.
- In practice, quality often evaluated by the a posteriori interest.

Source: H. Chen

Fairness



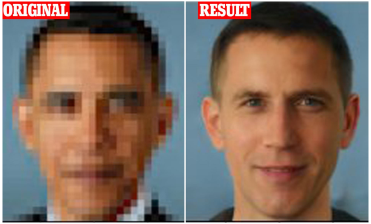
Source: Harvard

Fairness?

- Very hard to specify criterion.
- No consensus on its definition:
  - faithful reproduction of the reality?
  - correction of its bias?
- Current approaches through constraints in the optimization.
- A posteriori verification unavoidable!

- Additional constraints that may limit performances.

What About the Data Bias?



Source: A. Damian

Central assumption: representativity of the data!

- Optimization made in this setting.
- Possible training data bias:
  - selection bias in the data
  - population evolution
  - (historical) bias in the targets
- Correction possible at least up to a certain point for the two first cases if one is aware of the situation.

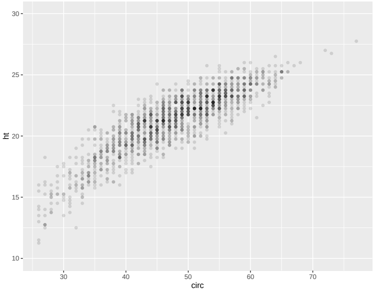
Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Outline

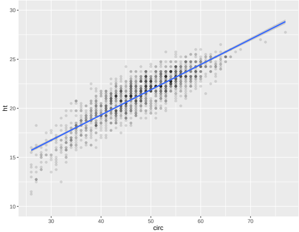
1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Eucalyptus



- Simple (and classical) dataset.
- Goal: predict the height from circumference
- $\underline{X} = \texttt{circ} =$ circumference.
- $Y = ht =$ height.

Eucalyptus



Linear Model

- Parametric model:

$$f_\beta(\mathtt{circ}) = \beta^{(1)} + \beta^{(2)}\mathtt{circ}$$

- How to choose $\beta = (\beta^{(1)}, \beta^{(2)})$?

Least Squares

Methodology

- Natural goodness criterion:

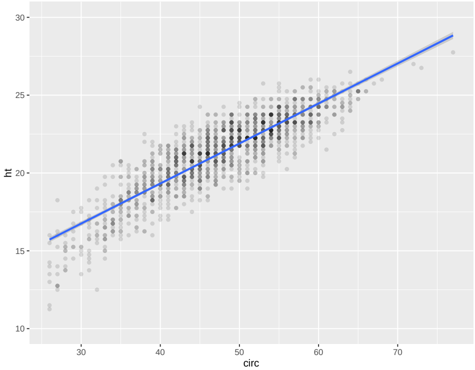
$$\sum_{i=1}^{n} |Y_i - f_\beta(\underline{X}_i)|^2 = \sum_{i=1}^{n} |\mathsf{ht}_i - f_\beta(\texttt{circ}_i)|^2$$
$$= \sum_{i=1}^{n} |\mathsf{ht}_i - (\beta^{(1)} + \beta^{(2)}\texttt{circ}_i)|^2$$

- Choice of β that minimizes this criterion!

$$\widehat{\beta} = \operatorname*{argmin}_{\beta\in\mathbb{R}^2} \sum_{i=1}^{n} |h_i - (\beta^{(1)} + \beta^{(2)}\texttt{circ}_i)|^2$$

- Easy minimization with an explicit solution!

Prediction



Prediction

- Linear prediction for the height:

$$\widehat{\texttt{ht}} = f_{\widehat{\beta}}(\texttt{circ}) = \widehat{\beta}^{(1)} + \widehat{\beta}^{(2)}\texttt{circ}$$

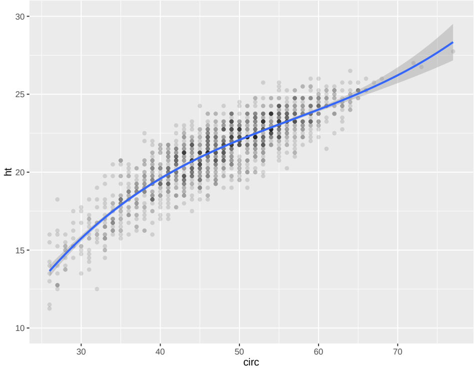
Heuristic

Linear Regression

- **Statistical model:** $(\mathtt{circ}_i, \mathtt{ht}_i)$ **i.i.d.** with the same law as a generic $(\mathtt{circ}, \mathtt{ht})$.
- **Performance criterion:** Look for f with a **small average error**
$$\mathbb{E}\left[|\mathtt{ht} - f(\mathtt{circ})|^2 \right]$$
- **Empirical criterion:** Replace the unknown law by its **empirical** counterpart
$$\frac{1}{n} \sum_{i=1}^{n} |\mathtt{ht}_i - f(\mathtt{circ}_i)|^2$$
- **Predictor model:** As the minimum over all function is 0 (if all the $\mathtt{circ}_i$ are different), **restrict** to the linear functions $f(\mathtt{circ}) = \beta^{(1)} + \beta^{(2)}\mathtt{circ}$ to avoid over-fitting.
- **Model fitting:** Explicit formula here.

- This model can be **too simple**!

Polynomial Regression

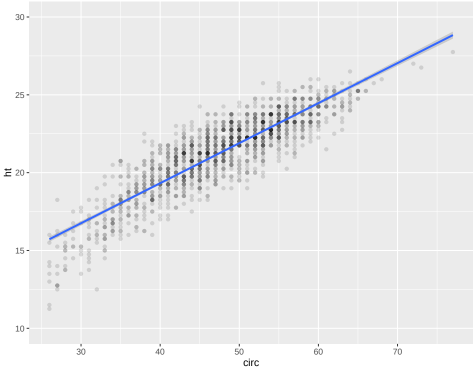


Polynomial Model

- Polynomial model: $f_\beta(\texttt{circ}) = \sum_{l=1}^{p} \beta^{(l)} \texttt{circ}^{l-1}$

- Linear in β.
- Easy least squares estimation for any degree!

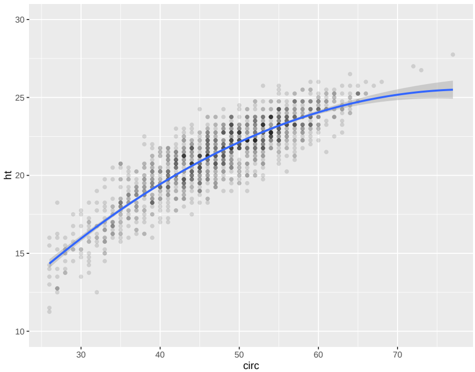
Which Degree?



Models

- Increasing degree = increasing complexity and better fit on the data

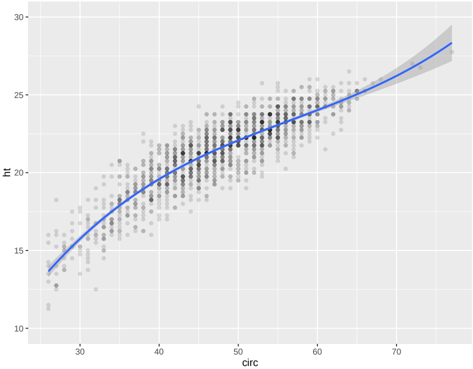
Which Degree?



Models

- Increasing degree = increasing complexity and better fit on the data

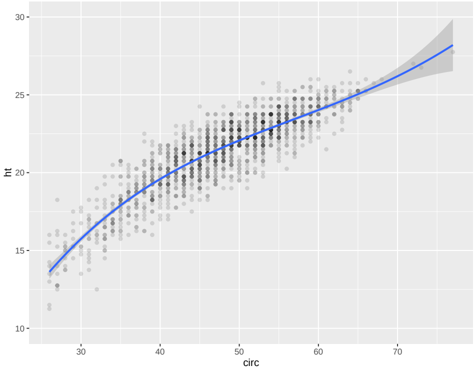
Which Degree?



Models

- Increasing degree = increasing complexity and better fit on the data

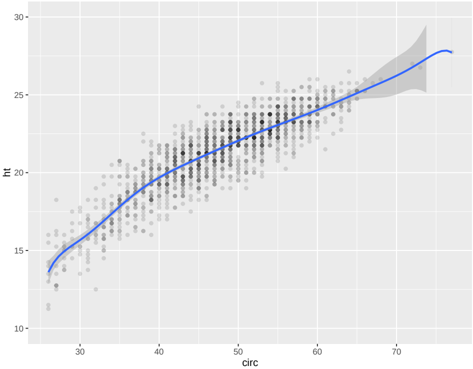
Which Degree?



Models

- Increasing degree = increasing complexity and better fit on the data

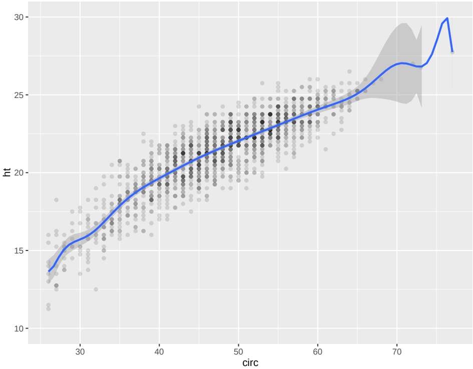
Which Degree?



Models

- Increasing degree = increasing complexity and better fit on the data

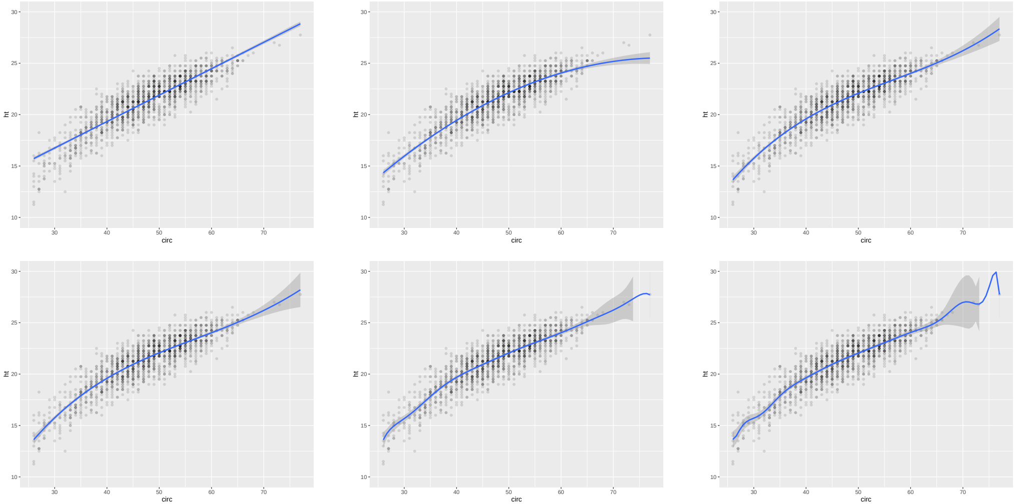
Which Degree?



Models

- Increasing degree = increasing complexity and better fit on the data

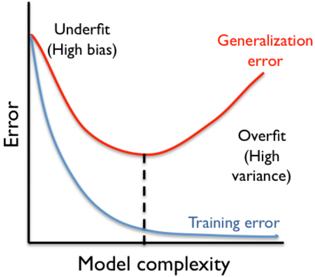
Which Degree?



Best Degree?

- How to choose among those solutions?

Over-fitting Issue



Risk behavior

- Training error (empirical error on the training set) decays when the complexity of the model increases.
- Quite different behavior when the error is computed on new observations (true risk / generalization error).

- Overfit for complex models: parameters learned are too specific to the learning set!
- General situation! (Think of polynomial fit. . .)
- Need to use another criterion than the training error!

Cross Validation and Penalization

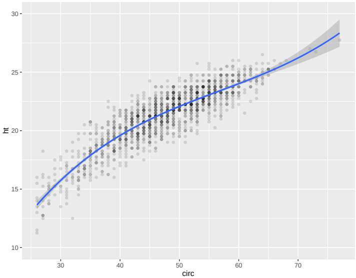
Two directions

- **How to estimate** the generalization error differently?
- Find a way to **correct** the empirical error?

Two Approaches

- **Cross validation:** Estimate the error on a different dataset:
  - Very efficient (and almost always used in practice!)
  - Need more data for the error computation.
- **Penalization approach:** Correct the optimism of the empirical error:
  - Require to find the correction (penalty).

Univariate Regression



Questions

- How to build a model?
- How to fit a model to the data?
- How to assess its quality?
- How to select a model among a collection?
- How to guaranty the quality of the selected model?

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Supervised Learning

Supervised Learning Framework

- Input measurement $\underline{X} \in \mathcal{X}$
- Output measurement $Y \in \mathcal{Y}$.
- $(\underline{X}, Y) \sim \mathbb{P}$ with $\mathbb{P}$ unknown.
- **Training data** : $\mathcal{D}_n = \{(\underline{X}_1, Y_1), \ldots, (\underline{X}_n, Y_n)\}$ (i.i.d. $\sim \mathbb{P}$)

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

Goal

- Construct a **good** predictor $\widehat{f}$ 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**: $\ell(Y, f(\underline{X}))$ measures the goodness of the prediction of Y by $f(\underline{X})$
- Examples:
  - 0/1 loss: $\ell(Y, f(\underline{X})) = \mathbf{1}_{Y \neq f(\underline{X})}$
  - Quadratic loss: $\ell(Y, f(\underline{X})) = |Y - f(\underline{X})|^2$

Risk function

- Risk measured as the average loss for a new couple:

$$\boldsymbol{\mathcal{R}(f) = \mathbb{E}_{(X,Y)\sim\mathbb{P}}[\ell(Y, f(\underline{X}))]}$$

- Examples:
  - 0/1 loss: $\mathbb{E}[\ell(Y, f(\underline{X}))] = \mathbb{P}(Y \neq f(\underline{X}))$
  - Quadratic loss: $\mathbb{E}[\ell(Y, f(\underline{X}))] = \mathbb{E}\left[|Y - f(\underline{X})|^2\right]$

- **Beware:** As $\widehat{f}$ depends on $\mathcal{D}_n$, $\mathcal{R}(\widehat{f})$ is a random variable!

Best Solution

- The best solution $f^\star$ (which is independent of $\mathcal{D}_n$) is

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

Bayes Predictor (explicit solution)

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

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

- In regression with the quadratic loss

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

Issue: Solution requires to **know** $Y|\underline{X}$ (or $Esp Y|\underline{X}$) for every value of $\underline{X}$!

Goal

Machine Learning

- Learn a rule to construct a **predictor** $\widehat{f} \in \mathcal{F}$ from the training data $\mathcal{D}_n$ s.t. **the risk** $\mathcal{R}(\widehat{f})$ 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

$$\widehat{f} = f_{\widehat{\theta}} = \operatorname*{argmin}_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(\underline{X}_i))$$

- Examples:
  - Linear regression
  - Linear classification with

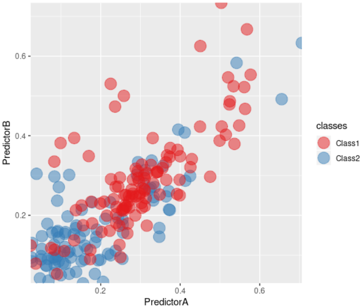
$$\mathcal{S} = \{\underline{x} \mapsto \mathrm{sign}\{\underline{x}^\top \beta + \beta^{(0)}\} \,/ \beta \in \mathbb{R}^d, \beta^{(0)} \in \mathbb{R}\}$$

Example: TwoClass Dataset

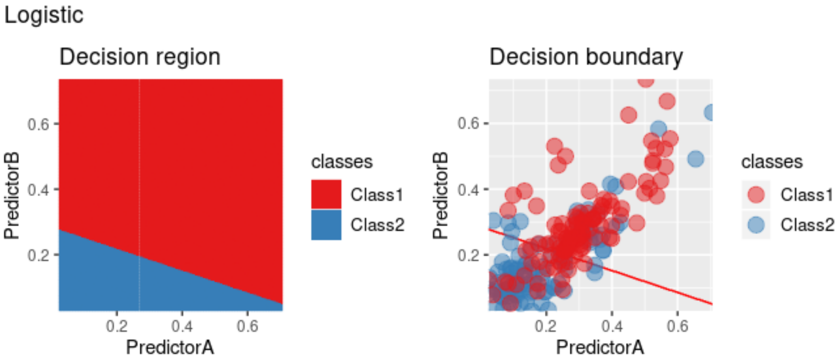
Synthetic Dataset

- Two features/covariates.
- Two classes.

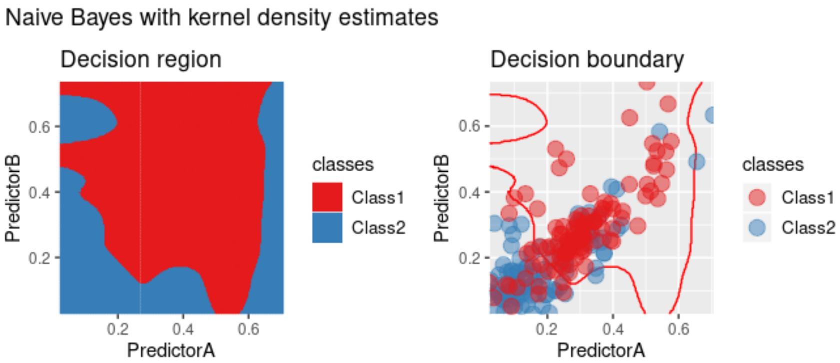
- Dataset from *Applied Predictive Modeling*, M. Kuhn and K. Johnson, Springer
- Numerical experiments with `R` and the {`caret`} package.



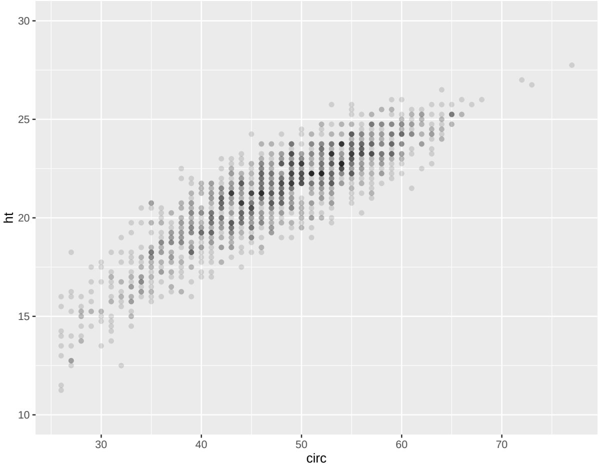
Example: Linear Classification



Example: More Complex Model



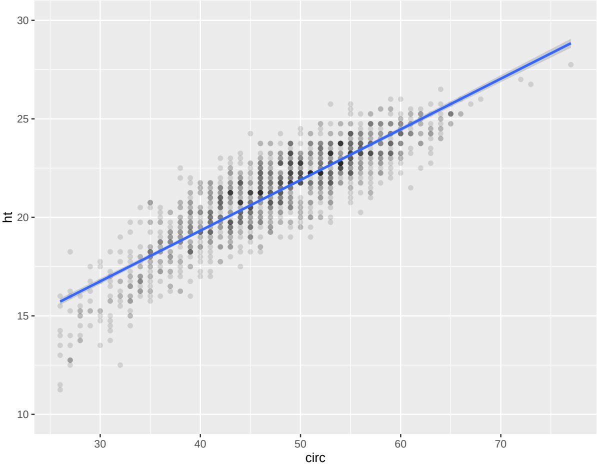
Eucalyptus



Dataset - P.A. Cornillon

- Real dataset of 1429 eucalyptus obtained by P.A. Cornillon:
  - $\underline{X}$: circumference / Y: height

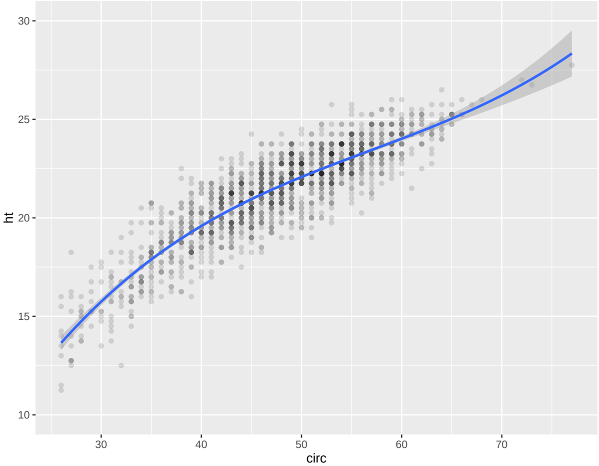
Eucalyptus



Dataset - P.A. Cornillon

- Real dataset of 1429 eucalyptus obtained by P.A. Cornillon:
  - $\underline{X}$: circumference / Y: height

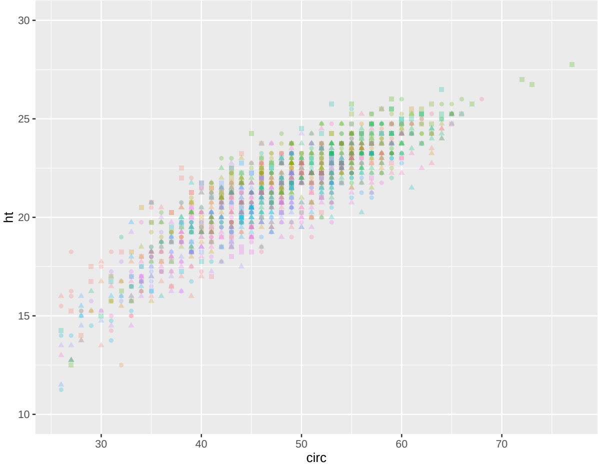
Eucalyptus



Dataset - P.A. Cornillon

- Real dataset of 1429 eucalyptus obtained by P.A. Cornillon:
  - $\underline{X}$: circumference / Y: height

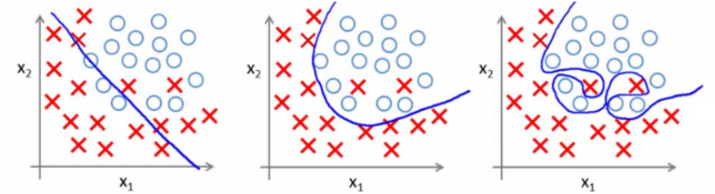
Eucalyptus



Dataset - P.A. Cornillon

- Real dataset of 1429 eucalyptus obtained by P.A. Cornillon:
  - $\underline{X}$: circumference, block, clone / Y: height

Under-fitting / Over-fitting Issue



$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$
(g = sigmoid function)

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2$
$+\theta_3 x_1^2 + \theta_4 x_2^2$
$+\theta_5 x_1 x_2)$

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2$
$+\theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2$
$+\theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \dots$

UNDERFITTING
(high bias)

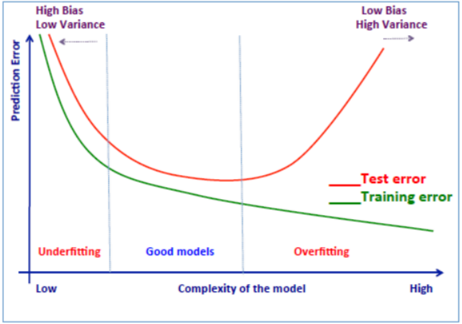
OVERFITTING
(high variance)

Model Complexity Dilemna

- What is best a simple or a complex model?
- Too simple to be good? Too complex to be learned?

Source: A. Ng

Under-fitting / Over-fitting Issue



Source: Unknown

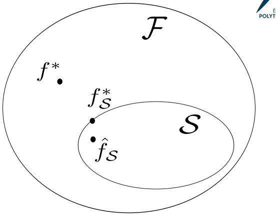
Under-fitting / Over-fitting

- **Under-fitting:** simple model are too simple.
- **Over-fitting:** complex model are too specific to the training set.

Bias-Variance Dilemma

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

A Better Point of View



Approximation error and estimation error (Bias-Variance)

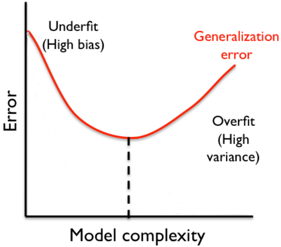
$$\mathcal{R}(\widehat{f}_{\mathcal{S}}) - \mathcal{R}(f^\star) = \underbrace{\mathcal{R}(f^\star_{\mathcal{S}}) - \mathcal{R}(f^\star)}_{\text{Approximation error}} + \underbrace{\mathcal{R}(\widehat{f}_{\mathcal{S}}) - \mathcal{R}(f^\star_{\mathcal{S}})}_{\text{Estimation error}}$$

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

Agnostic approach

- No assumption (so far) on the law of $(\underline{X}, Y)$.

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 $\Longleftrightarrow$ avoid **overfitting** and **underfitting**

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

Source: Unknown

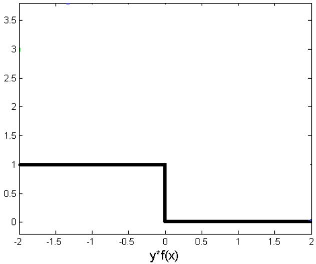
Theoretical Analysis

Statistical Learning Analysis

- Error decomposition:
$$\mathcal{R}(\widehat{f}_\mathcal{S}) - \mathcal{R}(f^\star) = \underbrace{\mathcal{R}(f^\star_\mathcal{S}) - \mathcal{R}(f^\star)}_{\text{Approximation error}} + \underbrace{\mathcal{R}(\widehat{f}_\mathcal{S}) - \mathcal{R}(f^\star_\mathcal{S})}_{\text{Estimation error}}$$
- Bound on the approximation term: approximation theory.
- Probabilistic bound on the estimation term: probability theory!
- **Goal: Agnostic bounds**, i.e. bounds that do not require assumptions on $\mathbb{P}$! (Statistical Learning?)

- Often need mild assumptions on $\mathbb{P}$... (Nonparametric Statistics?)

Binary Classification Loss Issue

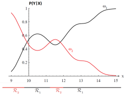


Empirical Risk Minimizer

$$\widehat{f} = \operatorname*{argmin}_{f \in \mathcal{S}} \frac{1}{n} \sum_{i=1}^{n} \ell^{0/1}(Y_i, f(\underline{X}_i))$$

- Classification loss: $\ell^{0/1}(y, f(\underline{x})) = \mathbf{1}_{y \neq f(\underline{x})}$
- Not convex and not smooth!

Probabilistic Point of View
Estimation and Plugin



- The best solution $f^\star$ (which is independent of $\mathcal{D}_n$) is

$$f^\star = \arg\min_{f\in\mathcal{F}} \mathcal{R}(f) = \arg\min_{f\in\mathcal{F}} \mathbb{E}[\ell(Y, f(\underline{X}))] = \arg\min_{f\in\mathcal{F}} \mathbb{E}_{\underline{X}}\Big[\mathbb{E}_{Y|\underline{X}}[\ell(Y, f(\underline{x}))]\Big]$$

Bayes Predictor (explicit solution)

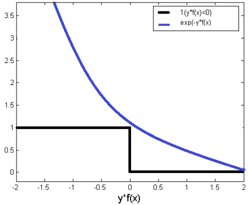
- In binary classification with $0-1$ loss:

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

- **Issue:** Solution requires to **know** $Y|\underline{X}$ for all values of $\underline{X}$!
- **Solution:** Replace it by an estimate and plug it in the Bayes predictor formula.

Source: A. Fermin

Optimization Point of View
Loss Convexification and Optimization



Minimizer of the risk

$$\widehat{f} = \operatorname*{argmin}_{f \in \mathcal{S}} \frac{1}{n} \sum_{i=1}^{n} \ell^{0/1}(Y_i, f(\underline{X}_i))$$

- **Issue:** Classification loss is not convex or smooth.
- **Solution:** Replace it by a convex majorant and find the best predictor for this surrogate problem.

Probabilistic and Optimization Framework

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

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

Canonical approach: $\widehat{f}_{\mathcal{S}} = \operatorname{argmin}_{f \in \mathcal{S}} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(\underline{X}_i))$

Problems

- How to choose $\mathcal{S}$?
- How to compute the minimization?

A Probabilistic Point of View

Solution: For $\underline{X}$, estimate $Y|\underline{X}$ and plug it in any Bayes classifier: **(Generalized) Linear Models, Kernel methods, k-nn, Naive Bayes, Tree, Bagging...**

An Optimization Point of View

Solution: Replace the loss ℓ by an upper bound $\bar{\ell}$ and minimize directly the corresponding emp. risk: **Neural Network, SVR, SVM, Tree, Boosting...**

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Outline

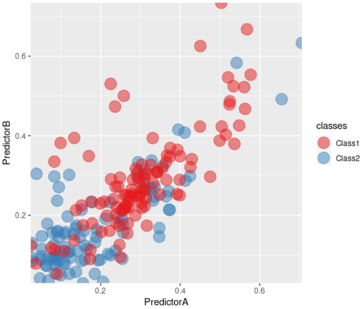
1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Example: TwoClass Dataset

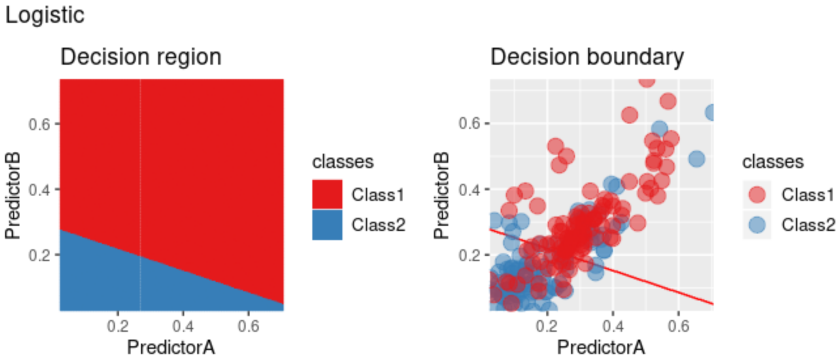
Synthetic Dataset

- Two features/covariates.
- Two classes.

