Statistical Learning vs Machine Learning in Classification

Ana Fermín (1) and Erwan Le Pennec (2)

(1) Modal'X Université Paris Ouest
(2)CMAP Ecole polytechnique

Labex MME-DII, 09/04/2015

Statistical vs Optimization Points of View in Classification

Ana Fermín (1) and Erwan Le Pennec (2)

(1) Modal'X Université Paris Ouest
(2)CMAP Ecole polytechnique

Labex MME-DII, 09/04/2015

Motivation

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

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

Motivation

Marketing: advertisement, recommendation...

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

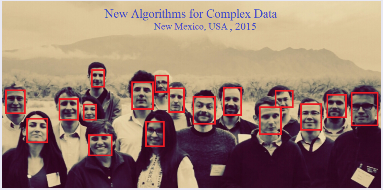
Motivation

Spam detection (Text classification)

- Data: email collection
- Input: email
- Output : Spam or No Spam

Motivation

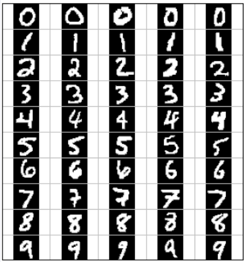
Face Detection



- Data: Annotated database of images
- Input : Sub window in the image
- Output : Presence or no of a face...

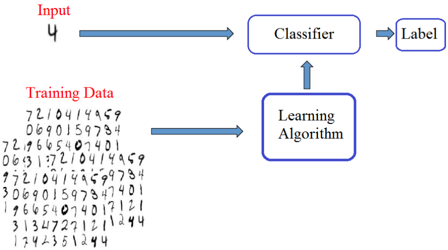
Motivation

Number Recognition



- Data: Annotated database of images (each image is represented by a vector of $28 \times 28 = 784$ pixel intensities)
- Input: Image
- Output: Corresponding number

Machine Learning



A definition by Tom Mitchell (`http://www.cs.cmu.edu/~tom/`)

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.

Machine Learning

With the explosion of "Big Data" problems, machine learning has become a very hot field in many scientific areas.

- It is important to understand the ideas behind the various techniques, in order to know how and when to use them.
- One has to understand the simpler methods first, in order to grasp the more sophisticated ones.
- This is an exciting research area, having important applications in science, industry and finance.
- Machine learning is a fundamental ingredient in the training of a modern **data scientist**.

Topics for Today

1. Supervised Classification (Part 1)
   - Binary Supervised Classifcation
   - Models
   - Statistical and Optimization Points of View
2. A Statistical Point of View (Part 1)
   - Logistic regression
   - Class by Class modeling
   - *k* Nearest Neighbors
3. An Optimization Point of View (Part 2)
   - SVM
   - (Deep) Neural Networks
   - Tree Based Methods
4. Models and Combinations (Part 2)
   - Models
   - Model Selection
   - Ensemble Methods
5. Big Data (Part 2)

Statistical Learning in Classification

Outline

Outline

Binary Supervised Classification

Supervised Learning Framework

- Input measurement $\mathbf{X} = (X^{(1)}, X^{(2)}, \ldots, X^{(d)}) \in \mathbb{R}^d$
- Output measurement $Y \in \{-1, 1\}$.
- $(\mathbf{X}, Y) \sim \mathbf{P}$ with $\mathbf{P}$ unknown.
- Training data : $\mathcal{D}_n = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$ (i.i.d. $\sim \mathbf{P}$)

- A classifier is a function in $\mathcal{F} = \{f : \mathbb{R}^d \to \{-1, 1\} \text{ measurable}\}$

Goal

- Construct a good classifier $\widehat{f}$ from the training data.

- Need to specify the meaning of good.

Binary Supervised Classification

Loss function and risk of a generic classifier

- Loss function : $\ell(f(x), y)$ measure how well $f(x)$ "predicts" y.
- For this talk $\ell(f(x), y) = \ell^{0/1}(f(x), y) = \mathbf{1}_{y \neq f(x)}$
- Risk measured as the average loss for a new couple:

$$\mathcal{R}(f) = \mathbb{E}_{(X,Y)\sim\mathbf{P}}\left[\ell^{0/1}(Y, f(\mathbf{X}))\right] = \mathbb{P}\left\{Y \neq f(\mathbf{X})\right\}$$

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

Goal

- Learn a rule to construct a classifier $\widehat{f} \in \mathcal{F}$ 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 $\mathcal{D}_n$.

Best Solution

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

$$f^* = \arg\min_{f \in \mathcal{F}} R(f) = \arg\min_{f \in \mathcal{F}} \mathbb{E}\left[\ell^{0/1}(Y, f(\mathbf{X}))\right]$$
$$= \arg\min_{f \in \mathcal{F}} \mathbb{E}_{\mathbf{X}}\left[\mathbb{E}_{Y|\mathbf{X}}\left[\ell^{0/1}(Y, f(\mathbf{x}))\right]\right]$$
$$f^*(\mathbf{x}) = \arg\max_{k} \mathbb{P}(Y = k | \mathbf{X} = \mathbf{x})$$

Binary Bayes Classifier (explicit solution)

In binary classification with $0-1$ loss:

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

Issue: Explicit solution requires to know $Y|\mathbf{x}$ for all $\mathbf{x}$!

Goal

Machine Learning

- Learn a rule to construct a classifier $\widehat{f} \in \mathcal{F}$ 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 $\mathcal{D}_n$.

Canonical example: Empirical Risk Minimizer

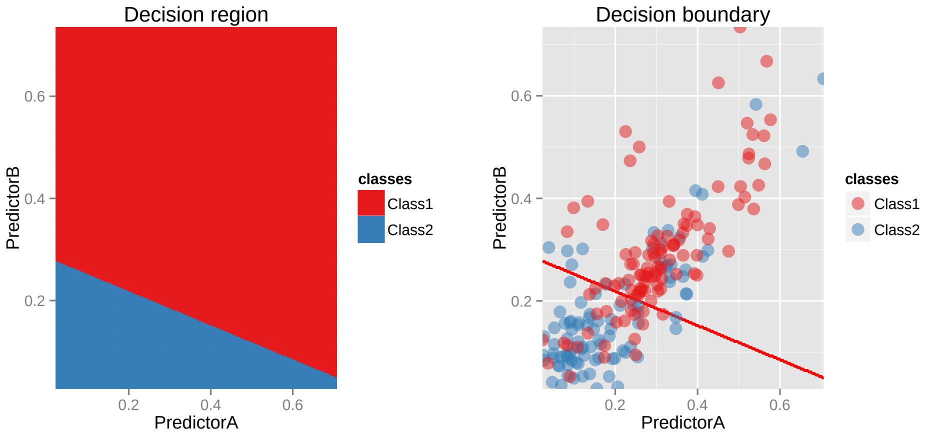
- One restricts f to a subset of functions $\mathcal{S} = \{f_\theta, \theta \in \Theta\}$
- One replaces the minimization of the average loss by the minimization of the empirical loss

$$\widehat{f} = f_{\widehat{\theta}} = \operatorname*{argmin}_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell^{0/1}(Y_i, f_\theta(\mathbf{X}_i))$$

- Example: Linear discrimination with

$$\mathcal{S} = \{\mathbf{x} \mapsto \mathtt{sign}\{\beta^T \mathbf{x} + \beta_0\} \,/ \beta \in \mathbb{R}^d, \beta_0 \in \mathbb{R}\}$$

Example: Linear Discrimination

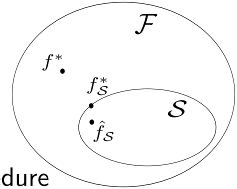


Outline

1. Supervised Classification
   - Binary Supervised Classification
   - Models
   - Statistical and Optimization Points of View
2. A Statistical Point of View
   - Logistic Modeling
   - Generative Modeling
   - k Nearest-Neighbors

Bias-Variance Dilemna

- General setting:
  - $\mathcal{F} = \{\text{measurable fonctions } \mathbb{R}^d \to \{-1,1\}\}$
  - Best solution: $f^* = \operatorname{argmin}_{f\in\mathcal{F}} \mathcal{R}(f)$
  - Class $\mathcal{S} \subset \mathcal{F}$ of functions
  - Ideal target in $\mathcal{S}$: $f^*_{\mathcal{S}} = \operatorname{argmin}_{f\in\mathcal{S}} \mathcal{R}(f)$
  - Estimate in $\mathcal{S}$: $\widehat{f}_{\mathcal{S}}$ obtained with some procedure



Approximation error and estimation error (Bias/Variance)

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

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

Agnostic approach

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

Theoretical Analysis

Statistical Learning Analysis

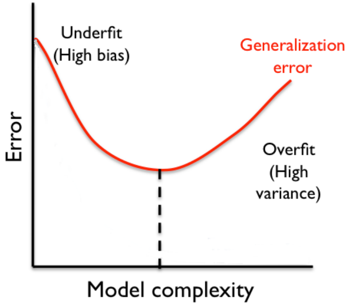
- Error decomposition:

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

- 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 **P**!

- Often need mild assumptions on **P**...
- Not our focus today!

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 contains a good ideal target but the estimation error ("variance") can be large (Over-fit)

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

Outline

1. Supervised Classification
   - Binary Supervised Classification
   - Models
   - Statistical and Optimization Points of View
2. A Statistical Point of View
   - Logistic Modeling
   - Generative Modeling
   - k Nearest-Neighbors

Statistical and Optimization Points of View

How to find a good function $f \in \mathcal{H}$ with a *small*

$$R(f) = \mathbb{E}\left[\ell^{0/1}(Y, f(X))\right] = \mathbb{P}\left\{Y \neq f(X)\right\} \quad ?$$

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

Problem: minimization impossible in practice for the 0-1 loss !

