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

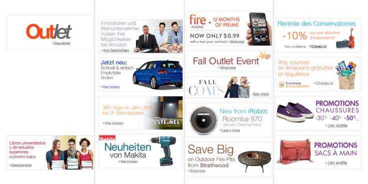
CIMFAV, 28/12/2014

Motivation

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

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

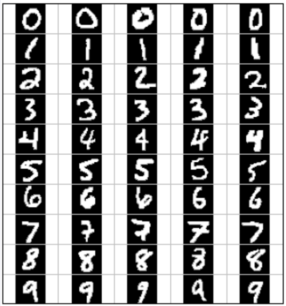
Marketing: advertisement, recommendation...



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

Motivation

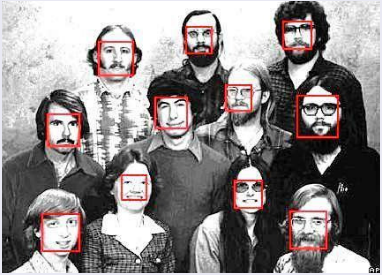
Number Recognition



- Data: Annotated database of images
- Input: Image.
- Output: Corresponding number.

Motivation

Face Detection



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

Motivation

Spam detection (Text classification)

- Data: 4601 emails sent to an individual (George, at HP labs, before 2000)
- Input: email
- Output : Spam/ No Spam

Motivation

Spam

WINNING NOTIFICATION
We are pleased to inform you of the result of the Lottery Winners International programs held on the 30th january 2005. [...] You have been approved for a lump sum pay out of 175,000.00 euros.
CONGRATULATIONS!!!

No Spam

Dear George,
Could you please send me the report #1248 on the project advancement? Thanks in advance.

Regards,
Cathia

goal: Detect spam in emails

input features: relative frequencies of the most commonly occurring words and punctuation marks in these email messages.
"George", "send", "Lottery", "project", "pay", "euros", "NOTIFICATION", "CONGRATULATIONS", "!", report, ...

Motivation

With the explosion of "Big Data" problems, statistical 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.
- Statistical 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 Machine Learning Framework
2. A Statistical Learner Point of View (Part 1)
   - Logistic regression
   - Class by Class modeling
   - *k* Nearest Neighbors
3. A Machine Learner Point of View (Part 2)
   - SVM
   - (Deep) Neural Networks
   - Tree Based Methods
4. Model and Variable Selection (Part 2)
   - Model Selection
   - Practical Variable Selection
   - Empirical Risk Minimization Analysis
5. Big Data (Part 2)

Statistical Learning in Classification

1. Supervised Classification
   - Binary Supervised Classification
   - Models
   - Statistical and Machine Learning Framework

2. A Statistical Learner Point of View
   - Logistic Modeling
   - Class by Class modeling
   - k Nearest-Neighbors

Outline

1. Supervised Classification
   - Binary Supervised Classification
   - Models
   - Statistical and Machine Learning Framework

2. A Statistical Learner Point of View
   - Logistic Modeling
   - Class by Class modeling
   - k Nearest-Neighbors

Outline

Binary Supervised Classification

- Output measurement $Y \in \{-1, 1\}$.
- Input measurement $\mathbf{X} = (X^{(1)}, X^{(2)}, \ldots, X^{(d)}) \in \mathbb{R}^d$
- $\{(\mathbf{X}_i, Y_i)\}_{i=1}^n$ are modeled as i.i.d random variables of a generic pair $(\mathbf{X}, Y) \in \mathbb{R}^d \times \{-1, 1\}$

- Training data : $\mathcal{D} = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$ (i.i.d. $\sim \mathbf{P}$)
- Classifier : $f : \mathbb{R}^d \to \{-1, 1\}$ measurable
- Cost/Loss function : $\ell(f(x), y)$ measure how well $f(x)$ "predicts" y For this talk $\ell(f(x), y) = \mathbf{1}_{Y \neq f(X)}$
- Goal : learn $f \in \mathcal{F} = \{\text{measurable fonctions } \mathbb{R}^d \to \{-1, 1\}\}$ s.t. the risk

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

is minimal.

Best solution

- The best solution f^* is

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

$$f^*(\mathbf{x}) = \arg\max_k \mathbb{P}(Y = k|\mathbf{X} = \mathbf{x})$$

Binary Bayes Classifier (explicit solution)

In binary classification with $0-1$ loss:

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

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

Empirical Risk minimisation

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

- Empirical risk:

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

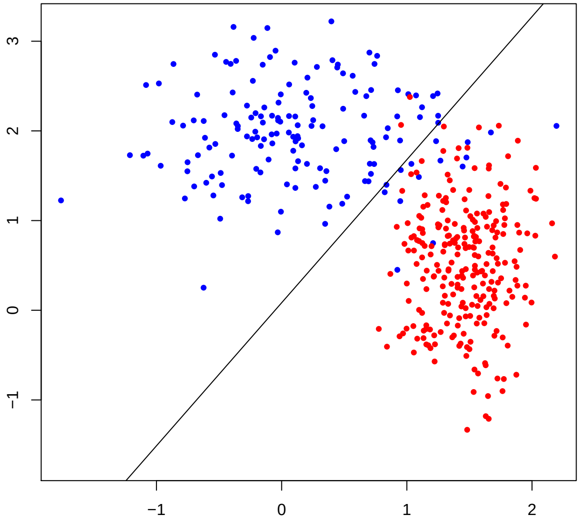
- Empirical risk minimizer over a model $\mathcal{S} \subset \mathcal{F}$:

$$\widehat{f}_{\mathcal{S}} = \operatorname*{argmin}_{f \in \mathcal{S}} \{\mathcal{R}_n(f)\}$$

- Exemple : linear discrimination

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

Example: linear discrimination



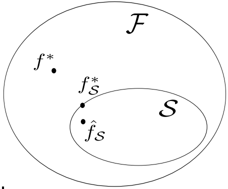
Outline

1. Supervised Classification
   - Binary Supervised Classification
   - Models
   - Statistical and Machine Learning Framework

2. A Statistical Learner Point of View
   - Logistic Modeling
   - Class by Class 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

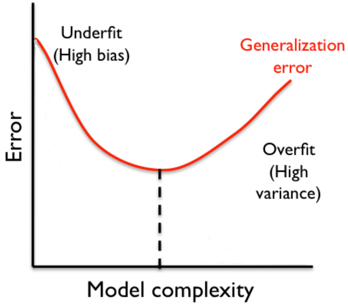


Approximation error and estimation error (Bias/Variance)

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

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

Under-fitting / Over-fitting Issue



- Different behavior for different model complexity
- Low complexity model are easily learned but the approximation large may remain large (Under-fit).
- High complexity model may contains a good ideal target but the one learned can be bad due to a high variance (Over-fit)

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

Outline

Statistical and Machine Learning Framework

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

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

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

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

Supervised Statistical Learning (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:
generalized linear models, k-nn, naive Bayes...

Supervised Machine Learning (E. Le Pennec)

Solution: Replace the loss ℓ by an upper bound ℓ' which allows the minimization: SVM, Neural Network, Boosting

Outline

Classification Rule / Algorithm

- Input: a data set $\mathcal{D}_n$
  Learn $Y|x$ or equivalently $p_k(\mathbf{x}) = \mathbb{P}\{Y = k|\mathbf{X} = \mathbf{x}\}$ (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. Class by class 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

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

Opposite of the log-lilkelihood formula

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

- Convex function in β!

Example: Edgar Anderson's Iris Data

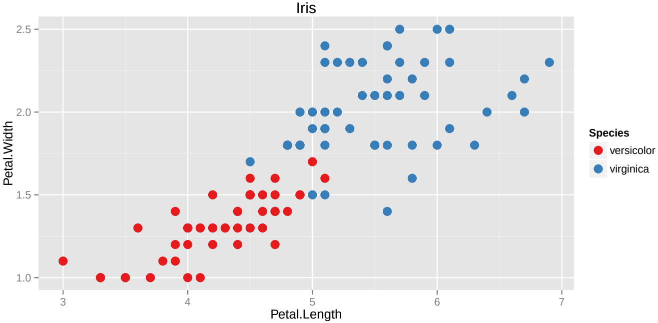
Description of this famous (Fisher's or Anderson's) dataset

- Measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris
- The species are Iris setosa, versicolor, and virginica.

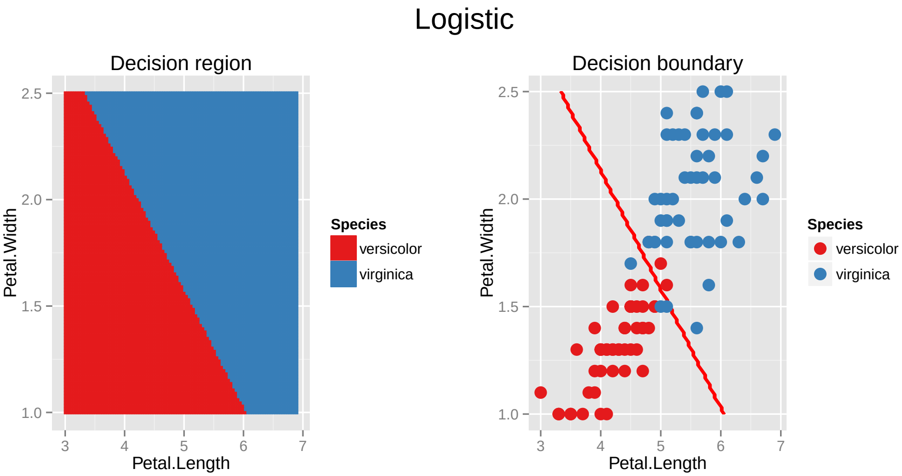
Example: Edgar Anderson's Iris Data

Simplified iris set

- Use on petal length and width.
- Restriction to two species versicolor, and virginica.



Example: Logistic



Outline

Class by Class Modeling

Bayes formula

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

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 $\mathbb{P}\{\mathbf{X}|Y\}$, we get different classifiers. Use your favorite density estimator...