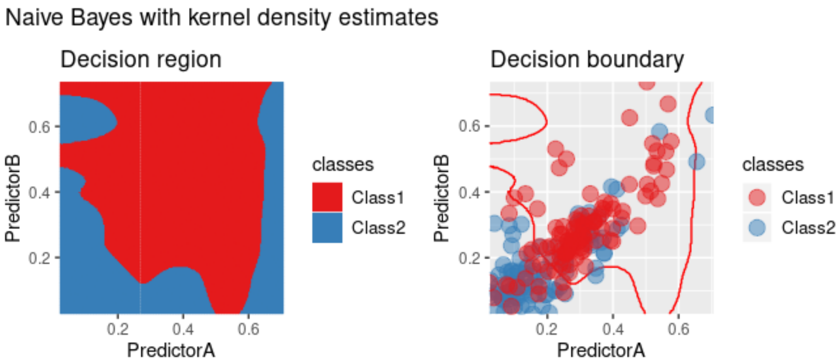
- Dataset from *Applied Predictive Modeling*, M. Kuhn and K. Johnson, Springer
- Numerical experiments with `R` and the {`caret`} package.



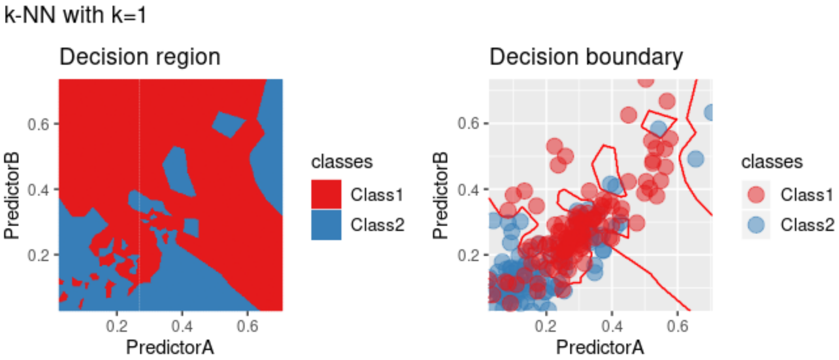
Example: Linear Classification



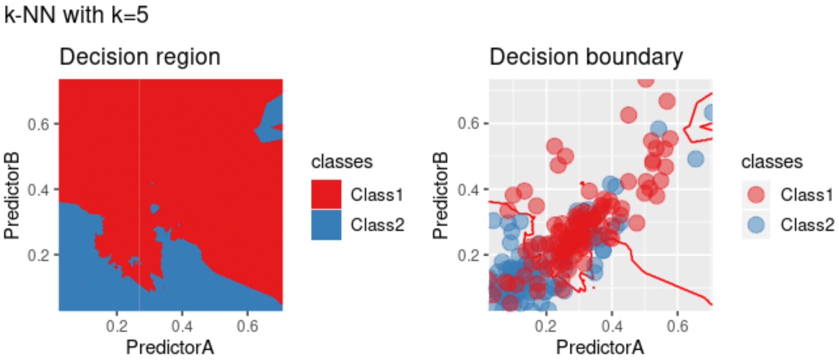
Example: More Complex Model



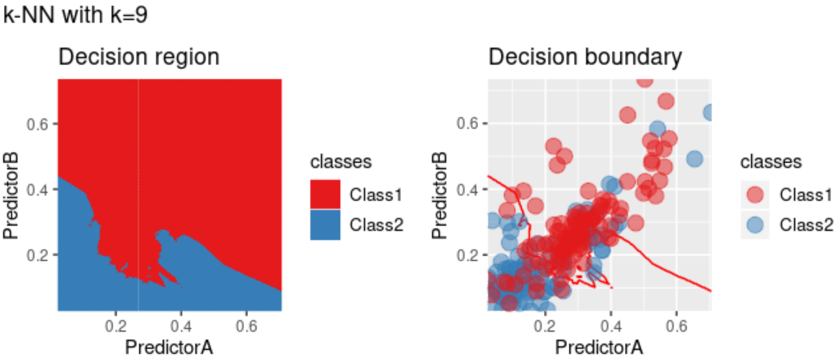
Example: KNN


k-NN with k=1
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

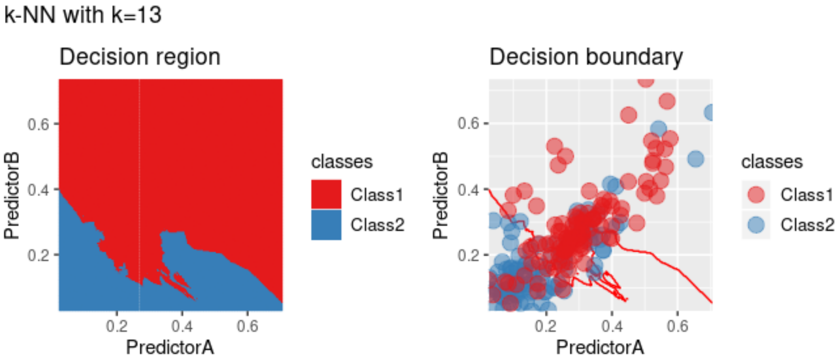
Example: KNN


k-NN with k=5
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

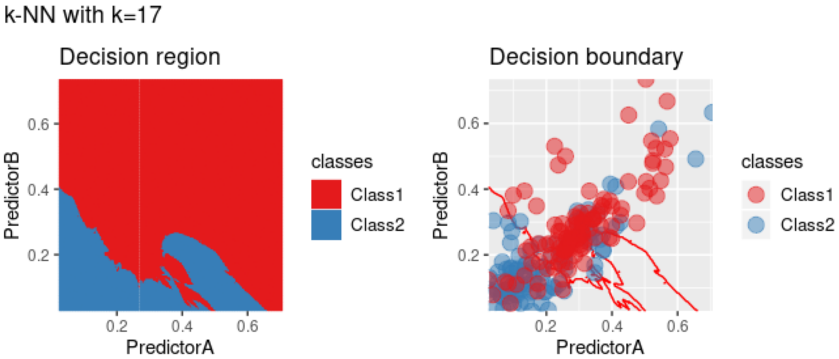
Example: KNN


k-NN with k=9
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

Example: KNN


k-NN with k=13
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: KNN



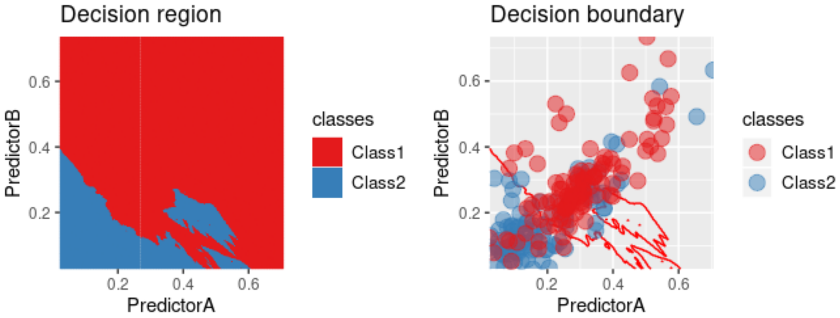
k-NN with k=17

Decision region

Decision boundary

Example: KNN

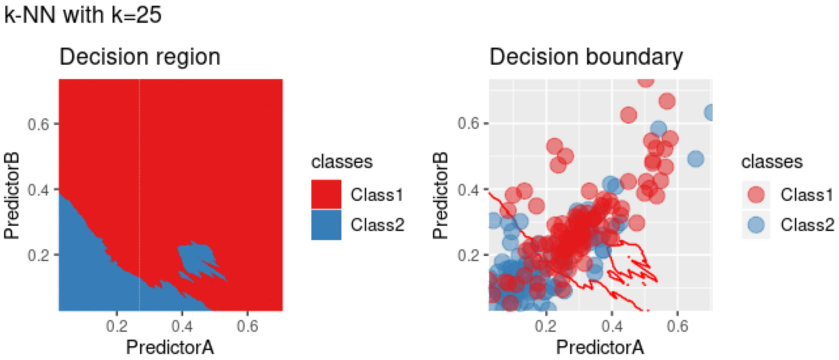
k-NN with k=21



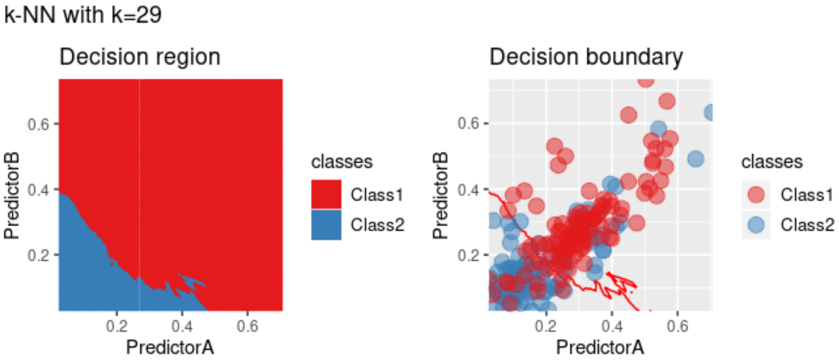
Decision region

Decision boundary

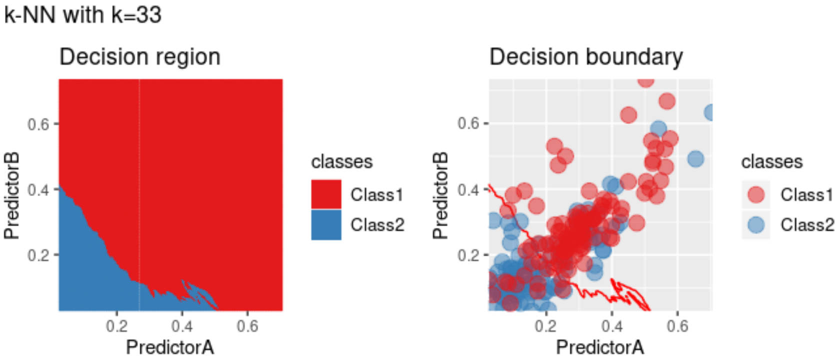
Example: KNN


k-NN with k=25
Decision region
PredictorB
0.6
0.4
0.2
PredictorA
0.2
0.4
0.6
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

Example: KNN

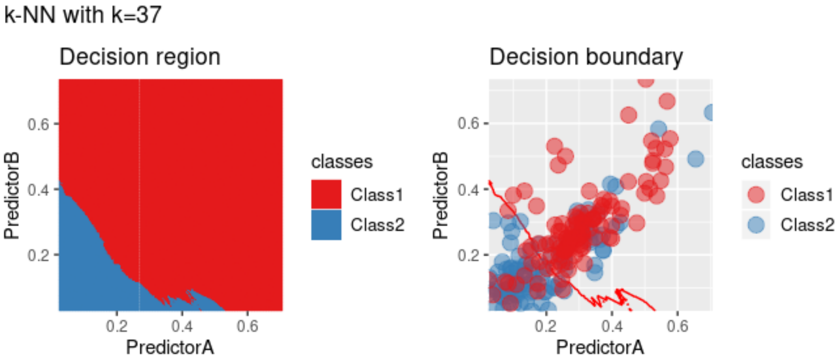

k-NN with k=29
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: KNN



k-NN with k=33

Example: KNN

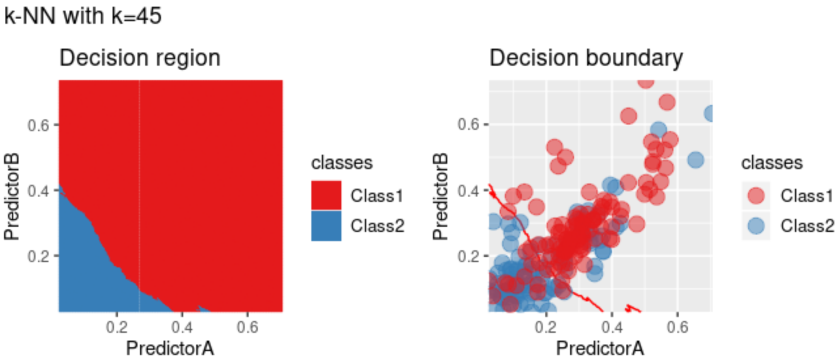


k-NN with k=37

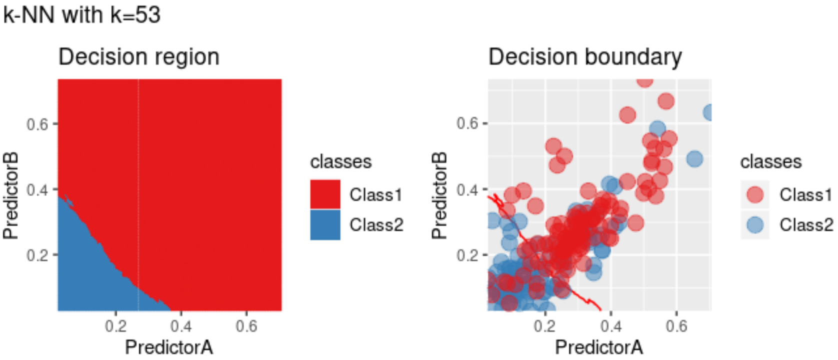
Decision region

Decision boundary

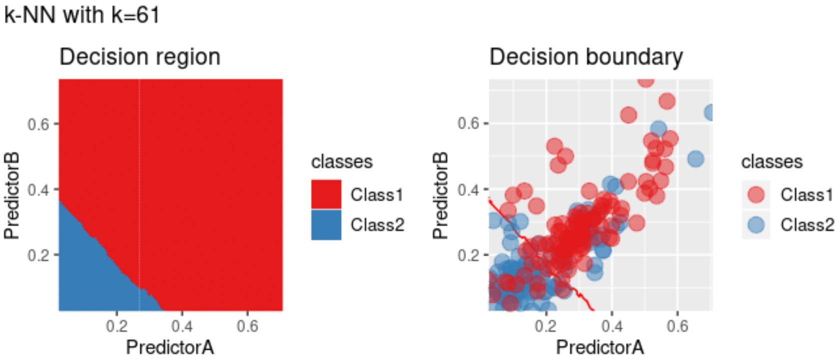
Example: KNN


k-NN with k=45
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

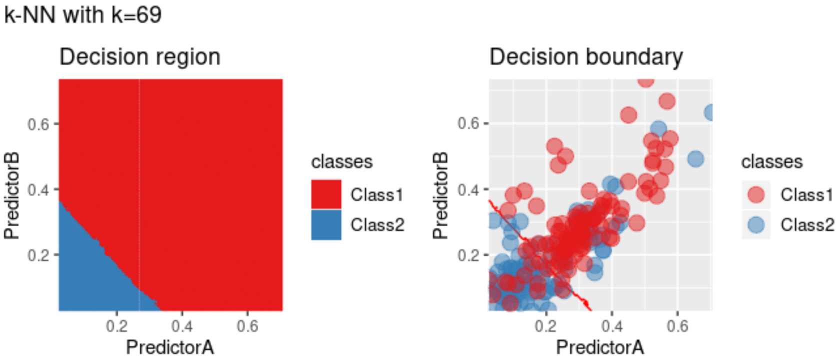
Example: KNN


k-NN with k=53
Decision region
Decision boundary
PredictorA
PredictorB
0.2
0.4
0.6
classes
Class1
Class2

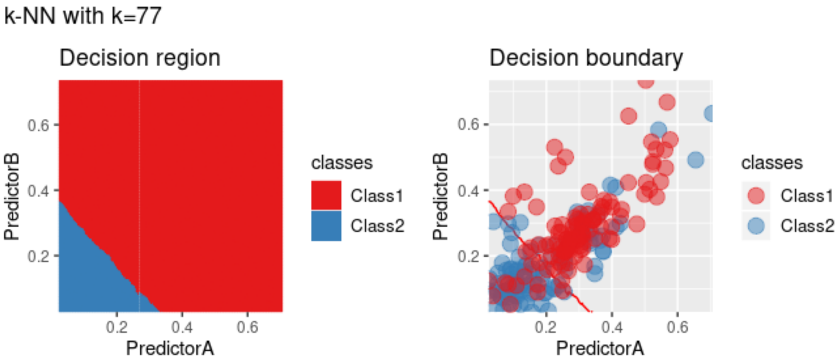
Example: KNN


k-NN with k=61
Decision region
PredictorB
0.2
0.4
0.6
PredictorA
0.2
0.4
0.6
classes
Class1
Class2
Decision boundary
PredictorB
0.2
0.4
0.6
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

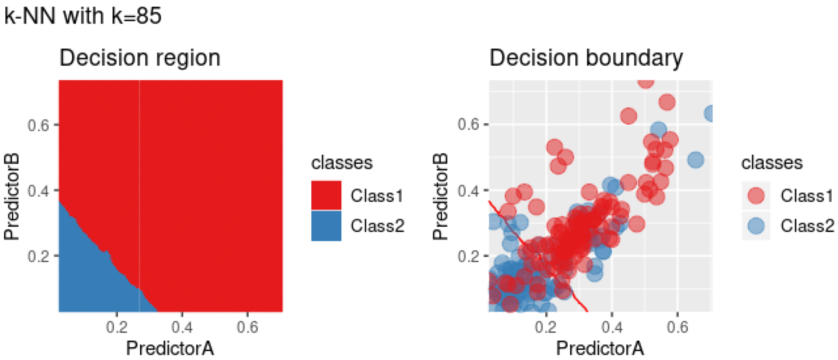
Example: KNN


k-NN with k=69
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

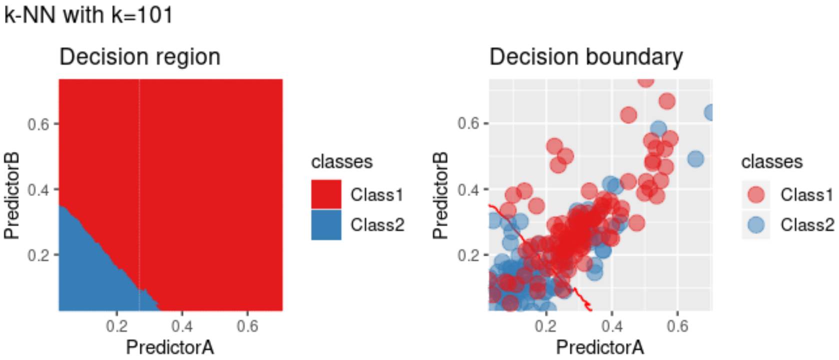
Example: KNN


k-NN with k=77
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

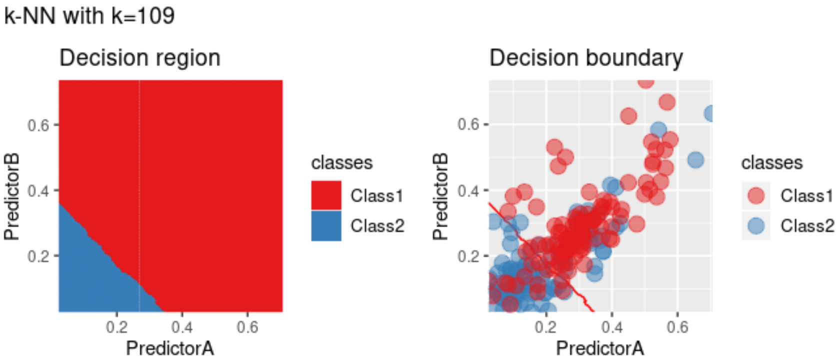
Example: KNN


k-NN with k=85
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: KNN

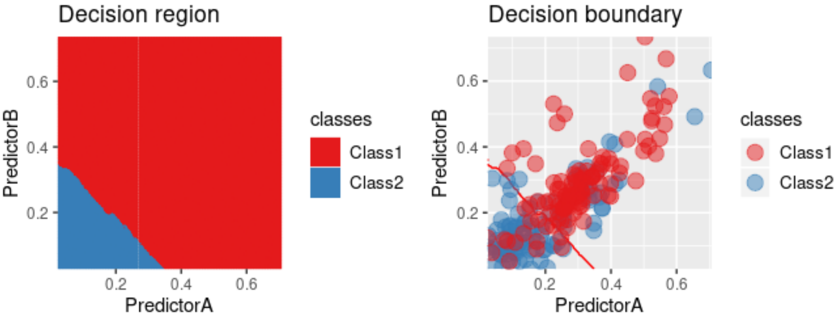

k-NN with k=101
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: KNN

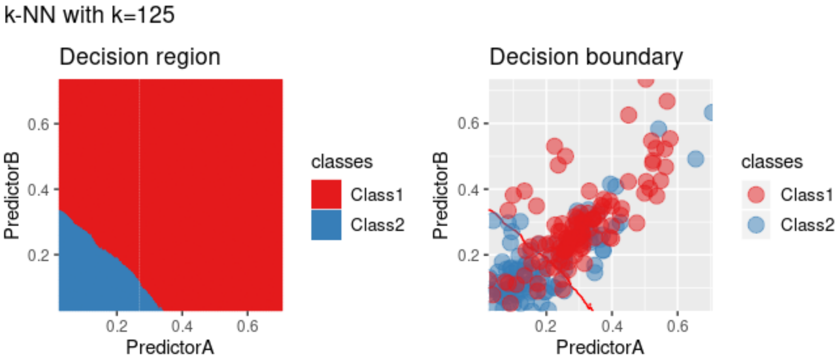

k-NN with k=109
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

Example: KNN

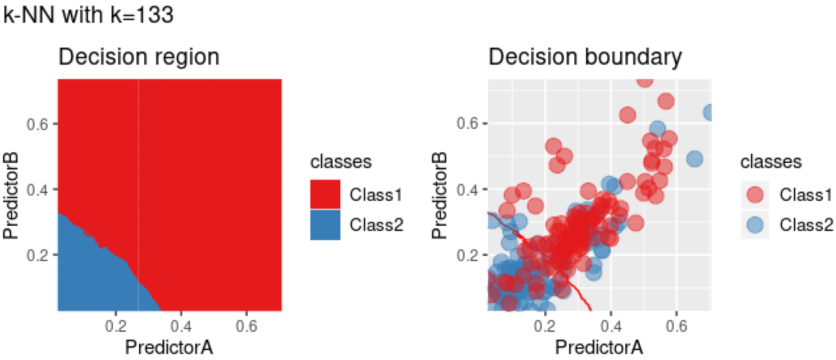
k-NN with k=117



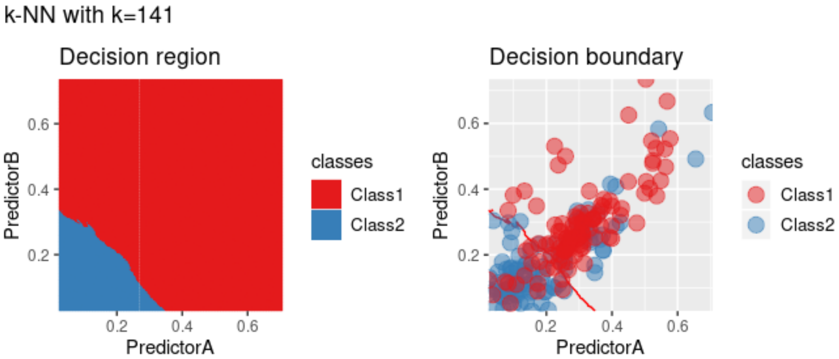
Example: KNN


k-NN with k=125
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

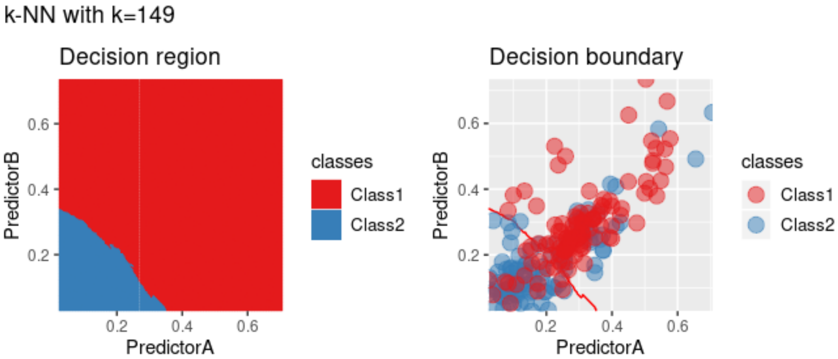
Example: KNN


k-NN with k=133
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

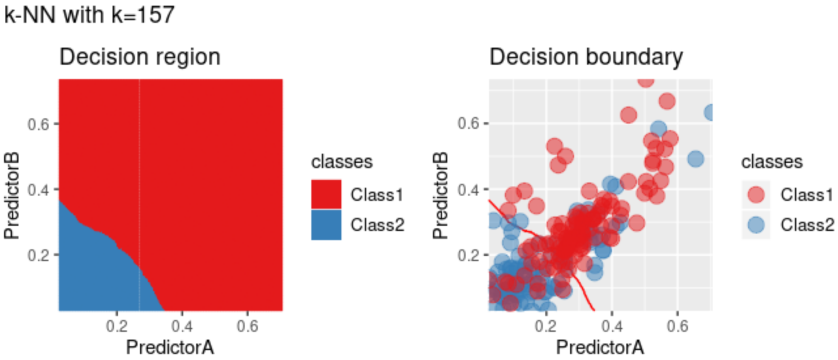
Example: KNN


k-NN with k=141
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

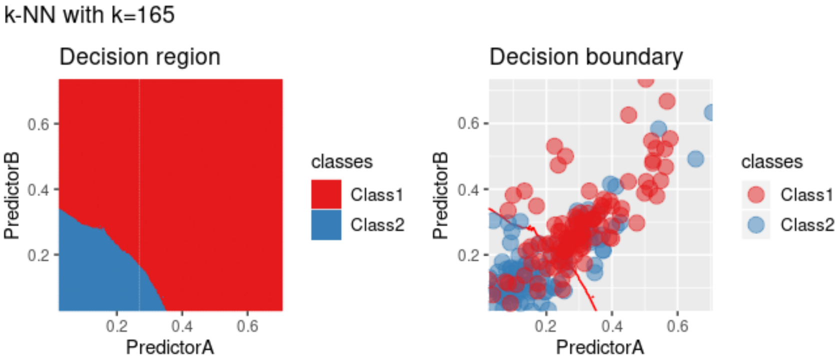
Example: KNN


k-NN with k=149
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
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=157
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: KNN

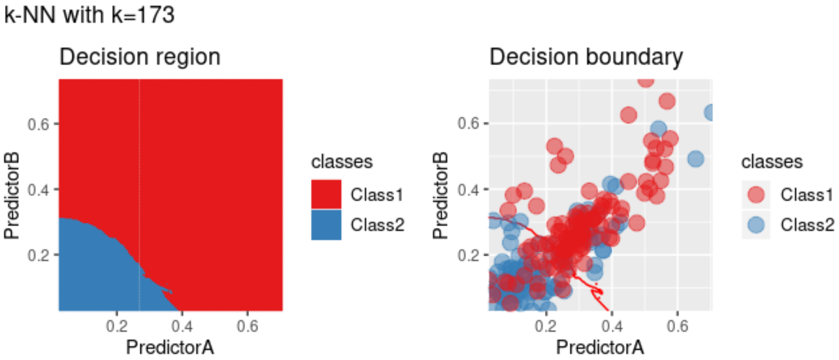


k-NN with k=165

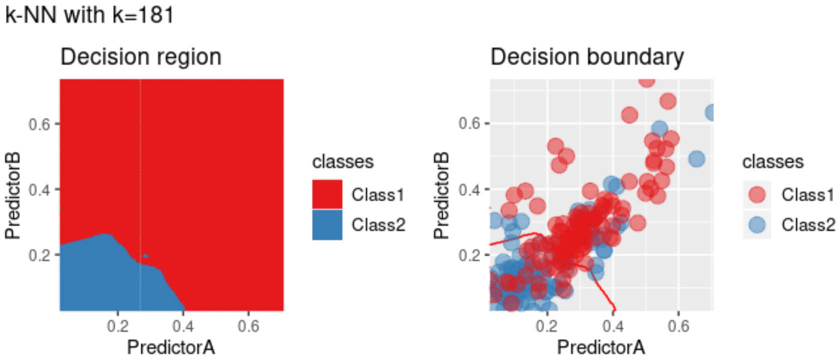
Decision region

Decision boundary

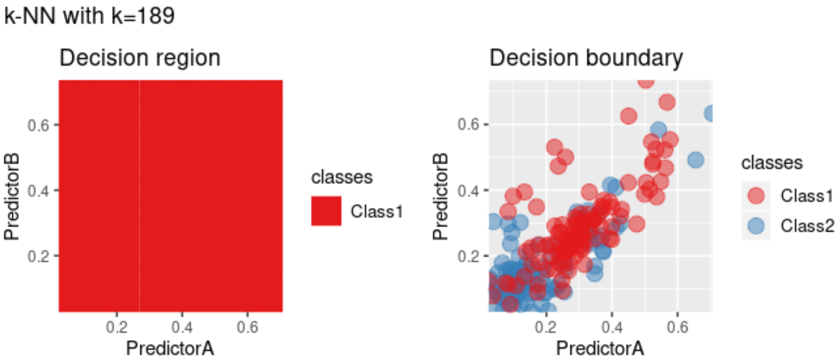
Example: KNN


k-NN with k=173
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

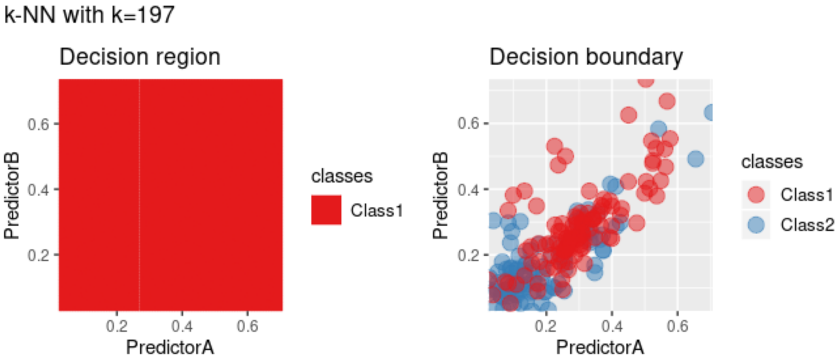
Example: KNN


k-NN with k=181
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

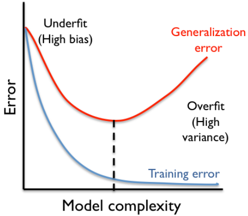
Example: KNN


k-NN with k=189
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
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=197
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Training Risk Issue



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!