A Statistical Point of View (A. Fermin)

Solution: For $\mathbf{x} \in \mathbb{R}^d$, estimate $\mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})$.
Learn $Y|X$ and plug this estimate in the Bayes classifier: gen. linear models, generative modeling, kernel methods, trees

An Optimization Point of View (E. Le Pennec)

Solution: Replace the loss $\ell^{0/1}$ by an upper bound ℓ' which allows the minimization: SVM, Neural Network, trees

Outline

Classification Rule / Algorithm

- Input: a data set $\mathcal{D}_n$
  Learn $Y|x$ or equivalently $p_k(\mathbf{x}) = \mathbb{P}\left\{Y = k|\mathbf{X} = \mathbf{x}\right\}$ (using the data set) and plug this estimate in the Bayes classifier
- Output: a classifier $\widehat{f} : \mathbb{R}^d \to \{-1, 1\}$

$$\hat{f}(\mathbf{x}) = \begin{cases} +1 & \text{if } \widehat{p}_{+1}(\mathbf{x}) \geq \widehat{p}_{-1}(\mathbf{x}) \\ -1 & \text{otherwise} \end{cases}$$

- Three instantiations:
  1. Logistic modeling (parametric method)
  2. Generative modeling (Bayes method)
  3. Nearest neighbors (kernel method)

Outline

Logistic Modeling

The Binary logistic model ($Y \in \{-1, 1\}$)

$$p_{+1}(\mathbf{x}) = \frac{e^{\beta^t \phi(\mathbf{x})}}{1 + e^{\beta^t \phi(\mathbf{x})}}$$

where $\phi(x)$ is a transformation of the individual $\mathbf{x}$

- In this model, one verifies that
$$p_{+1}(\mathbf{x}) \geq p_{-1}(\mathbf{x}) \quad \Leftrightarrow \quad \beta^t \phi(\mathbf{x}) \geq 0$$
- True $Y|x$ may not belong to this model $\Rightarrow$ maximum likelihood of β only finds a good approximation!
- Binary Logistic classifier:
$$\widehat{f}_L(\mathbf{x}) = \begin{cases} +1 & \text{if } \widehat{\beta}^t \phi(\mathbf{x}) \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

where $\widehat{\beta}$ is estimated by maximum likelihood.

Logistic Modeling

- Logistic model: approximation of $\mathcal{B}(p_1(\mathbf{x}))$ by $\mathcal{B}(h(\beta^t\phi(\mathbf{x})))$ with $h(t) = \frac{e^t}{1+e^t}$.

Opposite of the log-likelihood formula

$$
\begin{aligned}
& -\frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{1}_{y_i=1}\log(h(\beta^t\phi(\mathbf{x}))) + \mathbf{1}_{y_i=-1}\log(1-h(\beta^t\phi(\mathbf{x})))\right)\\
&= -\frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{1}_{y_i=1}\log\frac{e^{\beta^t\phi(\mathbf{x})}}{1+e^{\beta^t\phi(\mathbf{x})}} + \mathbf{1}_{y_i=-1}\log\frac{1}{1+e^{\beta^t\phi(\mathbf{x})}}\right)\\
&= \frac{1}{n}\sum_{i=1}^{n}\log\left(1+e^{-y_i(\beta^t\phi(\mathbf{x}))}\right)
\end{aligned}
$$

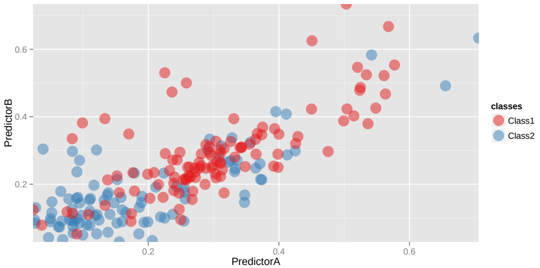
- Convex function in β!
- **Remark:** You can also use your favorite parametric model...

Example: TwoClass Dataset

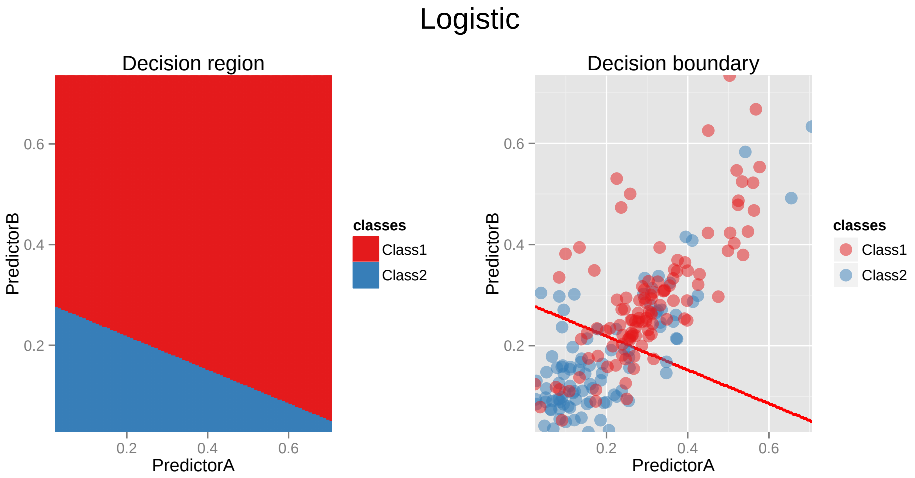
Synthetic Dataset

- Two features/covariates.
- Two classes.

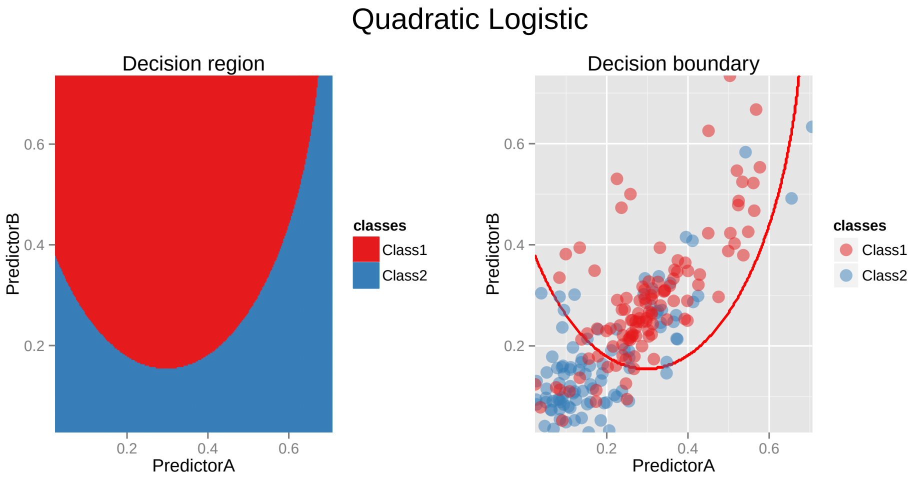
- Dataset from *Applied Predictive Modeling*, M. Kuhn and K. Johnson, Springer
- Numerical experiments with **R** and the package **caret**.



Example: Logistic



Example: Quadratic Logistic



Outline

Generative Modeling

Bayes formula

$$p_k(\mathbf{x}) = \frac{\mathbb{P}\left\{\mathbf{X} = \mathbf{x} | Y = k\right\} \mathbb{P}\left\{Y = k\right\}}{\mathbb{P}\left\{\mathbf{X} = \mathbf{x}\right\}}$$

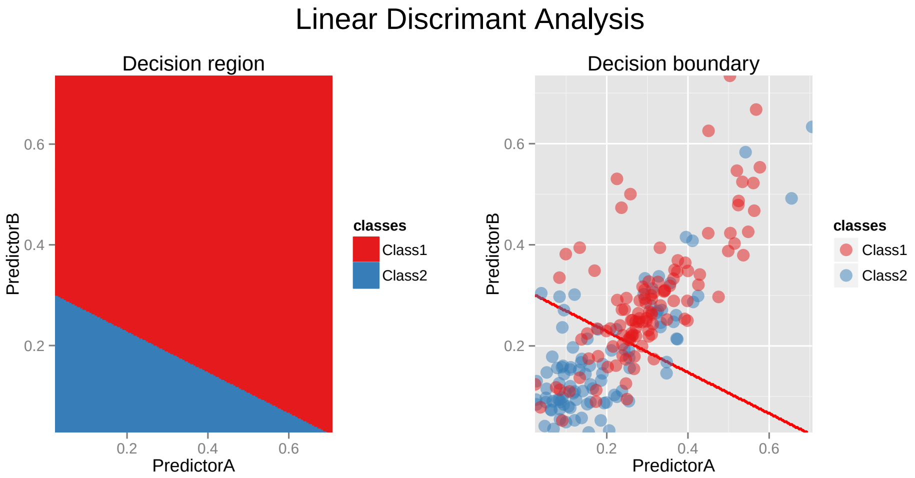
Remark: If one knows the law of X given y and the law of Y then everything is easy!

- Binary Bayes classifier (the best solution)

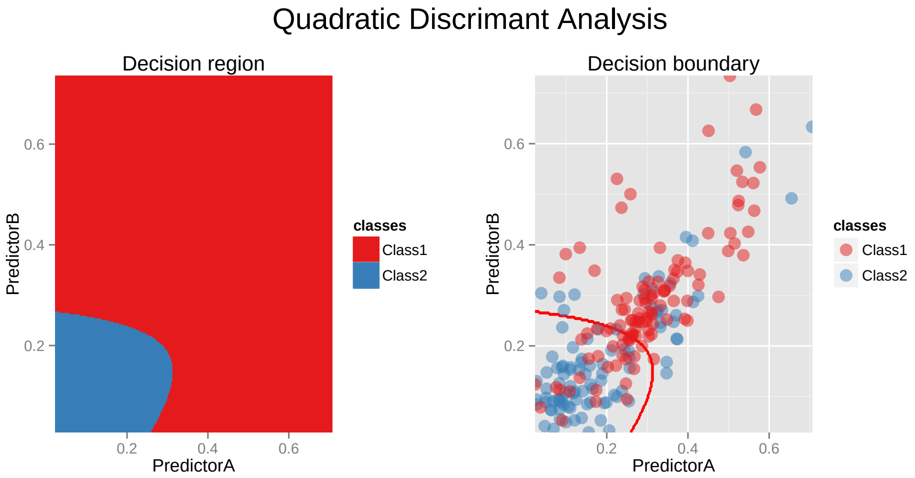
$$f^*(\mathbf{x}) = \begin{cases} +1 & \text{if } p_{+1}(\mathbf{x}) \geq p_{-1}(\mathbf{x}) \\ -1 & \text{otherwise} \end{cases}$$

- **Heuristic**: Estimate those quantities and plug the estimations.
- By using different models for $\mathbb{P}\left\{\mathbf{X} | Y\right\}$, we get different classifiers.
- **Remark:** You can also use your favorite density estimator...