Discriminant Analysis

Discriminant Analysis (Gaussian model)

- The densities are modeled as multivariate normal, i.e.,

$$\mathbb{P}\{X|Y=k\} \sim \mathcal{N}_{\mu_k,\Sigma_k}$$

- Discriminants fonctions:
  $g_k(\mathbf{x}) = \ln(\mathbb{P}\{X|Y=k\}) + \ln(\mathbb{P}\{Y=k\})$

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

- QDA (differents 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

Estimation

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

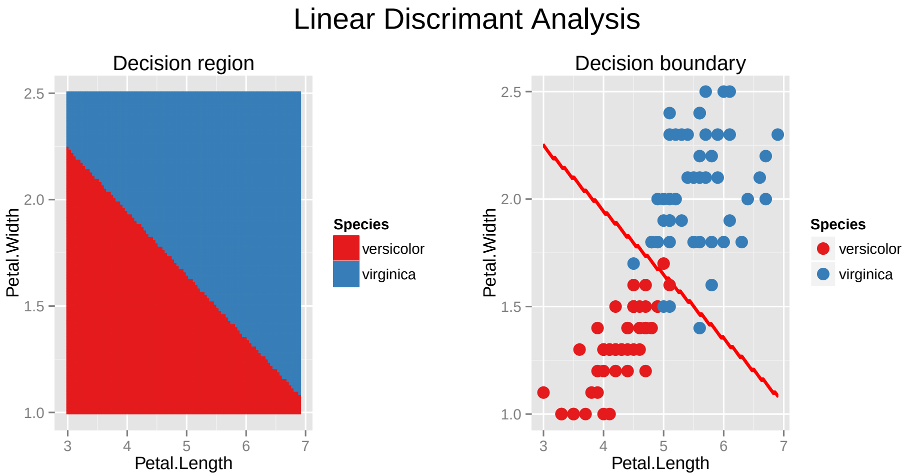
- The estimate proportion $\widehat{\mathbb{P}_k} = \frac{n_k}{n} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{\{Y_i=k\}}$
- Maximum likelihood estimate of $\widehat{\mu_k}$ and $\widehat{\Sigma_k}$ (explicit formulas)

- DA classifier

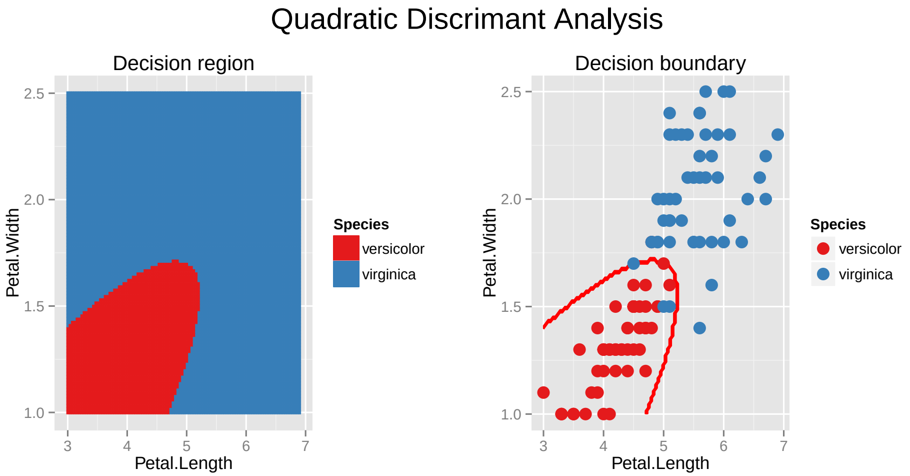
$$\widehat{f}_G(\mathbf{x}) = \begin{cases} +1 & \text{if } \widehat{g}_{+1} \geq \widehat{g}_{-1} \\ -1 & \text{otherwise} \end{cases}$$

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

Example: LDA


Linear Discriminant Analysis
Decision region
Decision boundary
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
3
4
5
6
7
Species
versicolor
virginica

Example: QDA


Quadratic Discrimant Analysis
Decision region
Decision boundary
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
3
4
5
6
7
Species
versicolor
virginica

Naive Bayes

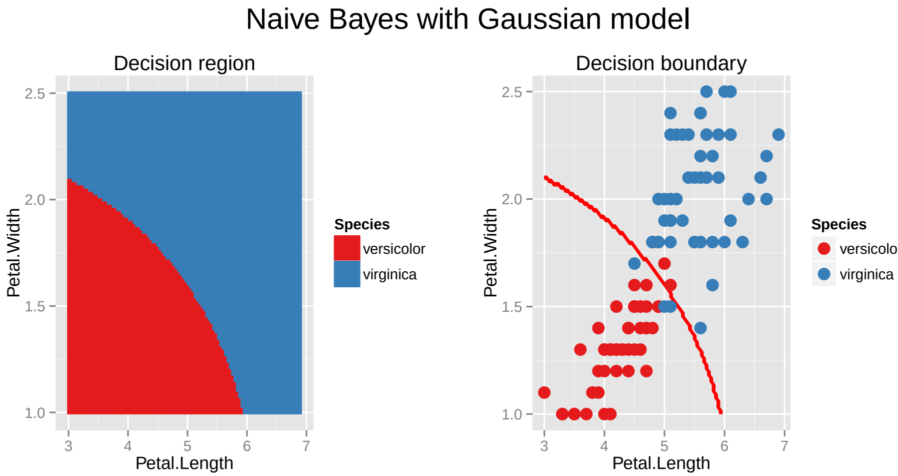
Naive Bayes

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

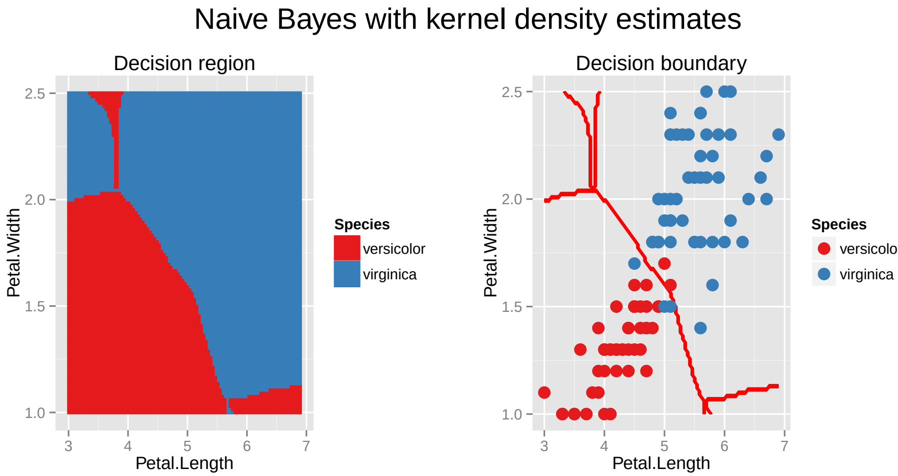
$$\mathbb{P}\{X|Y\} = \prod_{i=1}^{d} \mathbb{P}\left\{X^{(i)}\middle|Y\right\}$$

  - 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

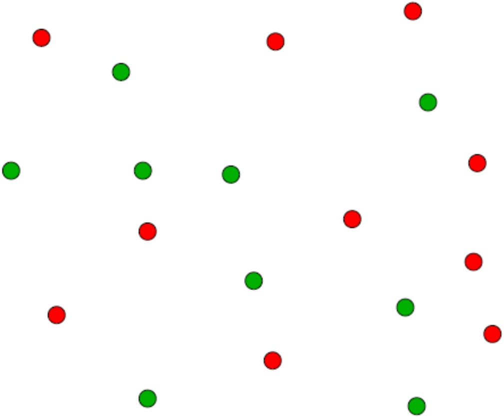


Example: Naive Bayes

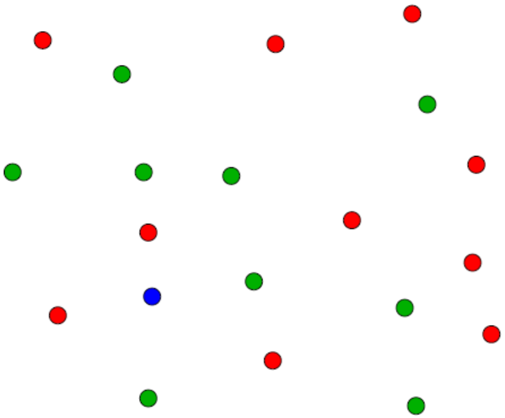


Outline

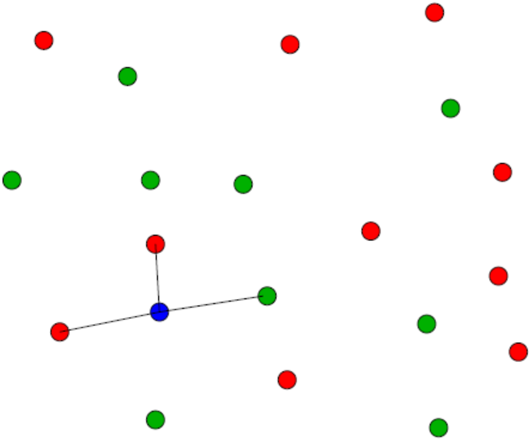
Example: k Nearest-Neighbors



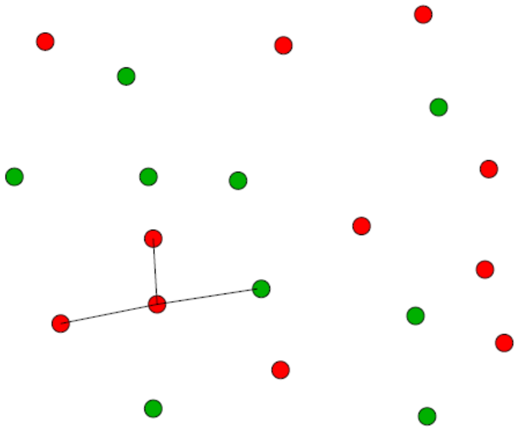
Example: k Nearest-Neighbors



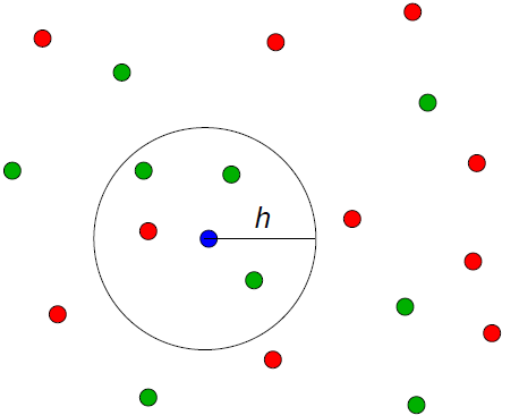
Example: k Nearest-Neighbors



Example: k Nearest-Neighbors



Example: k Nearest-Neighbors



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

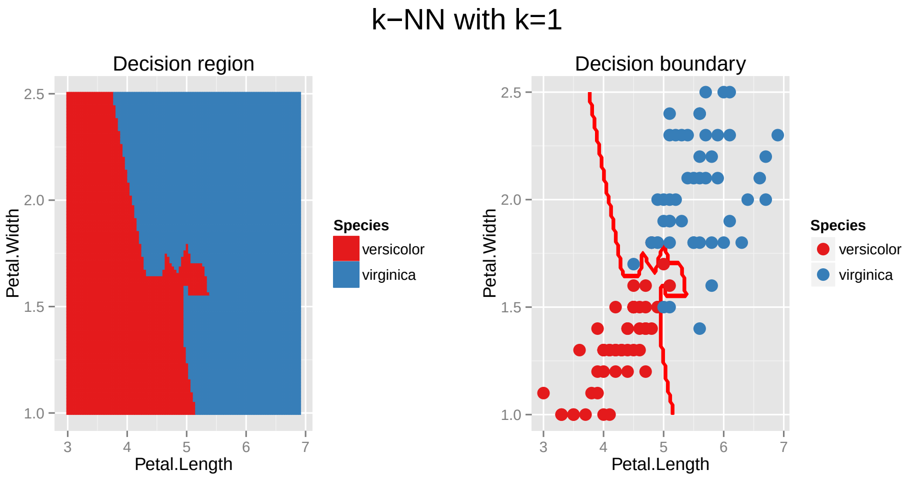
$$\widehat{p}_{+1}(\mathbf{x}) = \frac{\sum_{\mathbf{x}_i \in \mathcal{V}_{\mathbf{x}}} \mathbf{1}_{\{y_i=+1\}}}{|\mathcal{V}_{\mathbf{x}}|}$$