Source: JMP

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

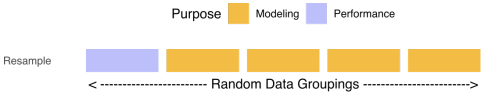
- **Cross validation:** Use empirical risk criterion but on independent data, very efficient (and almost always used in practice!) but slightly biased as its target uses only a fraction of the data.
- **Correction approach:** use empirical risk criterion but *correct* it with a term increasing with the complexity of $\mathcal{S}$
$$R_n(\widehat{f_{\mathcal{S}}}) \to R_n(\widehat{f_{\mathcal{S}}}) + \mathrm{cor}(\mathcal{S})$$
and choose the method with the smallest corrected risk.

Which loss is use?

- The loss used in the risk!
- Not the loss used in the training!

- Other performance measure can be used.

Cross Validation


Purpose
Modeling
Performance
Resample
< ----------------------- Random Data Groupings ---------------------->

- **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 ϵn is too small ?

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

Source: M. Kühn

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 $\widehat{f}^{HO}$ from the subset $\mathcal{D}_{\text{train}}$.
- Compute the empirical risk on the subset $\mathcal{D}_{\text{test}}$:

$$\mathcal{R}_n^{HO}(\widehat{f}^{HO}) = \frac{1}{n\epsilon} \sum_{(\underline{X}_i, Y_i) \in \mathcal{D}_{\text{test}}} \ell(Y_i, \widehat{f}^{HO}(\underline{X}_i))$$

Predictor Risk Estimation

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

Method Selection by Cross Validation

- Compute $\mathcal{R}_n^{HO}(\widehat{f}_{\mathcal{S}}^{HO})$ for all the considered methods,
- Select the method with the smallest CV risk,
- Reestimate the $\widehat{f}_{\mathcal{S}}$ 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 $\widehat{f}^{HO}$ from the subset $\mathcal{D}_{\text{train}}$.
- Compute the empirical risk on the subset $\mathcal{D}_{\text{test}}$:

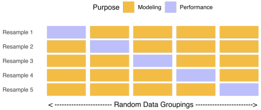
$$\mathcal{R}_n^{HO}(\widehat{f}^{HO}) = \frac{1}{n\epsilon} \sum_{(\underline{X}_i, Y_i) \in \mathcal{D}_{\text{test}}} \ell(Y_i, \widehat{f}^{HO}(\underline{X}_i))$$

- 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 $\mathcal{R}_n^{HO}(\widehat{f}^{HO})$ not taken into account.

V-fold Cross Validation



Principle

- Split the dataset $\mathcal{D}$ in V sets $\mathcal{D}_v$ of almost equals size.
- For $v \in \{1, .., V\}$:
  - Learn $\widehat{f}^{-v}$ from the dataset $\mathcal{D}$ minus the set $\mathcal{D}_v$.
  - Compute the empirical risk:

$$\mathcal{R}_n^{-v}(\widehat{f}^{-v}) = \frac{1}{n_v} \sum_{(\underline{X}_i, Y_i) \in \mathcal{D}_v} \ell(Y_i, \widehat{f}^{-v}(\underline{X}_i))$$

- Compute the average empirical risk:

$$\mathcal{R}_n^{CV}(\widehat{f}) = \frac{1}{V} \sum_{v=1}^{V} \mathcal{R}_n^{-v}(\widehat{f}^{-v})$$

- 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 $\mathcal{R}_n^{-v}(\widehat{f}^{-v})$ are identically distributed variables but are not independent!
- Consequence:

$$\mathbb{E}\left[\mathcal{R}_n^{CV}(\widehat{f})\right] = \mathbb{E}\left[\mathcal{R}_n^{-v}(\widehat{f}^{-v})\right]$$

$$\mathbb{V}\text{ar}\left[\mathcal{R}_n^{CV}(\widehat{f})\right] = \frac{1}{V}\,\mathbb{V}\text{ar}\left[\mathcal{R}_n^{-v}(\widehat{f}^{-v})\right] + (1-\frac{1}{V})\,\mathbb{C}\text{ov}\left[\mathcal{R}_n^{-v}(\widehat{f}^{-v}), \mathcal{R}_n^{-v'}(\widehat{f}^{-v'})\right]$$

- 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

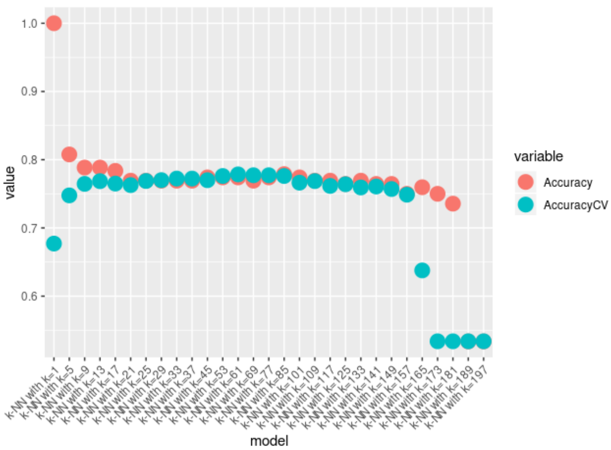
- **Prop:** for the least squares linear regression,

$$\widehat{f}^{-i}(\underline{X}_i) = \frac{\widehat{f}(\underline{X}_i) - h_{ii} Y_i}{1 - h_{ii}}$$

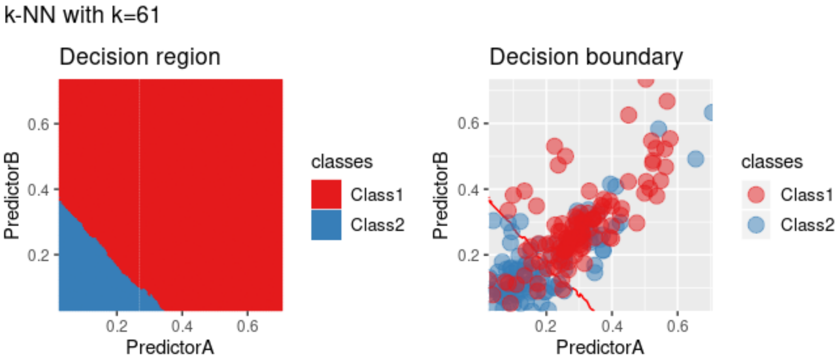
  with h_{ii} the ith diagonal coefficient of the **hat** (projection) matrix.
- Proof based on linear algebra!
- Leads to a fast formula for LOO:

$$\mathcal{R}_n^{LOO}(\widehat{f}) = \frac{1}{n} \sum_{i=1}^{n} \frac{|Y_i - \widehat{f}(\underline{X}_i)|^2}{(1 - h_{ii})^2}$$

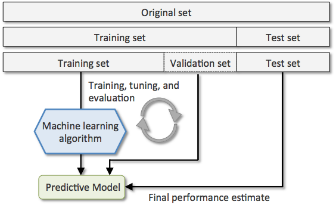
Cross Validation



Example: KNN ($\widehat{k} = 61$ using cross-validation)



Train/Validation/Test


Original set
Training set
Test set
Training set
Validation set
Test set
Training, tuning, and evaluation
Machine learning algorithm
Predictive Model
Final performance estimate

- **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!

Source: Shan-Hung Wu & DataLab

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 over a collection of models (Θ_m)

$$\min_{\theta \in \Theta_m} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(\underline{X}_i)) + \text{pen}(\Theta_m)$$

where $\text{pen}(\Theta)$ is a risk correction (penalty) depending on the model.

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. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
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

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

- **Question:** are the means $\overline{e^{(l)}}$ 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 $\mathbb{E}\left[e^{(1)}\right] = \mathbb{E}\left[e^{(2)}\right]$?
- **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.

Comparison of Two Means

Non i.i.d. case

- **Assumption:** $e_i^{(l)}$ are i.d. for each l but not necessarily independent.
- **Test formulation:** Can we reject the null hypothesis that $\mathbb{E}\left[e^{(1)}\right] = \mathbb{E}\left[e^{(2)}\right]$?
- **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

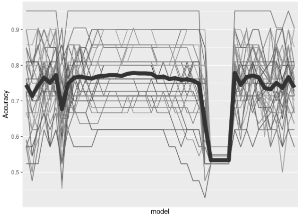
Comparison of Several Means

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 $\mathbb{E}\left[e^{(l)}\right]$ 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



CV Risk, Methods and Predictors

- Cross-Validation risk: estimate of the average risk of a ML method.
- No risk bound on the predictor obtained in practice.

Probabibly-Approximately-Correct (PAC) Approach

- Replace the control on the average risk by a probabilistic bound

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

- 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.)

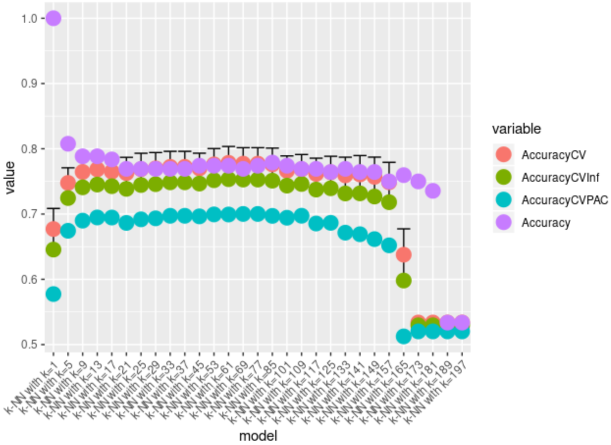
- Compute the empirical variance and divide it by the number of folds to construct an asymptotic Gaussian confidence interval,
- Select the simplest model whose value falls into the confidence interval of the model having the smallest CV risk.

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.

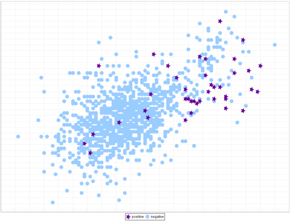
Cross Validation



Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Unbalanced and Rebalanced Dataset



Source: University of Granada

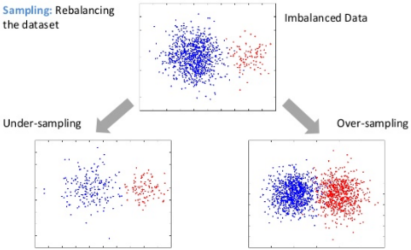
Unbalanced Class

- **Setting:** One of the classes 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!**

Source: Oracle

Resampling Effect

Testing

- Testing class prob.: $\pi_{\text{test}}(k)$
- Testing risk target:
  $$\mathbb{E}_{\text{test}}[\ell(Y, f(\underline{X}))] = \sum_k \pi_{\text{test}}(k) \mathbb{E}[\ell(Y, f(\underline{X}))|Y = k]$$

Training

- Training class prob.: $\pi_{\text{train}}(k)$
- Training risk target:
  $$\mathbb{E}_{\text{train}}[\ell(Y, f(\underline{X}))] = \sum_k \pi_{\text{train}}(k) \mathbb{E}[\ell(Y, f(\underline{X}))|Y = k]$$

Implicit Testing Risk Using the Training One

- Amounts to use a weighted loss:
  $$\begin{aligned}\mathbb{E}_{\text{train}}[\ell(Y, f(\underline{X}))] &= \sum_k \pi_{\text{train}}(k) \mathbb{E}[\ell(Y, f(\underline{X}))|Y = k] \\ &= \sum_k \pi_{\text{test}}(k) \mathbb{E}\left[\frac{\pi_{\text{train}}(k)}{\pi_{\text{test}}(k)} \ell(Y, f(\underline{X})) \middle| Y = k \right] \\ &= \mathbb{E}_{\text{test}}\left[\frac{\pi_{\text{train}}(Y)}{\pi_{\text{test}}(Y)} \ell(Y, f(\underline{X})) \right]\end{aligned}$$
- Put more weight on less probable classes!

Weighted Loss

- 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:**

$$\ell(Y, f(\underline{X})) \to C(Y)\ell(Y, f(\underline{X}))$$

- Weighted risk target:

$$\mathbb{E}[C(Y)\ell(Y, f(\underline{X}))]$$

- **Rk:** Strong link with ℓ as C is independent of f.
- Often allow reusing algorithm constructed for ℓ.
- C may also depend on $\underline{X}$. . .

Weighted Loss, $\ell^{0/1}$ loss and Bayes Classifier

- The Bayes classifier is now:
$$f^\star = \operatorname{argmin} \mathbb{E}[C(Y)\ell(Y, f(\underline{X}))] = \operatorname{argmin} \mathbb{E}_{\underline{X}}\left[\mathbb{E}_{Y|\underline{X}}[C(Y)\ell(Y, f(\underline{X}))]\right]$$

Bayes Predictor

- For $\ell^{0/1}$ loss,
$$f^\star(\underline{X}) = \operatorname*{argmax}_k C(k)\mathbb{P}(Y = k|\underline{X})$$
- Same effect than a threshold modification for the binary setting!

- Allow putting more emphasis on some classes than others.

Linking Weights and Proportions

Cost and Proportions

- Testing risk target:
$$\mathbb{E}_{\text{test}}[C_{\text{test}}(Y)\ell(Y,f(\underline{X}))] = \sum_k \pi_{\text{test}}(k) C_{\text{test}}(k) \mathbb{E}[\ell(Y,f(\underline{X}))|Y=k]$$
- Training risk target
$$\mathbb{E}_{\text{train}}[C_{\text{train}}(Y)\ell(Y,f(\underline{X}))] = \sum_k \pi_{\text{train}}(k) C_{\text{train}}(k) \mathbb{E}[\ell(Y,f(\underline{X}))|Y=k]$$
- **Coincide if**
$$\pi_{\text{test}}(k) C_{\text{test}}(k) = \pi_{\text{train}}(k) C_{\text{train}}(k)$$

- Lots of flexibility in the choice of C_t, C_{train} or π_{train}.

Combining Weights and Resampling

Weighted Loss and Resampling

- **Weighted loss:** choice of a weight $C_{\text{test}} \neq 1$.
- **Resampling:** use a $\pi_{\text{train}} \neq \pi_{\text{test}}$.

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

Combining Weights and Resampling

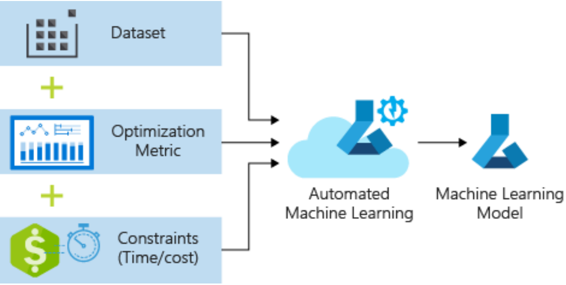
- **Weighted loss:** use $C_{\text{train}} = C_{\text{test}}$ as $\pi_{\text{train}} = \pi_{\text{test}}$.
- **Resampling:** use an implicit $C_{\text{test}}(k) = \pi_{\text{train}}(k)/\pi_{\text{test}}(k)$.
- **Combined:** use $C_{\text{train}}(k) = C_{\text{test}}(k)\pi_{\text{test}}(k)/\pi_{\text{train}}(k)$

- Most ML methods allow such weights!

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Auto ML

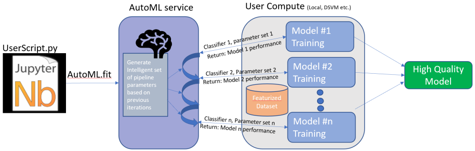


Source: softwareengineeringdaily.com/2019/05/15/introduction-to-automated-machine-learning-automl/

Auto ML

- Automatically propose a good predictor
- Rely heavily on risk evaluations
- **Pros:** easy way to obtain an excellent baseline
- **Cons:** black box that can be abused. . .

Auto ML Task

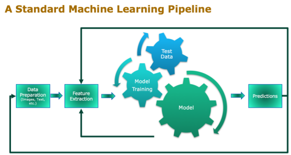

AutoML service
User Compute (Local, DSVM etc.)
UserScript.py
Jupyter
Nb
AutoML.fit
Generate Intelligent set of pipeline parameters based on previous iterations
Classifier 1, parameter set 1
Return: Model 1 performance
Classifier 2, Parameter set 2
Return: Model 2 performance
Classifier n, Parameter set n
Return: Model n performance
Model #1 Training
Model #2 Training
Featurized Dataset
Model #n Training
High Quality Model

Source: Microsoft

Auto ML Task

- Input:
  - a dataset $\mathcal{D} = (\underline{X}_i, Y_i)$
  - a loss function $\ell(Y, f(\underline{X}))$
  - a set of possible predictors $f_{l,h,\theta}$ corresponding to a method l in a list, with hyperparameters h and parameters θ
- Output:
  - a predictor f equal to $f_{\hat{l},\hat{h},\hat{\theta}}$ or combining several such functions.

Predictors


A Standard Machine Learning Pipeline
Data Preparation (Images, Text, etc.)
Feature Extraction
Test Data
Model Training
Model
Predictions

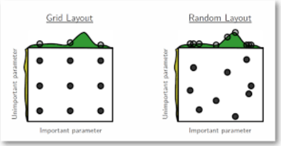
Predictors, a.k.a fitted pipelines

- Preprocessing:
  - Feature design: normalization, coding, kernel. . .
  - Missing value strategy
  - Feature selection method
- ML Method:
  - Method itself
  - Hyperparameters and architecture
  - Fitted parameters (includes optimization algorithm)

- Quickly amounts to 20 to 50 design decisions!
- **Bruteforce exploration impossible!**

Source: Microsoft

Auto ML and Hyperparameter Optimization

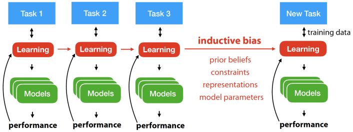


Most Classical Approach of Auto ML

- Task rephrased as an optimization on the discrete/continous space of methods/hyperparameters/parameters.
- Parameters obtained by classical minimization.
- Optimization of methods/hyperparameters much more challenging.
- Approaches:
  - Bruteforce: Grid search and random search
  - Clever exploration: Evolutionary algorithm
  - Surrogate based: Bayesian search and Reinforcement learning

Source: J. Borgstra

Auto ML and Meta-Learning



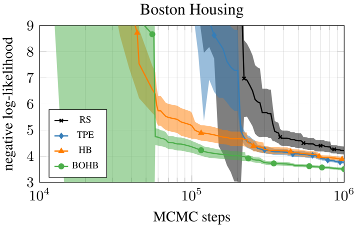
Learn from other Learning Tasks

- Consider the choice of the method from a dataset and a metric as a learning task.
- Requires a way to describe the problems (or to compute a similarity).
- Descriptor often based on a combination of dataset properties and fast method results.
- May output a list of candidates instead of a single method.

- Promising but still quite experimental!

Source: NeurIPS 2018 Tutorial on Automatic Machine Learning

Auto ML and Time Budget

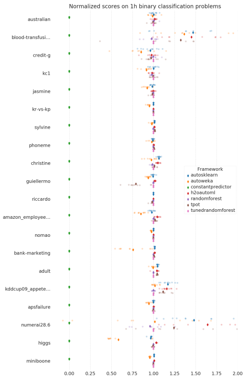


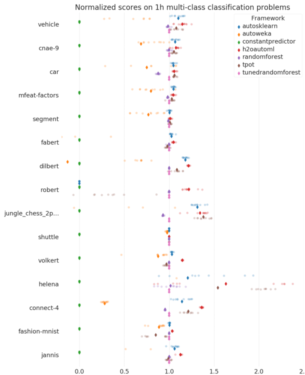
How to obtain a good result with a time constraint?

- Brute force: Time out and methods screening with Meta-Learning (less exploration at the beginning)
- Surrogate based: Bayesian optimization (exploration/exploitation tradeoff)
- Successive elimination: Fast but not accurate performance evaluation at the beginning to eliminate the worst models (more exploration at the beginning)
- Combined strategy: Bandit strategy to obtain a more accurate estimate of risks only for the promising models (exploration/exploitation tradeoff)

Source: A. Biedenkapp

Auto ML benchmark


Normalized scores on 1h binary classification problems


Normalized scores on 1h multi-class classification problems

Benchmark

- Almost always (slightly) better than a good random forest or gradient boosting predictor.
- Worth the try!

Source: AutoML Benchmark

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Three Classical Methods in a Nutshell

Logistic Regression

- Let $f_\theta(\underline{X}) = \underline{X}^\top \beta + \beta^{(0)}$ with $\theta = (\beta, \beta^{(0)})$.
- Let $\mathbb{P}_\theta(Y = 1|\underline{X}) = e^{f_\theta(\underline{X})}/(1 + e^{f_\theta(\underline{X})})$
- Estimate θ by $\hat{\theta}$ using a Maximum Likelihood.
- Classify using $\mathbb{P}_{\hat{\theta}}(Y = 1|\underline{X}) > 1/2$

k Nearest Neighbors

- For any $\underline{X}'$, define $\mathcal{V}_{\underline{X}'}$ as the k closest samples X_i from the dataset.
- Compute a score $g_k = \sum_{X_i \in \mathcal{V}_{\underline{X}'}} \mathbf{1}_{Y_i = k}$
- Classify using $\arg\max g_k$ (majority vote).

Three Classical Methods in a Nutshell

Quadratic Discrimant Analysis

- For each class, estimate the mean μ_k and the covariance matrix Σ_k.
- Estimate the proportion $\mathbb{P}(Y=k)$ of each class.
- Compute a score $\ln(\mathbb{P}(\underline{X}|Y=k)) + \ln(\mathbb{P}(Y=k))$

$$\begin{aligned} g_k(\underline{X}) = &-\frac{1}{2}(\underline{X}-\mu_k)^\top \Sigma_k^{-1}(\underline{X}-\mu_k) \\ &-\frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln(|\Sigma_k|) + \ln(\mathbb{P}(Y=k)) \end{aligned}$$

- Classify using $\arg\max g_k$

- Those three methods rely on a similar heuristic: the probabilistic point of view!
- Focus on classification, but similar methods for regression: Gaussian Regression, k Nearest Neighbors, Gaussian Processes...

Best Solution

- The best solution $f^\star$ (which is independent of $\mathcal{D}_n$) is

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

Bayes Predictor (explicit solution)

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

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

- In regression with the quadratic loss

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

Issue: Explicit solution requires to **know** $Y|\underline{X}$ for all values of $\underline{X}$!

Plugin Predictor

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

Plugin Bayes Predictor

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

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

- In regression with the quadratic loss

$$\widehat{f}(\underline{X}) = \mathbb{E}\left[\widehat{Y|\underline{X}}\right]$$

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

Plugin Predictor

- How to estimate $Y|\underline{X}$?

Three main heuristics

- **Parametric Conditional modeling:** Estimate the law of $Y|\underline{X}$ by a **parametric** law $\mathcal{L}_\theta(\underline{X})$: *(generalized) linear regression...*
- **Non Parametric Conditional modeling:** Estimate the law of $Y|\underline{X}$ by a **non parametric** estimate: *kernel methods, loess, nearest neighbors...*
- **Fully Generative modeling:** Estimate the law of $(\underline{X}, Y)$ and use the **Bayes formula** to deduce an estimate of $Y|\underline{X}$: *LDA/QDA, Naive Bayes, Gaussian Processes...*

- More than one loss can be minimized for a given estimate of $Y|X$ (quantiles, cost based loss...)

Plugin Classifier

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

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

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

- Can we guaranty that the classifier is good if $Y|\underline{X}$ is well estimated?

Classification Risk Analysis

Theorem

- If $\widehat{f} = \text{sign}(2\widehat{p}_{+1} - 1)$ then

$$\begin{aligned}\mathbb{E}\left[\ell^{0,1}(Y, \widehat{f}(\underline{X}))\right] - \mathbb{E}\left[\ell^{0,1}(Y, f^\star(\underline{X}))\right] &\leq \mathbb{E}\left[\|\widehat{Y|\underline{X}} - Y|\underline{X}\|_1\right] \\ &\leq \left(\mathbb{E}\left[2\,\text{KL}(Y|\underline{X}, \widehat{Y|\underline{X}})\right]\right)^{1/2}\end{aligned}$$

- If one estimates $\mathbb{P}(Y = 1|\underline{X})$ well then one estimates $f^\star$ well!
- Link between a *conditional density estimation* task and a *classification* one!
- **Rk:** Conditional density estimation is more complicated than classification:
  - Need to be good for all values of $\mathbb{P}(Y = 1|\underline{X})$ while the classification task focus on values around the decision boundary.
  - But several losses can be optimized simultaneously jointly.
- In **regression**, (often) direct control of the quadratic loss. . .

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Parametric Conditional Density Models

- **Idea:** Estimate directly $Y|\underline{X}$ by a parametric conditional density $\mathbb{P}_\theta(Y|\underline{X})$.

Maximum Likelihood Approach

- Classical choice for θ:

$$\widehat{\theta} = \operatorname*{argmin}_{\theta} - \sum_{i=1}^{n} \log \mathbb{P}_\theta(Y_i|\underline{X}_i)$$

- **Goal:** *Minimize* the Kullback-Leibler divergence between the conditional law of $Y|\underline{X}$ and $\mathbb{P}_\theta(Y|\underline{X})$

$$\mathbb{E}[\mathrm{KL}\left(Y|\underline{X}, \mathbb{P}_\theta(Y|\underline{X})\right)]$$

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

Linear Conditional Density Models

Linear Models

- **Classical choice:** $\theta = (\theta', \varphi)$
$$\mathbb{P}_\theta(Y|\underline{X}) = \mathbb{P}_{\underline{X}^\top \beta, \varphi}(Y)$$
- **Very strong modeling 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 $\mathbb{P}(Y=+1|\underline{X})$ by $h(\underline{X}^\top\beta+\beta^{(0)})$ with h non decreasing.
- $h(\underline{X}^\top\beta+\beta^{(0)}) > 1/2 \Leftrightarrow \underline{X}^\top\beta+\beta^{(0)} - h^{-1}(1/2) > 0$
- Linear Classifier: $\operatorname{sign}(\underline{X}^\top\beta+\beta^{(0)} - h^{-1}(1/2))$

Plugin Linear Classifier Estimation

- Classical choice for h:

$$h(t) = \frac{e^t}{1+e^t} \qquad \text{logit or logistic}$$

$$h(t) = F_{\mathrm{N}}(t) \qquad \text{probit}$$

$$h(t) = 1 - e^{-e^t} \qquad \text{log-log}$$

- Choice of the *best* β from the data.

- Extension to multi-class with multinomial parametric model.

Maximum Likelihood Estimate

Probabilistic Model

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

Maximum Likelihood Approach

- Minimization of the negative log-likelihood:

$$-\sum_{i=1}^{n}\log(\mathbb{P}(Y_i|\underline{X}_i)) = -\sum_{i=1}^{n}\left(\mathbf{1}_{Y_i=1}\log(h(\underline{X}_i{}^\top\beta))+\mathbf{1}_{Y_i=-1}\log(1-h(\underline{X}_i{}^\top\beta))\right)$$

- Minimization possible if h is regular...

Maximum Likelihood Estimate

KL Distance and negative log-likelihood

- *Natural* probalistic *distance*: Kullback-Leibler divergence

$$
\begin{aligned}
\mathrm{KL}(\mathcal{B}(\mathbb{P}(Y=1|\underline{X})), \mathcal{B}(h(\underline{X}^\top\beta))
&= \mathbb{E}_{\underline{X}}\left[\mathbb{P}(Y=1|\underline{X}) \log \frac{\mathbb{P}(Y=1|\underline{X})}{h(\underline{X}^\top\beta)}\right.\\
&\qquad \left. + \mathbb{P}(Y=-1|\underline{X}) \log \frac{1-\mathbb{P}(Y=1|\underline{X})}{1-h(\underline{X}^\top\beta)}\right]\\
&= \mathbb{E}_{\underline{X}}\left[-\mathbb{P}(Y=1|\underline{X}) \log(h(\underline{X}^\top\beta))\right.\\
&\qquad \left. -\mathbb{P}(Y=-1|\underline{X}) \log(1-h(\underline{X}^\top\beta))\right] + C_{\underline{X},Y}
\end{aligned}
$$

- Empirical counterpart = negative log-likelihood (up to $1/n$ factor):

$$
-\frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{1}_{Y_i=1}\log(h(\underline{X}_i{}^\top\beta)) + \mathbf{1}_{Y_i=-1}\log(1-h(\underline{X}_i{}^\top\beta))\right)
$$

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
$$\frac{\mathbb{P}(Y=1)}{\mathbb{P}(Y=-1)} = e^t \Leftrightarrow \log \frac{\mathbb{P}(Y=1)}{\mathbb{P}(Y=-1)} = t$$
- Interpretation in term of odd.
- Logistic model: linear model on the logarithm of the odd
$$\log \frac{\mathbb{P}(Y=1|\underline{X})}{\mathbb{P}(Y=-1|\underline{X})} = \underline{X}^\top \beta$$

Associated Classifier

- Plugin strategy:
$$f_\beta(\underline{X}) = \begin{cases} 1 & \text{if } \frac{e^{\underline{X}^\top \beta}}{1+e^{\underline{X}^\top \beta}} > 1/2 \Leftrightarrow \underline{X}^\top \beta > 0 \\ -1 & \text{otherwise} \end{cases}$$

Logistic Regression and Minimization

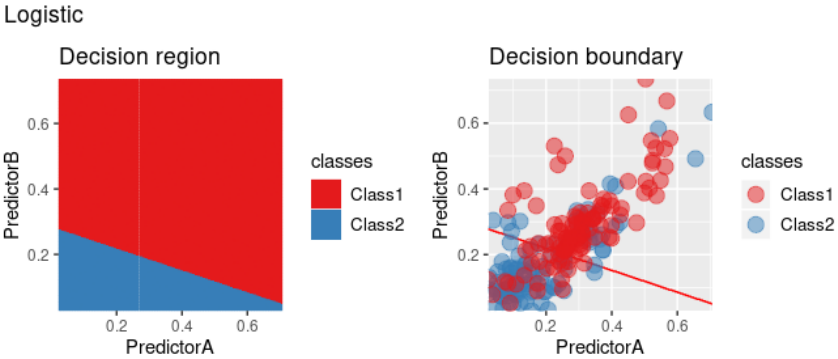
Likelihood Rewriting

- Negative log-likelihood:

$$
\begin{aligned}
&-\frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{1}_{Y_i=1}\log(h(\underline{X}_i{}^\top\beta)) + \mathbf{1}_{Y_i=-1}\log(1-h(\underline{X}_i{}^\top\beta))\right)\\
&=-\frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{1}_{Y_i=1}\log\frac{e^{\underline{X}_i{}^\top\beta}}{1+e^{\underline{X}_i{}^\top\beta}} + \mathbf{1}_{Y_i=-1}\log\frac{1}{1+e^{\underline{X}_i{}^\top\beta}}\right)\\
&=\frac{1}{n}\sum_{i=1}^{n}\log\left(1+e^{-Y_i(\underline{X}_i{}^\top\beta)}\right)
\end{aligned}
$$

- Convex and smooth function of β
- Easy optimization.