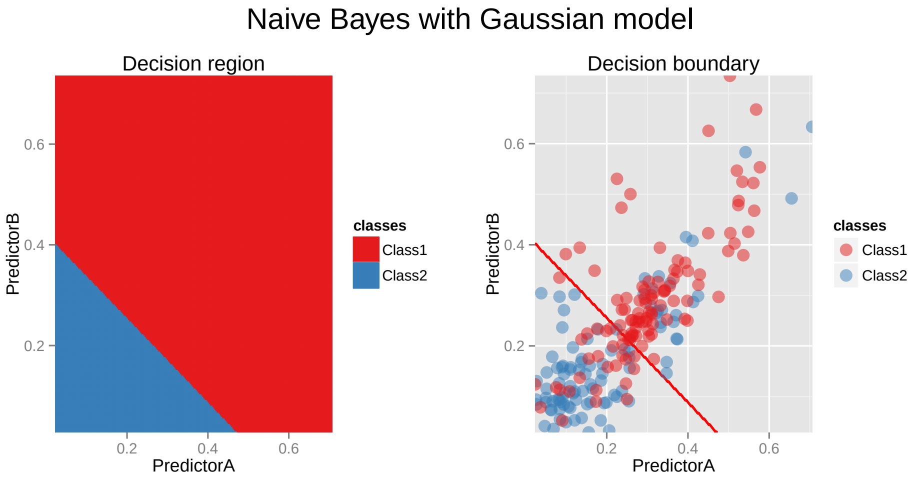
Example: LDA


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

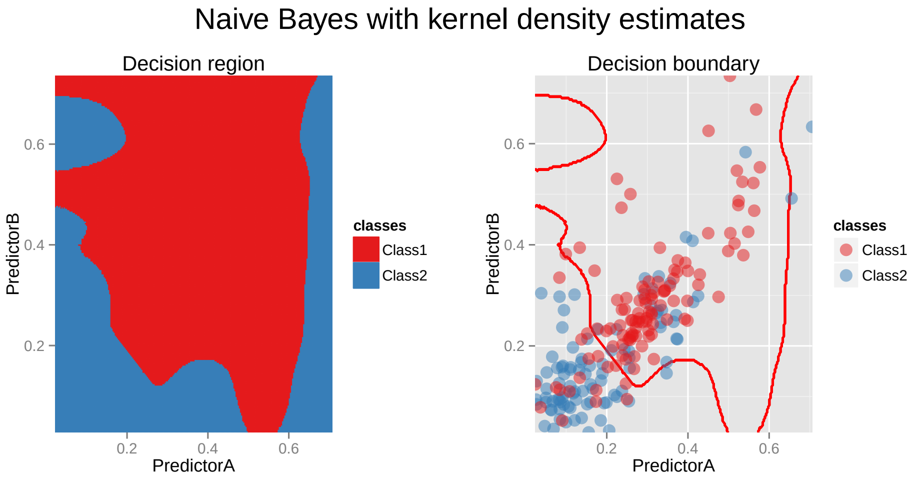
Example: QDA


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

Example: Naive Bayes

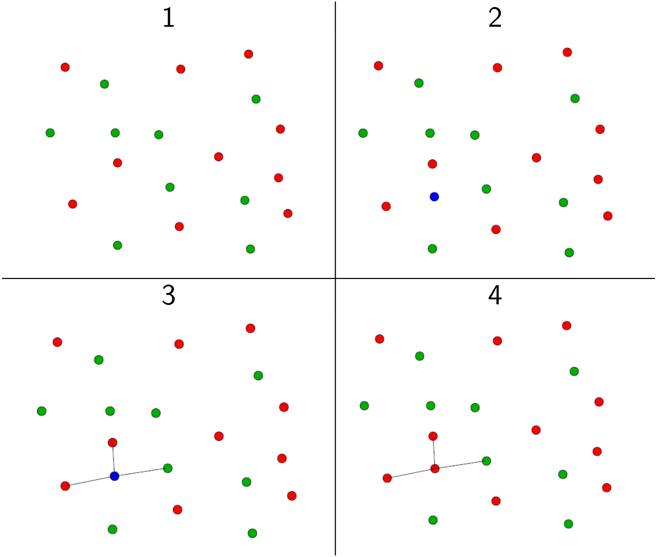


Example: Naive Bayes

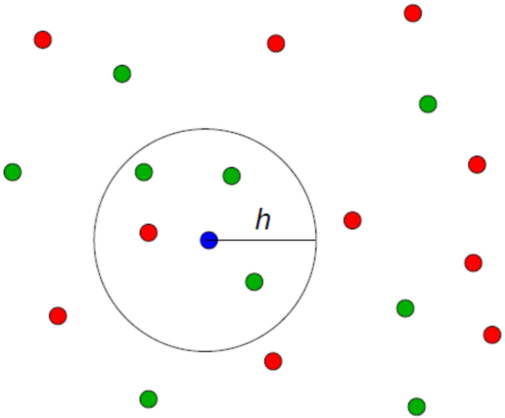


Outline

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



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



k Nearest-Neighbors

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

k-NN as local conditional density estimate

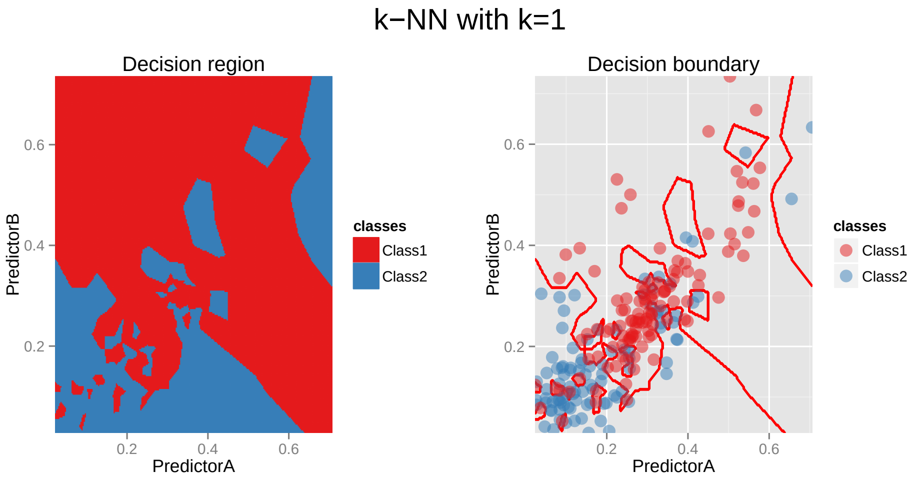
$$\widehat{p}_{+1}(\mathbf{x}) = \frac{\sum_{\mathbf{x}_i \in \mathcal{V}_{\mathbf{x}}} \mathbf{1}_{\{y_i=+1\}}}{|\mathcal{V}_{\mathbf{x}}|}$$

- KNN Classifier:

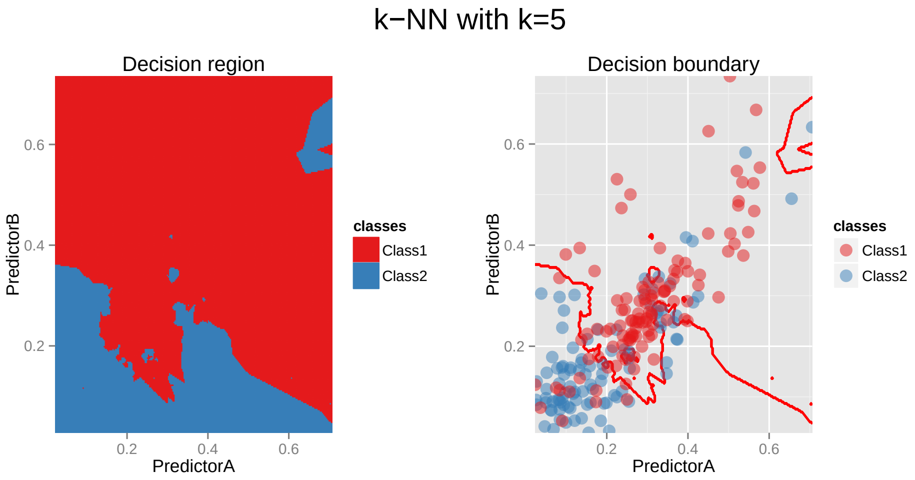
$$\widehat{f}_{KNN}(\mathbf{x}) = \begin{cases} +1 & \text{if } \widehat{p}_{+1}(\mathbf{x}) \geq \widehat{p}_{-1}(\mathbf{x}) \\ -1 & \text{otherwise} \end{cases}$$