- KNN Classifier:

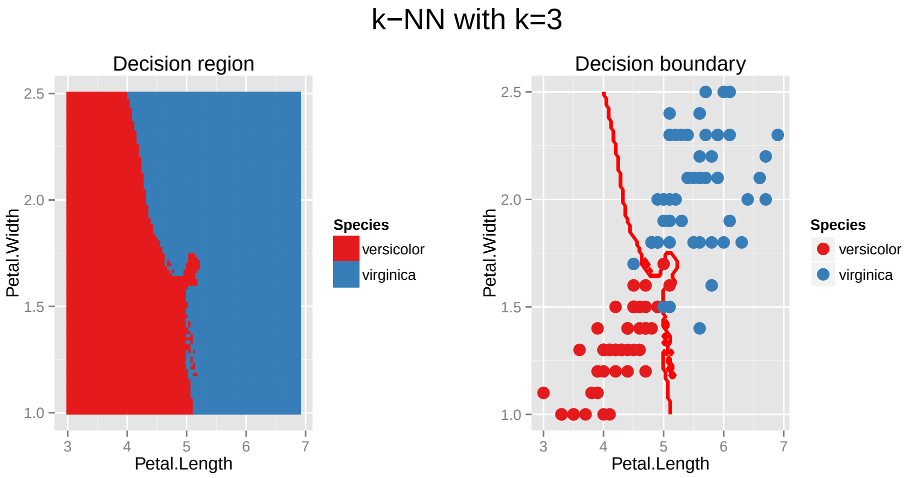
$$\widehat{f}_{KNN}(\mathbf{x}) = \begin{cases} +1 & \text{if } \widehat{p}_{+1}(\mathbf{x}) \geq \widehat{p}_{-1}(\mathbf{x}) \\ -1 & \text{otherwise} \end{cases}$$

- Remark: any kernel density estimate can be used...

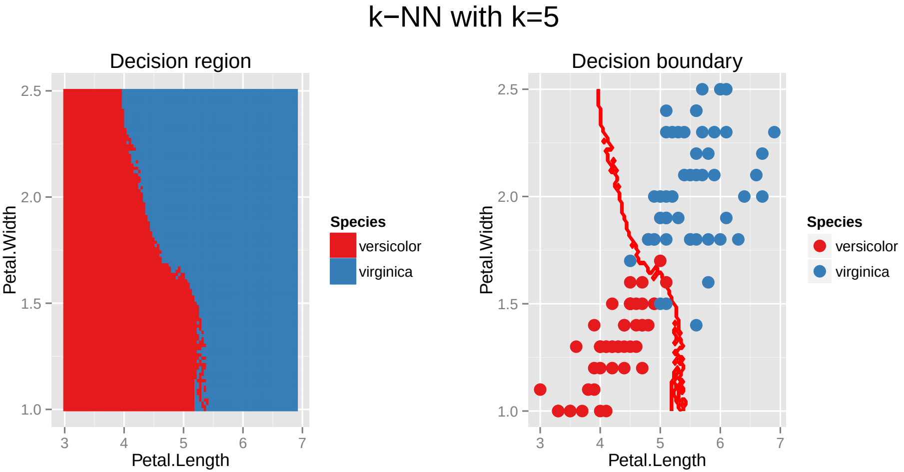
Example: KNN


k−NN with k=1
Decision region
Petal.Width
Petal.Length
Species
versicolor
virginica
Decision boundary
Petal.Width
Petal.Length
Species
versicolor
virginica

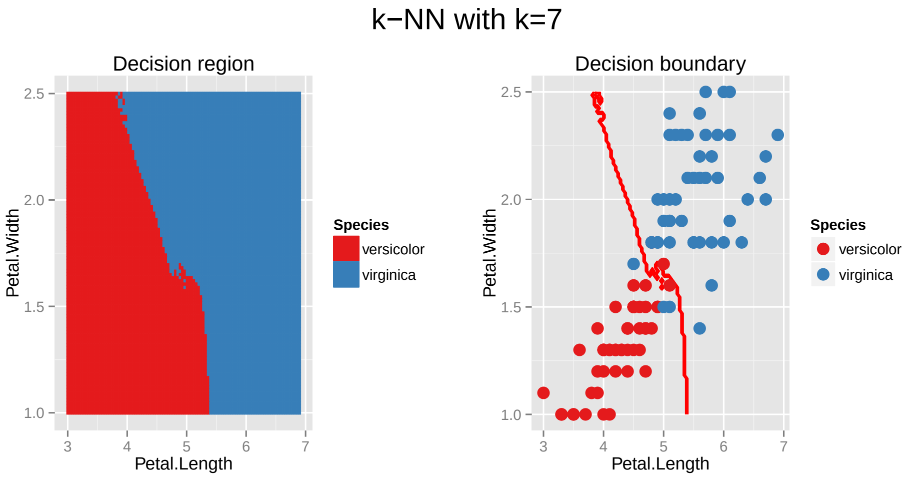
Example: KNN


k−NN with k=3
Decision region
Decision boundary
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
3
4
5
6
7
Species
versicolor
virginica

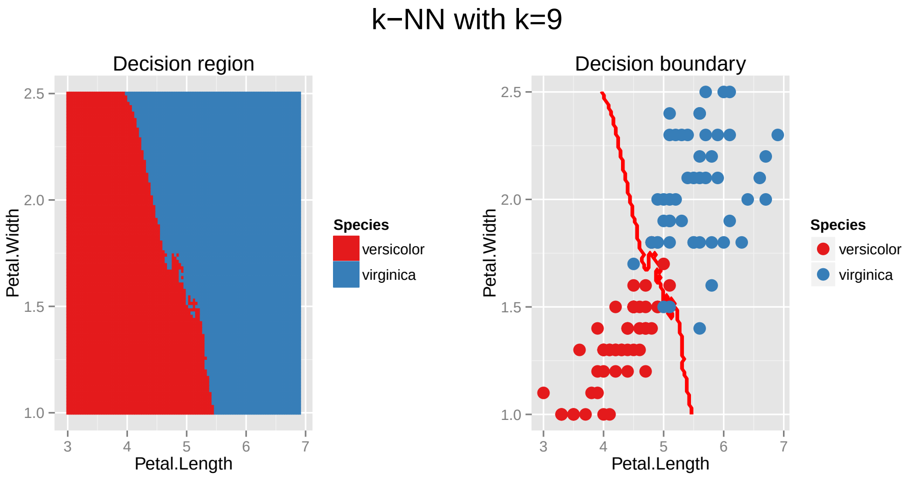
Example: KNN


k−NN with k=5
Decision region
Decision boundary
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
3
4
5
6
7
Species
versicolor
virginica

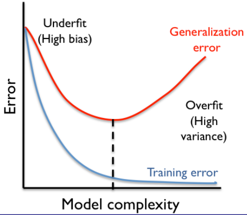
Example: KNN


k−NN with k=7
Decision region
Decision boundary
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
3
4
5
6
7
Species
versicolor
virginica

Example: KNN


k−NN with k=9
Decision region
Decision boundary
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
3
4
5
6
7
Species
versicolor
virginica

Over-fitting Issue

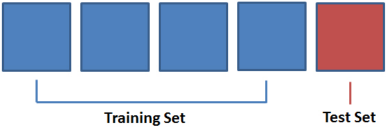


Error behaviour

- 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



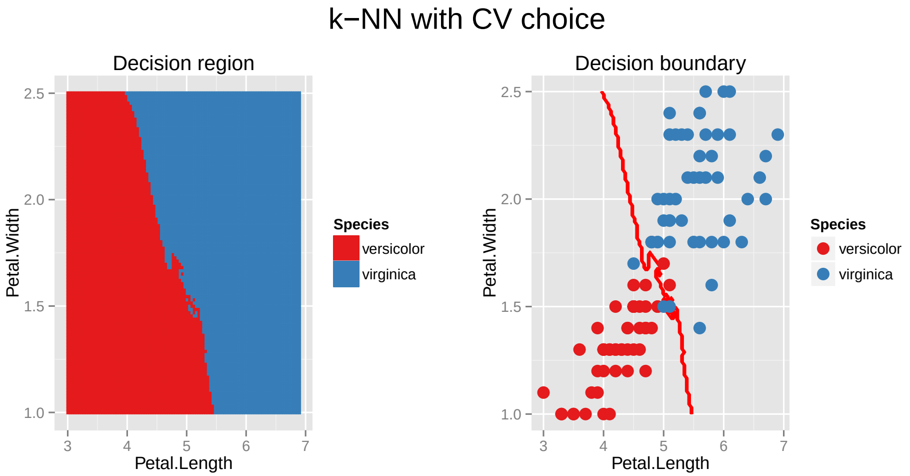
- **Very simple idea:** use a second learning/verification set to compute a verification error.
- Sufficient to avoid over-fitting!

Cross Validation

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

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

Example: KNN ($\hat{k} = 9$ using cross-validation)



Machine Learning in Classification

Statistical and Machine Learning Framework

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

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

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

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

Supervised Statistical Learning (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:
generalized linear models, k-nn, naive Bayes...