Example: Logistic



Feature Design

Transformed Representation

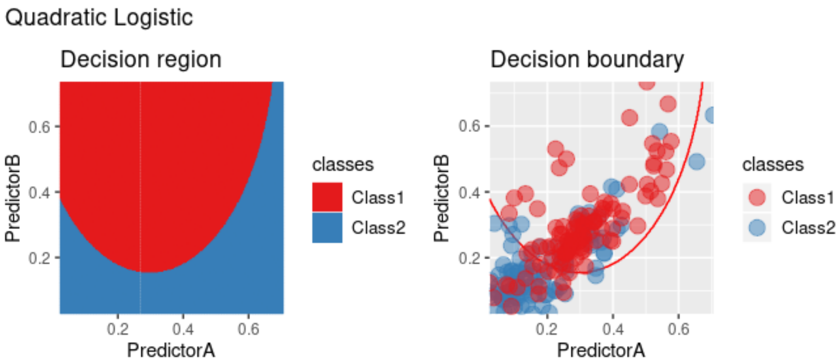
- From $\underline{X}$ to $\Phi(\underline{X})$!
- New description of $\underline{X}$ leads to a different **linear** model:

$$f_\beta(\underline{X}) = \Phi(\underline{X})^\top \beta$$

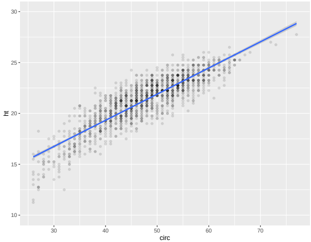
Feature Design

- Art of choosing Φ.
- Examples:
  - Renormalization, (domain specific) transform
  - Basis decomposition
  - Interaction between different variables...

Example: Quadratic Logistic



Gaussian Linear Regression



Gaussian Linear Model

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

- In regression, estimation of $\mathbb{E}[Y|\underline{X}]$ 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) $(v(\mathbb{E}_\theta[Y]) = \theta$ with v invertible).
- Exponential family: Probability law family P_θ such that the density can be written
$$f(y,\theta,\varphi) = e^{\frac{y\theta - v(\theta)}{\varphi} + w(y,\varphi)}$$
where φ is a nuisance parameter and w a function independent of θ.
- 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: $Y|\underline{X} \sim P_{\underline{x}^\top \beta}\ldots$

- Maximum likelihood fit of the parameters

Outline

1 Introduction
- Machine Learning
- Motivation

2 A Practical View
- Method or Models
- Interpretability
- Metric Choice

3 A Better Point of View
- The Example of Univariate Linear Regression
- Supervised Learning

4 Risk Estimation and Method Choice
- Cross Validation
- Cross Validation and Test
- Cross Validation and Weights
- Auto ML

5 A Probabilistic Point of View
- Parametric Conditional Density Modeling
- Non Parametric Conditional Density Modeling
- Generative Modeling

6 Optimization Point of View
- (Deep) Neural Networks
- Regularization
- Another Perspectivce on Bias-Variance Tradeoff
- SVM
- Tree Based Methods

7 Ensemble Methods
- Bagging and Random Forests
- Boosting

8 Empirical Risk Minimization
- Empirical Risk Minimization
- ERM and PAC Bayesian Analysis
- Hoeffding and Finite Class
- McDiarmid and Rademacher Complexity
- VC Dimension
- Structural Risk Minimization

9 References

Non Parametric Conditional Estimation

- **Idea:** Estimate $Y|\underline{X}$ directly without resorting to an explicit parametric model.

Non Parametric Conditional Estimation

- Two heuristics:
  - $Y|\underline{X}$ is almost constant (or simple) in a neighborhood of $\underline{X}$. (Kernel methods)
  - $Y|\underline{X}$ 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

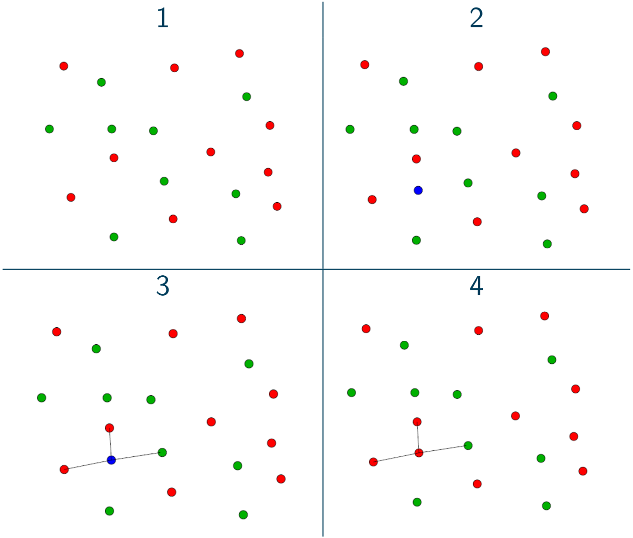
- **Idea:** The behavior of $Y|\underline{X}$ is locally *constant* or simple!

Kernel

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

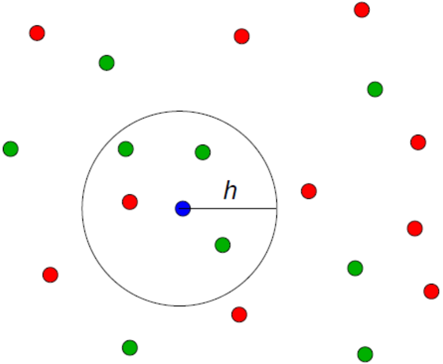
- In regression, an estimate of $\mathbb{E}[Y|\underline{X}]$ is easily obtained from an estimate of $Y|X$.
- Lazy learning: computation for a new point requires the full training dataset.

Example: k Nearest-Neighbors (with $k = 3$)



Source: Unknown

Example: k Nearest-Neighbors (with $k = 4$)



Source: Unknown

k Nearest-Neighbors

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

k-NN as local conditional density estimate

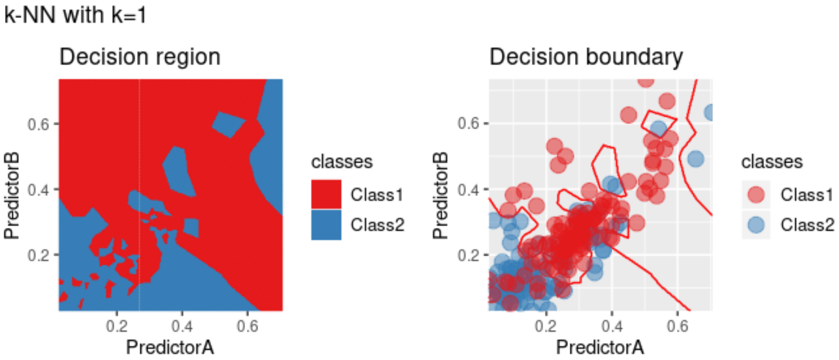
$$\widehat{\mathbb{P}(Y=1|\underline{X})} = \frac{\sum_{\underline{X}_i \in \mathcal{V}_{\underline{X}}} \mathbf{1}_{\{Y_i=+1\}}}{|\mathcal{V}_{\underline{X}}|}$$

- KNN Classifier:

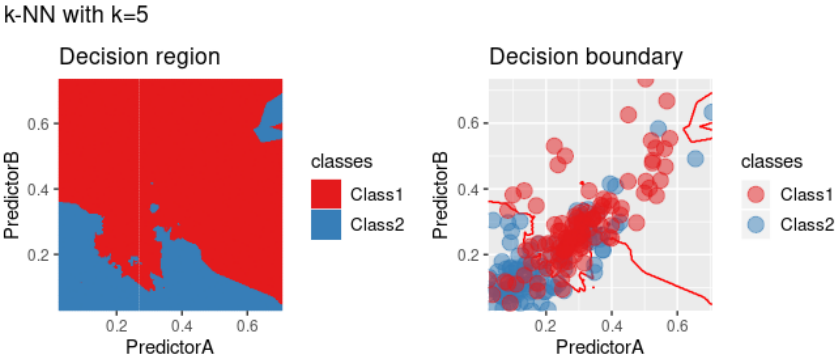
$$\widehat{f}_{KNN}(\underline{X}) = \begin{cases} +1 & \text{if } \widehat{\mathbb{P}(Y=1|\underline{X})} \geq \widehat{\mathbb{P}(Y=-1|\underline{X})} \\ -1 & \text{otherwise} \end{cases}$$

- **Lazy learning**: all the computations have to be done at prediction time.
- Easily extend to the multi-class setting.
- **Remark:** You can also use your favorite kernel estimator. . .

Example: KNN


k-NN with k=1
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: KNN

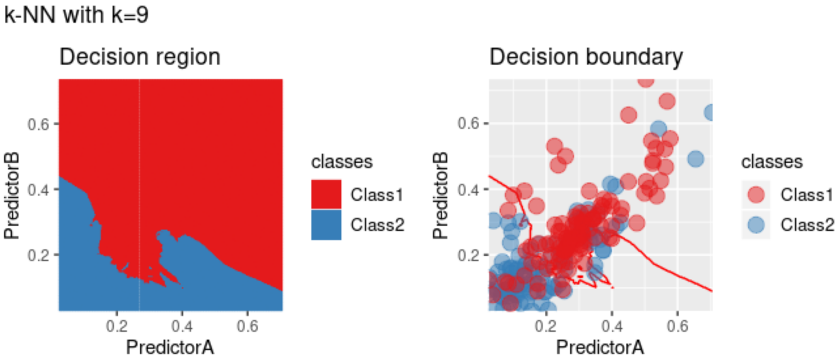


k-NN with k=5

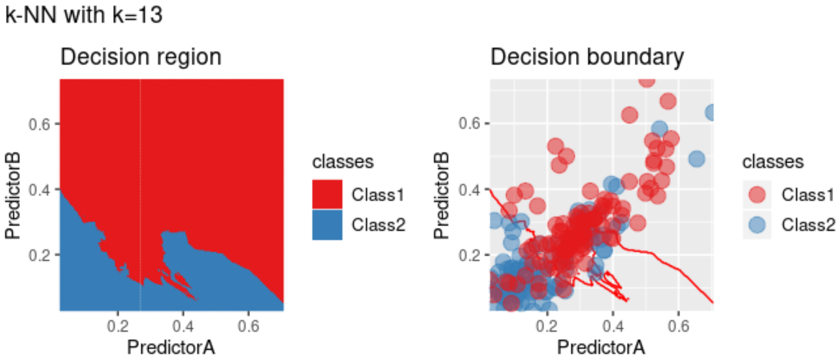
Decision region

Decision boundary

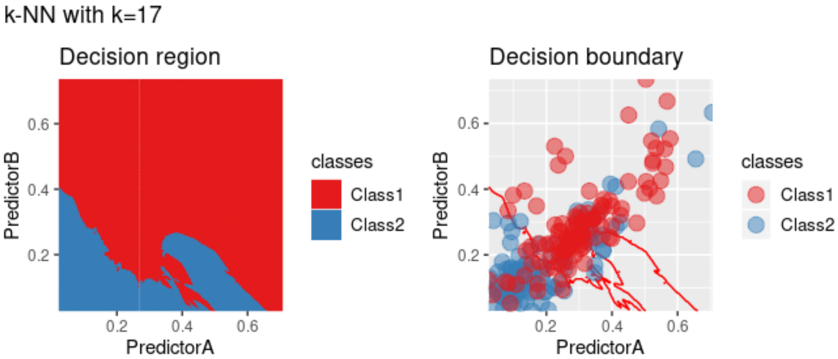
Example: KNN


k-NN with k=9
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

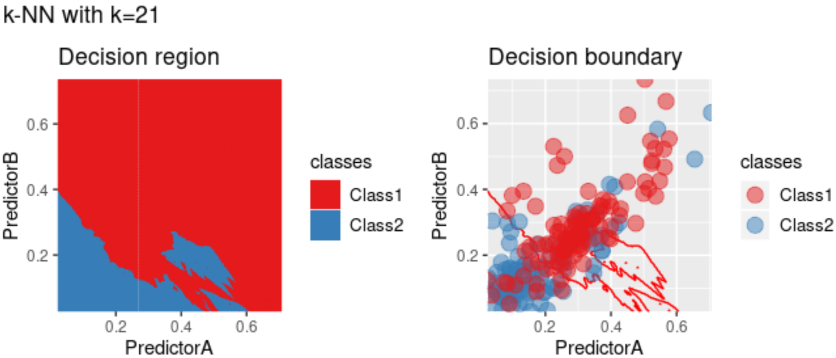
Example: KNN


k-NN with k=13
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

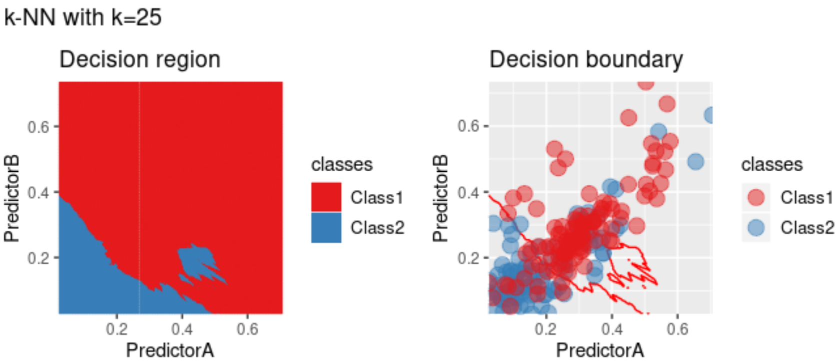
Example: KNN


k-NN with k=17
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
🟥 Class1
🟦 Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
● Class1
● Class2

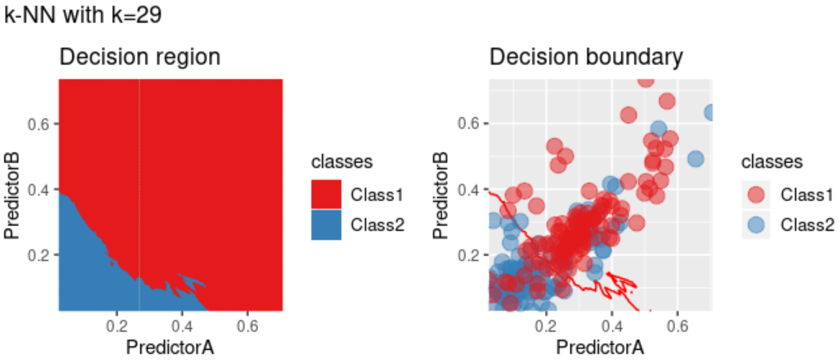
Example: KNN


k-NN with k=21
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

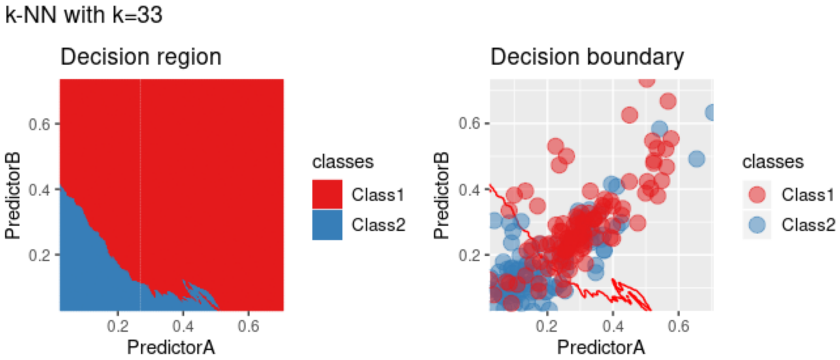
Example: KNN


k-NN with k=25
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

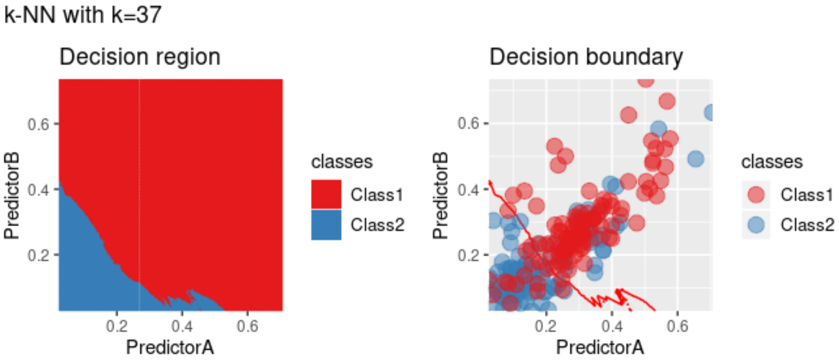
Example: KNN


k-NN with k=29
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

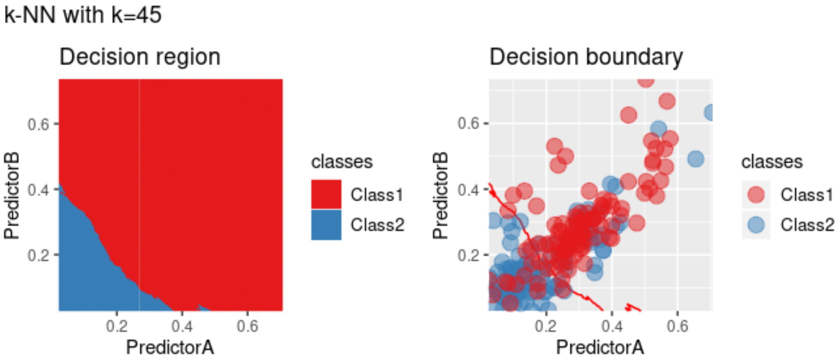
Example: KNN


k-NN with k=33
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

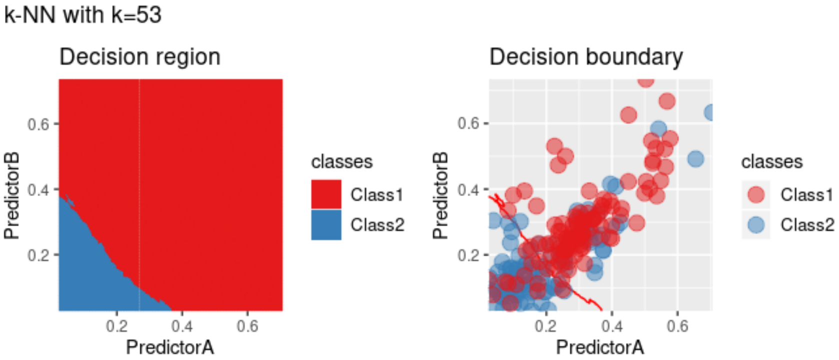
Example: KNN


k-NN with k=37
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: KNN

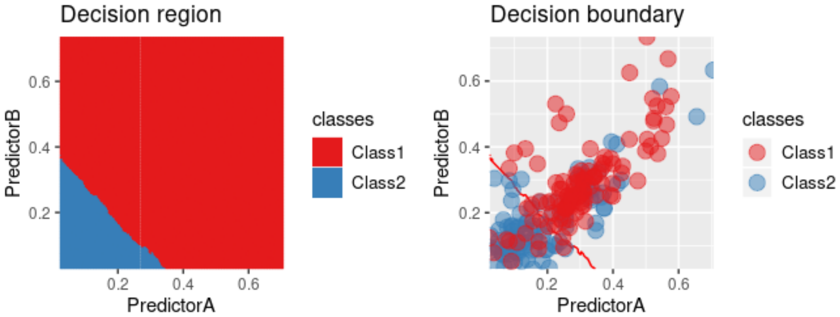

k-NN with k=45
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
🟥 Class1
🟦 Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
● Class1
● Class2

Example: KNN

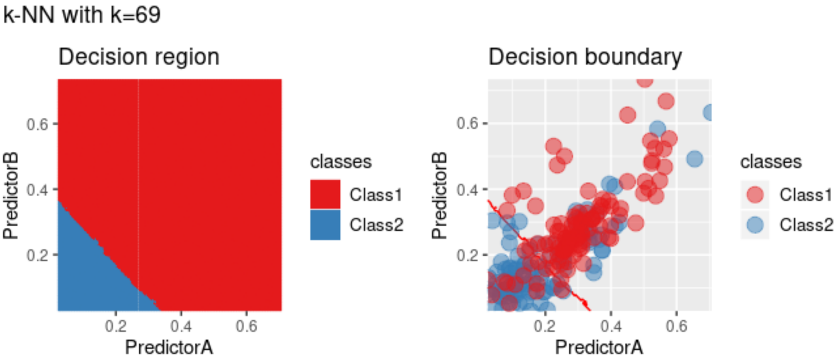

k-NN with k=53
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

Example: KNN

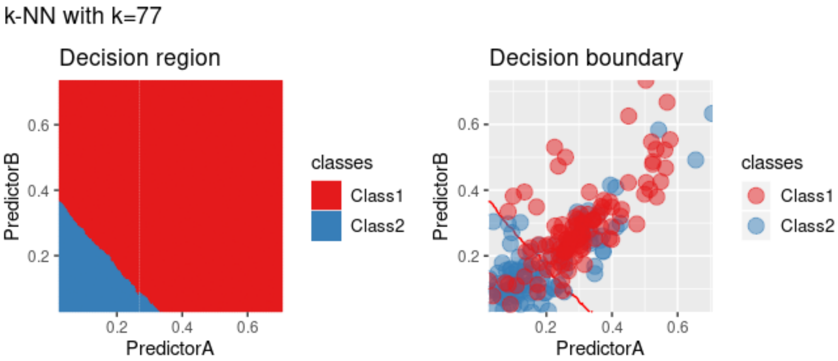
k-NN with k=61



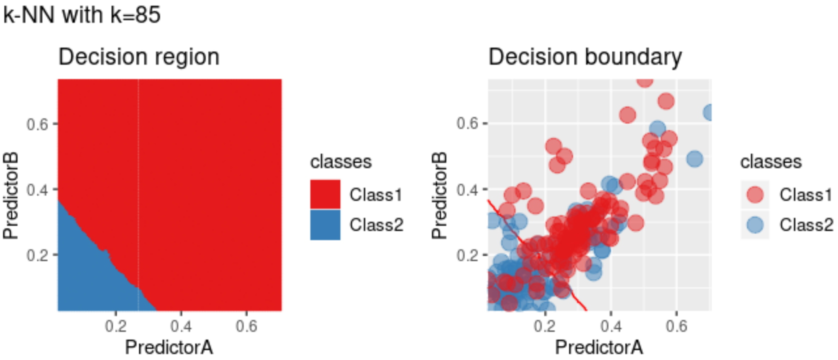
Example: KNN


k-NN with k=69
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

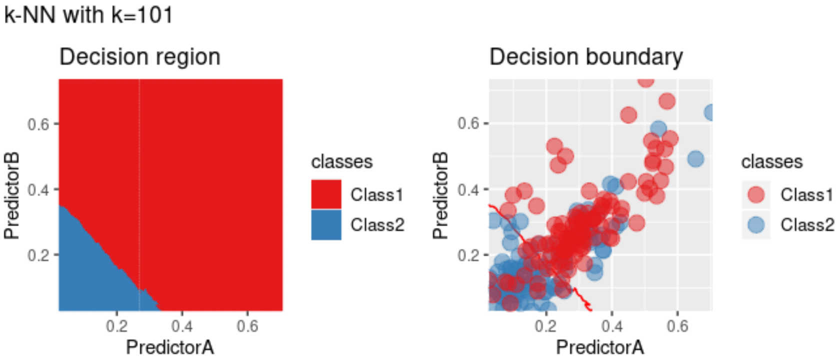
Example: KNN


k-NN with k=77
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

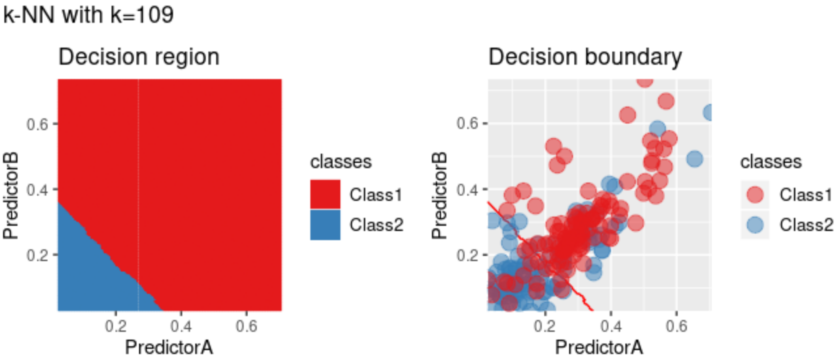
Example: KNN


k-NN with k=85
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

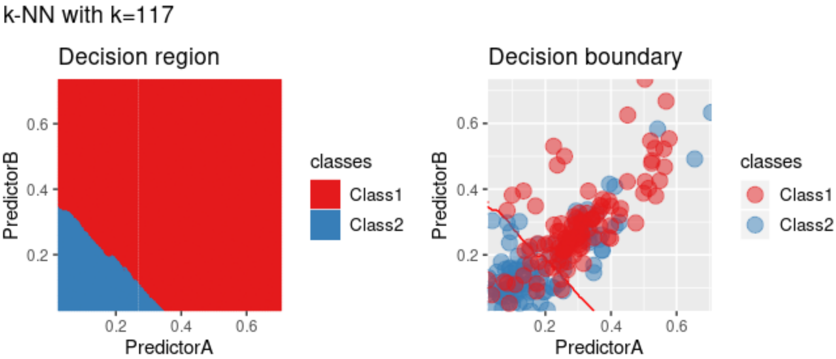
Example: KNN


k-NN with k=101
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

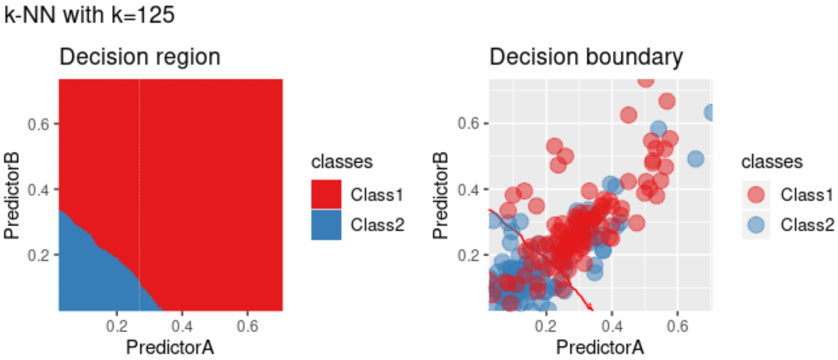
Example: KNN


k-NN with k=109
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: KNN

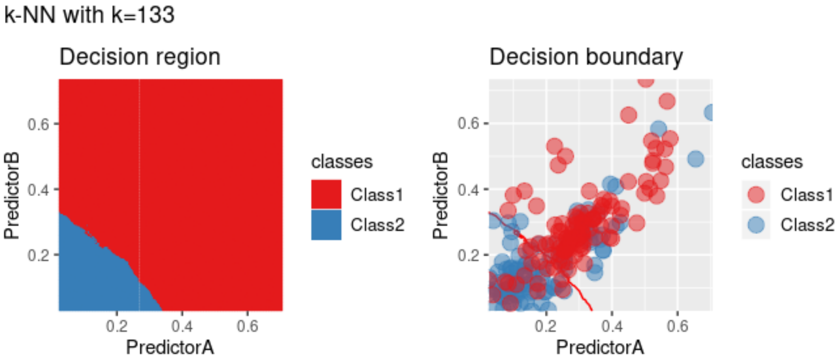

k-NN with k=117
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: KNN

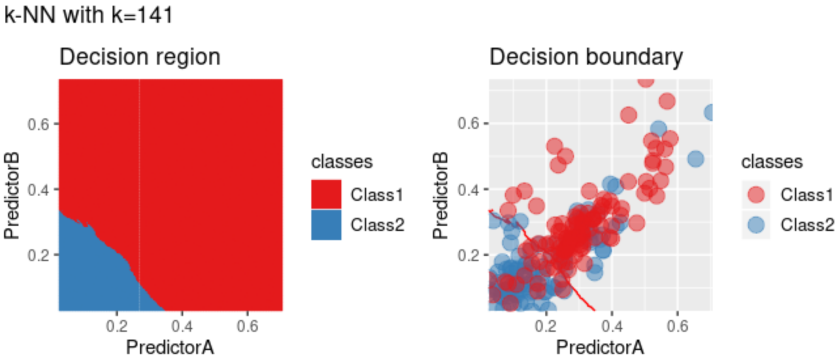


k-NN with k=125

Example: KNN


k-NN with k=133
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: KNN

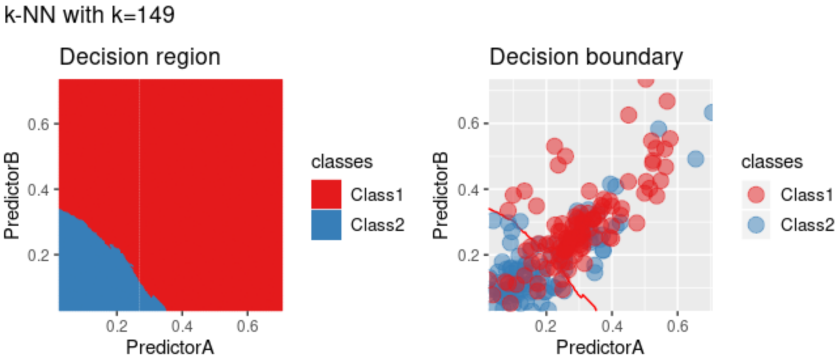


k-NN with k=141

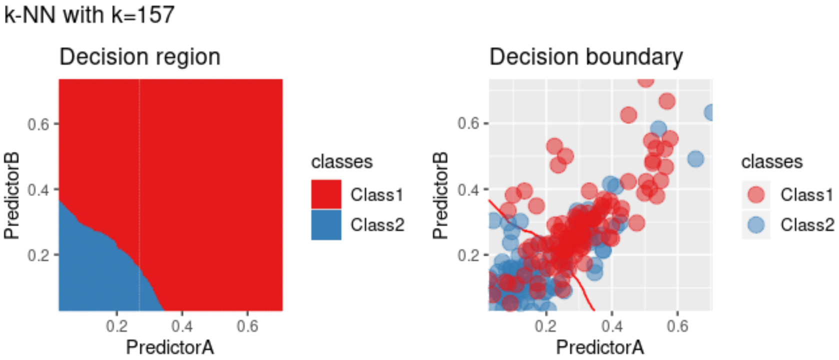
Decision region

Decision boundary

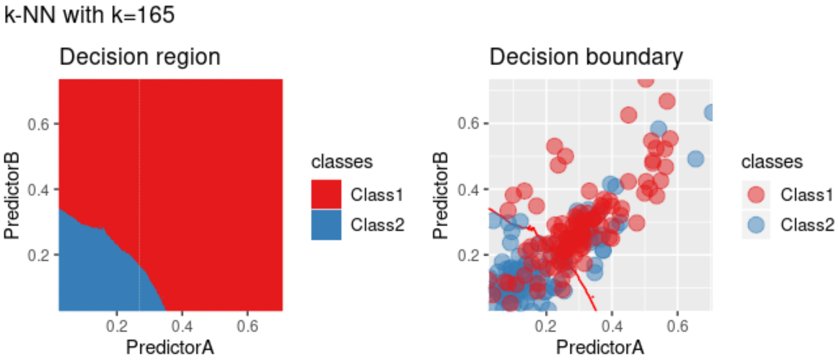
Example: KNN


k-NN with k=149
Decision region
0.6
0.4
0.2
PredictorB
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
0.6
0.4
0.2
PredictorB
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