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

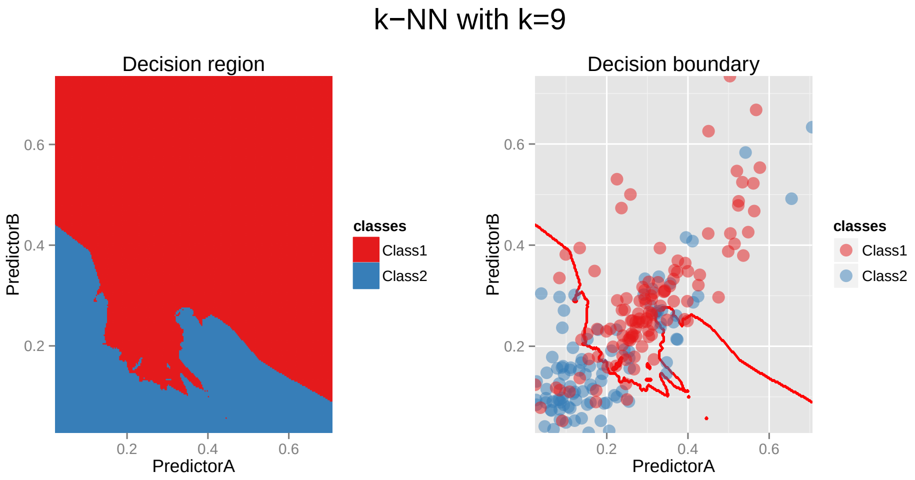
Example: KNN


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

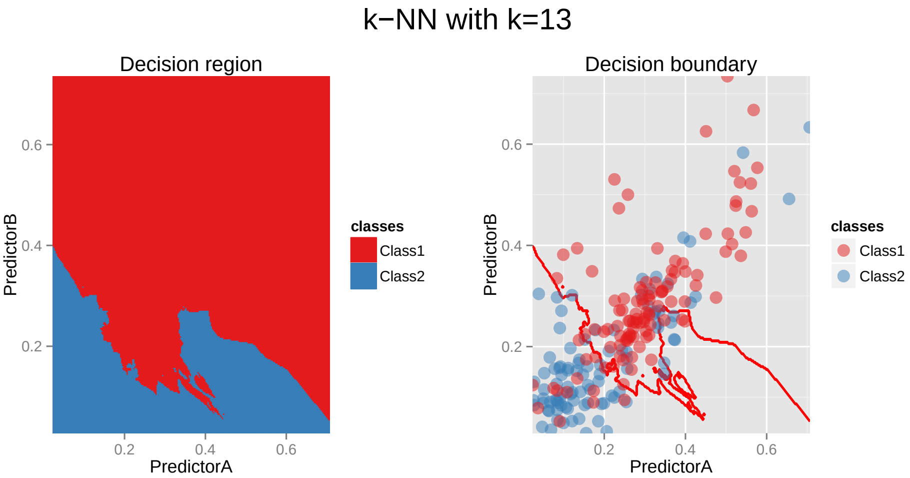
Example: KNN


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

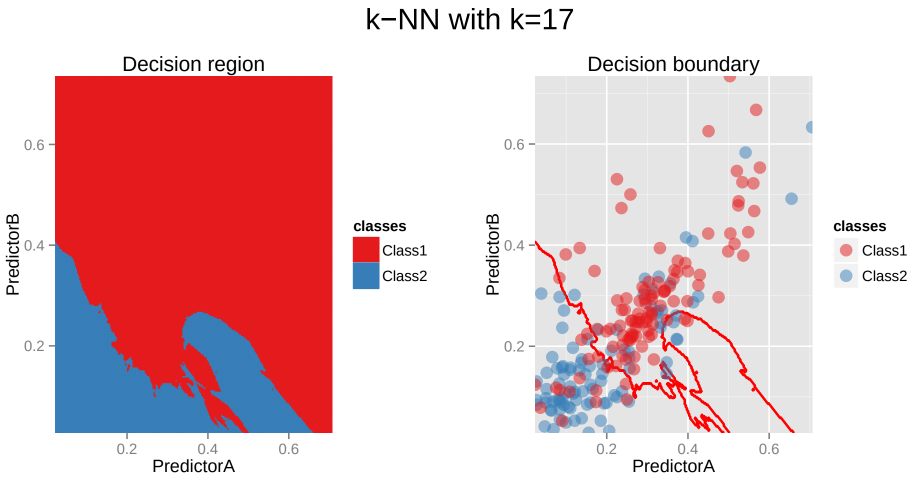
Example: KNN


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

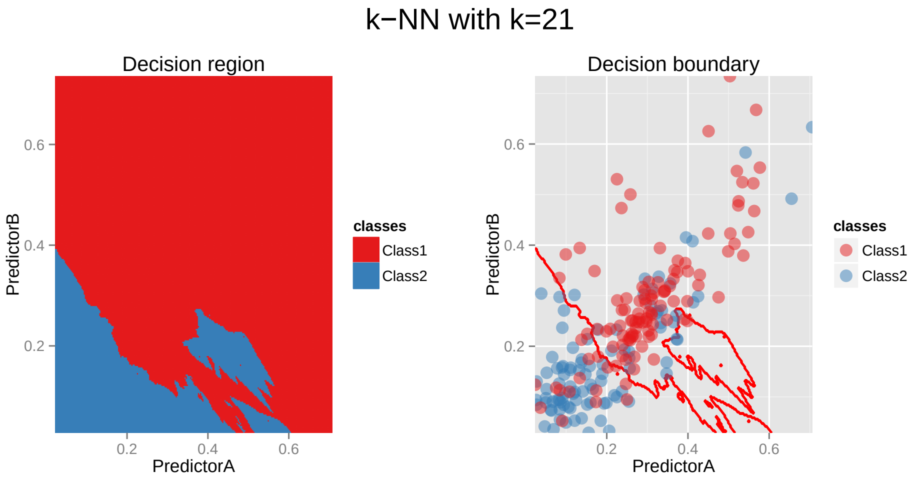
Example: KNN


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

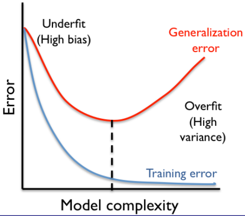
Example: KNN


k−NN with k=17
Decision region
Decision boundary
PredictorA
PredictorB
0.2
0.4
0.6
classes
Class1
Class2

Example: KNN


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

Over-fitting Issue

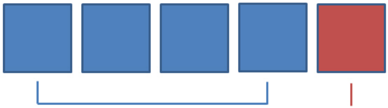


Error behaviour

- Learning/training error (error made on the learning/training set) decays when the complexity of the model increases.
- Quite different behavior when the error is computed on new observations (generalization error).

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

Cross Validation



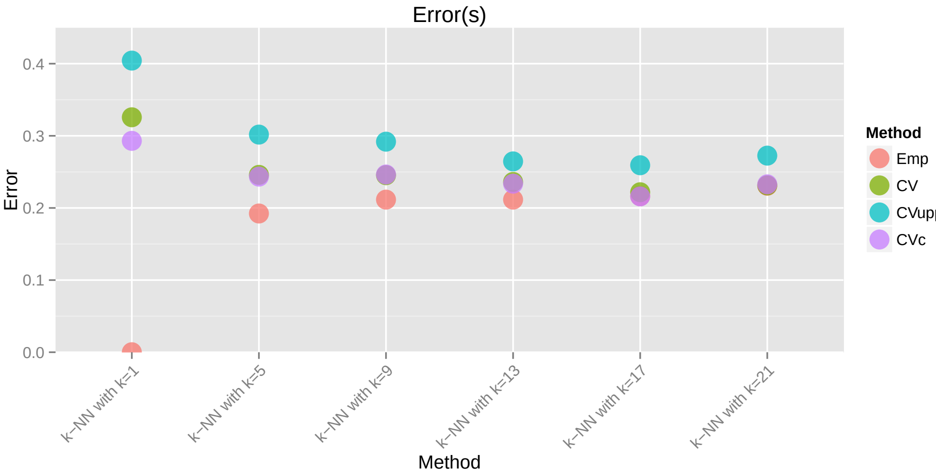
- **Very simple idea:** use a second learning/verification set to compute a verification error.
- Sufficient to avoid over-fitting!

Cross Validation

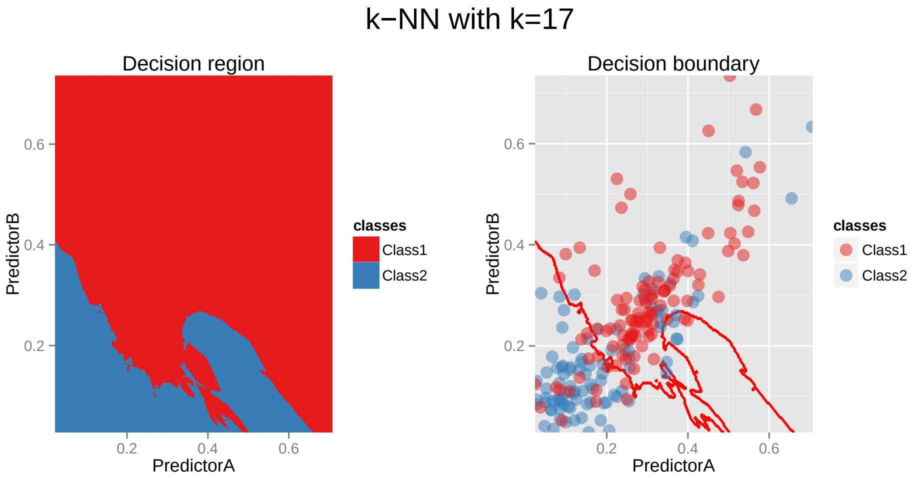
- Use $\frac{V-1}{V}n$ observations to train and $\frac{1}{V}n$ to verify!
- Validation for a learning set of size $(1-\frac{1}{V}) \times n$ instead of n!

- Most classical variations:
  - Leave One Out,
  - V-fold cross validation.
- Accuracy/Speed tradeoff: $V=5$ or $V=10$!

Example: Cross Validation for KNN



Example: KNN ($\hat{k} = 17$ using cross-validation)



An Optimization Point of View

Statistical and Optimization Points of View

How to find a good function $f \in \mathcal{H}$ that makes small

$$R(f) = \mathbb{E}\left[\ell^{0/1}(Y, f(X))\right] = \mathbb{P}\left\{Y \neq f(X)\right\} \quad ?$$

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

Problem: minimization impossible in practice for the 0-1 loss !

A Statistical Point of View (A. Fermin)

Solution: For $\mathbf{x} \in \mathbb{R}^d$, estimate $\mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})$.
Learn $Y|X$ and plug this estimate in the Bayes classifier: gen. linear models, generative modeling, kernel methods, trees

An Optimization Point of View (E. Le Pennec)

Solution: Replace the loss $\ell^{0/1}$ by an upper bound ℓ' which allows the minimization: SVM, Neural Network, trees

Outline

Outline

Empirical Risk Minimization

- The best solution f^* is the one minimizing

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

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 empirical loss

$$\widehat{f} = f_{\widehat{\theta}} = \underset{f_\theta, \theta \in \Theta}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f_\theta(x_i))$$

- Unusable for the $\ell^{0/1}$ loss!
- Solution: convexification/regularization of the risk...
- Examples: SVM, (Deep) Neural Networks, Trees

Logistic Revisited

- Ideal solution:

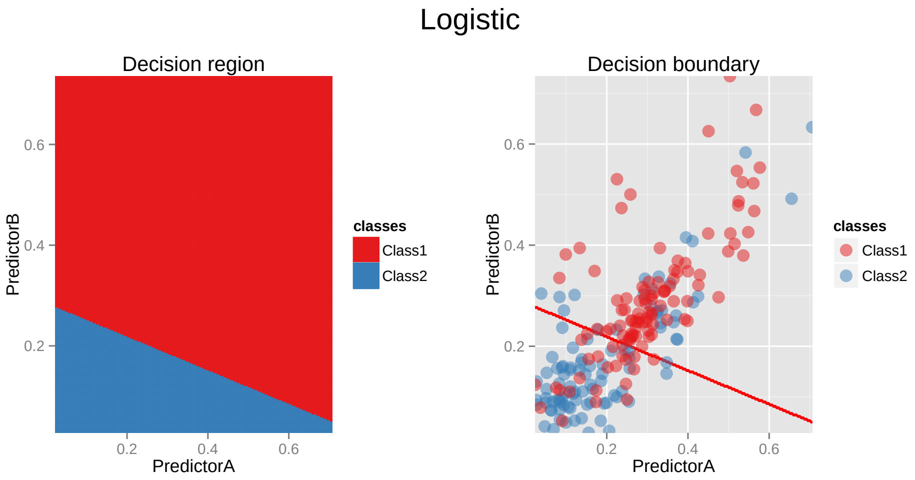
$$\widehat{f} = \operatorname*{argmin}_{f \in \mathcal{S}} \frac{1}{n} \sum_{i=1}^{n} \ell^{0/1}(y_i, f(x_i))$$

Logistic regression

- Use $f(x) = \langle \beta, x \rangle + b$.
- Use the logistic loss $\ell'(y, f) = \log_2(1 + e^{-yf})$, i.e. the -log-likelihood.