Supervised Machine Learning (E. Le Pennec)

Solution: Replace the loss ℓ by an upper bound ℓ' which allows the minimization: SVM, Neural Network, Boosting

Outline

3. A Machine Learner Point of View
   - SVM
   - (Deep) Neural Networks
   - Tree Based Methods

4. Model and Variable Selection
   - Model Selection
   - Practical Variable Selection
   - Empirical Risk Minimization Analysis

5. Big Data

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

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

- Plus convexification/regularization of the risk...
- Examples: SVM, Trees and (Deep) Neural Networks

Logistic Revisited

- Ideal solution:

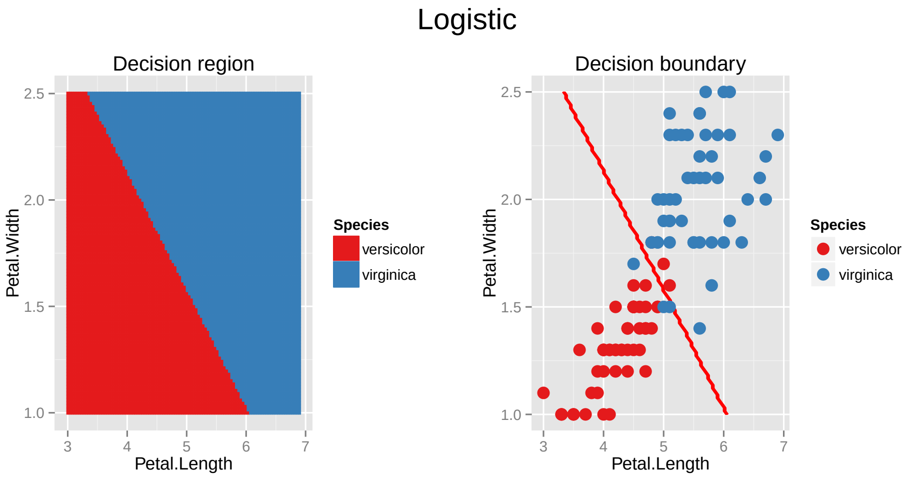
$$\widehat{f} = \operatorname*{argmin}_{f \in \mathcal{S}} \frac{1}{n} \sum_{i=1}^{n} \ell^{0/1}(y_i, f(x_i))$$

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.

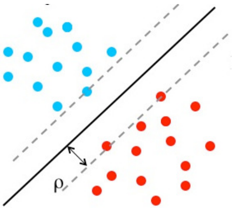
- Different vision than the statistician but same algorithm!

Logistic Revisited



Outline

Ideal Separable Case

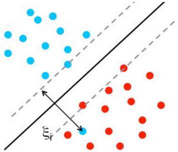


- Linear classifier: $\mathrm{sign}(\langle \beta, x \rangle + b)$
- Separable case: $\exists (\beta, b), \forall i, y_i(\langle \beta, x \rangle + b) > 0!$

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:

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

- 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:

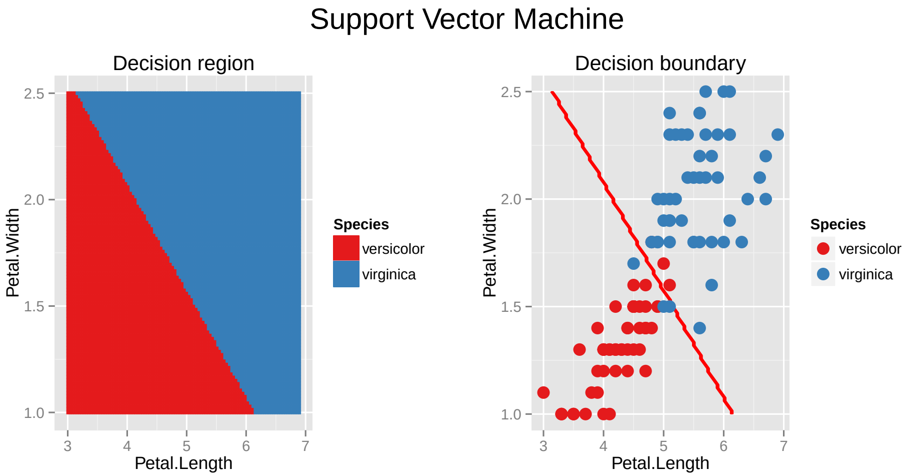
$$\begin{aligned} &\operatorname{argmin} \|\beta\|^2 + C\frac{1}{n}\sum_{i=1}^{n} \max(1 - y_i(\langle \beta, x\rangle + b), 0) \\ &= \operatorname{argmin} \frac{1}{n}\sum_{i=1}^{n} \max(1 - y_i(\langle \beta, x\rangle + b), 0) + \frac{1}{C}\|\beta\|^2 \end{aligned}$$

- **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!)

$$\begin{aligned} &\frac{1}{n}\sum_{i=1}^{n} \ell^{0/1}(y_i, \operatorname{sign}(\langle \beta, x\rangle + b)) \\ &\qquad \leq \frac{1}{n}\sum_{i=1}^{n} \max(1 - y_i(\langle \beta, x\rangle + b), 0) + \frac{1}{C}\|\beta\|^2 \end{aligned}$$

SVM


Support Vector Machine
Decision region
Decision boundary
Petal.Width
Petal.Length
Species
versicolor
virginica
2.5
2.0
1.5
1.0
3
4
5
6
7

Mercer Theorem and Scalar Product

- **Mercer Theorem:** the minimizer in β of

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

  is a linear combination of the input points $\sum_{i=1}^{n}\alpha'_i x_i$.
- **Duality theory:** $\alpha'_i = \alpha_i y_i$ where

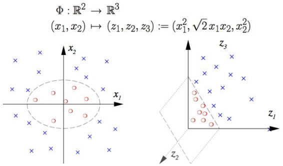
$$\alpha = \arg\max \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_i y_i y_k \langle x_i, x_j\rangle$$

  under the constraints $\sum_{i=1}^{n}\alpha_i y_i = 0$ and $0 \leq \alpha_i \leq C/n$.

Dual formulation

- α_i are Lagrangian multipliers and are equal to 0 as soon as $y_i(\langle\beta,x_i\rangle+b)\geq 1$ + Explicit formula for b.
- Data involved only through scalar product $\langle x,y\rangle$!

The Kernel Trick



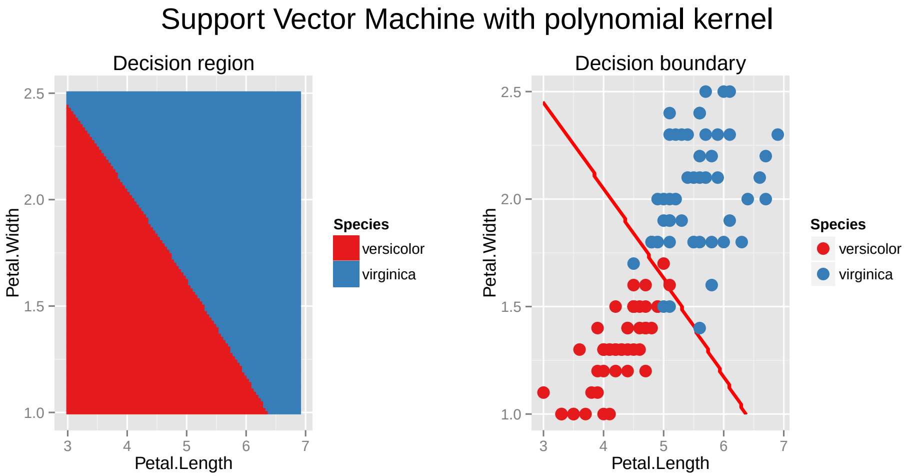
- Non linear separation: just replace x by a non linear $\Phi(x)$...

Kernel trick

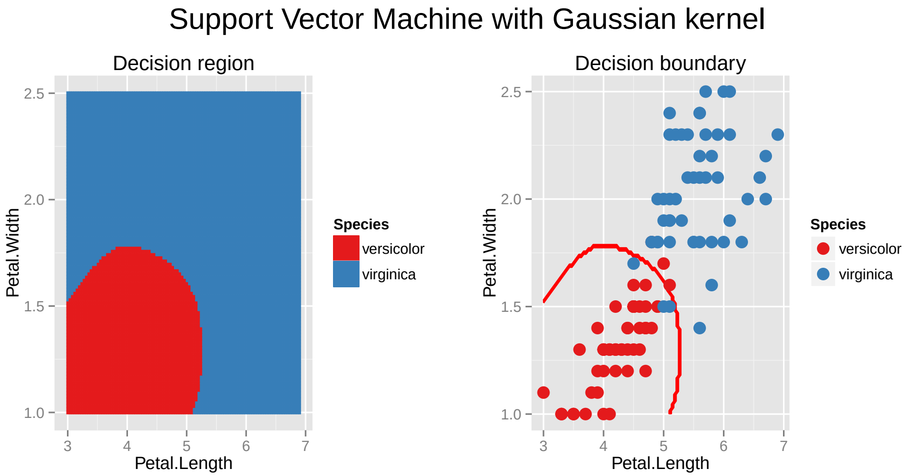
- Computing $k(x, y) = \langle \Phi(x), \Phi(y) \rangle$ may be easier than computing $\Phi(x)$, $\Phi(y)$ and then the scalar product!
- Φ can be specified through its definite positive kernel k.