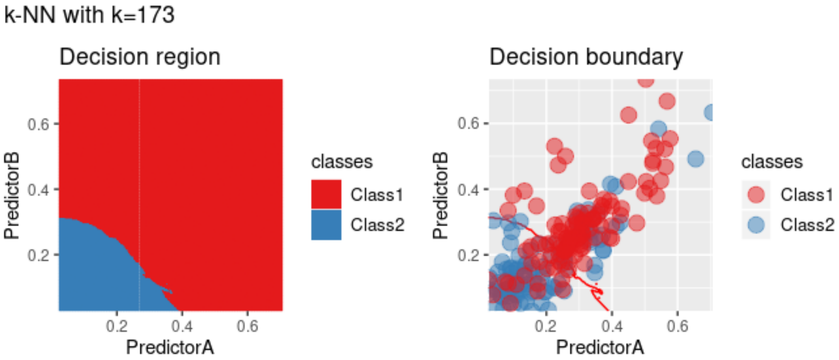
Example: KNN


k-NN with k=157
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

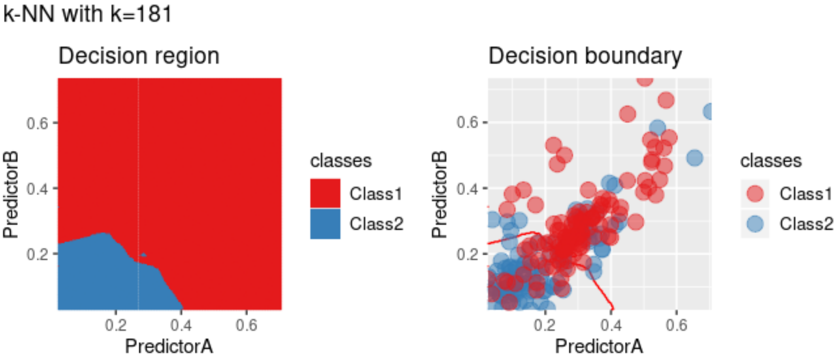
Example: KNN


k-NN with k=165
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: KNN

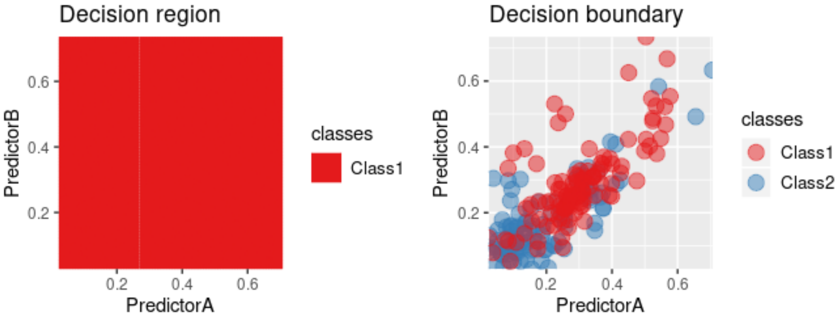

k-NN with k=173
Decision region
0.6
0.4
0.2
PredictorB
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
0.6
0.4
0.2
PredictorB
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: KNN

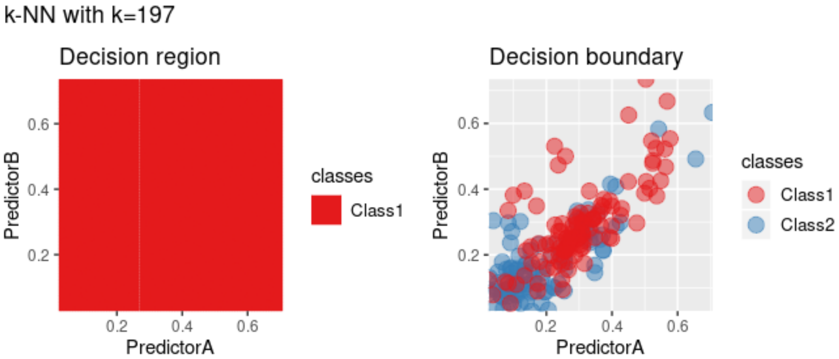

k-NN with k=181
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

Example: KNN

k-NN with k=189



Example: KNN


k-NN with k=197
Decision region
0.6
0.4
0.2
PredictorB
0.2
0.4
0.6
PredictorA
classes
Class1
Decision boundary
0.6
0.4
0.2
PredictorB
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Regression and Local Averaging

A naive idea

- $\mathbb{E}[Y|\underline{X}]$ can be approximated by a local average in a neighborhood $\mathcal{N}(\underline{X})$ of $\underline{X}$:

$$\widehat{f}(\underline{X}) = \frac{1}{|\{\underline{X}_i \in \mathcal{N}(\underline{X})\}|} \sum_{\underline{X}_i \in \mathcal{N}(\underline{X})} Y_i$$

- **Heuristic:**
  - If $\underline{X} \to \mathbb{E}[Y|\underline{X}]$ is regular then

$$\mathbb{E}[Y|\underline{X}] \simeq \mathbb{E}\left[\mathbb{E}\left[Y|\underline{X}'\right]|\underline{X}' \in \mathcal{N}(\underline{X})\right] = \mathbb{E}\left[Y|\underline{X}' \in \mathcal{N}(\underline{X})\right]$$

  - Replace an expectation by an empirical average:

$$\mathbb{E}\left[Y|\underline{X}' \in \mathcal{N}(\underline{X})\right] \simeq \frac{1}{|\{\underline{X}_i \in \mathcal{N}(\underline{X})\}|} \sum_{\underline{X}_i \in \mathcal{N}(\underline{X})} Y_i$$

Conditional Density Interpretation

- Amount to use as in classification,

$$\widehat{Y|X} = \frac{1}{|\{\underline{X}_i \in \mathcal{N}(\underline{X})\}|} \sum_{\underline{X}_i \in \mathcal{N}(\underline{X})} \mathbf{1}_{Y=Y_i}$$

Regression and Local Averaging

Neighborhood and Size

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

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 $\mathbb{E}[Y|\underline{X}] \simeq \mathbb{E}\left[Y|\underline{X}' \in \mathcal{N}(\underline{X})\right]$ is more accurate **(small bias)**.

Weighted Local Averaging

Weighted Local Average

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

$$\widehat{f}(\underline{X}) = \frac{\sum_i w(\underline{X}, \underline{X}'_i) Y_i}{\sum_i w(\underline{X}, \underline{X}'_i)}.$$

Kernel

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

A Density Estimation Point of View?

Nadaraya-Watson Heuristic

- Provided all the **densities** exist

$$Y|\underline{X} \sim \frac{p(\underline{X}, Y)}{p(\underline{X})} dY \qquad \text{and} \qquad \mathbb{E}[Y|\underline{X}] = \frac{\int Y p(\underline{X}, Y) dY}{(\underline{X})}$$

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

$$\widehat{p}(\underline{X}) = \frac{1}{n} \sum_{i=1}^{n} K(\underline{X} - \underline{X}_i)$$

$$\widehat{p}(\underline{X}, Y) = \frac{1}{n} \sum_{i=1}^{n} K(\underline{X} - \underline{X}_i) K'(Y - Y_i)$$

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

$$\int Y \widehat{p}(\underline{X}, Y) dY = \frac{1}{n} \sum_{i=1}^{n} K(\underline{X} - \underline{X}_i) Y_i$$

A Density Estimation Point of View?

Nadaraya-Watson

- Resulting estimator of $\mathbb{E}[Y|\underline{X}]$
$$\widehat{f}(\underline{X}) = \frac{\sum_{i=1}^{n} Y_i K_h(\underline{X} - \underline{X}_i)}{\sum_{i=1}^{n} K_h(\underline{X} - \underline{X}_i)}$$
- 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!

- Probabilistic approach POV!

Local Linear Estimation

Another Point of View on Kernel

- Nadaraya-Watson estimator:

$$\widehat{f}(\underline{X}) = \frac{\sum_{i=1}^{n} Y_i K_h(\underline{X} - \underline{X}_i)}{\sum_{i=1}^{n} K_h(\underline{X} - \underline{X}_i)}$$

- Can be view as a **minimizer** of

$$\sum_{i=1}^{n} |Y_i - \beta|^2 K_h(\underline{X} - \underline{X}_i)$$

- **Local regression** of order 0.

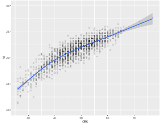
Local Linear Model

- Estimate $\mathbb{E}[Y|\underline{X}]$ by $\widehat{f}(\underline{X}) = \phi(\underline{X})^\top \widehat{\beta}(\underline{X})$ where ϕ is any function of $\underline{X}$ and $\widehat{\beta}(\underline{X})$ is the minimizer of

$$\sum_{i=1}^{n} |Y_i - \phi(\underline{X}_i)^\top \beta|^2 K_h(\underline{X} - \underline{X}_i).$$

- Very similar to a piecewise modeling approach.

LOESS: LOcal polynomial regrESSion



1D Nonparametric Regression

- Assume that $\underline{X} \in \mathbb{R}$ and let $\phi(\underline{X}) = (1, \underline{X}, \ldots, \underline{X}^d)$.
- **LOESS estimate:** $\widehat{f}(\underline{X}) = \sum_{j=0}^{d} \widehat{\beta}(\underline{X}^{(j)})\underline{X}^j$ with $\widehat{\beta}(\underline{X})$ minimizing
$$\sum_{i=1}^{n} |Y_i - \sum_{j=0}^{d} \beta^{(j)} \underline{X}_i^j|^2 K_h(\underline{X} - \underline{X}_i).$$
- Most classical kernel used: Tricubic kernel
$$K(t) = \max(1 - |t|^3, 0)^3$$
- Most classical degree: 2...
- Local bandwidth choice such that a proportion of points belongs to the window.

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Fully Generative Modeling

- **Idea:** If one knows the law of $(\underline{X}, Y)$ everything is easy!

Bayes formula

- With a slight abuse of notation,

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

- **Generative Modeling:**
  - Propose a model for $(\underline{X}, Y)$ (or equivalently $\underline{X}|Y$ 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 $(\underline{X}, Y)$ rather than only $Y|\underline{X}$!
- Great flexibility in the model design but may lead to complex computation.

Fully Generative Modeling

- Simpler setting in classification!

Bayes formula

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

- Binary Bayes classifier (the best solution)

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

- **Heuristic**: Estimate those quantities and plug the estimations.
- By using different models/estimators for $\mathbb{P}(\underline{X}|Y)$, we get different classifiers.
- **Rk:** No need to renormalize by $\mathbb{P}(\underline{X})$ to take the decision!

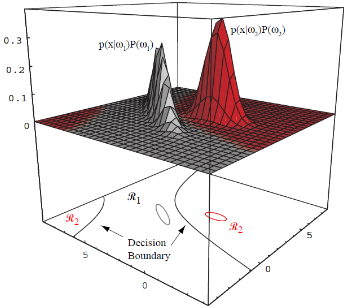
Discriminant Analysis

Discriminant Analysis (Gaussian model)

- The densities are modeled as multivariate normal, i.e.,
$$\mathbb{P}(\underline{X}|Y=k) \sim \mathrm{N}_{\mu_k,\Sigma_k}$$
- Discriminant functions: $\boldsymbol{g_k(\underline{X}) = \ln(\mathbb{P}(\underline{X}|Y=k)) + \ln(\mathbb{P}(Y=k))}$
$$\begin{aligned} g_k(\underline{X}) = &-\frac{1}{2}(\underline{X}-\mu_k)^\top \Sigma_k^{-1}(\underline{X}-\mu_k) \\ &-\frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln(|\Sigma_k|) + \ln(\mathbb{P}(Y=k)) \end{aligned}$$
- QDA (different Σ_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

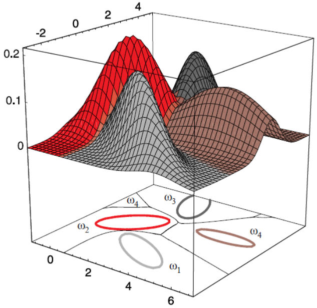


Quadratic Discriminant Analysis

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

Source: A. Fermin

Discriminant Analysis



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

Source: A. Fermin

Discriminant Analysis

Estimation

In practice, we will need to estimate μ_k, Σ_k and $\mathbb{P}_k := \mathbb{P}(Y = k)$

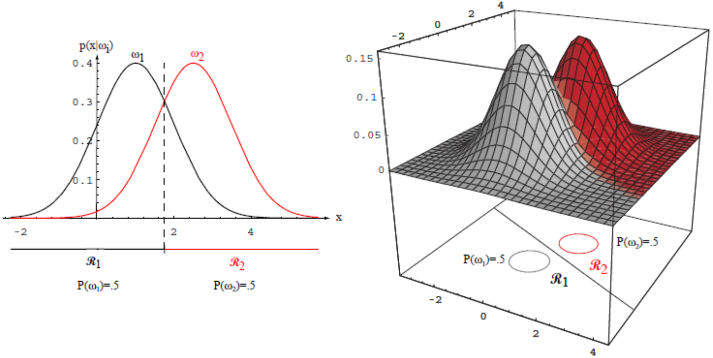
- The estimate proportion $\mathbb{P}\widehat{(Y = k)} = \frac{n_k}{n} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{\{Y_i = k\}}$
- Maximum likelihood estimate of $\widehat{\mu_k}$ and $\widehat{\Sigma_k}$ (explicit formulas)

- DA classifier

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

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

Discriminant Analysis

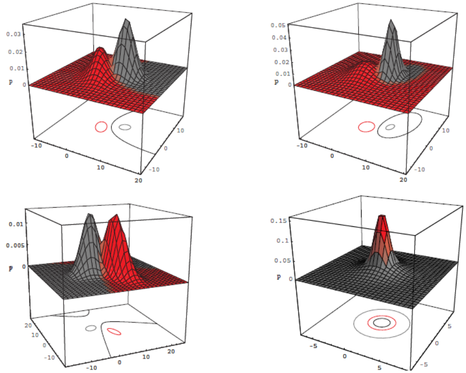


Linear Discriminant Analysis

- $\Sigma_{\omega_1} = \Sigma_{\omega_2} = \Sigma$
- The decision boundaries are linear hyperplanes

Source: A. Fermin

Discriminant Analysis

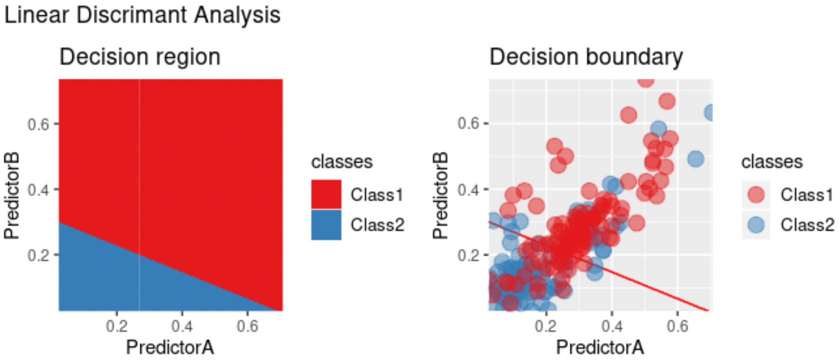


Quadratic Discriminant Analysis

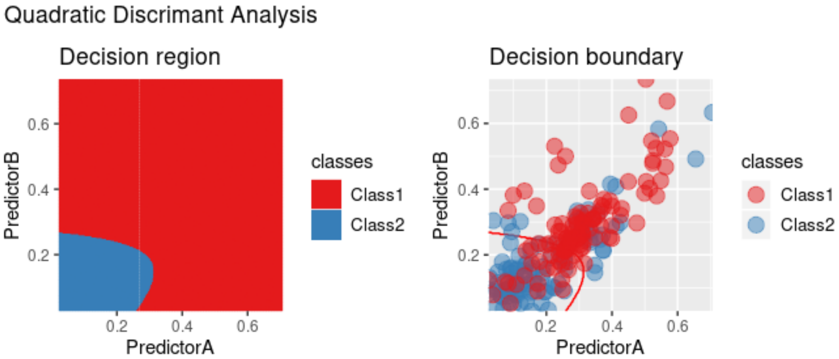
- $\Sigma_{\omega_1} \neq \Sigma_{\omega_2}$
- Arbitrary Gaussian distributions lead to Bayes decision boundaries that are general quadratics.

Source: A. Fermin

Example: LDA


Linear Discriminant Analysis
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Example: QDA


Quadratic Discrimant Analysis
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

Naive Bayes

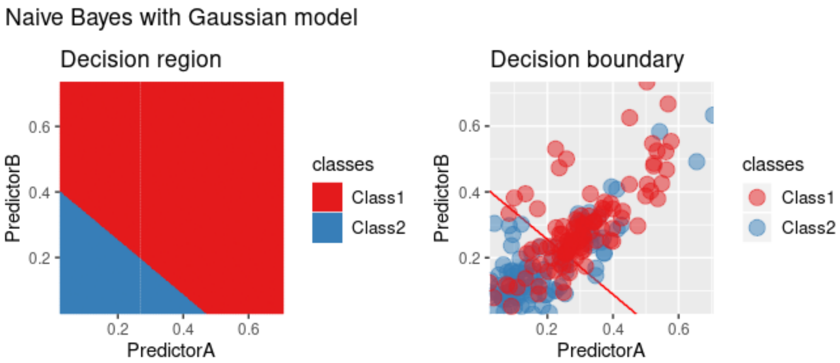
Naive Bayes

- Classical algorithm using a crude modeling for $\mathbb{P}(\underline{X}|Y)$:
  - Feature **independence** assumption:

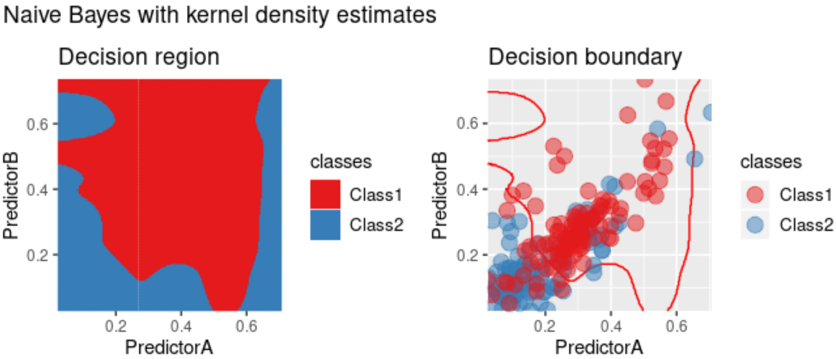
$$\mathbb{P}(\underline{X}|Y) = \prod_{l=1}^{d} \mathbb{P}\left(\underline{X}^{(l)}\middle|Y\right)$$

  - 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!**

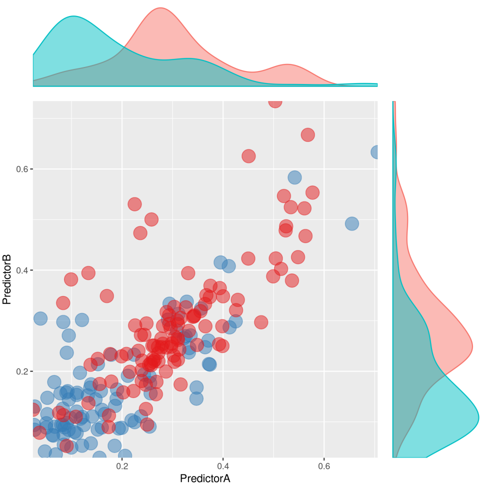
Example: Naive Bayes



Example: Naive Bayes



Naive Bayes with Density Estimation



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. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Probabilistic and Optimization Framework

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

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

Canonical approach: $\widehat{f}_{\mathcal{S}} = \operatorname{argmin}_{f \in \mathcal{S}} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(\underline{X}_i))$

Problems

- How to choose $\mathcal{S}$?
- How to compute the minimization?

A Probabilistic Point of View

Solution: For $\underline{X}$, estimate $Y|\underline{X}$ and plug it in any Bayes classifier: **(Generalized) Linear Models, Kernel methods, k-nn, Naive Bayes, Tree, Bagging...**

An Optimization Point of View

Solution: Replace the loss ℓ by an upper bound $\bar{\ell}$ and minimize directly the corresponding emp. risk: **Neural Network, SVR, SVM, Tree, Boosting...**

Three Classical Methods in a Nutshell

Deep Learning

- Let $f_\theta(\underline{X})$ 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 $-\frac{1}{n}\sum_{i=1}^{n}\log\left(f_\theta(\underline{X}_i)^{(Y_i)}\right)$
- Classify using $\operatorname{sign}(f_{\hat{\theta}})$

Regularized Logistic Regression

- Let $f_\theta(\underline{X}) = \underline{X}^\top\beta + \beta^{(0)}$ with $\theta = (\beta, \beta^{(0)})$.
- Find $\hat{\theta} = \arg\min \frac{1}{n}\sum_{i=1}^{n}\log\left(1 + e^{-Y_i f_\theta(\underline{X}_i)}\right) + \lambda\|\beta\|_1$
- Classify using $\operatorname{sign}(f_{\hat{\theta}})$

Three Classical Methods in a Nutshell

Support Vector Machine

- Let $f_\theta(\underline{X}) = \underline{X}^\top \beta + \beta^{(0)}$ with $\theta = (\beta, \beta^{(0)})$.
- Find $\hat{\theta} = \arg\min \frac{1}{n} \sum_{i=1}^{n} \max\left(1 - Y_i f_\theta(\underline{X}_i), 0\right) + \lambda \|\beta\|_2^2$
- Classify using $\operatorname{sign}(f_{\hat{\theta}})$

- Those three methods rely on a similar heuristic: the optimization point of view!
- Focus on classification, but similar methods for regression: Deep Learning, Regularized Regrssion, Support Vector Regression. . .

Empirical Risk Minimization

- The best solution $f^\star$ is the one minimizing

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

Empirical Risk Minimization

- 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 average empirical loss

$$\widehat{f} = f_{\widehat{\theta}} = \operatorname*{argmin}_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(\underline{X}_i))$$

- Often tractable for the quadratic loss in regression.
- Intractable for the $0/1$ loss in classification!

Convexification Strategy

Risk Convexification

- Replace the loss $\ell(Y, f_\theta(\underline{X}))$ by a convex upperbound $\bar{\ell}(Y, f_\theta(\underline{X}))$ (surrogate loss).
- Minimize the average of the surrogate empirical loss

$$\tilde{f} = f_{\widehat{\theta}} = \operatorname*{argmin}_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \bar{\ell}(Y_i, f_\theta(\underline{X}_i))$$

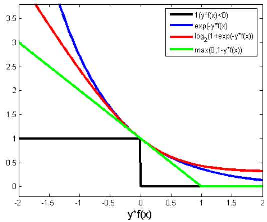
- Use $\widehat{f} = \operatorname{sign}(\tilde{f})$

- Much easier optimization.

Instantiation

- Logistic (Revisited)
- (Deep) Neural Network
- Support Vector Machine
- Boosting

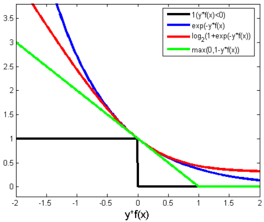
Classification Loss and Convexification



Convexification

- Replace the loss $\ell^{0/1}(Y, f(\underline{X}))$ by
$$\bar{\ell}(Y, f(\underline{X})) = l(Yf(\underline{X}))$$
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: $\bar{\ell}(Y, f(\underline{X})) = \log_2(1 + e^{-Yf(\underline{X})})$ (Logistic / NN)
- Hinge loss: $\bar{\ell}(Y, f(\underline{X})) = (1 - Yf(\underline{X}))_+$ (SVM)
- Exponential loss: $\bar{\ell}(Y, f(\underline{X})) = e^{-Yf(\underline{X})}$ (Boosting. . .)

Properties

The Target is the Bayes Classifier

- The minimizer of
$$\mathbb{E}\left[\bar{\ell}(Y, f(\underline{X}))\right] = \mathbb{E}[l(Yf(\underline{X}))]$$
is the Bayes classifier $f^\star = \operatorname{sign}(2\eta(\underline{X}) - 1)$

Control of the Excess Risk

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

- Multi-class generalizations of convexification lead to similar controls, but not necessarily a direct upper bound of the loss.
- Direct (approximate) optimization of the predictor, but for a single loss.
- Connection with the probabilistic POV when the (surrogate) loss used is the opposite of the log-likelihood.

Logistic Revisited

- Ideal solution:

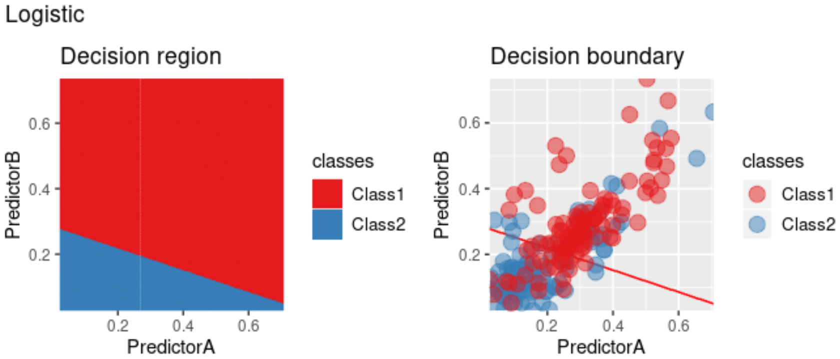
$$\widehat{f} = \operatorname*{argmin}_{f \in \mathcal{S}} \frac{1}{n} \sum_{i=1}^{n} \ell^{0/1}(Y_i, f(\underline{X}_i))$$

Logistic regression

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

- Different vision than the statistician but same algorithm!
- In regression, a similar approach will be to minimize the least square criterion without making the Gaussian noise assumption.

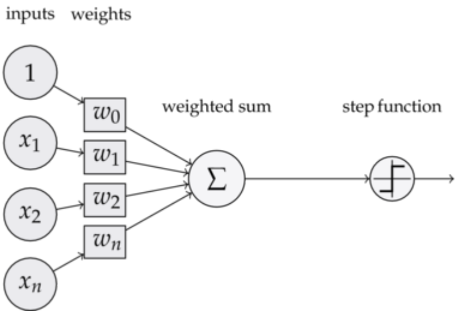
Logistic Revisited



Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Perceptron

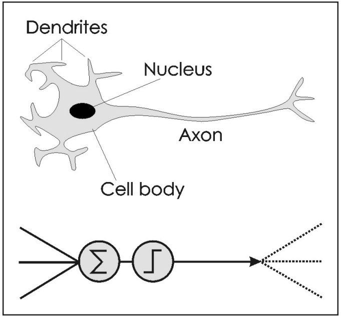


Perceptron (Rosenblatt 1957)

- Inspired from biology.
- Very simple (linear) model!
- Physical implementation and proof of concept.

Source: Tikz

Perceptron

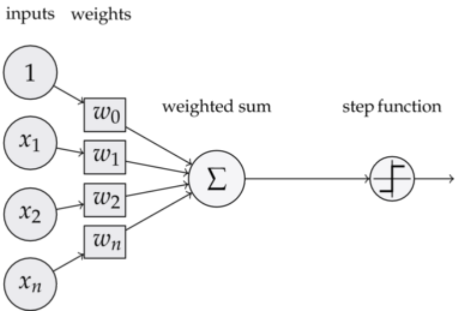


Perceptron (Rosenblatt 1957)

- Inspired from biology.
- Very simple (linear) model!
- Physical implementation and proof of concept.

Source: Tikz

Perceptron

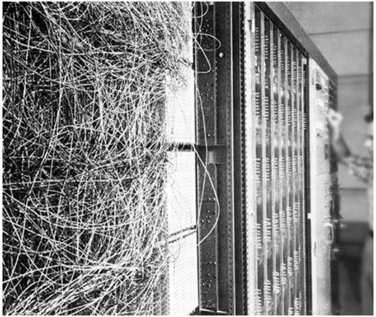


Perceptron (Rosenblatt 1957)

- Inspired from biology.
- Very simple (linear) model!
- Physical implementation and proof of concept.

Source: Tikz

Perceptron

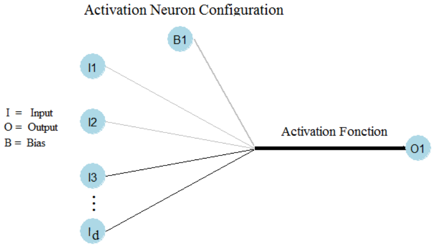


Source: Avin Calspan Advanced Technology Center

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.

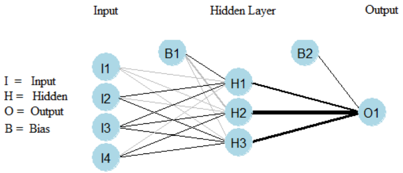
Logistic unit

- Structure:
  - Mix inputs with a **weighted sum**,
  - Apply the **logistic function** $\sigma(t) = e^t/(1+e^t)$,
  - Threshold at 1/2 to make a decision!
- Logistic weights learned by minimizing the -log-likelihood.