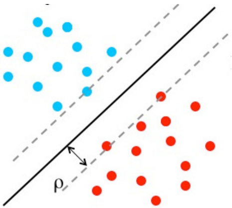
- Different vision than the statistician but same algorithm!

Logistic Revisited



Outline

Ideal Separable Case

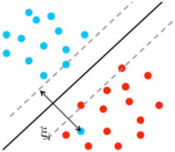


- Linear classifier: $\text{sign}(\langle \beta, x \rangle + b)$
- Separable case: $\exists (\beta, b), \forall i, y_i(\langle \beta, x \rangle + b) > 0$!

How to choose (β, b) so that the separation is maximal?

- Strict separation: $\exists (\beta, b), \forall i, y_i(\langle \beta, x \rangle + b) \geq 1$
- Maximize the distance between $\langle \beta, x \rangle + b = 1$ and $\langle \beta, x \rangle + b = -1$.
- Equivalent to the minimization of $\|\beta\|^2$.

Non Separable Case



- What about the non separable case?
- Relax the assumption that $\forall i, y_i(\langle \beta, x \rangle + b) \geq 1$.
- Naive attempt:

$$\operatorname{argmin} \|\beta\|^2 + C\frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{y_i(\langle \beta, x \rangle + b) \leq 1}$$

- Non convex minimization.

SVM: better convex relaxation!

$$\operatorname{argmin} \|\beta\|^2 + C\frac{1}{n}\sum_{i=1}^{n} \max(1 - y_i(\langle \beta, x \rangle + b), 0)$$

SVM as a Penalized Convex Relaxation

- Convex relaxation:

$$\operatorname{argmin} \|\beta\|^2 + C\frac{1}{n}\sum_{i=1}^{n} \max(1 - y_i(\langle\beta, x\rangle + b), 0)$$

$$= \operatorname{argmin} \frac{1}{n}\sum_{i=1}^{n} \max(1 - y_i(\langle\beta, x\rangle + b), 0) + \frac{1}{C}\|\beta\|^2$$

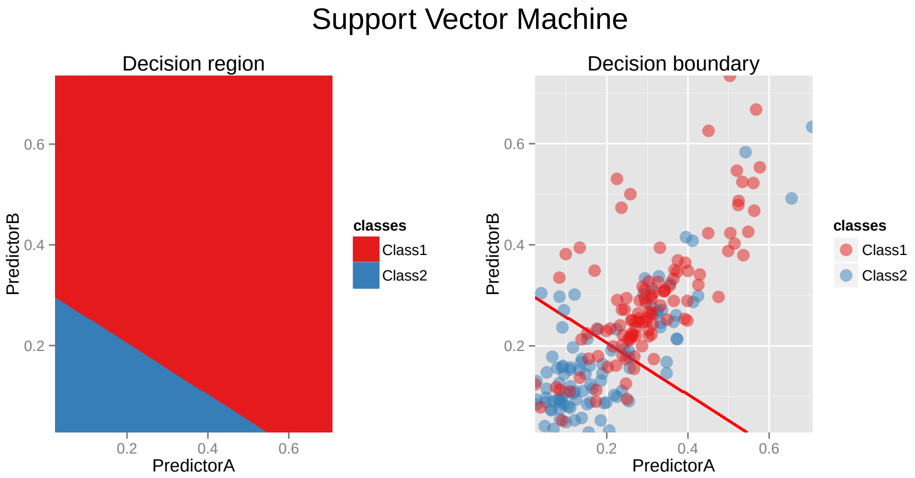
- **Prop:** $\ell^{0/1}(y_i, \operatorname{sign}(\langle\beta, x\rangle + b)) \leq \max(1 - y_i(\langle\beta, x\rangle + b), 0)$

Penalized convex relaxation (Tikhonov!)

$$\frac{1}{n}\sum_{i=1}^{n}\ell^{0/1}(y_i, \operatorname{sign}(\langle\beta, x\rangle + b))$$

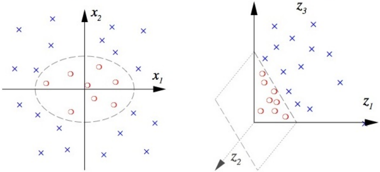
$$\leq \frac{1}{n}\sum_{i=1}^{n} \max(1 - y_i(\langle\beta, x\rangle + b), 0) + \frac{1}{C}\|\beta\|^2$$

SVM


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

The Kernel Trick

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



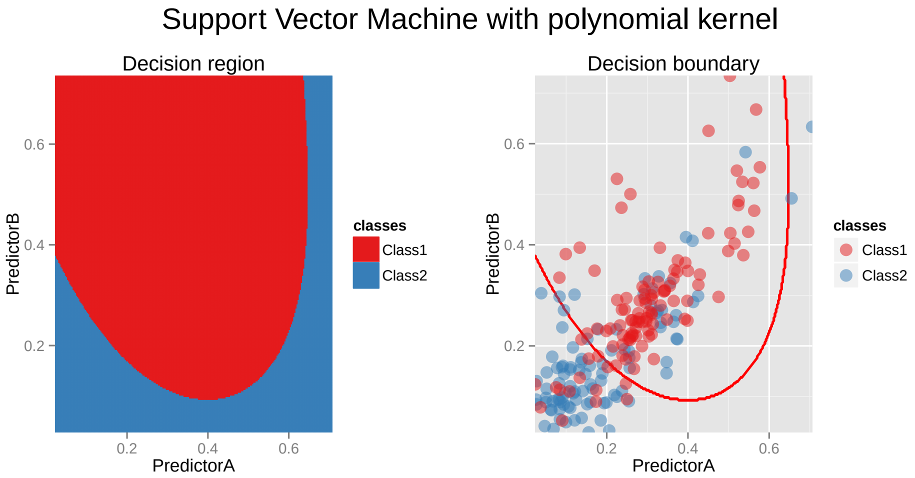
- Non linear separation: just replace x by a non linear $\Phi(x)$...

Kernel trick

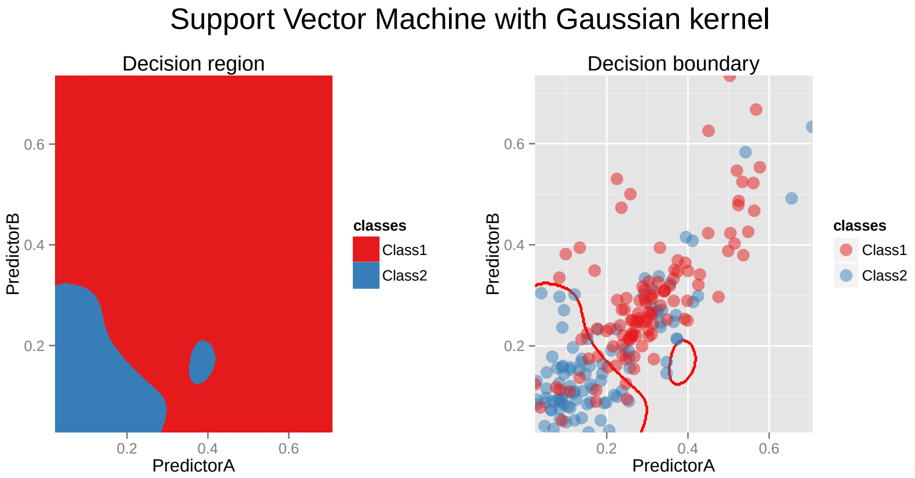
- Computing $k(x, y) = \langle \Phi(x), \Phi(y) \rangle$ may be easier than computing $\Phi(x)$, $\Phi(y)$ and then the scalar product!
- Φ can be specified through its definite positive kernel k.

- Examples: Polynomial kernel $k(x, y) = (1 + \langle x, y \rangle)^d$, Gaussian kernel $k(x, y) = e^{-\|x-y\|^2/2}$,...
- RKHS setting!
- Can be used in (logistic) regression and more...

SVM


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

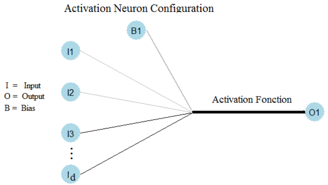
SVM


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

Outline

3. An Optimizer Point of View
   - SVM
   - (Deep) Neural Networks
   - Tree Based Methods
4. Model and Variable Selection
   - Models
   - Model Selection
5. Big Data

Artificial Neuron and Logistic Regression



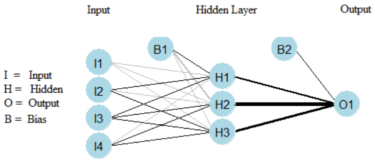
Artificial neuron

- Structure:
  - Mix inputs with a weighted sum,
  - Apply a (non linear) transfer function to this sum,
  - Eventually 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.