- 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

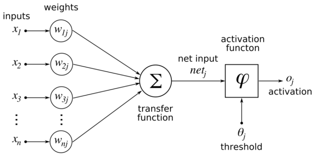

Support Vector Machine with polynomial kernel
Decision region
Decision boundary
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
3
4
5
6
7
Species
versicolor
virginica

SVM


Support Vector Machine with Gaussian kernel
Decision region
Decision boundary
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
3
4
5
6
7
Species
versicolor
virginica

Outline

Artificial Neuron and Logistic Regression



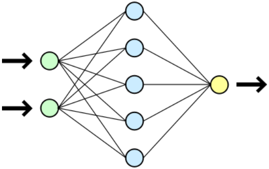
Artificial neuron

- 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

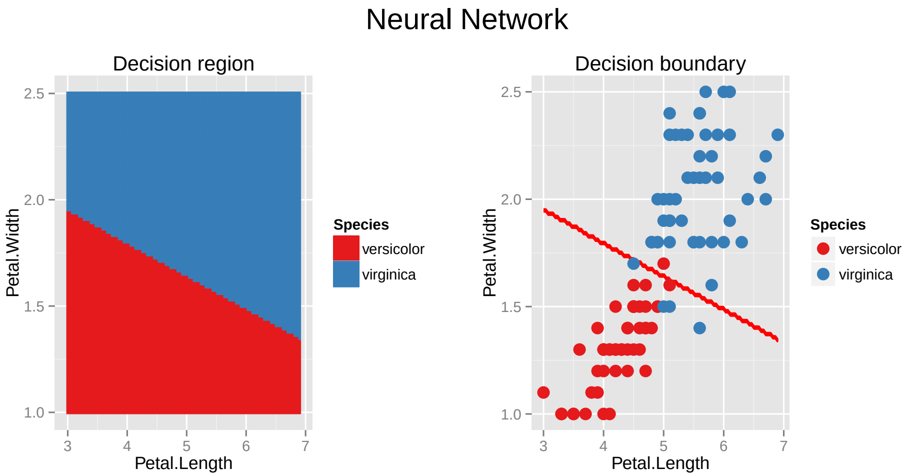


Neural network structure

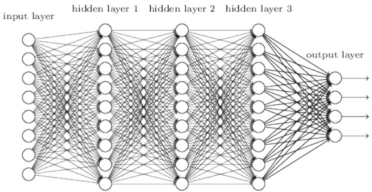
- Cascade of artificial neurons organized in layers
- Thresholding decision only at the output layer

- 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

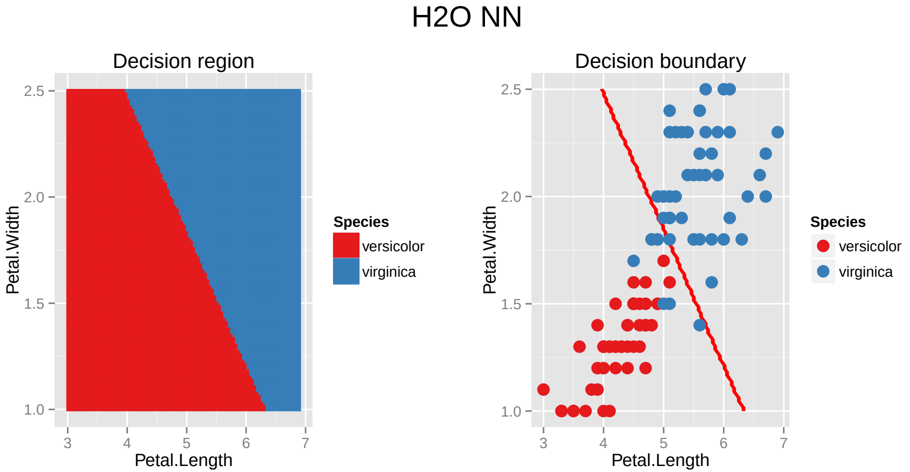


Deep Neural Network structure

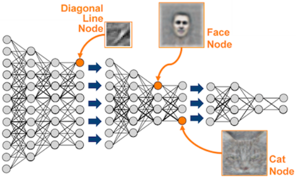
- Deep cascade of layers!

- No conceptual novelty but initialization becomes a crucial issue.
- Bunch of solutions proposed on a greedy initialization of the layers starting from the deepest one.
- Very impressive results!

Deep Neural Network



Deep Learning



Family of Machine Learning algorithm combining:

- a (deep) multilayered structure,
- a clever (often unsupervised) initalization,
- a more classical final fine tuning optimization.

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

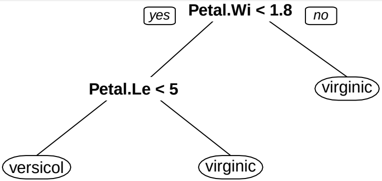
Outline

3. A Machine Learner Point of View
   - SVM
   - (Deep) Neural Networks
   - Tree Based Methods

4. Model and Variable Selection
   - Model Selection
   - Practical Variable Selection
   - Empirical Risk Minimization Analysis

5. Big Data

Classification and Regression Trees

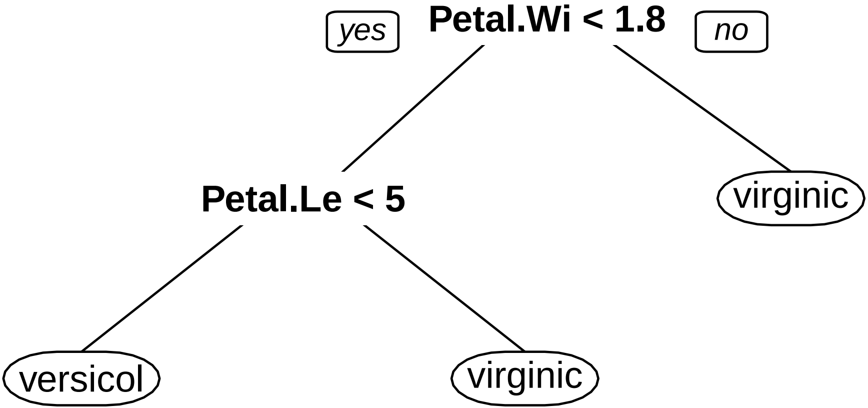


Tree principle

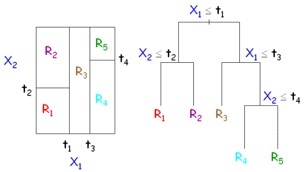
- Construction of a recursive partition through a tree structured set of questions (splits around a given value of avariable),
- Use a simple majority vote in each leaf.

- 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



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 selection within the (rooted) subtrees of the 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

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

Pruning

Examples of criterion satisfying this assumptions

- AIC type criterion:

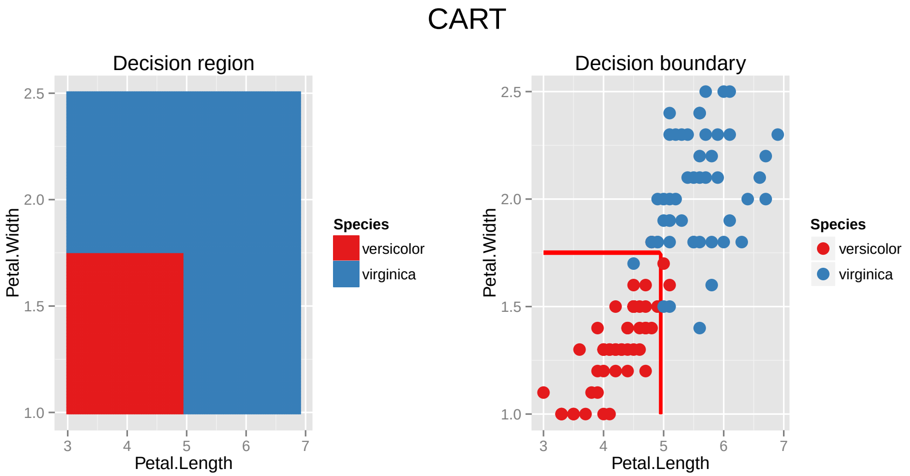
$$\sum_{i=1}^{n} \ell'(y_i, f_{\mathcal{L}(x_i)}(x_i) + \lambda|\mathcal{T}| = \sum_{\mathcal{L}\in\mathcal{T}} \left(\sum_{x_i\in\mathcal{L}} \ell'(y_i, f_{\mathcal{L}}(x_i) + \lambda \right)$$

- Simple cross-Validation (with (x_i', y_i') a different dataset):

$$\sum_{i=1}^{n'} \ell'(y_i', f_{\mathcal{L}}(x_i') = \sum_{\mathcal{L}\in\mathcal{T}} \left(\sum_{x_i'\in\mathcal{L}} \ell'(y_i', f_{\mathcal{L}}(x_i') \right)$$

- Limits over-fitting...

CART


CART
Decision region
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
3
4
5
6
7
Species
versicolor
virginica
Decision boundary
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
3
4
5
6
7
Species
versicolor
virginica

Extensions

Recursive Partitioning methods

- Recursive construction of a partition
- Use of simple local model on each part of the partition

- Examples:
  - CART, ID3, C4.5, C5
  - MARS (local linear regression models)
  - Piecewise polynomial model with a dyadic partition...
- Book: *Recursive Partitioning and Applications* by Zhang and Singer

Stabilization by Independent Average

Very simple idea to obtain a more stable estimator

- Vote/average of B predictors $f_1, \dots, f_B$ 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: $\mathbb{E}\left[f_{\text{agr}}(x)\right] = \mathbb{E}\left[f_b(x)\right]$ and $\mathbb{V}\left[f_{\text{agr}}(x)\right] = \frac{\mathbb{V}[f_b(x)]}{B}$
- Prediction: more complex analysis

- Averaging leads to variance reduction, i.e. stability!
- Issue: cost of obtaining B independent datasets of size n!

Bagging and Bootstrap

Bagging: Bootstrap Aggregation(Breiman)

- Instead of using B independent dataset of size n, draw B datasets from a single one using a uniform with replacement scheme (Bootstrap).
- The f_b are identically distributed but not independent anymore.

- Price for the non independence: $\mathbb{E}\left[f_{\mathrm{agr}}(x)\right] = \mathbb{E}\left[f_b(x)\right]$ and
$$\mathbb{V}\left[f_{\mathrm{agr}}(x)\right] = \frac{\mathbb{V}\left[f_b(x)\right]}{B} + \left(1 - \frac{1}{B}\right)\rho(x)$$
with $\rho(x) = \mathrm{Cov}\left[f_b(x), f_{b'}(x)\right]$ with $b \neq b'$.
- On average, a fraction of $(1 - 1/e) \simeq .63$ examples are unique among each drawn dataset...
- Better aggregation scheme exists...

Randomized Predictors

- Correlation leads to less variance reduction:

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

with $\rho(x) = \text{Cov}\left[f_b(x), f_{b'}(x)\right]$ with $b \neq b'$.

Idea

- Reduce the correlation by adding more randomness in the predictor.

- **Randomized predictors:** construct predictors that depends on a randomness source R that may be chosen independently for all bootstrap samples.
- This reduces the correlation between the estimates...
- But may **modify heavily** the estimates themselves!