- Equivalent to linear regression when using a linear activation function!

Source: Unknown

Multilayer Perceptron



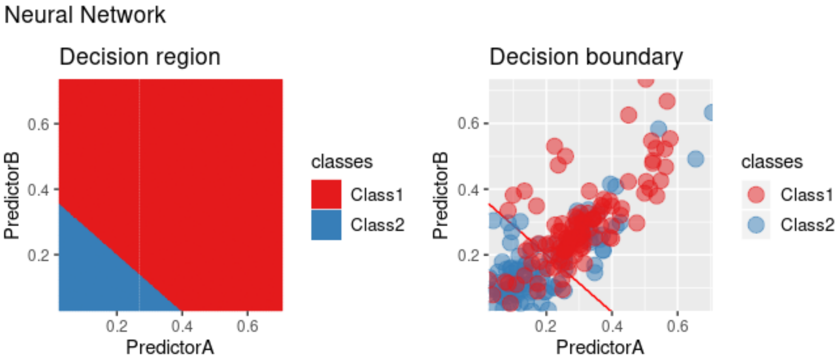
MLP (Rumelhart, McClelland, Hinton - 1986)

- 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.
- Minimized loss chosen among the classical losses in both classification and regression.
- Non convex optimization problem!

Source: Unknown

Multilayer Perceptron



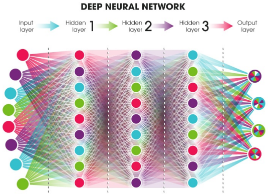
Universal Approximation Theorem

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.

Deep Neural Network



neuralnetworksanddeeplearning.com - Michael Nielsen, Yoshua Bengio, Ian Goodfellow, and Aaron Courville, 2016.

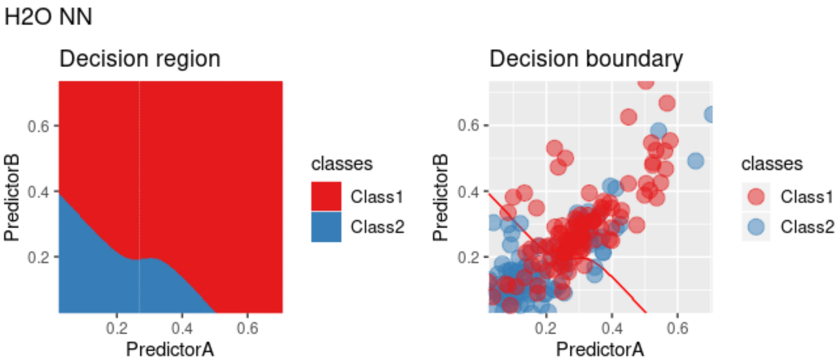
Deep Neural Network structure

- Deep cascade of layers!

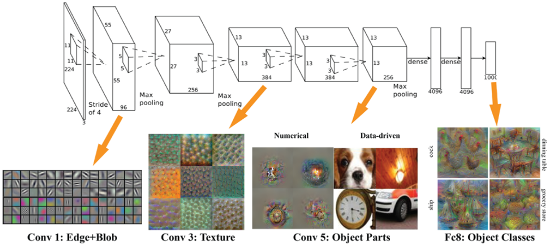
- No conceptual novelty. . .
- But a **lot of tricks** allowing to obtain a good solution: clever initialization, better activation function, weight regularization, accelerated stochastic gradient descent, early stopping. . .
- Use of GPU and a lot of data. . .
- Very impressive results!

Source: Nielsen, Bengio, Goodfellow and Courville

Deep Neural Network



Deep Learning



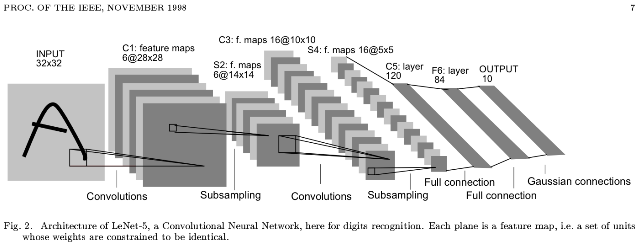
Family of Machine Learning algorithm combining:

- 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 a pretrained net as initialization.
- Very efficient and still evolving!

Source: J. Hays

Convolutional Network



PROC. OF THE IEEE, NOVEMBER 1998 7

Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

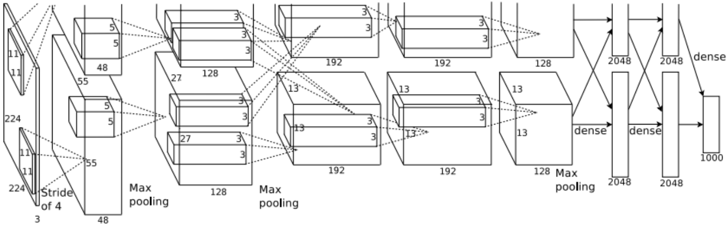
Le Net - Y. LeCun (1989)

- 6 hidden layer architecture.
- Drastic reduction of the number of parameters through a translation invariance principle (convolution).

- Required 3 days of training for 60 000 examples!
- Tremendous improvement.
- Representation learned through the task.

Source: Y. LeCun

Deep Convolutional Networks



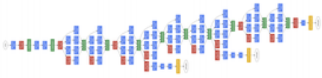
Alexnet - A. Krizhevsky, I. Sutskever, G. Hinton (2012)

- Bigger and deeper layers and thus much more parameters.
- Clever intialization scheme, RELU, renormalization and use of GPU.

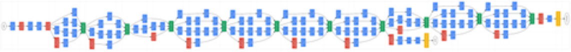
- 6 days of training for 1.2 millions images.
- Tremendous improvement. . .

Source: A. Krizhevsky

Deep Convolutional Networks

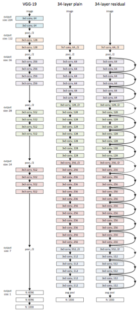


[1]Inception 5 (GoogLeNet)



Inception 7a

[1]Going Deeper with Convolutions, [C. Szegedy et al, CVPR 2015]



Trends

- Bigger and bigger networks! (GoogLeNet / Residual Neural Network / Transformers. . .)
- More computational power to learn better representation.

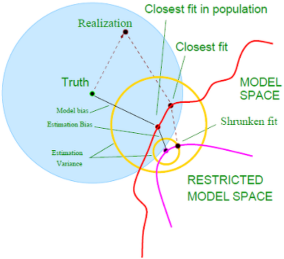
- Work in Progess!

Sources: Google / Microsoft

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. **Optimization Point of View**
   - (Deep) Neural Networks
   - **Regularization**
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

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 $h(\underline{x}^\top \beta)$.

Model coefficients

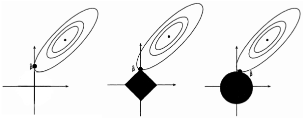
- Model entirely specified by β.
- Coefficientwise:
  - $\beta^{(i)} = 0$ means that the ith covariate is not used.
  - $\beta^{(i)} \sim 0$ means that the ith covariate as a *low* influence...

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

Submodels

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

Norms and Sparsity



Sparsity

- β is sparse if its number of non-zero coefficients (ℓ_0) is small. . .
- Easy interpretation in terms of dimension/complexity.

Norm Constraint and Sparsity

- Sparsest solution obtained by definition with the ℓ_0 norm.
- No induced sparsity with the ℓ_2 norm. . .
- Sparsity with the ℓ_1 norm (can even be proved to be the same as with the ℓ_0 norm under some assumptions).
- Geometric explanation.

Source: Tibshirani et al.

Constraint and Lagrangian Relaxation

Constrained Optimization

- Choose a constant C.
- Compute β as

$$\operatorname*{argmin}_{\beta \in \mathbb{R}^d, \|\beta\|_p \leq C} \frac{1}{n} \sum_{i=1}^{n} \bar{\ell}(Y_i, h(\underline{x}_i{}^\top \beta))$$

Lagrangian Relaxation

- Choose λ and compute β as

$$\operatorname*{argmin}_{\beta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \bar{\ell}(Y_i, h(\underline{x}_i{}^\top \beta)) + \lambda \|\beta\|_p^{p'}$$

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

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

Regularization

Regularized Linear Model

- Minimization of

$$\underset{\beta \in \mathbb{R}^d}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \bar{\ell}(Y_i, h(\underline{x}_i{}^\top \beta)) + \operatorname{reg}(\beta)$$

  where $\operatorname{reg}(\beta)$ is a (sparsity promoting) regularisation term (regularization penalty).

- Variable selection if β is sparse.

Classical Regularization Penalties

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

- Easy optimization if reg (and the loss) is convex. . .
- **Need to specify λ to define an ML method!**

Regularized Gen. Linear Models

Classical Examples

- Regularized Least Squares
- Regularized Logistic Regression
- Regularized Maximum Likelihood
- SVM
- Tree pruning

- Sometimes used even if the parameterization is not linear. . .

Regularization and Cross-Validation

Practical Selection Methodology

- Choose a regularization penalty family reg_λ.
- Compute a CV risk for the regularization penalty reg_λ for all $\lambda \in \Lambda$.
- Determine $\widehat{\lambda}$ the λ minimizing the CV risk.
- Compute the final model with the regularization penalty $\text{reg}_{\widehat{\lambda}}$.

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

Why not using directly a parameter grid?

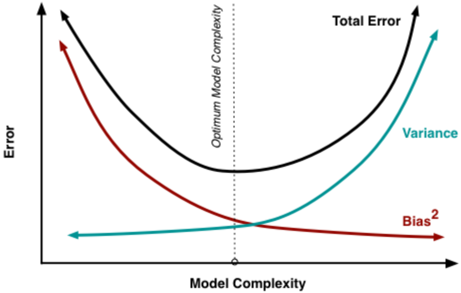
- Grid size scales exponentially with the dimension!
- **If the regularized minimization is easy**, much cheaper to compute the CV risk for all $\lambda \in \Lambda$...
- CV performs best when the set of candidates is not too big (or is structured...)

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

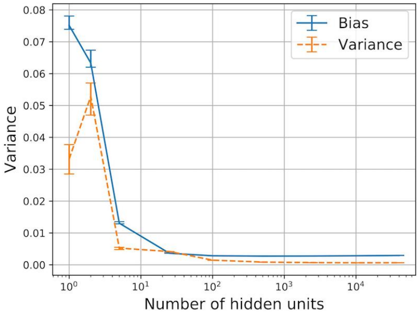
NN and Bias-Variance Tradeoff

Traditional view



vs

NN reality

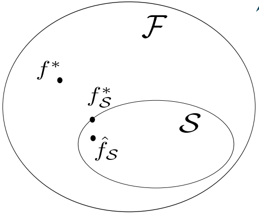


No Bias-Variance Tradeoff in NN ?

- Simultaneous decay of the variance and the bias!
- Contradiction with the bias-variance tradeoff intuition ?

Source: S. Formann-Roe/B. Neal

Bias-Variance Dilemma



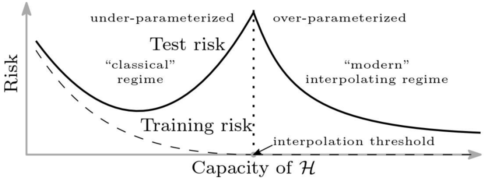
- General setting:
  - $\mathcal{F} = \{\text{measurable functions } \mathcal{X} \to \mathcal{Y}\}$
  - Best solution: $f^\star = \operatorname{argmin}_{f \in \mathcal{F}} \mathcal{R}(f)$
  - Class $\mathcal{S} \subset \mathcal{F}$ of functions
  - Ideal target in $\mathcal{S}$: $f_{\mathcal{S}}^\star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

Approximation error and estimation error (Bias-Variance)

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

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

Approximation-Estimation Dilemna?



Approximation error and estimation error ($\neq$ predictor bias-variance)

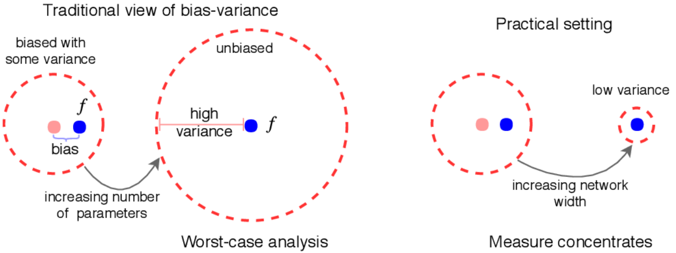
$$\mathcal{R}(\widehat{f}_{\mathcal{S}}) - \mathcal{R}(f^\star) = \underbrace{\mathcal{R}(f^\star_{\mathcal{S}}) - \mathcal{R}(f^\star)}_{\text{Approximation error}} + \underbrace{\mathcal{R}(\widehat{f}_{\mathcal{S}}) - \mathcal{R}(f^\star_{\mathcal{S}})}_{\text{Estimation error}}$$

- Approx. error can be large if the model $\mathcal{S}$ is not suitable.
- Estimation error
  - can be large if the model is complex,
  - but may be small for complex model if it is easy to find a model having a performance similar to the best one!

- Small estimation errors scenario seem the most probable scenario in deep learning.

Source: M. Belkin

A Refined View



Source: B. Neal

Traditional View

- Single good target
- Difficulty to be close grows with complexity.
- Bias-Variance analysis in the predictor space.

Refined View

- Many good targets
- Difficulty to be close from one may decrease with complexity.
- Bias-Variance analysis in the loss space.

- Importance of (cross) validation!

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Support Vector Machine

$$f_\theta(\underline{X}) = \underline{X}^\top \beta + \beta^{(0)} \quad \text{with} \quad \theta = (\beta, \beta^{(0)})$$

$$\hat{\theta} = \arg\min \frac{1}{n} \sum_{i=1}^{n} \max\left(1 - Y_i f_\theta(\underline{X}_i), 0\right) + \lambda \|\beta\|_2^2$$

Support Vector Machine

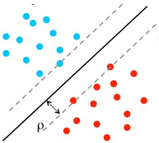
- Convexification of the 0/1-loss with the hinge loss:

$$\mathbf{1}_{Y_i f_\theta(\underline{X}_i) < 0} \leq \max\left(1 - Y_i f_\theta(\underline{X}_i), 0\right)$$

- Regularization 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



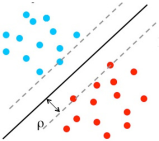
- Linear classifier: $\mathrm{sign}(\underline{X}^\top \beta + \beta^{(0)})$
- Separable case: $\exists (\beta, \beta^{(0)}), \forall i, Y_i(\underline{X}_i{}^\top \beta + \beta^{(0)}) > 0$

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

- Strict separation: $\exists (\beta, \beta^{(0)}), \forall i, Y_i(\underline{X}_i{}^\top \beta + \beta^{(0)}) \geq 1$
- Distance between $\underline{X}^\top \beta + \beta^{(0)} = 1$ and $\underline{X}^\top \beta + \beta^{(0)} = -1$:
$$\frac{2}{\|\beta\|}$$
- Maximizing this distance is equivalent to minimizing $\frac{1}{2}\|\beta\|^2$.

Source: M. Mohri et al.

Ideal Separable Case



Separable SVM

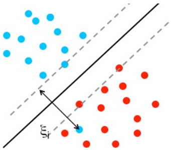
- Constrained optimization formulation:

$$\min \frac{1}{2}\|\beta\|^2 \quad \text{with} \quad \forall i, Y_i(\underline{X}_i{}^\top \beta + \beta^{(0)}) \geq 1$$

- Quadratic Programming setting.
- Efficient solver available. . .

Source: M. Mohri et al.

Non Separable Case



- What about the non separable case?

SVM relaxation

- Relax the assumptions

$$\forall i, Y_i(\underline{X}_i{}^\top \beta + \beta^{(0)}) \geq 1 \quad \text{to} \quad \forall i, Y_i(\underline{X}_i{}^\top \beta + \beta^{(0)}) \geq 1 - s_i$$

  with the **slack variables** $s_i \geq 0$

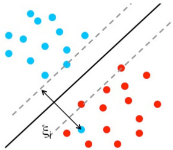
- Keep those slack variables as small as possible by minimizing

$$\frac{1}{2}\|\beta\|^2 + C \sum_{i=1}^{n} s_i$$

  where $C > 0$ is the **goodness-of-fit strength**

Source: M. Mohri et al.

Non Separable Case



SVM

- Constrained optimization formulation:

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

- **Hinge Loss** reformulation:

$$\min \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n} \underbrace{\max(0, 1 - Y_i(\underline{X}_i{}^\top\beta + \beta^{(0)}))}_{\text{Hinge Loss}}$$

- Constrained convex optimization algorithms vs gradient descent algorithms.

Source: M. Mohri et al.

SVM as a Regularized Convex Relaxation

- Convex relaxation:

$$\operatorname{argmin} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^n \max(1 - Y_i(\underline{X}_i{}^\top\beta + \beta^{(0)}), 0)$$

$$= \operatorname{argmin} \frac{1}{n}\sum_{i=1}^n \max(1 - Y_i(\underline{X}_i{}^\top\beta + \beta^{(0)}), 0) + \frac{1}{Cn}\frac{1}{2}\|\beta\|^2$$

- **Prop:** $\ell^{0/1}(Y_i, \operatorname{sign}(\underline{X}_i{}^\top\beta + \beta^{(0)})) \leq \max(1 - Y_i(\underline{X}_i{}^\top\beta + \beta^{(0)}), 0)$

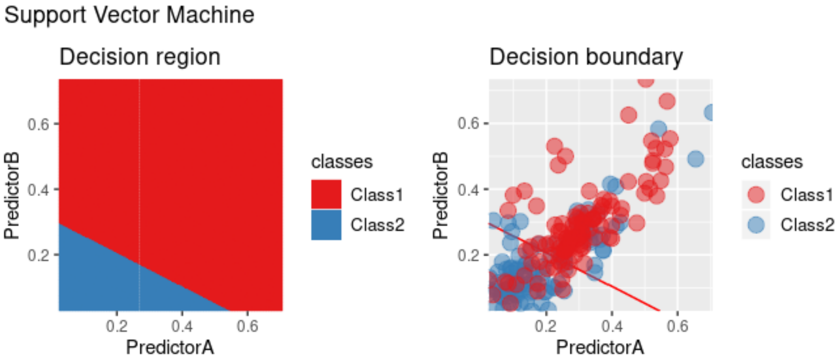
Regularized convex relaxation (Tikhonov!)

$$\frac{1}{n}\sum_{i=1}^n \ell^{0/1}(Y_i, \operatorname{sign}(\underline{X}_i{}^\top\beta + \beta^{(0)})) + \frac{1}{Cn}\frac{1}{2}\|\beta\|^2$$

$$\leq \frac{1}{n}\sum_{i=1}^n \max(1 - Y_i(\underline{X}_i{}^\top\beta + \beta^{(0)}), 0) + \frac{1}{Cn}\frac{1}{2}\|\beta\|^2$$

- No straightforward extension to multi-class classification.
- Extension to regression using $\ell(f(X), Y) = |Y - X|$.

SVM


Support Vector Machine
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2

Constrained Minimization

Constrained Minimization

- Goal:

$$\min_x f(x)$$

$$\text{with} \begin{cases} h_j(x) = 0, & j = 1, \dots p \\ g_i(x) \leq 0, & i = 1, \dots q \end{cases}$$

- or rather with argmin!

Different Setting

- f, h_j, g_i **differentiable**
- f **convex**, h_j **affine** and g_i **convex**.

Feasibility

- x is **feasible** if $h_j(x) = 0$ and $g_i(x) \leq 0$.
- **Rk:** The set of feasible points may be empty

Lagrangian

Constrained Minimization

- Goal:

$$p^\star = \min_x f(x) \quad \text{with} \quad \begin{cases} h_j(x) = 0, & j = 1, \dots p \\ g_i(x) \leq 0, & i = 1, \dots q \end{cases}$$

Lagrangian

- **Def:**

$$\mathcal{L}(x, \lambda, \mu) = f(x) + \sum_{j=1}^{p} \lambda_j h_j(x) + \sum_{i=1}^{q} \mu_i g_i(x)$$

  with $\lambda \in \mathbb{R}^p$ and $\mu \in (\mathbb{R}^+)^q$.
- The λ_j and μ_i are called the dual (or Lagrange) variables.
- **Prop:**

$$\max_{\lambda \in \mathbb{R}^p,\ \mu \in (\mathbb{R}^+)^q} \mathcal{L}(x, \lambda, \mu) = \begin{cases} f(x) & \text{if } x \text{ is feasible} \\ +\infty & \text{otherwise} \end{cases}$$

$$\min_x \max_{\lambda \in \mathbb{R}^p,\ \mu \in (\mathbb{R}^+)^q} \mathcal{L}(x, \lambda, \mu) = p^\star$$

Lagrangial Dual

Lagrangian

- **Def:**

$$\mathcal{L}(x, \lambda, \mu) = f(x) + \sum_{j=1}^{p} \lambda_j h_j(x) + \sum_{i=1}^{q} \mu_i g_i(x)$$

with $\lambda \in \mathbb{R}^p$ and $\mu \in (\mathbb{R}^+)^q$.

Lagrangian Dual

- Lagrangian dual function:

$$Q(\lambda, \mu) = \min_x \mathcal{L}(x, \lambda, \mu)$$

- **Prop:**

$$Q(\lambda, \mu) \leq f(x), \text{ for all feasible } x$$

$$\max_{\lambda \in \mathbb{R}^p,\ \mu \in (\mathbb{R}^+)^q} Q(\lambda, \mu) \leq \min_{x \text{ feasible}} f(x)$$

Duality

Primal

- Primal:

$$p^\star = \min_{x \in \mathcal{X}} f(x) \text{ with } \begin{cases} h_j(x) = 0, & j = 1, \dots p \\ g_i(x) \leq 0, & i = 1, \dots q \end{cases}$$

Dual

- Dual:

$$q^\star = \max_{\lambda \in \mathbb{R}^p,\ \mu \in (\mathbb{R}^+)^q} Q(\lambda, \mu) = \max_{\lambda \in \mathbb{R}^p,\ \mu \in (\mathbb{R}^+)^q} \min_x \mathcal{L}(x, \lambda, \mu)$$

Duality

- Always **weak duality:**

$$q^\star \leq p^\star$$

$$\max_{\lambda \in \mathbb{R}^p,\ \mu \in (\mathbb{R}^+)^q} \min_x \mathcal{L}(x, \lambda, \mu) \leq \min_x \max_{\lambda \in \mathbb{R}^p,\ \mu \in (\mathbb{R}^+)^q} \mathcal{L}(x, \lambda, \mu)$$

- Not always strong duality $q^\star = p^\star$.

Strong Duality

Strong Duality

- **Strong duality:**

$$q^\star = p^\star$$

$$\max_{\lambda \in \mathbb{R}^p,\ \mu \in (\mathbb{R}^+)^q} \min_x \mathcal{L}(x, \lambda, \mu) = \min_x \max_{\lambda \in \mathbb{R}^p,\ \mu \in (\mathbb{R}^+)^q} \mathcal{L}(x, \lambda, \mu)$$

- Allow to compute the solution of one problem from the other.
- Requires some assumptions!

Strong Duality under Convexity and Slater's Condition

- f **convex**, h_j **affine** and g_i **convex**.
- **Slater's condition:** it exists a feasible point such that $h_j(x) = 0$ for all j and $g_i(x) < 0$ for all i.
- Sufficient to prove **strong duality**.
- **Rk:** If the g_i are affine, it suffices to have $h_j(x) = 0$ for all j and $g_i(x) \leq 0$ for all i.

KKT

Karush-Kuhn-Tucker Condition

- Stationarity:
$$\nabla_x \mathcal{L}(x^\star, \lambda, \mu) = \nabla f(x^\star) + \sum_j \lambda_j \nabla h_j(x^\star) + \sum_i \mu_i \nabla g_i(x^\star) = 0$$
- Primal admissibility:
$$h_j(x^\star) = 0 \quad \text{and} \quad g_i(x^\star) \leq 0$$
- Dual admissibility:
$$\mu_i \geq 0$$
- Complementary slackness:
$$\mu_i g_i(x^\star) = 0$$

KKT Theorem

- If f **convex**, h_j **affine** and g_i **convex**, all are differentiable and **strong duality** holds then $x^\star$ is a **solution** of the primal problem **if and only if** the **KKT condition holds**

SVM and Lagrangian

SVM

- Constrained optimization formulation:

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

SVM Lagrangian

- Lagrangian:

$$\begin{aligned} \mathcal{L}(\beta, \beta^{(0)}, s, \alpha, \mu) = {} & \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n} s_i \\ & + \sum_i \alpha_i (1 - s_i - Y_i(\underline{X}_i{}^\top \beta + \beta^{(0)})) - \sum_i \mu_i s_i \end{aligned}$$

SVM and KKT

KKT Optimality Conditions

- Stationarity:

$$\nabla_\beta \mathcal{L}(\beta, \beta^{(0)}, s, \alpha, \mu) = \beta - \sum_i \alpha_i Y_i \underline{X}_i = 0$$

$$\nabla_{\beta^{(0)}} \mathcal{L}(\beta, \beta^{(0)}, s, \alpha, \mu) = - \sum_i \alpha_i = 0$$

$$\nabla_{s_i} \mathcal{L}(\beta, \beta^{(0)}, s, \alpha, \mu) = C - \alpha_i - \mu_i = 0$$

- Primal and dual admissibility:

$$(1 - s_i - Y_i(\underline{X}_i{}^\top \beta + \beta^{(0)})) \leq 0, \quad s_i \geq 0, \quad \alpha_i \geq 0, \text{ and } \mu_i \geq 0$$

- Complementary slackness:

$$\alpha_i(1 - s_i - Y_i(\underline{X}_i{}^\top \beta + \beta^{(0)})) = 0 \quad \text{and} \quad \mu_i s_i = 0$$

Consequence

- $\beta^\star = \sum_i \alpha_i Y_i \underline{X}_i$ and $0 \leq \alpha_i \leq C$.
- If $\alpha_i \neq 0$, $\underline{X}_i$ is called a **support vector** and either
  - $s_i = 0$ and $Y_i(\underline{X}_i{}^\top \beta^\star + \beta^{(0)*}) = 1$ (margin hyperplane),
  - or $\alpha_i = C$ (outliers).
- $\beta^{(0)*} = Y_i - \underline{X}_i{}^\top \beta^\star$ for any support vector with $0 < \alpha_i < C$.

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 $\forall i, \alpha_i + \mu_i = C$,

$$Q(\alpha,\mu) = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i\alpha_j Y_i Y_j \underline{X}_i{}^\top \underline{X}_j$$

SVM Dual problem

- Dual problem is a Quadratic Programming problem:

$$\max_{\alpha\geq 0,\mu\geq 0} Q(\alpha,\mu) \Leftrightarrow \max_{0\leq\alpha\leq C} \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i\alpha_j Y_i Y_j \underline{X}_i{}^\top \underline{X}_j$$

- Involves the $\underline{X}_i$ only through their scalar products.

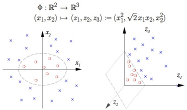
Mercer Theorem

Mercer Representation Theorem

- For any loss $\bar{\ell}$ and any increasing function Φ, the minimizer in β of
$$\sum_{i=1}^{n} \bar{\ell}(Y_i, \underline{X}_i^\top \beta + \beta^{(0)}) + \Phi(\|\beta\|_2)$$
is a linear combination of the input points $\beta^\star = \sum_{i=1}^{n} \alpha'_i \underline{X}_i$.
- Minimization problem in α':
$$\sum_{i=1}^{n} \bar{\ell}(Y_i, \sum_{j} \alpha'_j \underline{X}_i^\top \underline{X}_j + \beta^{(0)}) + \Phi(\|\beta\|_2)$$
involving only the scalar product of the data.

- Optimal predictor requires only to compute scalar products.
$$\hat{f}^\star(\underline{X}) = \underline{X}^\top \beta^\star + \beta^{(0),*} = \sum_{i} \alpha'_i \underline{X}_i^\top \underline{X}$$
- Transform a problem in dimension $\dim(\mathcal{X})$ in a problem in dimension n.
- Direct minimization in β can be more efficient...

Feature Map


$\Phi : \mathbb{R}^2 \to \mathbb{R}^3$
$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2)$
x_2
x_1
z_3
z_1
z_2

Feature Engineering

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

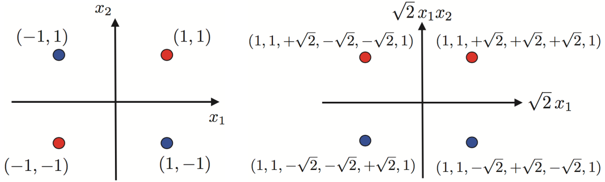
Feature Map

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

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

Source: Unknown

Polynomial Mapping


x_2
$(-1,1)$
$(1,1)$
x_1
$(-1,-1)$
$(1,-1)$
$\sqrt{2}\,x_1x_2$
$(1,1,+\sqrt{2},-\sqrt{2},-\sqrt{2},1)$
$(1,1,+\sqrt{2},+\sqrt{2},+\sqrt{2},1)$
$\sqrt{2}\,x_1$
$(1,1,-\sqrt{2},-\sqrt{2},+\sqrt{2},1)$
$(1,1,-\sqrt{2},+\sqrt{2},-\sqrt{2},1)$

Polynomial Mapping of order 2

- $\phi : \mathbb{R}^2 \to \mathbb{R}^6$

$$\phi(\underline{X}) = \left((\underline{X}^{(1)})^2, (\underline{X}^{(2)})^2, \sqrt{2}\underline{X}^{(1)}\underline{X}^{(2)}, \sqrt{2}\underline{X}^{(1)}, \sqrt{2}\underline{X}^{(2)}, 1\right)$$

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

Polynomial Mapping and Scalar Product

- **Prop:**

$$\phi(\underline{X})^\top \phi(\underline{X}') = (1 + \underline{X}^\top \underline{X}')^2$$

Source: M. Mohri et al.