Neural network

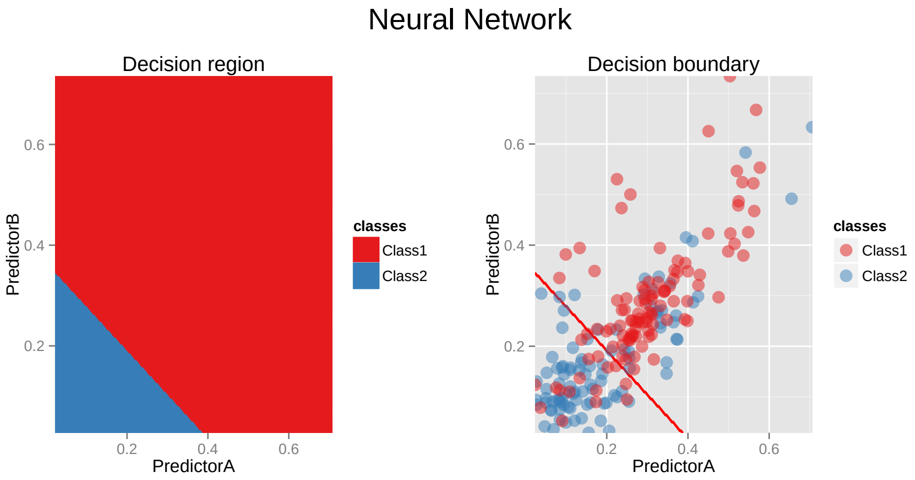


Neural network structure

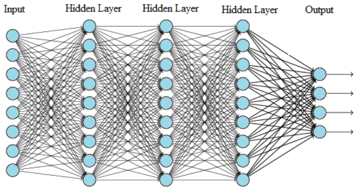
- Cascade of artificial neurons organized in layers
- Thresholding decision only at the output layer

- Most classical case use logistic neurons and the -log-likelihood as the criterion to minimize.
- Classical (stochastic) gradient descent algorithm (Back propagation)
- Non convex and thus may be trapped in local minima.

Neural network



Deep Neural Network

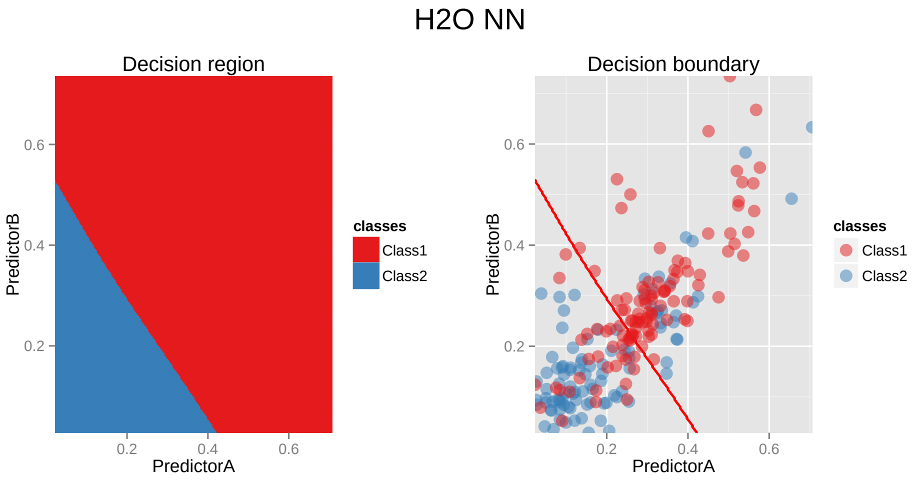


Deep Neural Network structure

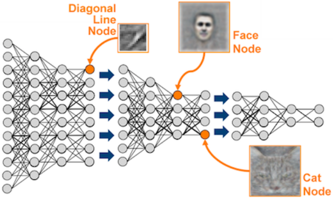
- Deep cascade of layers!

- No conceptual novelty!
- Bet on (clever?) randomized initialization and stochastic optimization scheme... and huge computational power!
- Very impressive results!

Deep Neural Network



Deep Learning



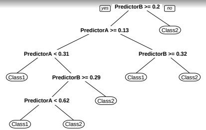
Family of Machine Learning algorithm combining:

- a (deep) multilayered structure,
- a (clever?) randomized initialization,
- a stochastic tuning optimization.

- Examples: Deep Neural Network, Deep (Restricted) Boltzman Machine, Stacked Encoder...
- Appears to be very efficient but lack of theoretical foundation!

Outline

Regression Trees

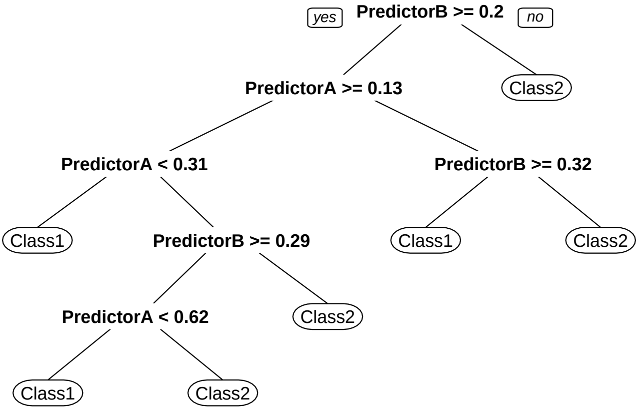


Tree principle

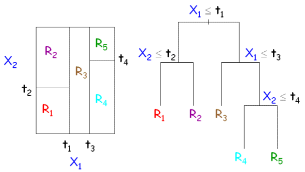
- Construction of a recursive partition through a tree structured set of questions (splits around a given value of a variable)
- For a given partition, statistical approach **and** optimization approach yields the same classifier!
- A simple majority vote in each leaf

- Quality of the prediction depends on the tree (the partition).
- Issue: Minim. of the (penalized) empirical error 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


yes
PredictorB >= 0.2
no
PredictorA >= 0.13
Class2
PredictorA < 0.31
PredictorB >= 0.32
Class1
PredictorB >= 0.29
Class1
Class2
PredictorA < 0.62
Class2
Class1
Class2

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 *homogeneous*

- CART: empirical loss based criterion

$$C(R,\overline{R}) = \sum_{x_i \in R} \ell(y_i, y(R)) + \sum_{x_i \in \overline{R}} \ell(y_i, y(\overline{R}))$$

- CART: Gini index (classification)

$$C(R,\overline{R}) = \sum_{x_i \in R} p(R)(1-p(R)) + \sum_{x_i \in \overline{R}} p(\overline{R})(1-p(\overline{R}))$$

- C4.5: entropy based criterion (Information Theory)

$$C(R,\overline{R}) = \sum_{x_i \in R} H(R) + \sum_{x_i \in \overline{R}} H(\overline{R})$$

- CART with Gini is probably the most used technique...
- Other criterion based on 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 one minimizing the criterion

- Variations: split at all categories of a categorical variables (ID3), split at a fixed position (median/mean)
- Stopping rules:
  - when a leaf/region contains less than a prescribed number of observations
  - when the region is sufficiently homogeneous...
- May lead to a quite complex tree / Over-fitting possible!

Pruning

- Model select. 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

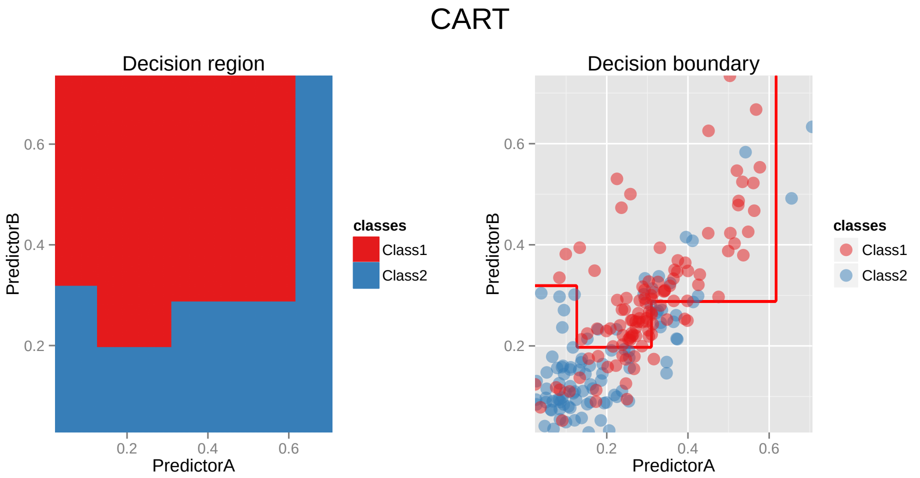
- The predictor in a leaf depends only on the values in this leaf.
- Efficient bottom-up (dynamic programming) algorithm if the criterion used satisfies an additive property

$$C(\mathcal{T}) = \sum_{\mathcal{L} \in \mathcal{T}} c(\mathcal{L})$$

- Example: AIC / CV.

- Limits over-fitting...

CART


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

Ensemble methods

- Lack of robustness for single trees.
- How to combine trees?

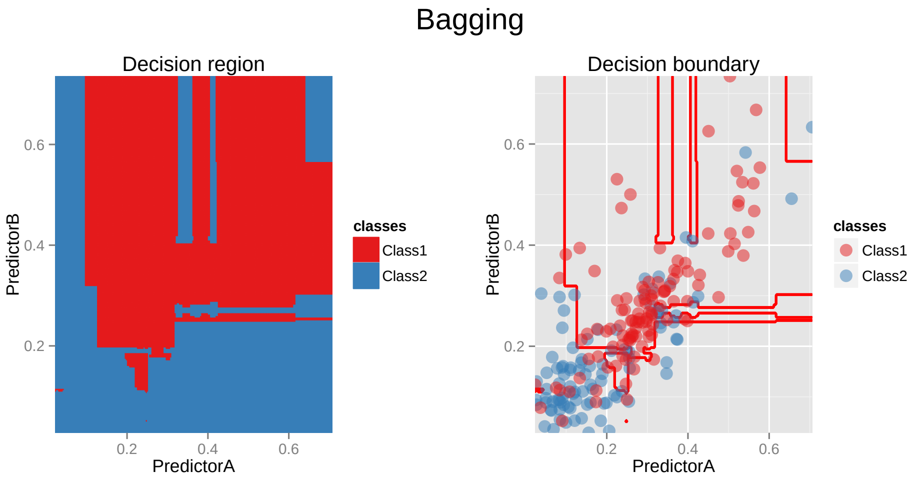
Parallel construction

- Construct several trees from bootstrapped samples and average the responses (bagging)
- Add more randomness in the tree construction (random forests)