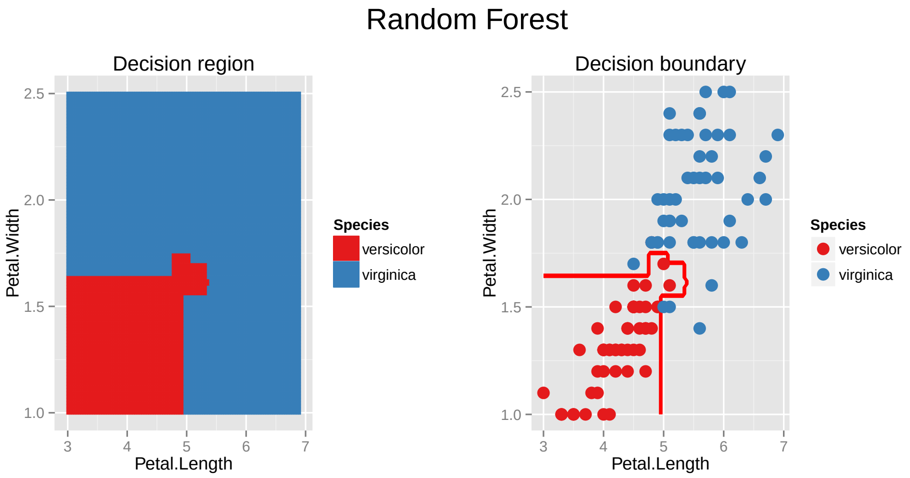
Random Forest

Tree based randomized predictors (Breiman)

- Draw B resampled datasets from a single one using a uniform with replacement scheme (Bootstrap)
- For each resampled datasets, construct a tree using a different randomly drawn subset of variables at each split.

- Most important parameter is the size of this subset:
  - 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 $1/\sqrt{d}$
  - Regression: use a proportion of $1/3$
- Often sloppier stopping rules and pruning...

Random Forest



AdaBoost

- Idea: learn a sequence of predictor trained on weighted dataset with weights depending on the loss so far.

Iterative scheme proposed by Schapire and Freud

- Set $w_1(i) = 1/n$; $t = 0$ and $f = 0$
- For $t = 1$ to $= T$
  - $t = t + 1$
  - $h_t = \text{argmin}_{h \in \mathcal{S}} \sum_{i=1}^{n} w_t(i) \ell^{0/1}(y_i, h(x_i))$
  - Set $\epsilon_t = \sum_{i=1}^{n} w_t(i) \ell^{0/1}(y_i, g(x_i))$ and $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$
  - let $w_i(t+1) = \frac{w_t(i) e^{-\alpha_t z_i h_t(x_i)}}{Z_{t+1}}$ where Z_{t+1} is a renormalization constant such that $\sum_{i=1}^{n} w_i(t+1) = 1$
  - $f = f + \alpha_t h_t$
- Use $f = \sum_{i=1}^{T} \alpha_t h_t$

- Now simple explanation of such a scheme!

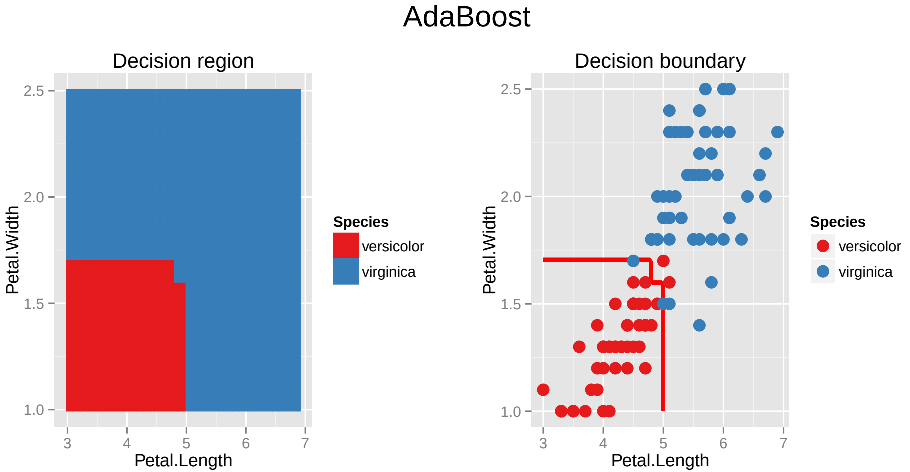
AdaBoost

Exponential Stagewise Additive Modeling

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

- Greedy optimization of a classifier as a linear combination of T classifier for the exponential loss.
- Those two algorithms are **equivalent!**
- Iterative scheme with only two parameters: the class $\mathcal{S}$ of *weak* classifier and the number of step T.
- In the literature, one can read that Adaboost does not overfit! This not true and T should be chosen with care...

AdaBoost


AdaBoost
Decision region
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
3
4
5
6
7
Species
versicolor
virginica
Decision boundary
Petal.Width
Petal.Length
2.5
2.0
1.5
1.0
3
4
5
6
7
Species
versicolor
virginica

Boosting

General greedy optimization strategy to combine *weak* predictors

- Set $t = 0$ and $f = 0$.
- For $t = 1$ to T,
  - $(h_t, \alpha_t) = \operatorname{argmin}_{h,\alpha} \sum_{i=1}^{n} \ell'(y_i, f(x_i) + \alpha h(x_i))$
  - $f = f + \alpha_t h_t$
- Use $f = \sum_{t=1}^{T} \alpha_t h_t$

- Forward Stagewise Additive Modeling:
  - AdaBoost with $\ell'(y,h) = e^{-yh}$
  - LogitBoost with $\ell'(y,h) = \log(1 + e^{-yh})$
  - L_2Boost with $\ell'(y,h) = (y-h)^2$ (Matching pursuit)
  - L_1Boost with $\ell'(y,h) = |y-h|$
  - HuberBoost with $\ell'(y,h) = |y-h|^2 \mathbf{1}_{|y-h|<\epsilon} + (2\epsilon|y-h| - \epsilon^2)\mathbf{1}_{|y-h|\geq\epsilon}$
- Simple principle but no easy numerical scheme except for AdaBoost and L_2Boost...

Gradient Boosting

- At each boosting step, one need to solve

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

- Gradient approximation $L(y, f + \alpha h) \sim L(y, f) + \alpha \langle \nabla f, h \rangle$.

Gradient boosting

Replace the minimization step by a *gradient descent* type step:

- Choose h_t as the best possible descent direction in $\mathcal{S}$
- Choose α_t that minimizes $L(y, f + \alpha h_t)$ (line search)

- Easy if finding the best descent direction is easy!
- Numerical scheme based on either explicit solution (classifier) or LS.

SVM

- Ideal solution:

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

SVM

- Replace $\ell(y, f) = \mathbf{1}_{y=f}$ by $\ell(y, f) = (1 - yf)_+$.
- Add a penalty $\lambda \|f\|_{\mathcal{S}}^2$

- Example:
  - $f(x) = \langle \beta, x \rangle$ and $\|f\|_{\mathcal{S}}^2$
  - $f(x) = \sum_{i=1}^{n} \alpha_i K(x, x_i)$ with $\|f\|_{\mathcal{S}}^2 = \alpha^t K \alpha$ (Kernel trick)...

(Deep) Neural Networks

- Ideal solution:

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

NN

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

- Canonical (logistic) example:

$$\sigma(x) = e^x/(1 + e^x) \quad \text{and} \quad \ell(y, f) = -y \log f - (1 - y) \log(1 - f)$$

- Deep Neural Networks: good initialization strategy.

Tree and Boosting

- Ideal solution:

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

Single tree

- Minimization of the loss / Conditional law estimation
- 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 $\ell(y, f) = e^{-yf}$ (machine learner point of view?)

Outline

Outline

Model Selection

Models

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

- Key to obtain good performance!

Approximation error and estimation error (Bias/Variance)

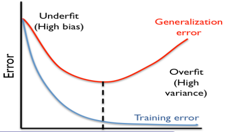
$$\mathcal{R}(\widehat{f}_{\mathcal{S}}) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(f^*_{\mathcal{S}}) - \mathcal{R}(f^*)}_{\text{Approximation error}} + \underbrace{\mathcal{R}(\widehat{f}_{\mathcal{S}}) - \mathcal{R}(f^*_{\mathcal{S}})}_{\text{Estimation error}}$$

- 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

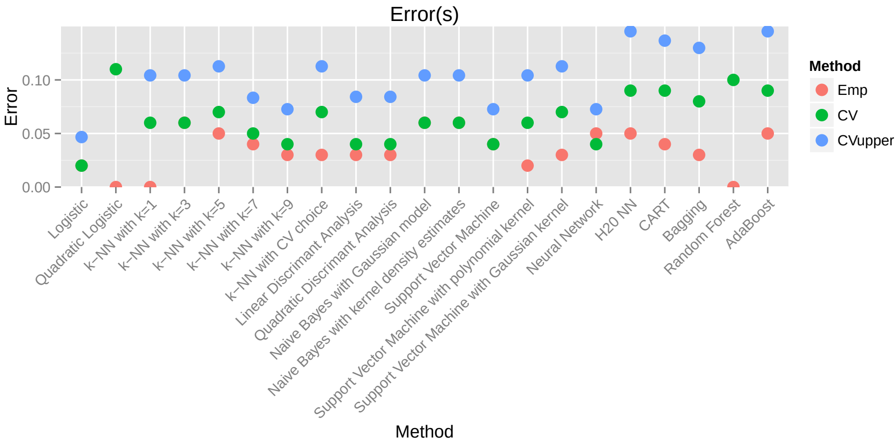
- **Cross validation:** Very efficient (and almost always used in practice!) but slightly biased as it target uses only a fraction of the data.
- **Penalization approach:** use empirical loss criterion but penalize it by a term increasing with the complexity of $\mathcal{S}$

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

and choose the model with the smallest penalized risk.

- Model mixing also possible...

Cross Validation



Penalized Maximum Likelihood Estimate

Penalized Maximum Likelihood

$$\widehat{\mathcal{S}} = \operatorname*{argmin}_{\mathcal{S}} \min_{f \in \mathcal{S}} \frac{1}{n} \sum_{i=1}^{n} -\log \mathbb{P}_f(y_i|x_i) + \text{pen}(\mathcal{S})$$

- AIC (An Information Criterion/Akaike Information Criterion):
  - *Wilks theorem*: if the true law belongs to $\mathcal{S}$
  $$\frac{1}{n}\sum_{i=1}^{n} \ell'(Y_i, \widehat{f}(x_i)) \rightarrow \frac{1}{n}\sum_{i=1}^{n} \ell'(Yi, \widetilde{f}(x_i)) + \frac{D_{\mathcal{S}}}{n}$$
- BIC (Bayesian Information Criterion):
  - Asymptotic approximation of Bayesian modeling:
  $$-\log \mathbb{P}\left\{\mathcal{S}|(x_i, y_i)\right\} \sim -\log \mathbb{P}\left\{y_i|x_i, \mathcal{S}\right\} + \frac{\log n}{2} D_{\mathcal{S}}$$
- MDL (Minimum Descrition Length):
  - Information-Theoretic approach: $\text{pen}(\mathcal{S})$ = length of code required to specify $f \in \mathcal{S}$ with enough precision ($\sim \frac{\log n}{2} D_{\mathcal{S}}$)
- Generally $\text{pen}(\mathcal{S}) \sim \lambda D_{\mathcal{S}}$!