SVM Primal and Dual

Primal, Lagrandian and Dual

- Primal:

$$\min \|\beta\|^2 + C \sum_{i=1}^{n} s_i \quad \text{with} \quad \begin{cases} \forall i, Y_i(\phi(\underline{X}_i)^\top \beta + \beta^{(0)}) \geq 1 - s_i \\ \forall i, s_i \geq 0 \end{cases}$$

- Lagrangian:

$$\begin{aligned} \mathcal{L}(\beta, \beta^{(0)}, s, \alpha, \mu) = & \frac{1}{2}\|\beta\|^2 + C \sum_{i=1}^{n} s_i \\ & + \sum_i \alpha_i (1 - s_i - Y_i(\phi(\underline{X}_i)^\top \beta + \beta^{(0)})) - \sum_i \mu_i s_i \end{aligned}$$

- Dual:

$$\max_{\alpha \geq 0, \mu \geq 0} Q(\alpha, \mu) \Leftrightarrow \max_{0 \leq \alpha \leq C} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j Y_i Y_j \phi(\underline{X}_i)^\top \phi(\underline{X}_j)$$

- Optimal $\phi(\underline{X})^\top \beta^\star + \beta^{(0),*} = \sum_i \alpha_i Y_i \phi(\underline{X})^\top \phi(\underline{X}_i)$

- Only need to know to compute $\phi(\underline{X})^\top \phi(\underline{X}')$ to obtain the solution.

From Map to Kernel

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

Kernel

- Any application

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

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

Kernel Trick

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

- Here k is defined from ϕ.
- Under some assumption on k, ϕ can be implicitly *defined* from k!

PDS Kernel

Positive Definite Symmetric Kernels

- A kernel k is PDS if and only if
  - k is symmetric, i.e.
  $$k(\underline{X}, \underline{X}') = k(\underline{X}', \underline{X})$$
  - for any $N \in \mathbb{N}$ and any $(\underline{X}_1, \ldots, \underline{X}_N) \in \mathcal{X}^N$,
  $$\boldsymbol{K} = [k(\underline{X}_i, \underline{X}_j)]_{1 \leq i,j \leq N}$$
  is positive semi-definite, i.e. $\forall u \in \mathbb{R}^N$
  $$u^\top \boldsymbol{K} u = \sum_{1 \leq i,j \leq N} u^{(i)} u^{(j)} k(\underline{X}_i, \underline{X}_j) \geq 0$$
  or equivalently all the eigenvalues of $\boldsymbol{K}$ are non-negative.

- The matrix $\boldsymbol{K}$ is called the **Gram matrix** associated to $(\underline{X}_1, \ldots, \underline{X}_N)$.

Reproducing Kernel Hilbert Space

Moore-Aronsajn Theorem

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

- By def., $\mathbb{H}$ is a **reproducing kernel Hilbert space** (RKHS).
- $\mathbb{H}$ is called the **feature space** associated to k and ϕ the **feature mapping**.
- No unicity in general.
- **Rk:** if $k(\underline{X}, \underline{X}') = \phi'(\underline{X})^\top \phi'(\underline{X}')$ with $\phi' : \mathcal{X} \to \mathbb{R}^p$ then
  - $\mathbb{H}$ can be chosen as $\{\underline{X} \mapsto \phi'(\underline{X})^\top \beta, \beta \in \mathbb{R}^p\}$ and $\|\underline{X} \mapsto \phi'(\underline{X})^\top \beta\|_{\mathbb{H}}^2 = \|\beta\|_2^2$.
  - $\phi(\underline{X}') : \underline{X} \mapsto \phi'(\underline{X})^\top \phi'(\underline{X}')$.

Kernel Construction Machinery

Separable Kernel

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

Kernel Stability

- For any PDS kernels k_1 and k_2, $k_1 + k_2$ and $k_1 k_2$ 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 $\sum_n a_n z^n$ with $a_n \geq 0$ and a convergence radius larger than r, $\displaystyle\sum_n a_n k^n$ is a PDS kernel.
- For any PDS kernel k, the renormalized kernel $k'(\underline{X}, \underline{X}') = \dfrac{k(\underline{X}, \underline{X}')}{\sqrt{k(\underline{X}, \underline{X})k(\underline{X}', \underline{X}')}}$ is a PDS kernel.

- Cauchy-Schwartz for k PDS: $k(\underline{X}, \underline{X}')^2 \leq k(\underline{X}, \underline{X})k(\underline{X}', \underline{X}')$

Classical Kernels

PDS Kernels

- Vanilla kernel:

$$k(\underline{X}, \underline{X}') = \underline{X}^\top \underline{X}'$$

- Polynomial kernel:

$$k(\underline{X}, \underline{X}') = (1 + \underline{X}^\top \underline{X}')^k$$

- Gaussian RBF kernel:

$$k(\underline{X}, \underline{X}') = \exp\left(-\gamma \|\underline{X} - \underline{X}'\|^2\right)$$

- Tanh kernel:

$$k(\underline{X}, \underline{X}') = \tanh(a\underline{X}^\top \underline{X}' + b)$$

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

Representer Theorem

Representer Theorem

- Let k be a PDS kernel and $\mathbb{H}$ its corresponding RKHS, for any increasing function Φ and any function $L : \mathbb{R}^n \to \mathbb{R}$, the optimization problem
$$\operatorname*{argmin}_{h \in \mathbb{H}} L(h(\underline{X}_1), \ldots, h(\underline{X}_n)) + \Phi(\|h\|)$$
admits only solutions of the form
$$\sum_{i=1}^{n} \alpha'_i k(\underline{X}_i, \cdot).$$

- Examples:
    - (kernelized) SVM
    - (kernelized) Regularized Logistic Regression (Ridge)
    - (kernelized) Regularized Regression (Ridge)

Kernelized SVM

Primal

- Constrained Optimization:

$$\min_{f\in\mathbb{H},\beta^{(0)},s} \|f\|_{\mathbb{H}}^2 + C\sum_{i=1}^{n} s_i \quad \text{with} \quad \begin{cases} \forall i,\ Y_i(f(\underline{X}_i)+\beta^{(0)}) \geq 1 - s_i \\ \forall i,\ s_i \geq 0 \end{cases}$$

- Hinge loss:

$$\min_{f\in\mathbb{H},\beta^{(0)}} \|f\|_{\mathbb{H}}^2 + C\sum_{i=1}^{n} \max(0, 1 - Y_i(f(\underline{X}_i)+\beta^{(0)}))$$

- Representer:

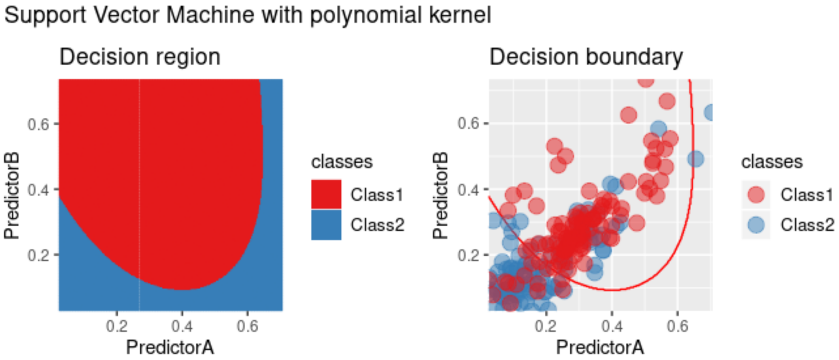
$$\min_{\alpha',\beta^{(0)}} \sum_{i,j} \alpha'_i\alpha'_j k(\underline{X}_i,\underline{X}_j) \\ + C\sum_{i=1}^{n} \max(0, 1 - Y_i(\sum_j \alpha'_j k(\underline{X}_j,\underline{X}_i)+\beta^{(0)}))$$

Dual

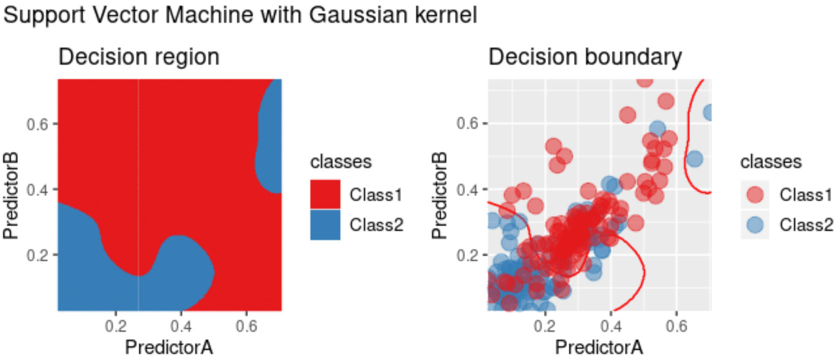
- Dual:

$$\max_{\alpha\geq 0,\mu\geq 0} Q(\alpha,\mu) \Leftrightarrow \max_{0\leq\alpha\leq C} \sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j Y_i Y_j k(\underline{X}_i,\underline{X}_j)$$

SVM


Support Vector Machine with polynomial kernel
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

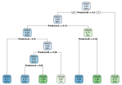
SVM


Support Vector Machine with Gaussian kernel
Decision region
Decision boundary
PredictorA
PredictorB
0.2
0.4
0.6
classes
Class1
Class2

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Classification And Regression Trees

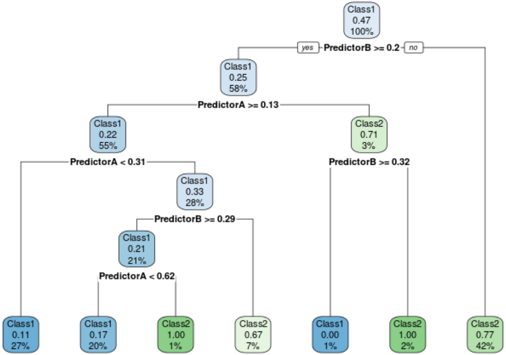


Tree principle (CART by Breiman (85) / ID3 by Quinlan (86))

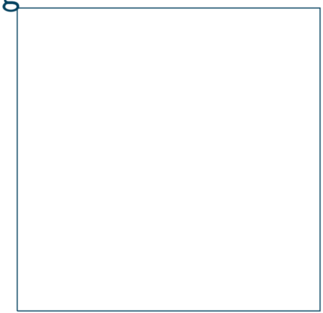
- Construction of a recursive partition through a tree structured set of questions (splits around a given value of a variable)
- For a given partition, probabilistic approach **and** optimization approach yield the same predictor!
- A simple majority vote/averaging in each leaf

- 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


Class1
0.47
100%
yes
PredictorB >= 0.2
no
Class1
0.25
58%
PredictorA >= 0.13
Class1
0.22
55%
PredictorA < 0.31
Class2
0.71
3%
PredictorB >= 0.32
Class1
0.33
28%
PredictorB >= 0.29
Class1
0.21
21%
PredictorA < 0.62
Class1
0.11
27%
Class1
0.17
20%
Class2
1.00
1%
Class2
0.67
7%
Class1
0.00
1%
Class2
1.00
2%
Class2
0.77
42%

Branching

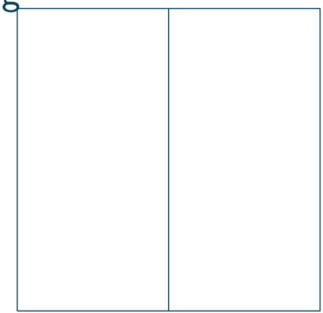


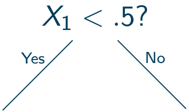
Greedy top-bottom approach

- 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



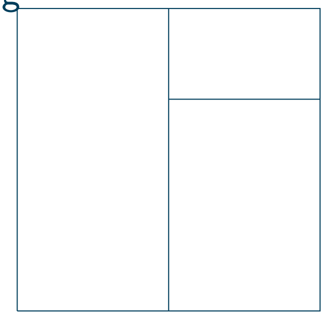


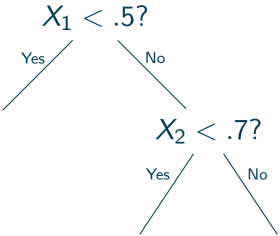
Greedy top-bottom approach

- 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



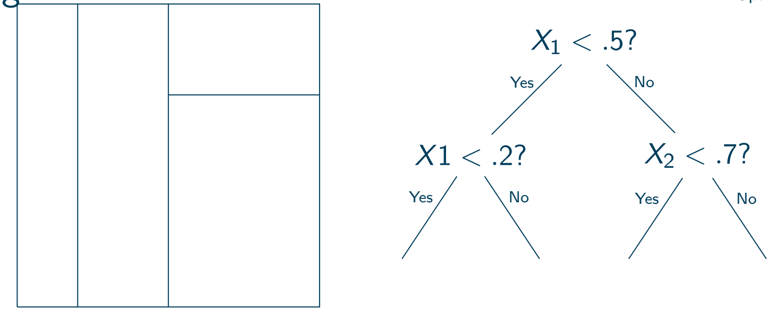


Greedy top-bottom approach

- 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 *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 in*homogeneous*

- **CART:** empirical loss based criterion (least squares/prediction error)
$$C(R,\overline{R}) = \sum_{\underline{x}_i \in R} \bar{\ell}(y_i, y(R)) + \sum_{\underline{x}_i \in \overline{R}} \bar{\ell}(y_i, y(\overline{R}))$$
- **CART:** Gini index (Classification)
$$C(R,\overline{R}) = \sum_{\underline{x}_i \in R} p(R)(1-p(R)) + \sum_{\underline{x}_i \in \overline{R}} p(\overline{R})(1-p(\overline{R}))$$
- **C4.5:** entropy based criterion (Information Theory)
$$C(R,\overline{R}) = \sum_{\underline{x}_i \in R} H(R) + \sum_{\underline{x}_i \in \overline{R}} H(\overline{R})$$

- CART with Gini is probably the most used technique. . . even in the multi-class setting where the entropy may be more natural.
- Other criterion based on χ^2 homogeneity or based on different local predictors (generalized linear models. . .)

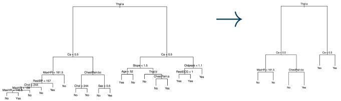
Branching

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 depth is equal to a prescribed maximum depth,
  - when the region is sufficiently homogeneous. . .
- May lead to a quite complex tree: over-fitting possible!
- Additional pruning often used.

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
$$C(\mathcal{T}) = \sum_{\mathcal{L}\in\mathcal{T}} c(\mathcal{L})$$
- Example: AIC / CV.

Pruning

Examples of criterion satisfying this assumptions

- AIC type criterion:

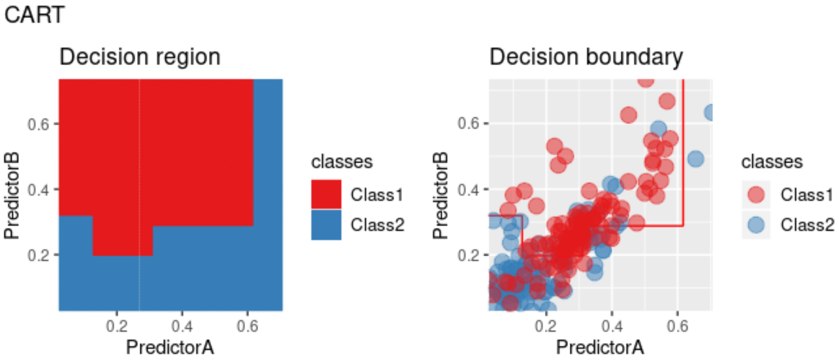
$$\sum_{i=1}^{n} \bar{\ell}(y_i, f_{\mathcal{L}(\underline{x}_i)}(\underline{x}_i)) + \lambda|\mathcal{T}| = \sum_{\mathcal{L}\in\mathcal{T}} \left(\sum_{\underline{x}_i \in \mathcal{L}} \bar{\ell}(y_i, f_{\mathcal{L}}(\underline{x}_i)) + \lambda \right)$$

- Simple cross-Validation (with $(\underline{x}'_i, y'_i)$ a different dataset):

$$\sum_{i=1}^{n'} \bar{\ell}(y'_i, f_{\mathcal{L}}(\underline{x}'_i)) = \sum_{\mathcal{L}\in\mathcal{T}} \left(\sum_{\underline{x}'_i \in \mathcal{L}} \bar{\ell}(y'_i, f_{\mathcal{L}}(\underline{x}'_i)) \right)$$

- Limit over-fitting for a single tree.
- **Rk:** almost never used when combining several trees. . .

CART


CART
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

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?

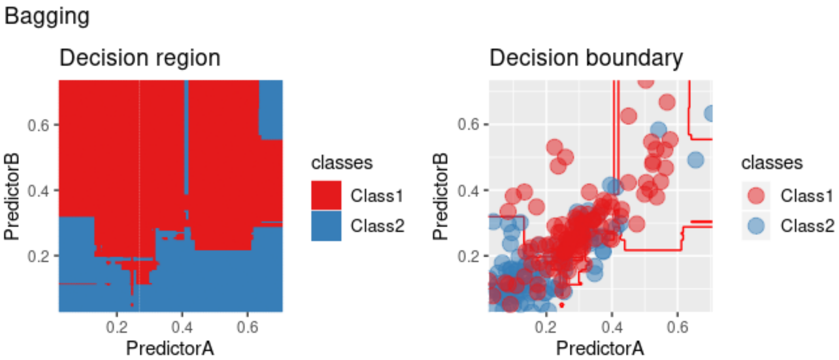
Parallel construction

- Construct several trees from bootstrapped samples and average the responses **(Bagging)**
- Add more randomness in the tree construction **(Random Forests)**

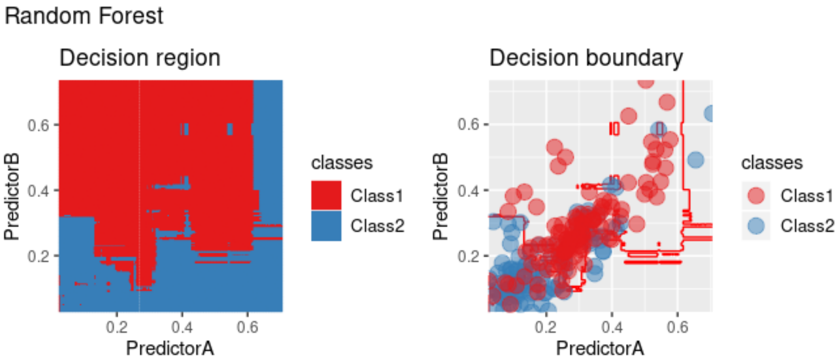
Sequential construction

- Construct a sequence of trees by reweighting sequentially the samples according to their difficulties **(AdaBoost)**
- Reinterpretation as a stagewise additive model **(Boosting)**

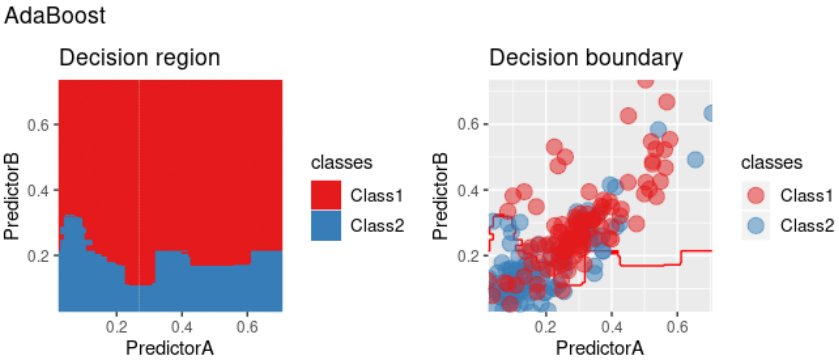
Ensemble methods



Ensemble methods



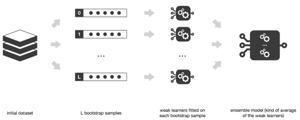
Ensemble methods

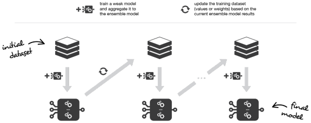


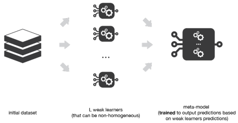
Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Ensemble Methods


0
1
L
initial dataset
L bootstrap samples
weak learners fitted on each bootstrap sample
ensemble model (kind of average of the weak learners)


train a weak model and aggregate it to the ensemble model
update the training dataset (values or weights) based on the current ensemble model results
initial dataset
final model


initial dataset
L weak learners (that can be non-homogeneous)
meta-model (**trained** to output predictions based on weak learners predictions)

Ensemble Methods

- **Averaging:** combine several models by averaging (bagging, random forests,. . .)
- **Boosting:** construct a sequence of (weak) classifiers (XGBoost, LightGBM, CatBoost, Histogram Gradient Boosting from `scikit-learn`)
- **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!

Source: J. Rocca

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Independent Average

Stability through averaging

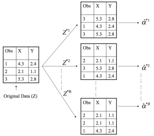
- Very simple idea to obtain a more stable estimator.
- **Vote/average** of B predictors $f_1, \ldots, f_B$ obtained with **independent datasets** of size n!

$$f_{\text{agr}} = \operatorname{sign}\left(\frac{1}{B}\sum_{b=1}^{B} f_b\right) \quad \text{or} \quad f_{\text{agr}} = \frac{1}{B}\sum_{i=1}^{B} f_b$$

- **Regression:** $\mathbb{E}[f_{\text{agr}}(x)] = \mathbb{E}[f_b(x)]$ and $\mathbb{V}\text{ar}\left[f_{\text{agr}}(x)\right] = \frac{\mathbb{V}\text{ar}[f_b(x)]}{B}$
- **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



Obs	X	Y
1	4.3	2.4
2	2.1	1.1
3	5.3	2.8

Original Data (Z)

Obs	X	Y
3	5.3	2.8
1	4.3	2.4
3	5.3	2.8

Obs	X	Y
2	2.1	1.1
3	5.3	2.8
1	4.3	2.4

Obs	X	Y
2	2.1	1.1
2	2.1	1.1
1	4.3	2.4

- Strategy proposed by Breiman in 1994.

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 $(1 - 1/e) \simeq .63$ examples are unique among each drawn dataset. . .
- The f_b are still identically distributed but **not independent** anymore.
- Price for the non independence: $\mathbb{E}[f_{\text{agr}}(x)] = \mathbb{E}[f_b(x)]$ and
$$\mathbb{V}\text{ar}\left[f_{\text{agr}}(x)\right] = \frac{\mathbb{V}\text{ar}\left[f_b(x)\right]}{B} + \left(1 - \frac{1}{B}\right)\rho(x)$$
with $\rho(x) = \mathbb{C}\text{ov}\left[f_b(x), f_{b'}(x)\right] \leq \mathbb{V}\text{ar}\left[f_b(x)\right]$ with $b \neq b'$.
- **Bagging:** Bootstrap Aggregation

- Better aggregation scheme exists. . .

Randomized Predictors

- Correlation leads to less variance reduction:

$$\mathbb{V}\text{ar}\left[f_{\text{agr}}(x)\right] = \frac{\mathbb{V}\text{ar}\left[f_b(x)\right]}{B} + \left(1 - \frac{1}{B}\right)\rho(x)$$

  with $\rho(x) = \mathbb{C}\text{ov}\left[f_b(x), f_{b'}(x)\right]$ 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

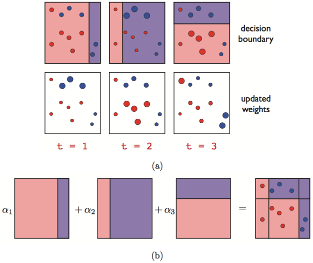
- **Importance:** Number of time used or criterion gain at each split can be used to rank the variables.
- **Permutation tests:** Difference between OOB estimate using the true value of the jth feature and a value drawn a random from the list of possible values.

- Up to OOB error, the permutation technique is not specific to trees.

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Boosting



Boosting

- Construct a sequence of predictors h_t and weights α_t so that the weighted sum
$$f_t = f_{t-1} + \alpha_t h_t$$
is better and better (at least on the training set!).

- Simple idea but no straightforward instanciation!
- First boosting algorithm: AdaBoost by Schapire and Freund in 1997.

AdaBoost

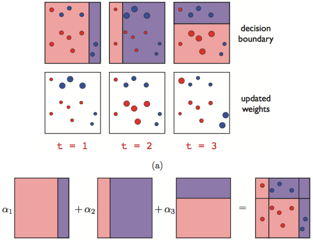
- **Idea:** learn a predictor in a sequential manner by training a correction term at each step with weighted dataset with weights depending on the error so far.

Iterative scheme proposed by Schapire and Freud

- Set $w_{1,i} = 1/n$; $t = 0$ and $f = 0$
- For $t = 1$ to $t = T$
  - $h_t = \operatorname{argmin}_{h \in \mathcal{H}} \sum_{i=1}^{n} w_{t,i} \ell^{0/1}(y_i, h(x_i))$
  - Set $\epsilon_t = \sum_{i=1}^{n} w_{t,i} \ell^{0/1}(y_i, h_t(x_i))$ and $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$
  - let $w_{t+1,i} = \frac{w_{t,i} e^{-\alpha_t y_i h_t(\underline{x}_i)}}{Z_{t+1}}$ 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 $f = \sum_{i=1}^{T} \alpha_t h_t$ or rather its sign.

- **Intuition:** $w_{t,i}$ measures the difficulty of learning the sample i up to step t and thus the importance of being good at this step...
- **Prop:** The resulting predictor can be proved to have a training risk of at most $2^T \prod_{t=1}^{T} \sqrt{\epsilon_t(1-\epsilon_t)}$.

AdaBoost



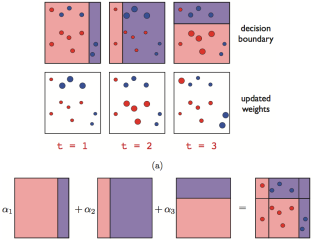
(a)

(b)

AdaBoost Intuition

- h_t obtained by minimizing a weighted loss
$$h_t = \operatorname*{argmin}_{h \in \mathcal{H}} \sum_{i=1}^{n} w_{t,i} \ell^{0/1}(y_i, h(\underline{x}_i))$$
- Update the current estimate with
$$f_t = f_{t-1} + \alpha_t h_t$$

AdaBoost



(a)

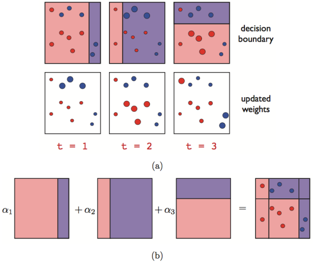
(b)

AdaBoost Intuition

- Weight $w_{t,i}$ should be large if $\underline{x}_i$ is not well-fitted at step $t-1$ and small otherwise.
- Use a weight proportional to $e^{-y_i f_{t-1}(\underline{x}_i)}$ so that it can be recursively updated by

$$w_{t+1,i} = w_{t,i} \times \frac{e^{-\alpha_t y_i h_t(\underline{x}_i)}}{Z_t}$$

AdaBoost



AdaBoost Intuition

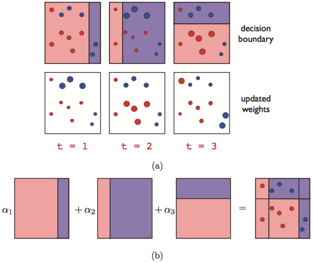
- Set α_t such that

$$\sum_{y_i h_t(\underline{x}i)=1} w_{t+1,i} = \sum_{y_i h_t(\underline{x}i)=-1} w_{t+1,i}$$

or equivalently

$$\left(\sum_{y_i h_t(\underline{x}i)=1} w_{t,i}\right) e^{-\alpha_t} = \left(\sum_{y_i h_t(\underline{x}i)=-1} w_{t,i}\right) e^{\alpha_t}$$

AdaBoost



AdaBoost Intuition

- Using

$$\epsilon_t = \sum_{y_i h_t(\underline{x}i)=-1} w_{t,i}$$

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,
  - $(h_t, \alpha_t) = \operatorname{argmin}_{h,\alpha} \sum_{i=1}^{n} e^{-y_i (f(\underline{x}_i) + \alpha h(\underline{x}_i))}$
  - $f = f + \alpha_t h_t$
- Use $f = \sum_{t=1}^{T} \alpha_t h_t$ or rather its sign.

- **Greedy optimization** of a classifier as a linear combination of T classifiers for the **exponential loss**.
- Additive Modeling can be traced back to the 70's.
- AdaBoost and Exponential Stagewise Additive Modeling are **exactly the same**!

Revisited AdaBoost

AdaBoost

- Set $t = 0$ and $f = 0$.
- For $t = 1$ to T,
  - $(h_t, \alpha_t) = \operatorname{argmin}_{h,\alpha} \sum_{i=1}^{n} e^{-y_i(f(\underline{x}_i)+\alpha h(\underline{x}_i))}$
  - $f = f + \alpha_t h_t$
- Use $f = \sum_{t=1}^{T} \alpha_t h_t$ or rather its sign.

- **Greedy iterative scheme** with only two parameters: the class $\mathcal{H}$ of *weak* classifiers and the number of steps T.
- In the literature, one can read that Adaboost does not overfit! This is not true and T should be chosen with care...

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,
    - $(h_t, \alpha_t) = \operatorname{argmin}_{h,\alpha} \sum_{i=1}^{n} \bar{\ell}(y_i, f(x_i) + \alpha h(x_i))$
    - $f = f + \alpha_t h_t$
  - Use $f = \sum_{t=1}^{T} \alpha_t h_t$
- AKA as **Forward Stagewise Additive Modeling**
  - AdaBoost with $\bar{\ell}(y, h) = e^{-yh}$
  - LogitBoost with $\bar{\ell}(y, h) = \log_2(1 + e^{-yh})$
  - L_2Boost with $\bar{\ell}(y, h) = (y - h)^2$ (Matching pursuit)
  - L_1Boost with $\bar{\ell}(y, h) = |y - h|$
  - HuberBoost with $\bar{\ell}(y, h) = |y - h|^2 \mathbf{1}_{|y-h|<\epsilon} + (2\epsilon|y - h| - \epsilon^2)\mathbf{1}_{|y-h|\geq\epsilon}$

- Extension to multi-class classification through surrogate losses.
- **No easy numerical scheme** except for AdaBoost and L_2Boost...