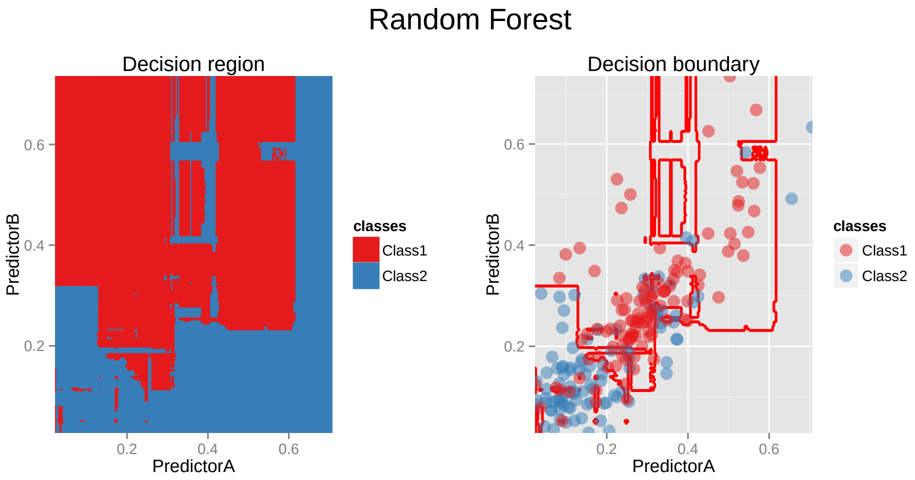
Sequential construction

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

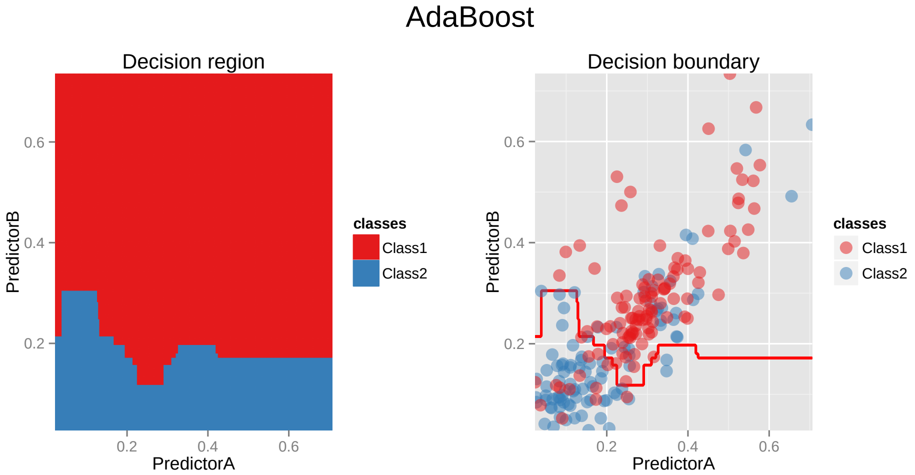
Ensemble methods



Ensemble methods



Ensemble methods



Outline

Outline

Logistic Regression

- Ideal solution:

$$f^*(x) = \operatorname{argmax} \mathbb{P}\{Y|x\}$$

Logistic

- Model $Y|X$ with a logistic model.
- Estimate its parameters with a Maximum Likelihood approach.
- Plug the estimate in the Bayes classifier.

- Model hyperparameters:
  - Features
  - Parametric model...

Generative Modeling

- Ideal solution:

$$f^*(x) = \operatorname{argmax} \mathbb{P}\left\{Y|x\right\}$$

Generative Modeling

- Estimate $X|Y$ with a density estimator as well as $\mathbb{P}\left\{Y\right\}$
- Deduce using the Bayes formula an estimate $Y|X$.
- Plug the estimate in the Bayes classifier.

- Model hyperparameters:
  - Features
  - Generative model

Kernel Method

- Ideal solution:

$$f^*(x) = \operatorname{argmax} \mathbb{P}\{Y|x\}$$

Kernel methods

- Estimate $Y|X$ with a kernel conditional density estimator.
- Plug the estimate in the Bayes classifier.

- Model hyperparameters:
  - Features
  - Bandwidth and kernel

Logistic Regression

- Ideal solution:

$$f^* = \operatorname*{argmin}_{f \in \mathcal{S}} \mathbb{E}\left[\ell^{0/1}(Y, f(X))\right]$$

Logistic

- Replace $\ell^{0/1}$ by the logistic loss.
- Add a penalty $\lambda \|f\|_p$
- Compute the minimizer.

- Model hyperparameters:
  - Features
  - Penalty and regularization parameter.

SVM

- Ideal solution:

$$f^* = \operatorname*{argmin}_{f \in \mathcal{S}} \mathbb{E}\left[\ell^{0/1}(Y, f(X))\right]$$

SVM

- Replace the expectation by its empirical counterpart.
- Replace $\ell^{0/1}(y, f) = \mathbf{1}_{y=f}$ by $\ell'(y, f) = (1 - yf)_+$.
- Add a penalty $\lambda \|f\|_{\mathcal{S}}^2$.
- Compute the minimizer.

- Model hyperparameters:
  - Features
  - $\mathcal{S}$ RKHS structure: features mapping and metric
  - Regularization parameters λ

(Deep) Neural Networks

- Ideal solution:

$$f^* = \operatorname*{argmin}_{f \in \mathcal{S}} \mathbb{E}\left[\ell^{0/1}(Y, f(X))\right]$$

NN

- Neuron: $x \mapsto \sigma(\langle \beta, x \rangle + b)$
- Neural Network: Convolution system of neurons.
- Replace $\ell^{0/1}(y, f)$ by a smooth/convex loss.
- Minimize the empirical loss using the backprop algorithm (gradient descent)

- Model hyperparameters:
  - Features
  - Net architecture, activation function
  - Initialization strategy
  - Optimization strategy (and regularization strategy)

Tree and Boosting

- Ideal solution:

$$f^*(x) = \operatorname{argmax} \mathbb{P}\left\{Y|x\right\} \quad \text{and} \quad f^* = \operatorname*{argmin}_{f \in \mathcal{S}} \mathbb{E}\left[\ell^{0/1}(Y, f(X))\right]$$

Single tree

- Greedy Partition construction.
- Local conditional density estimation / loss minimization.
- Suboptimal tree optimization through a relaxed criterion

Bagging/Random Forest

- Averaging of several predictors (statistical point of view)

Boosting

- Best interpretation as a minimization of the exponential loss $\ell(y, f) = e^{-yf}$ (optimization point of view)

Outline

Model Selection

Models

- How to design models? (Model/feature design)
- How to chose among several models? (Model/feature selection)

- Key to obtain good performance!

Approximation error and estimation error (Bias/Variance)

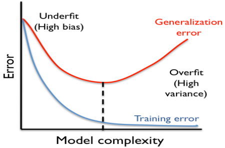
$$\mathcal{R}(\widehat{f}_{\mathcal{S}}) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(f^*_{\mathcal{S}}) - \mathcal{R}(f^*)}_{\text{Approximation error}} + \underbrace{\mathcal{R}(\widehat{f}_{\mathcal{S}}) - \mathcal{R}(f^*_{\mathcal{S}})}_{\text{Estimation error}}$$

- Approximation error can be large for not suitable model $\mathcal{S}$!
- Estimation error can be large if the model is complex!

- Need to find the good balance automatically!

Model Selection

- Empirical error biased toward complex models!



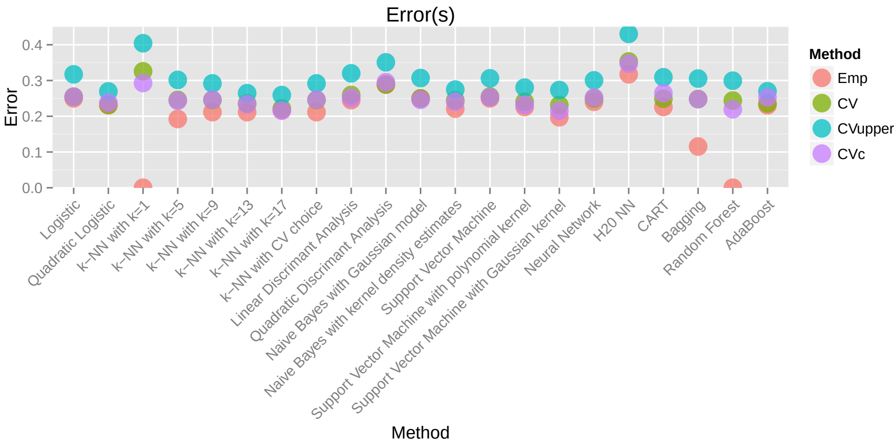
Selection criterion

- **Cross validation:** Very efficient (and almost always used in practice!) but slightly biased as it target uses only a fraction of the data.
- **Penalization approach:** use empirical loss criterion but penalize it by a term increasing with the complexity of $\mathcal{S}$

$$R_n(\widehat{f}_{\mathcal{S}}) \rightarrow R_n(\widehat{f}_{\mathcal{S}}) + \text{pen}(\mathcal{S})$$

and choose the model with the smallest penalized risk.

Cross Validation



Ensemble methods

- How to combine several predictors (models)?
- Two strategies: mixture or sequential

Mixture

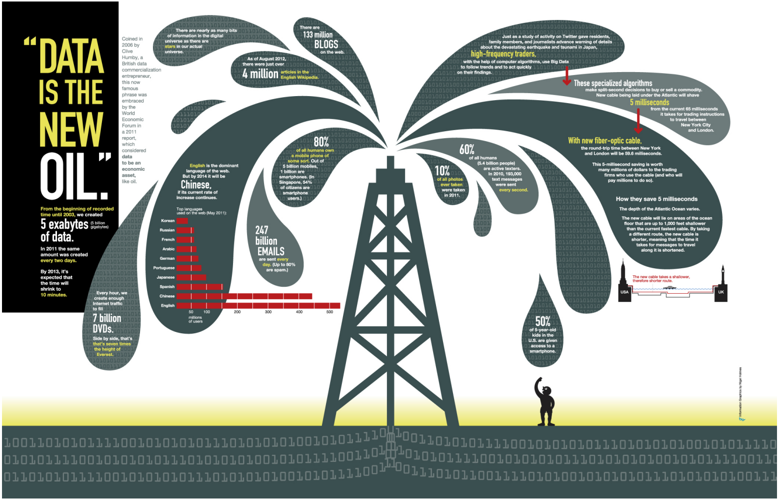
- Model averaging
- Data dependent model averaging (learn mixture weights)

Stagewise

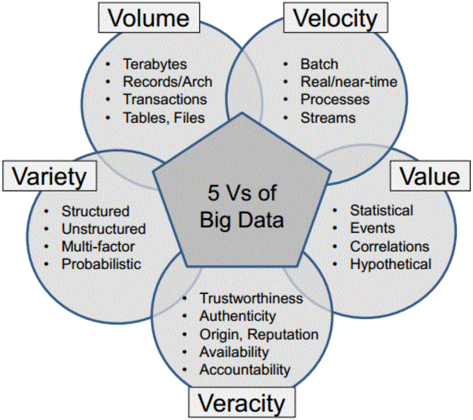
- Modify learning procedure according to current results.
- Boosting, Cascade...