Complexity Theory

Typical PAC type result

- With probability larger than $1 - \eta$

$$R(\widehat{f}_{\mathcal{S}}) \leq R_n(\widehat{f}_{\mathcal{S}}) + \sqrt{\frac{\epsilon(n, \eta, \mathcal{S})}{n}}$$

- Use then $\text{pen}(\mathcal{S}) = \sqrt{\epsilon(n, \eta, \mathcal{S})/n}$ to obtain an upper bound of the risk!

- Example:
  - Vapnik-Chervonenkis theorem: with prob. larger than $1 - \eta$

$$R(\widehat{f}_{\mathcal{S}}) \leq R_n(\widehat{f}_{\mathcal{S}}) + \sqrt{\frac{h_{\mathcal{S}}(\log(2n/h_{\mathcal{S}}) + 1) - \log(\eta/4)}{n}}$$

    where $h_{\mathcal{S}}$ is the VC dimension of $\mathcal{S}$ (maximum number of points that can be shattered by $f \in \mathcal{S}$)
  - Similar results with different definition of the dimension...

Model Collection Complexity

- Upper bound of the risk of type: with probability larger than $1-\eta$, for a single model $\mathcal{S}$

$$R(\widehat{f}_{\mathcal{S}}) \leq R_n(\widehat{f}_{\mathcal{S}}) + \sqrt{\frac{\epsilon(n,\eta,\mathcal{S})}{n}}$$

- Selection requires a simultaneous control over all models!

Union bounds type control

- With probability $1-\sum_{\mathcal{S}} \eta_{\mathcal{S}}$, $\forall$ model $\mathcal{S}$

$$R(\widehat{f}_{\mathcal{S}}) \leq R_n(\widehat{f}_{\mathcal{S}}) + \sqrt{\frac{\epsilon(n,\eta_{\mathcal{S}},\mathcal{S})}{n}}$$

- Larger penalty required for complex model collections!
- Visible in MDL approach as a cost to specify the model...

Outline

General Setting

- Prediction for $x \in \mathbb{R}^d$
- All the coordinates of x may not be useful!

Variable Selection

- How to choose as a subset of indices / a subset of variables in a given statistical model?

- Curse of dimensionality: number of possible subsets 2^d!
- Even worse as in practice $\Phi(x)$ is often used instead of x!
- **Remark:** Competition between different statistical models only possible by exhaustive exploration...

Exhaustive Exploration

- Brute force approach!

Strategy

- Exhaustive exploration of all subsets
- Computation of a criterion for all subsets (CV,AIC,...)
- Choice of the model minimizing the criterion

- Only possible when d is small.

Clever Exploration

- Minimization of a criterion but without an exhaustive exploration of the subsets.

Generic strategy

- Start with a pool of subsets of size P
- Create a larger pool of size PC by adding and/or removing variables from the previous subset
- Keep only the best P subset according to the criterion and iterate

- Variations on the size of the subsets, the initial subsets, the rule to add and remove variables, the criterion...

Clever Exploration

Forward strategy

- Start with an empty model
- At each step, create a larger collection by creating models equal to the current one plus any variable not used in the current model (one at a time)
- Modify the current model if the best model within the new collection leads to a reduction of the criterion.

Backward strategy

- Start with the full model.
- At each step, create a larger collection by creating models equal to the current one minus any variable used in the current model (one at a time)
- Modify the current model if the best model within the new collection leads to a reduction of the criterion.

Clever Exploration

Forward/Backward strategy

- Start with the full model.
- At each step, create a larger collection by creating models equal to the current one plus any variable not used in the current model (one at a time) and to the current one minus any variable used in the current model (one at a time)
- Modify the current model if the best model within the new collection leads to a reduction of the criterion.

- Various Stochastic (Genetic) Algorithm...
- Stability issue...

Linear Model and (Convex) Penalty

- In (generalized) linear model, prediction depends only on $x^t\beta$ with $\beta \in \mathbb{R}^d$.

Penalization on β

- Subset selection $\Leftrightarrow$ Support selection for β!
- Combine the empirical loss minimization with a (sparsity promoting) penalty:

$$\frac{1}{n}\sum_{i=1}^{n} \ell'(y_i, f(x^t\beta)) + \text{pen}(\beta)$$

- Penalty choices
  - AIC: $\text{pen}(\beta) = \lambda\|\beta\|_0$ (non convex / sparsity)
  - Ridge: $\text{pen}(\beta) = \lambda\|\beta\|_2^2$ (convex / no sparsity)
  - Lasso: $\text{pen}(\beta) = \lambda\|\beta\|_1$ (convex / sparsity)
  - Elastic net: $\text{pen}(\beta) = \lambda_1\|\beta\|_1 + \lambda_2\|\beta\|_2^2$ (convex / sparsity)
- Efficient algorithm as soon as ℓ' and pen are convex.

Variable Filtering

- Heuristic screening of the variables used when there is a lot of variables.

Two different strategies to associate a importance factor to a variable

- Independent criterion for each feature
- Criterion obtained by combining several variable selections on (smaller) variable subsets

- Filtering: Removing the variables whose criterion is small

Variable Filtering

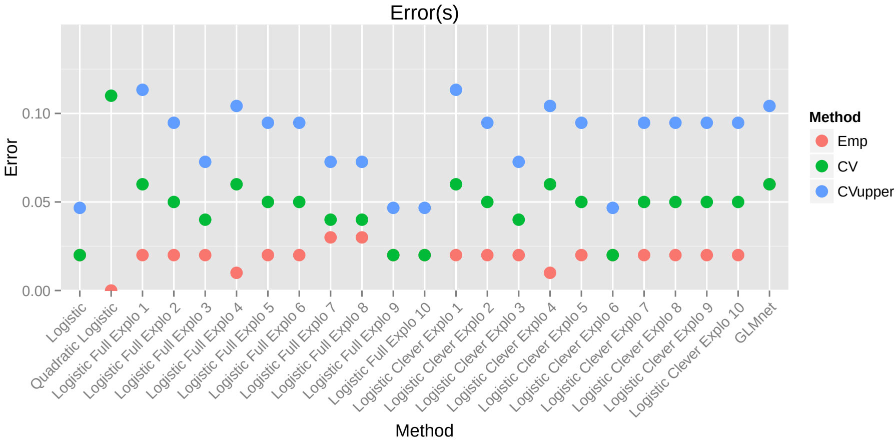
Independent criterions

- Correlation of $X^{(i)}$ with Y (continuous/continuous)
- Information Gain based on entropy criterion $H(X^{(i)}) + H(Y) - H(X^{(i)}, Y)$ (continuous or discrete/continuous or discrete)
- χ^2-test of independence between $X^{(i)}$ and Y (discrete/discrete)
- ...

Variable filtering based on variable selection

- Penalty based exploration
- Random forest
- ...

Cross Validation



Outline

Empirical Risk Minimization and Concentration

- Let the risk be $R(f) = \mathbb{E}\left[\ell(Y, f(X))\right]$ and its empirical counterpart $R_n = \sum_{i=1}^n \ell(y_i, f(x_i))$.
- Let $\widetilde{f} = \operatorname{argmin}_{f \in \mathcal{S}} R(f)$ and $\widehat{f} = \operatorname{argmin}_{f \in \mathcal{S}} R_n(f)$ (Empirical Risk Minimization).
- If $\forall f \in \mathcal{S}, R(f) - R_n(f) \leq \epsilon$ and $R_n(\widetilde{f}) - R(\widetilde{f}) \leq \epsilon$ then

$$\begin{aligned} R(\widehat{f}) &\leq R_n(\widehat{f}) + \epsilon \\ &\leq R_n(\widetilde{f}) + \epsilon \\ &\leq R(\widetilde{f}) + 2\epsilon \end{aligned}$$

  and the ERM is optimal up to 2ϵ.
- Two different bounds in one:
  - $R_n(\widehat{f}) + \epsilon$ is a data driven upper bound of the risk (Penalization type)
  - $R_n(\widetilde{f}) + 2\epsilon$ is a oracle type upper bound of the risk.

Empirical Risk Minimization and Concentration

- If $\ell = \ell^{0/1}$ then we can easily prove (Hoeffding) that for any $f \in \mathcal{S}$

$$\mathbb{P}\left\{R(f) - R_n(f) \leq \epsilon\right\} \geq 1 - e^{-2n\epsilon^2}$$
$$\mathbb{P}\left\{R_n(f) - R(f) \leq \epsilon\right\} \geq 1 - e^{-2n\epsilon^2}$$

- Union bound technique for finite set $\mathcal{S}$:

$$\begin{aligned}
&\mathbb{P}\left\{\forall f \in \mathcal{S}, R(f) - R_n(f) \leq \epsilon\right\} \\
&\quad = 1 - \mathbb{P}\left\{\exists f \in \mathcal{S}, R(f) - R_n(f) \geq \epsilon\right\} \\
&\quad \geq 1 - \sum_{f \in \mathcal{S}} \mathbb{P}\left\{R(f) - R_n(f) \geq \epsilon\right\} \\
&\quad \geq 1 - |\mathcal{S}| e^{-2n\epsilon^2}
\end{aligned}$$

Empirical Risk Minimization and Concentration

- If we let $\epsilon = \sqrt{\frac{\log|\mathcal{S}|+\log(1/\delta)}{2n}}$, we deduced (with a trick) that with a probability greater than $1-2\delta$,

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

- We also have

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

Empirical Risk Minimization and Concentration

and with the non optimal choice $\delta = 1/\sqrt{n}$

$$\mathbb{E}\left[R(\widehat{f})\right] \leq R(\widetilde{f}) + 2\sqrt{\frac{\log|\mathcal{S}| + \frac{1}{2}\log n}{2n}} + \sqrt{\frac{1}{n}}$$