Gradient Boosting

- **Issue:** At each boosting step, one need to solve

$$(h_t, \alpha_t) = \operatorname*{argmin}_{h,\alpha} \sum_{i=1}^{n} \bar{\ell}(y_i, f(x_i) + \alpha h(x_i)) = L(y, f + \alpha h)$$

- **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 α_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:

$$\nabla L(y, f) \quad \text{with} \quad \nabla_i L(y, f) = \frac{\partial}{df(x_i)} \left(\sum_{i'=1}^{n} \bar{\ell}(y_{i'}, f(x_{i'})) \right)$$
$$= \frac{\partial}{df(x_i)} \bar{\ell}(y_i, f(x_i))$$

Best Direction within $\mathcal{H}$

- Direct formulation:

$$h_t \in \operatorname*{argmin}_{h \in \mathcal{H}} \frac{\sum_{i=1}^{n} \nabla_i L(y, f) h(x_i)}{\sqrt{\sum_{i=1}^{n} |h(x_i)|^2}} \left(= \frac{\langle \nabla L(y, f), h \rangle}{\|h\|} \right)$$

- Equivalent (least-squares) formulation: $h_t = -\beta_t h'_t$ with

$$(\beta_t, h'_t) \in \operatorname*{argmin}_{(\beta, h) \in \mathbb{R} \times \mathcal{H}} \sum_{i=1}^{n} |\nabla_i L(y, f) - \beta h(x_i)|^2 \left(= \|\nabla L - \beta h\|^2 \right)$$

- 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 $\|h\|^2 = n$.
  - $\bar{\ell}(y, f(x)) = l(yf(x))$ 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

$$h_t = \operatorname*{argmin}_{h \in \mathcal{H}} \sum_i \nabla_i L(y, f) h(x_i)$$

AdaBoost Type Minimization

- Best direction rewriting

$$\begin{aligned} h_t &= \operatorname*{argmin}_{h \in \mathcal{H}} \sum_i l'(y_i f(x_i)) y_i h(x_i) \\ &= \operatorname*{argmin}_{h \in \mathcal{H}} \sum_i (-l')(y_i f(x_i))(2\ell^{0/1}(y_i, h(x_i)) - 1) \end{aligned}$$

- **AdaBoost type weighted loss minimization** as soon as $(-l')(y_i f(x_i) \geq 0$:

$$h_t = \operatorname{argmin} \sum_i (-l')(y_i f(x_i)) \ell^{0/1}(y_i, h(x_i))$$

Gradient Boosting of Classifiers

Gradient Boosting

- **(Gradient) AdaBoost:** $\bar{\ell}(y,f) = \exp(-yf)$
    - $l(x) = \exp(-x)$ and thus $(-l')(y_i f(x_i)) = e^{-y_i f(x_i)} \geq 0$
    - h_t is the same as in AdaBoost
    - α_t also... (explicit computation)
- **LogitBoost:** $\bar{\ell}(y,f) = \log_2(1+e^{-yf})$
    - $l(x) = \log_2(1+e^{-x})$ and thus $(-l')(y_i f(x_i)) = \frac{e^{-y_i f(x_i)}}{\log(2)(1+e^{-y_i f(x_i)})} \geq 0$
    - Less weight on misclassified samples than in AdaBoost...
    - No explicit formula for α_t (line search)
    - Different path than with the (non-computable) classical boosting!
- **SoftBoost:** $\bar{\ell}(y,f) = \max(1-yf,0)$
    - $l(x) = \max(1-x,0)$ and $(-l')(y_i f(x_i)) = \mathbf{1}_{y_i f(x_i) \leq 1} \geq 0$
    - Do not use the samples that are sufficiently well classified!

Gradient Boosting and Least Squares

- Least squares formulation is preferred when $|h| \neq 1$.

Least Squares Gradient Boosting

- Find $h_t = -\beta_t h'_t$ with

$$(\beta_t, h'_t) \in \operatorname*{argmin}_{(\beta,h)\in\mathbb{R}\times\mathcal{H}} \sum_{i=1}^{n} |\nabla_i L(y, f) - \beta h(x_i)|^2$$

- 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_2Boost:**
  - $\ell(y,f) = |y-f|^2$ and $\nabla_i L(y_i, f(x_i)) = -2(y_i - f(x_i))$:
  $$(\beta_t, h'_t) \in \operatorname*{argmin}_{(\beta,h)\in\mathbb{R}\times\mathcal{H}} \sum_{i=1}^{n} |2y_i - 2(f(x_i) - \beta/2h(x_i))|^2$$
  - $\alpha_t = -\beta_t/2$
  - Equivalent to classical L_2-Boosting
- **Gradient L_1Boost:**
  - $\ell(y,f) = |y-f|$ and $\nabla_i L(y_i, f(x_i)) = -\mathrm{sign}(y_i - f(x_i))$:
  $$(\beta_t, h'_t) \in \operatorname*{argmin}_{(\beta,h)\in\mathbb{R}\times\mathcal{H}} \sum_{i=1}^{n} |-\mathrm{sign}(y_i - f(x_i)) - \beta h(x_i)|^2$$
  - 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:

$$(\beta_t, h'_t) \in \underset{(\beta,h)\in\mathbb{R}\times\mathcal{H}}{\text{argmin}} \sum_{i=1}^{n} |y_i l'(y_i f(x_i)) - \beta h(x_i)|^2$$

- **Intuition:** Modify misclassified examples without modifying too much the well-classified ones...
- Most classical optimization choice nowadays!
- Also true for the extensions to multi-class classification.

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 (regularized) 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 regularization.

- Excellent baseline for tabular data (and time series)!
- Lightgbm, CatBoost, and Histogram Gradient Boosting from `scikit-learn` are also excellent similar choices!

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Empirical Risk Minimization

Empirical Risk Minimizer (ERM)

- For any loss ℓ and function class $\mathcal{S}$,

$$\widehat{f} = \operatorname*{argmin}_{f \in \mathcal{S}} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(\underline{X}_i)) = \operatorname*{argmin}_{f \in \mathcal{S}} \mathcal{R}_n(f)$$

- Key property:

$$\mathcal{R}_n(\widehat{f}) \leq \mathcal{R}_n(f), \forall f \in \mathcal{S}$$

- **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. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

ERM and PAC Analysis

- Theoretical control of the random (error estimation) term:
$$\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}})$$

Probably Almost Correct Analysis

- **Theoretical guarantee** that
$$\mathbb{P}\left(\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) \leq \epsilon_{\mathcal{S}}(\delta)\right) \geq 1 - \delta$$
for a suitable $\epsilon_{\mathcal{S}}(\delta) \geq 0$.
- Implies:
  - $\mathbb{P}\left(\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star) \leq \mathcal{R}(f^\star_{\mathcal{S}}) - \mathcal{R}(f^\star) + \epsilon_{\mathcal{S}}(\delta)\right) \geq 1 - \delta$
  - $\mathbb{E}\left[\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}})\right] \leq \int_0^{+\infty} \delta_{\mathcal{S}}(\epsilon) d\epsilon$

- The result should hold without any assumption on the law ***P***!

A General Decomposition

- By construction:

$$\begin{aligned}\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) &= \mathcal{R}(\widehat{f}) - \mathcal{R}_n(\widehat{f}) + \mathcal{R}_n(\widehat{f}) - \mathcal{R}_n(f^\star_{\mathcal{S}}) + \mathcal{R}_n(f^\star_{\mathcal{S}}) - \mathcal{R}(f^\star_{\mathcal{S}})\\ &\leq \mathcal{R}(\widehat{f}) - \mathcal{R}_n(\widehat{f}) + \mathcal{R}_n(f^\star_{\mathcal{S}}) - \mathcal{R}(f^\star_{\mathcal{S}})\\ &\leq \left(\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}})\right) - \left(\mathcal{R}_n(\widehat{f}) - \mathcal{R}_n(f^\star_{\mathcal{S}})\right)\end{aligned}$$

Four possible upperbounds

- $\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) \leq \sup_{f\in\mathcal{S}} ((\mathcal{R}(f) - \mathcal{R}(f^\star_{\mathcal{S}})) - (\mathcal{R}_n(f) - \mathcal{R}_n(f^\star_{\mathcal{S}})))$
- $\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) \leq \sup_{f\in\mathcal{S}} (\mathcal{R}(f) - \mathcal{R}_n(f)) + (\mathcal{R}_n(f^\star_{\mathcal{S}}) - \mathcal{R}(f^\star_{\mathcal{S}}))$
- $\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) \leq \sup_{f\in\mathcal{S}} (\mathcal{R}(f) - \mathcal{R}_n(f)) + \sup_{f\in\mathcal{S}} (\mathcal{R}_n(f) - \mathcal{R}(f))$
- $\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) \leq 2 \sup_{f\in\mathcal{S}} |\mathcal{R}(f) - \mathcal{R}_n(f)|$

- Supremum of centered random variables!
- **Key:** Concentration of each variable...

Risk Bounds

- By construction, for any $f' \in \mathcal{S}$,

$$\mathcal{R}(f') = \mathcal{R}_n(f') + (\mathcal{R}(f') - \mathcal{R}_n(f'))$$

A uniform upper bound for the risk

- Simultaneously $\forall f' \in \mathcal{S}$,

$$\mathcal{R}(f') \leq \mathcal{R}_n(f') + \sup_{f \in \mathcal{S}} (\mathcal{R}(f) - \mathcal{R}_n(f))$$

- Supremum of centered random variables!
- **Key:** Concentration of each variable...
- Can be interpreted as a justification of the ERM!

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. **Empirical Risk Minimization**
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - **Hoeffding and Finite Class**
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Concentration of the Empirical Loss

- Empirical loss:

$$\mathcal{R}_n(f) = \frac{1}{n}\sum_{i=1}^{n} \ell^{0/1}(Y_i, f(\underline{X}_i))$$

Properties

- $\ell^{0/1}(Y_i, f(\underline{X}_i))$ are i.i.d. random variables in $[0,1]$.

Concentration

$$\mathbb{P}(\mathcal{R}(f) - \mathcal{R}_n(f) \leq \epsilon) \geq 1 - e^{-2n\epsilon^2}$$

$$\mathbb{P}(\mathcal{R}_n(f) - \mathcal{R}(f) \leq \epsilon) \geq 1 - e^{-2n\epsilon^2}$$

$$\mathbb{P}(|\mathcal{R}_n(f) - \mathcal{R}(f)| \leq \epsilon) \geq 1 - 2e^{-2n\epsilon^2}$$

- Concentration of sum of bounded independent variables!
- Hoeffding theorem.
- Equiv. to $\mathbb{P}\left(\mathcal{R}(f) - \mathcal{R}_n(f) \leq \sqrt{\log(1/\delta)/(2n)}\right) \geq 1 - \delta$

Hoeffding

Theorem

- Let Z_i be a sequence of ind. centered r.v. supported in $[a_i, b_i]$ then

$$\mathbb{P}\left(\sum_{i=1}^{n} Z_i \geq \epsilon\right) \leq e^{-\frac{2\epsilon^2}{\sum_{i=1}^{n}(b_i - a_i)^2}}$$

- Proof ingredients:
  - Chernov bounds:

$$\mathbb{P}\left(\sum_{i=1}^{n} Z_i \geq \epsilon\right) \leq \frac{\mathbb{E}\left[e^{\lambda \sum_{i=1}^{n} Z_i}\right]}{e^{\lambda\epsilon}} \leq \frac{\prod_{i=1}^{n} \mathbb{E}\left[e^{\lambda Z_i}\right]}{e^{\lambda\epsilon}}$$

  - Exponential moment bounds: $\mathbb{E}\left[e^{\lambda Z_i}\right] \leq e^{\frac{\lambda^2 (b_i - a_i)^2}{8}}$
  - Optimization in λ
- **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

$$\mathbb{P}\left(\sum_{i=1}^{n} Z_i \geq \epsilon\right) \leq e^{-\frac{2\epsilon^2}{\sum_{i=1}^{n}(b_i-a_i)^2}}$$

- $Z_i = \frac{1}{n}\left(\mathbb{E}\left[\ell^{0/1}(Y, f(\underline{X}))\right] - \ell^{0/1}(Y_i, f(\underline{X}_i))\right)$
- $\mathbb{E}[Z_i] = 0$ and $Z_i \in [\frac{1}{n}\left(\mathbb{E}\left[\ell^{0/1}(Y, f(\underline{X}))\right] - 1\right), \frac{1}{n}\mathbb{E}\left[\ell^{0/1}(Y, f(\underline{X}))\right]]$
- 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 $\mathcal{S}$ is finite of cardinality $|\mathcal{S}|$,

$$\mathbb{P}\left(\sup_f \left(\mathcal{R}(f) - \mathcal{R}_n(f)\right) \leq \sqrt{\frac{\log|\mathcal{S}| + \log(1/\delta)}{2n}}\right) \geq 1 - \delta$$

$$\mathbb{P}\left(\sup_f \left|\mathcal{R}_n(f) - \mathcal{R}(f)\right| \leq \sqrt{\frac{\log|\mathcal{S}| + \log(1/\delta)}{2n}}\right) \geq 1 - 2\delta$$

- Control of the supremum by a quantity depending on the cardinality and the probability parameter δ.
- Simple combination of Hoeffding and a union bound.

Finite Class Case

PAC Bounds

- If $\mathcal{S}$ is finite of cardinality $|\mathcal{S}|$, with proba greater than $1 - 2\delta$

$$\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) \leq \sqrt{\frac{\log|\mathcal{S}| + \log(1/\delta)}{2n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$$

$$\leq 2\sqrt{\frac{\log|\mathcal{S}| + \log(1/\delta)}{2n}}$$

- If $\mathcal{S}$ is finite of cardinality $|\mathcal{S}|$, with proba greater than $1 - \delta$, simultaneously $\forall f' \in \mathcal{S}$,

$$\mathcal{R}(f') \leq \mathcal{R}_n(f') + \sqrt{\frac{\log|\mathcal{S}| + \log(1/\delta)}{2n}}$$

$$\leq \mathcal{R}_n(f') + \sqrt{\frac{\log|\mathcal{S}|}{2n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$$

Finite Class Case

PAC Bounds

- If $\mathcal{S}$ is finite of cardinality $|\mathcal{S}|$, with proba greater than $1 - 2\delta$

$$\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) \leq \sqrt{\frac{\log |\mathcal{S}|}{2n}} + \sqrt{\frac{2\log(1/\delta)}{n}}$$

- If $\mathcal{S}$ is finite of cardinality $|\mathcal{S}|$, with proba greater than $1 - \delta$, simultaneously $\forall f' \in \mathcal{S}$,

$$\mathcal{R}(f') \leq \mathcal{R}_n(f') + \sqrt{\frac{\log |\mathcal{S}|}{2n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$$

- Risk increases with the cardinality of $\mathcal{S}$.
- Similar issue in cross-validation!
- No direct extension for an infinite $\mathcal{S}$...

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Concentration of the Supremum of Empirical Losses

- Supremum of Empirical losses:

$$\Delta_n(\mathcal{S})(\underline{X}_1,\dots,\underline{X}_n) = \sup_{f\in\mathcal{S}} \mathcal{R}(f) - \mathcal{R}_n(f)$$

$$= \sup_{f\in\mathcal{S}} \left(\mathbb{E}\left[\ell^{0/1}(Y,f(\underline{X}))\right] - \frac{1}{n}\sum_{i=1}^{n} \ell^{0/1}(Y_i,f(\underline{X}_i)) \right)$$

Properties

- Bounded difference:

$$|\Delta_n(\mathcal{S})(\underline{X}_1,\dots,\underline{X}_i,\dots\underline{X}_n) - \Delta_n(\mathcal{S})(\underline{X}_1,\dots\underline{X}'_i,\dots,\underline{X}_n)| \leq 1/n$$

Concentration

$$\mathbb{P}(\Delta_n(\mathcal{S}) - \mathbb{E}[\Delta_n(\mathcal{S})] \leq \epsilon) \geq 1 - e^{-2n\epsilon^2}$$

- 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

$$\forall (\underline{X}_i)_{i=1}^n, (\underline{X}'_i)_{i=1}^n \in \mathbb{R},$$
$$\left| g(\underline{X}_1, \ldots, \underline{X}_i, \ldots, \underline{X}_n) - g(\underline{X}_1, \ldots, \underline{X}'_i, \ldots, \underline{X}_n) \right| \leq c_i$$

Theorem

- If g is a bounded difference function and $\underline{X}_i$ are independent random variables then

$$\mathbb{P}(g(\underline{X}_1, \ldots, \underline{X}_n) - \mathbb{E}[g(\underline{X}_1, \ldots, \underline{X}_n)] \geq \epsilon) \leq e^{\frac{-2\epsilon^2}{\sum_{i=1}^n c_i^2}}$$

$$\mathbb{P}(\mathbb{E}[g(\underline{X}_1, \ldots, \underline{X}_n)] - g(\underline{X}_1, \ldots, \underline{X}_n) \geq \epsilon) \leq e^{\frac{-2\epsilon^2}{\sum_{i=1}^n c_i^2}}$$

- Proof ingredients:
  - Chernov bounds
  - Martingale decomposition...

McDiarmid Inequality

Theorem

- If g is a bounded difference function and $\underline{X}_i$ are independent random variables then

$$\mathbb{P}(g(\underline{X}_1,\ldots,\underline{X}_n)-\mathbb{E}[g(\underline{X}_1,\ldots,\underline{X}_n)]\geq\epsilon)\leq e^{\frac{-2\epsilon^2}{\sum_{i=1}^n c_i^2}}$$

- Using $g=\Delta_n(\mathcal{S})$ for which $c_i=1/n$ yields immediately

$$\mathbb{P}(\Delta_n(\mathcal{S})-\mathbb{E}[\Delta_n(\mathcal{S})]\geq\epsilon)\leq e^{\frac{-2\epsilon^2}{\sum_{i=1}^n c_i^2}}=e^{-2n\epsilon^2}$$

- We derive then

$$\mathbb{P}(\Delta_n(\mathcal{S})\geq\mathbb{E}[\Delta_n(\mathcal{S})]+\epsilon)\leq e^{\frac{-2\epsilon^2}{\sum_{i=1}^n c_i^2}}=e^{-2n\epsilon^2}$$

- It remains to upperbound

$$\mathbb{E}[\Delta_n]=\mathbb{E}\left[\sup_{f\in\mathcal{S}}\mathcal{R}(f)-\mathcal{R}_n(f)\right]$$

Rademacher Complexity

Theorem

- Let σ_i be a sequence of i.i.d. random symmetric Bernoulli variables (Rademacher variables):

$$\mathbb{E}\left[\sup_{f\in\mathcal{S}}(\mathcal{R}(f)-\mathcal{R}_n(f))\right] \leq 2\mathbb{E}\left[\sup_{f\in\mathcal{S}}\frac{1}{n}\sum_{i=1}^{n}\sigma_i\ell^{0/1}(Y_i,f(\underline{X}_i))\right]$$

Rademacher complexity

- Let $B\subset\mathbf{R}^n$, 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** $B_n(\mathcal{S})=\{(\ell^{0/1}(Y_i,f(\underline{X}_i)))_{i=1}^{n}, f\in\mathcal{S}\}$.
- **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 $B = B_n(\mathcal{S}) = \{(\ell^{0/1}(Y_i, f(\underline{X}_i)))_{i=1}^n, f \in \mathcal{S}\}$, we have $M = 1$ and thus

$$R_n(B) \leq \sqrt{\frac{2 \log |B_n(\mathcal{S})|}{n}}$$

- We obtain immediately

$$\mathbb{E}\left[\sup_{f \in \mathcal{S}} (\mathcal{R}(f) - \mathcal{R}_n(f))\right] \leq \mathbb{E}\left[\sqrt{\frac{8 \log |B_n(\mathcal{S})|}{n}}\right].$$

Finite Set Rademacher Complexity Bound

Theorem

- With probability greater than $1 - 2\delta$,

$$\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) \leq \mathbb{E}\left[\sqrt{\frac{8 \log |B_n(\mathcal{S})|}{n}}\right] + \sqrt{\frac{2 \log(1/\delta)}{n}}$$

- With probability greater than $1 - \delta$, simultaneously $\forall f' \in \mathcal{S}$

$$\mathcal{R}(f') \leq \mathcal{R}_n(f') + \mathbb{E}\left[\sqrt{\frac{8 \log |B_n(\mathcal{S})|}{n}}\right] + \sqrt{\frac{\log(1/\delta)}{2n}}$$

- This is a direct consequence of the previous bound.

Finite Set Rademacher Complexity Bound

Corollary

- If $\mathcal{S}$ is finite then with probability greater than $1-2\delta$

$$\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) \leq \sqrt{\frac{8\log|\mathcal{S}|}{n}} + \sqrt{\frac{2\log(1/\delta)}{n}}$$

- If $\mathcal{S}$ is finite then with probability greater than $1-\delta$, simultaneously $\forall f' \in \mathcal{S}$

$$\mathcal{R}(f') \leq \mathcal{R}_n(f') + \sqrt{\frac{8\log|\mathcal{S}|}{n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$$

- It suffices to notice that

$$|B_n(\mathcal{S})| = |\{(\ell^{0/1}(Y_i, f(\underline{X}_i)))_{i=1}^n, f \in \mathcal{S}\}| \leq |\mathcal{S}|$$

Finite Set Rademacher Complexity Bound

- Same result with Hoeffding but with **better** constants!

$$\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) \leq \sqrt{\frac{\log|\mathcal{S}|}{2n}} + \sqrt{\frac{2\log(1/\delta)}{n}}$$

$$\mathcal{R}(f') \leq \mathcal{R}_n(f') + \sqrt{\frac{\log|\mathcal{S}|}{2n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$$

- Difference due to the *crude* upperbound of

$$\mathbb{E}\left[\sup_{f\in\mathcal{S}}\left(\mathcal{R}(f) - \mathcal{R}_n(f)\right)\right]$$

- **Why bother?:** We do not have to assume that $\mathcal{S}$ is finite!

$$|B_n(\mathcal{S})| \leq 2^n$$

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Back to the Bound

Theorem

$$\mathbb{E}\left[\sup_{f\in\mathcal{S}}\left(\mathcal{R}(f)-\mathcal{R}_n(f)\right)\right]\leq\mathbb{E}\left[\sqrt{\frac{8\log|B_n(\mathcal{S})|}{n}}\right]$$

- Key quantity: $\mathbb{E}\left[\sqrt{\frac{8\log|B_n(\mathcal{S})|}{n}}\right]$
- Hard to control due to its structure!

A first data dependent upperbound

$$\mathbb{E}\left[\sqrt{\frac{8\log|B_n(\mathcal{S})|}{n}}\right]\leq\sqrt{\frac{8\log\mathbb{E}[|B_n(\mathcal{S})|]}{n}}\qquad\text{(Jensen)}$$

- Depends on the unknown ***P***!

Shattering Coefficient

Shattering Coefficient (or Growth Function)

- The shattering coefficient of the class $\mathcal{S}$, $s(\mathcal{S}, n)$, is defined as
$$s(\mathcal{S}, n) = \sup_{\left((\underline{X}_1, Y_1), \ldots, (\underline{X}_n, Y_n)\right) \in (\mathcal{X} \times \{-1,1\})^n} \left| \{ (\ell^{0/1}(Y_i, f(\underline{X}_i)))_{i=1}^n, f \in \mathcal{S} \} \right|$$

- By construction, $|B_n(\mathcal{S})| \leq s(\mathcal{S}, n) \leq \min(2^n, |\mathcal{S}|)$.

A data independent upperbound

$$\mathbb{E}\left[\sqrt{\frac{8 \log |B_n(\mathcal{S})|}{n}}\right] \leq \sqrt{\frac{8 \log s(\mathcal{S}, n)}{n}}$$

Shattering Coefficient

Theorem

- With probability greater than $1 - 2\delta$,

$$\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_\mathcal{S}) \le \sqrt{\frac{8 \log s(\mathcal{S}, n)}{n}} + \sqrt{\frac{2 \log(1/\delta)}{n}}$$

- With probability greater than $1 - \delta$, simultaneously $\forall f' \in \mathcal{S}$,

$$\mathcal{R}(f') \le \mathcal{R}_n(f') + \sqrt{\frac{8 \log s(\mathcal{S}, n)}{n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$$

- Depends only on the class $\mathcal{S}$!

Vapnik-Chervonenkis Dimension

VC Dimension

- The VC dimension d_{VC} of $\mathcal{S}$ is defined as the largest integer d such that
$$s(\mathcal{S}, d) = 2^d$$

- The VC dimension can be infinite!

VC Dimension and Dimension

- **Prop:** If $\text{span}(\mathcal{S})$ 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 $\mathcal{S}$ is finite

$$s(\mathcal{S}, n) \leq \begin{cases} 2^n & \text{if } n \leq d_{VC} \\ \left(\frac{en}{d_{VC}}\right)^{d_{VC}} & \text{if } n > d_{VC} \end{cases}$$

- **Cor.:** $\log s(\mathcal{S}, n) \leq d_{VC} \log\left(\frac{en}{d_{VC}}\right)$ if $n > d_{VC}$.

VC Dimension and PAC Bounds

PAC Bounds

- If $\mathcal{S}$ is of VC dimension d_{VC} then if $n > d_{VC}$
- With probability greater than $1 - 2\delta$,

$$\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) \leq \sqrt{\frac{8 d_{VC} \log\left(\frac{en}{d_{VC}}\right)}{n}} + \sqrt{\frac{2\log(1/\delta)}{n}}$$

- With probability greater than $1 - \delta$, simultaneously $\forall f' \in \mathcal{S}$,

$$\mathcal{R}(f') \leq \mathcal{R}_n(f') + \sqrt{\frac{8 d_{VC} \log\left(\frac{en}{d_{VC}}\right)}{n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$$

- **Rk:** If $d_{VC} = +\infty$ no uniform PAC bounds exists!

Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

Countable Collection and Non Uniform PAC Bounds

PAC Bounds

- Let $\pi_f > 0$ such that $\sum_{f \in \mathcal{S}} \pi_f = 1$
- With proba greater than $1 - 2\delta$,

$$\mathcal{R}(\widehat{f}) - \mathcal{R}(f^\star_{\mathcal{S}}) \leq \sqrt{\frac{\log(1/\pi_f)}{2n}} + \sqrt{\frac{2\log(1/\delta)}{n}}$$

- With proba greater than $1 - \delta$, simultaneously $\forall f' \in \mathcal{S}$,

$$\mathcal{R}(f') \leq \mathcal{R}_n(f') + \sqrt{\frac{\log(1/\pi_f)}{2n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$$

- 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 $(\mathcal{S}_m)_{m\in\mathcal{M}}$ and let π_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 $f \in \mathcal{S}_m$,
$$\mathcal{R}(f) \leq \mathcal{R}_n(f) + \mathbb{E}\left[\sqrt{\frac{8\log|B_n(\mathcal{S}_m)|}{n}}\right] + \sqrt{\frac{\log(1/\pi_m)}{2n}} + \sqrt{\frac{\log(1/\delta)}{2n}}$$

Structural Risk Minimization

- Choose $\hat{f}$ as the minimizer over $m \in \mathcal{M}$ and $f \in \mathcal{S}_m$ of
$$\mathcal{R}_n(f) + \mathbb{E}\left[\sqrt{\frac{8\log|B_n(\mathcal{S}_m)|}{n}}\right] + \sqrt{\frac{\log(1/\pi_m)}{2n}}$$

- Mimics the minimization of the integrated risk!

SRM and PAC Bound

PAC Bound

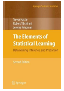
- If $\hat{f}$ is the SRM minimizer then with probability $1 - 2\delta$,
$$\mathcal{R}(\widehat{f}) \leq \inf_{m \in \mathcal{M}} \inf_{f \in \mathcal{S}_m} \left(\mathcal{R}(f) + \mathbb{E}\left[\sqrt{\frac{8 \log |B_n(\mathcal{S}_m)|}{n}} \right] + \sqrt{\frac{\log(1/\pi_m)}{2n}} \right) + \sqrt{\frac{2 \log(1/\delta)}{n}}$$

- The SRM minimizer balances the risk $\mathcal{R}(f)$ and the upper bound on the estimation error $\mathbb{E}\left[\sqrt{\frac{8 \log |B_n(\mathcal{S}_m)|}{n}} \right] + \sqrt{\frac{\log(1/\pi_m)}{2n}}$.
- $\mathbb{E}\left[\sqrt{\frac{8 \log |B_n(\mathcal{S}_m)|}{n}} \right]$ can be replaced by an upper bound (for instance a VC based one)...

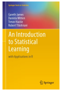
Outline

1. Introduction
   - Machine Learning
   - Motivation
2. A Practical View
   - Method or Models
   - Interpretability
   - Metric Choice
3. A Better Point of View
   - The Example of Univariate Linear Regression
   - Supervised Learning
4. Risk Estimation and Method Choice
   - Cross Validation
   - Cross Validation and Test
   - Cross Validation and Weights
   - Auto ML
5. A Probabilistic Point of View
   - Parametric Conditional Density Modeling
   - Non Parametric Conditional Density Modeling
   - Generative Modeling
6. Optimization Point of View
   - (Deep) Neural Networks
   - Regularization
   - Another Perspectivce on Bias-Variance Tradeoff
   - SVM
   - Tree Based Methods
7. Ensemble Methods
   - Bagging and Random Forests
   - Boosting
8. Empirical Risk Minimization
   - Empirical Risk Minimization
   - ERM and PAC Bayesian Analysis
   - Hoeffding and Finite Class
   - McDiarmid and Rademacher Complexity
   - VC Dimension
   - Structural Risk Minimization
9. References

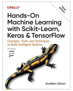
References



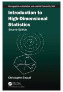
T. Hastie, R. Tibshirani, and J. Friedman.
The Elements of Statistical Learning.
Springer Series in Statistics, 2009



G. James, D. Witten, T. Hastie, and R. Tibshirani.
An Introduction to Statistical Learning with Applications in R.
Springer, 2014

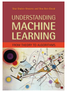


geron22Âš

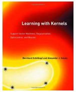


Ch. Giraud.
Introduction to High-Dimensional Statistics (2nd ed.)
CRC Press, 2021

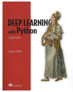
M. Mohri, A. Rostamizadeh, and A. Talwalkar.
Foundations of Machine Learning (2nd ed.)
MIT Press, 2018



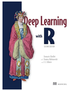
S. Shalev-Shwartz and S. Ben-David.
Understanding Machine Learning.
Cambridge University Press, 2014



B. Schölkopf and A. Smola.
Learning with kernels.
The MIT Press, 2002



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

Licence and Contributors

Contributors

- Main contributor: E. Le Pennec
- Contributors: S. Boucheron, A. Dieuleveut, A.K. Fermin, S. Gadat, S. Gaiffas, A. Guilloux, Ch. Keribin, E. Matzner, M. Sangnier, E. Scornet.