Outline

Data is the new Oil!

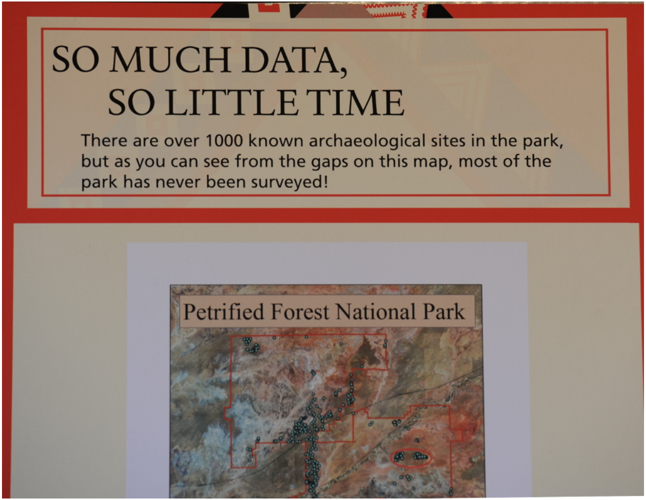


The 5 Vs of Big Data



Lots of Words!

Petrified Forest!



Doing Data Science

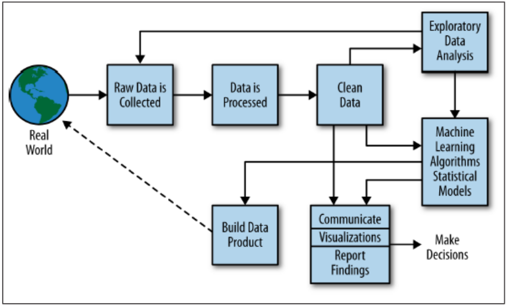


Figure 2-2. The data science process

Doing Data Science: Straight talk from the frontline

- Rachel Schutt, Cathy O'Neil - O'Reilly
- Art of data driven decision / evaluation.

A new Context

Data everywhere

- Huge volume,
- Huge variety...

Affordable computation units

- Cloud computing
- Graphical Processor Units (GPU)...

- Growing academic and industrial interest!

Big Data is (quite) Easy

Example of *off the shelves* solution

Spark   amazon web services

```
def run(params: Params) {
  val conf = new SparkConf()
      .setAppName(s"BinaryClassification with $params")
  val sc = new SparkContext(conf)

  Logger.getRootLogger.setLevel(Level.WARN)

  val examples = MLUtils.loadLibSVMFile(sc, params.input).cache()

  val splits = examples.randomSplit(Array(0.8, 0.2))
  val training = splits(0).cache()
  val test = splits(1).cache()
  val numTraining = training.count()
  val numTest = test.count()
  println(s"Training: $numTraining, test: $numTest.")
  examples.unpersist(blocking = false)

  val updater = params.regType match {
    case L1 => new L1Updater()
    case L2 => new SquaredL2Updater()
  }

  val algorithm = new LogisticRegressionWithSGD()
      algorithm.optimizer
        .setNumIterations(params.numIterations)
        .setStepSize(params.stepSize)
        .setUpdater(updater)
        .setRegParam(params.regParam)
  val model = algorithm.run(training).clearThreshold()

  val prediction = model.predict(test.map(_.features))
  val predictionAndLabel = prediction.zip(test.map(_.label))

  val metrics = new BinaryClassificationMetrics(predictionAndLabel)
  val myMetrics = new MyBinaryClassificationMetrics(predictionAndLabel)

  println(s"Empirical CrossEntropy = ${myMetrics.crossEntropy()}.")
  println(s"Test areaUnderPR = ${metrics.areaUnderPR()}.")
  println(s"Test areaUnderROC = ${metrics.areaUnderROC()}.")

  sc.stop()
}
```

Big Data is (quite) Easy

Example of *off the shelves* solution

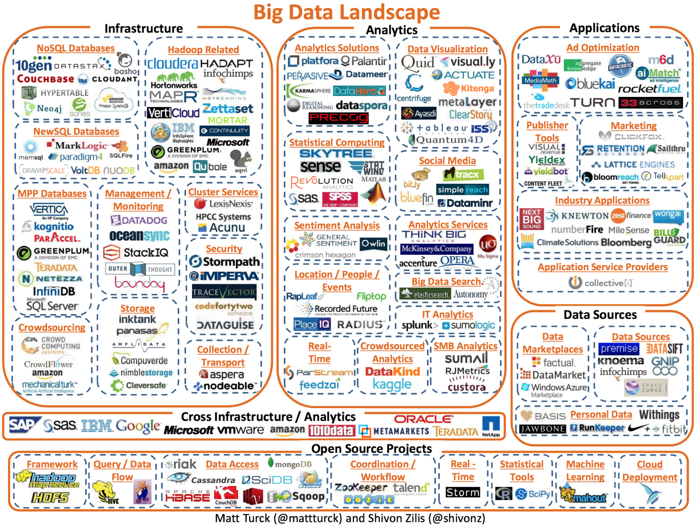
Spark   amazon web services

```
export AWS_ACCESS_KEY_ID=<your-access-keyid>
export AWS_SECRET_ACCESS_KEY=<your-access-key-secret>
cellule/spark/ec2/sparl-ec2 -i cellule.pem -k cellule -s <number of machines> launch <cluster-name>
ssh -i cellule.pem root@<your-cluster-master-dns>
spark-ec2/copy-dir ephemeral-hdfs/conf
ephemeral-hdfs/bin/hadoop distcp s3n://celluledecalcul/dataset/raw/train.csv /data/train.csv
scp -i cellule.pem cellule/challenge/target/scala-2.10/target/scala-2.10/challenges_2.10-0.0.jar

cellule/spark/bin/spark-submit \
        --class fr.cc.challenge.Preprocess \
        challenges_2.10-0.0.jar \
        /data/train.csv \
        /data/train2.csv

cellule/spark/bin/spark-submit \
       --class fr.cc.sparktest.LogisticRegression \
       challenges_2.10-0.0.jar \
       /data/train2.csv
```

$\Rightarrow$ Logistic regression for arbitrary large dataset!

A Complex Ecosystem!

A Complex Ecosystem!



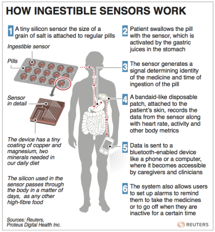
New Interdisciplinary Challenges

- Applied math **AND** Computer science
- Huge importance of domain specific knowledge: physics, signal processing, biology, health, marketing...

Some joint math/computer science challenges

- Data acquisition
- Unstructured data and their representation
- Huge dataset and computation
- High dimensional data and model selection
- Learning with less supervision
- Visualization

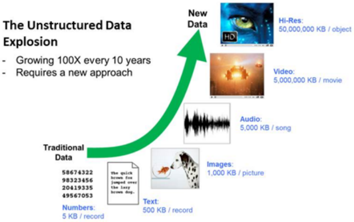
Data acquisition

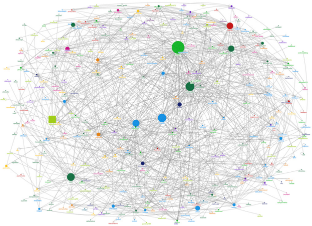


Some challenges

- How to measure new things?
- How to choose what to measure?
- How to deal with distributed sensors?
- How to look for new sources of informations?

Unstructured Data

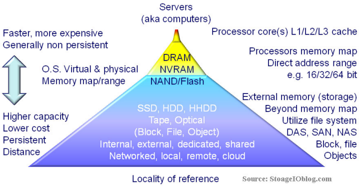


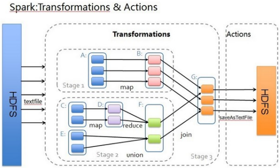


Some challenges

- How to store efficiently the data?
- How to describe (model) them to be able to process them?
- How to combine data of different nature?
- How to learn dynamics?

Huge Dataset

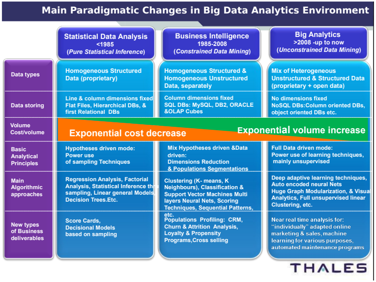




Some challenges

- How to take into account the locality of the data?
- How to construct distributed architectures?
- How to design adapted algorithms?

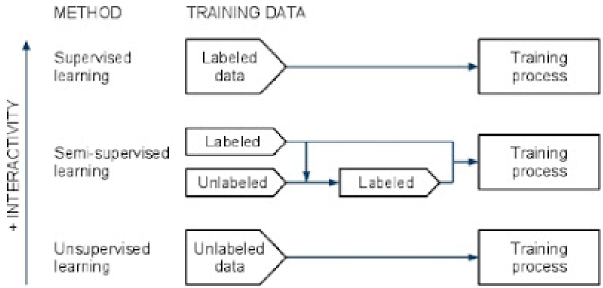
High Dimensional Data



Some challenges

- How to describe (model) the data?
- How to reduce the data dimensionality?
- How to select/mix models?

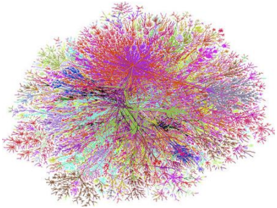
Learning and Supervision

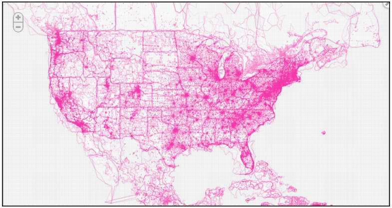


Some challenges

- How to learn with the less possible interactions?
- How to learn simultaneously several related tasks?

Visualization





Some challenges

- How to look at the data?
- How to present results?
- How to help taking better informed decision?

Bibliography

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

G. James, D. Witten, T. Hastie and R. Tibshirani (2013)
An Introduction to Statistical Learning with Applications in R
Springer Series in Statistics.

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

R. Schutt, and C. O'Neil (2014)
Doing Data Science: Straight talk from the frontline
O'Reilly