Empirical Risk Minimization and Concentration

- If $\mathcal{S}$ is not finite then if $\mathcal{S}(\eta)$ is a finite subset such that

$$\forall f \in \mathcal{S}, \exists f' \in \mathcal{S}(\eta), |R(f) - R(f')| \leq \eta \text{ and } R_n(f') \leq R_n(f) + \eta$$

then, with a control on $\mathcal{S}(\eta)$, with probability $1 - \eta$

$$\begin{aligned} R(\widehat{f}) &\leq R(\widehat{f}') + \eta \leq R_n(\widehat{f}') + \epsilon(\eta) + \eta \\ &\leq \min_{f' \in \mathcal{S}(\eta)} R_n(f') + \epsilon(\eta) + 2\eta \\ &\leq \min_{f' \in \mathcal{S}(\eta)} R(f') + 2\epsilon(\eta) + 2\eta \\ &\leq R(\widetilde{f}) + 2\epsilon(\eta) + 3\eta \end{aligned}$$

and along the same line

$$R(\widehat{f}) \leq R_n(\widehat{f}) + \epsilon(\eta) + 3\eta$$

Empirical Risk Minimization and Concentration

where $\epsilon(\eta) = \sqrt{\frac{\log|\mathcal{S}(\eta)|+\log(1/\eta)}{2n}}$

- In a usual parametric setting, $\log|\mathcal{S}(\eta)| \leq C + D_{\mathcal{S}} \log(1/\eta)$ so that

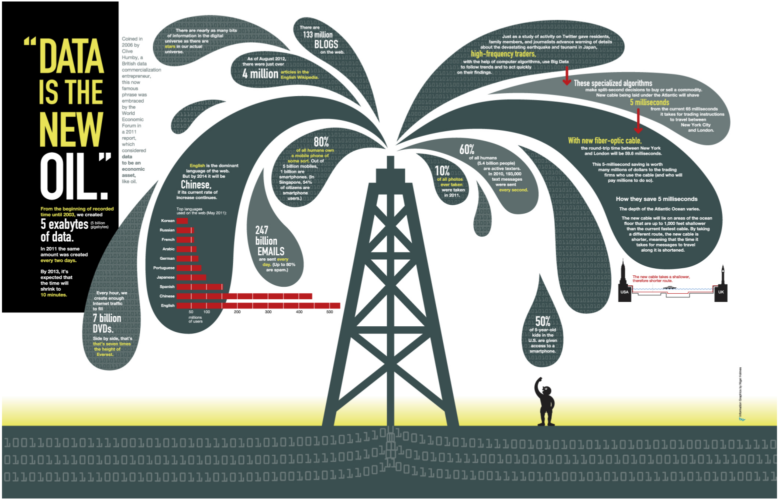
$$\min_{\eta} 2\epsilon(\eta) + 3\eta \leq \min_{\eta} 2\sqrt{\frac{C + D_{\mathcal{S}} \log(1/\eta) + \log(1/\eta)}{2n}} + \eta$$

and using the non optimal choice $\eta = \sqrt{\frac{\dim_{\mathcal{S}}}{2n}}$

$$\begin{aligned}\min_{\eta} 2\epsilon(\eta) + 3\eta &\leq 2\sqrt{\frac{C + \frac{1}{2} D_{\mathcal{S}} \log(2n/D_{\mathcal{S}}) + \log(1/\eta)}{2n}} + 3\sqrt{\frac{D_{\mathcal{S}}}{2n}} \\ &\leq 2\sqrt{\frac{C + D(\mathcal{S})(9/4 + \frac{1}{2}\log(2n/D_{\mathcal{S}})) + \log(1/\eta)}{2n}}\end{aligned}$$

Outline

Data is the new Oil!



Lots of Words!

Doing Data Science

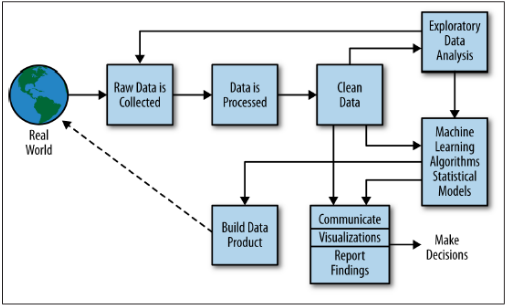
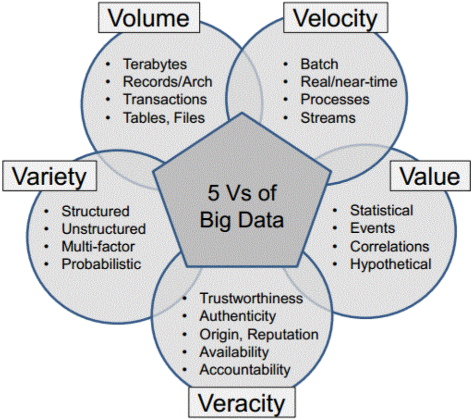


Figure 2-2. The data science process

Doing Data Science: Straight talk from the frontline

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

The 5 Vs of Big Data



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

```
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

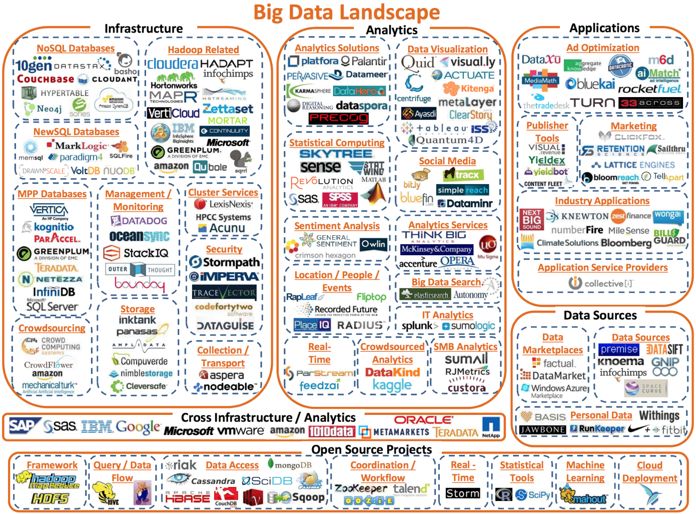
Example of *off the shelves* solution

```
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!



Matt Turck (@mattturck) and Shivon Zilis (@shivonz)

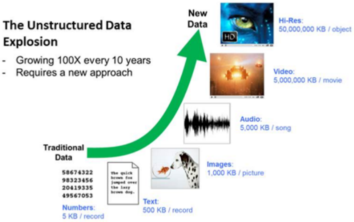
New Interdisciplinary Challenges

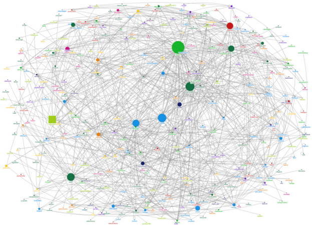
- Applied math **AND** Computer science
- Strong link with domain specific applications: marketing, signal processing, genomic, biology, health...

Some joint math/computer science challenges

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

Unstructured Data

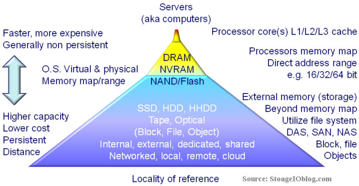


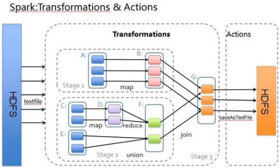


Some challenges

- How to store efficiently the data?
- How to describe them to be able to process them?
- How to combine data of different nature?

Huge Dataset

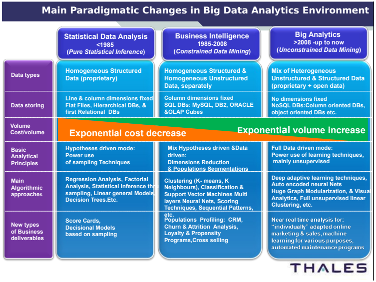




Some challenges

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

High Dimensional Data



Main Paradigmatic Changes in Big Data Analytics Environment

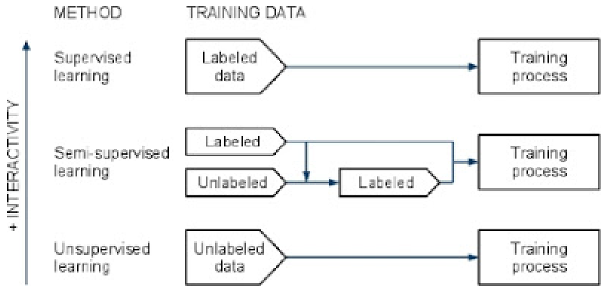
	Statistical Data Analysis <1985 (*Pure Statistical Inference*)	Business Intelligence 1985-2008 (*Constrained Data Mining*)	Big Analytics >2008 -up to now (*Unconstrained Data Mining*)
Data types	Homogeneous Structured Data (proprietary)	Homogeneous Structured & Homogeneous Unstructured Data, separately	Mix of Heterogeneous Unstructured & Structured Data (proprietary + open data)
Data storing	Line & column dimensions fixed Flat Files, Hierarchical DBs, & first Relational DBs	Column dimensions fixed SQL DBs: MySQL, DB2, ORACLE &OLAP Cubes	No dimensions fixed NoSQL DBs:Column oriented DBs, object oriented DBs etc.
Volume Cost/volume	Exponential cost decrease		Exponential volume increase
Basic Analytical Principles	Hypotheses driven mode: Power use of sampling Techniques	Mix Hypotheses driven &Data driven: Dimensions Reduction & Populations Segmentations	Full Data driven mode: Power use of learning techniques, mainly unsupervised
Main Algorithmic approaches	Regression Analysis, Factorial Analysis, Statistical Inference thr sampling, Linear general Models, Decision Trees.Etc.	Clustering (K- means, K Neighbours), Classification & Support Vector Machines Multi layers Neural Nets, Scoring Techniques, Sequential Patterns, etc.	Deep adaptive learning techniques, Auto encoded neural Nets Huge Graph Modularization, & Visua Analytics, Full unsupervised linear Clustering, etc.
New types of Business deliverables	Score Cards, Decisional Models based on sampling	Populations Profiling: CRM, Churn & Attrition Analysis, Loyalty & Propensity Programs,Cross selling	Near real time analysis for: "individually" adapted online marketing & sales, machine learning for various purposes, automated maintenance programs

THALES

Some challenges

- How to describe the data?
- How to reduce the data dimensionality?
- How to select models?

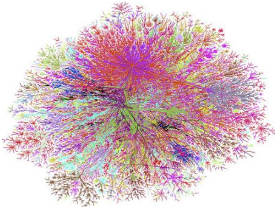
Learning and Supervision

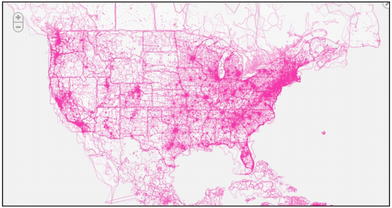


Some challenges

- 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?

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