Statistical and Optimization Approaches in Classification

A. Fermin - MODAL'X, Université Paris Ouest
E. Le Pennec - CMAP, École polytechnique

RTE - 30/06/2015

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

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

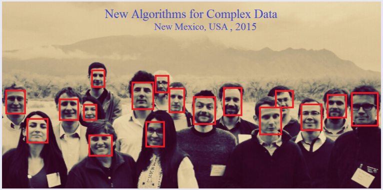
Marketing: advertisement, recommendation...

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

Spam detection (Text classification)

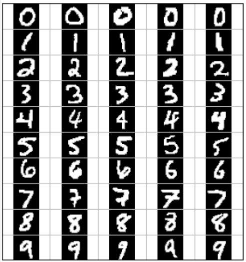
- Data: email collection
- Input: email
- Output : Spam or No Spam

Face Detection

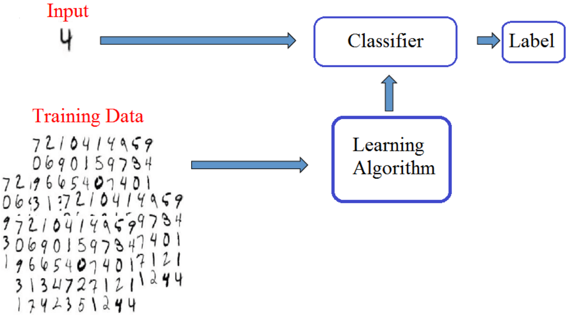


- Data: Annotated database of images
- Input : Sub window in the image
- Output : Presence or no of a face...

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



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.

Big Data, Data Science and Machine Learning

- **Big Data**: buzzword to raise money (or data sets too large or too complex to be handled by the current system)
- **Data Science**: art (or science) of the generalizable extraction of knowledge from data.
- **Machine Learning**: construction and study of algorithms that can learn from and make predictions on data.

- Exciting challenges in the industrial **and** the academic worlds.

Machine Learning

- Fundamental ingredient in data science.
- Necessity for a Data Scientist to understand the principle of the simplest methods to grasp the more sophisticated ones.

Supervised Learning Framework

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

- A classifier is a function in $\mathcal{F} = \{f : \mathcal{X} \to \mathcal{Y} \text{ measurable}\}$

Goal

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

- Need to specify the meaning of good.
- Formally, classification and regression are the same problem!

Loss function

- Loss function : $\ell(f(x), y)$ measure how well $f(x)$ "predicts" y.
- Examples:
  - Prediction loss: $\ell(Y, f(\mathbf{X})) = \mathbf{1}_{Y \neq f(\mathbf{X})}$
  - Quadratic loss: $\ell(Y, \mathbf{X}) = |Y - f(\mathbf{X})|^2$

Risk of a generic classifier

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

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

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

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

Experience, Task and Performance measure

- Training data : $\mathcal{D} = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$ (i.i.d. $\sim \mathbf{P}$)
- Predictor: $f : \mathcal{X} \to \mathcal{Y}$ measurable
- Cost/Loss function : $\ell(f(\mathbf{X}), Y)$ measure how well $f(\mathbf{X})$ "predicts" Y
- Risk:

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

- Often $\ell(f(\mathbf{X}), Y) = |f(\mathbf{X}) - Y|^2$ or $\ell(f(\mathbf{X}), Y) = \mathbf{1}_{Y \neq f(\mathbf{X})}$

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 $\mathcal{D}_n$.

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

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

Bayes Classifier (explicit solution)

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

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

- In regression with the quadratic loss

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

Issue: Explicit solution requires to know $\mathbb{E}\left[Y|\mathbf{X}\right]$ for all values of $\mathbf{X}$!

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

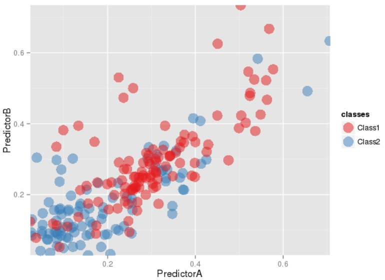
$$\widehat{f} = f_{\widehat{\theta}} = \operatorname*{argmin}_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(\mathbf{X}_i))$$

- Examples:
  - Linear regression
  - Linear discrimination with
  $$\mathcal{S} = \{\mathbf{x} \mapsto \mathtt{sign}\{\beta^T \mathbf{x} + \beta_0\} \,/ \beta \in \mathbb{R}^d, \beta_0 \in \mathbb{R}\}$$

Synthetic Dataset

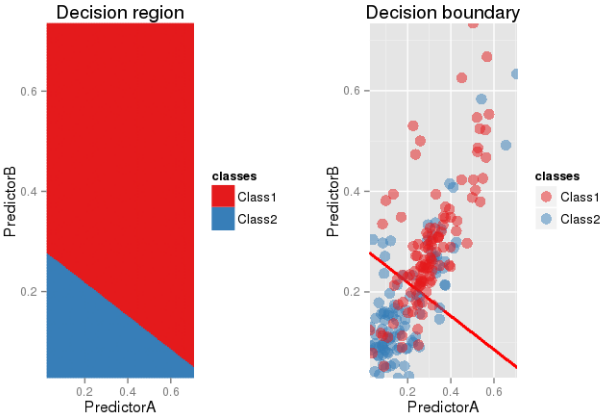
- Two features/covariates.
- Two classes.

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

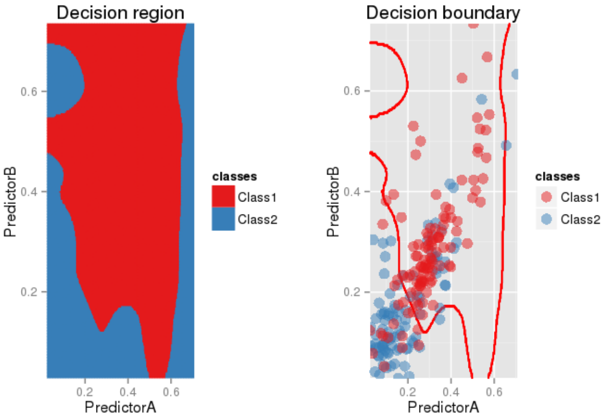


Supervised Learning

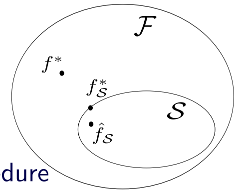
Example: Linear Discrimination



Supervised Learning
Example: More complex model



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



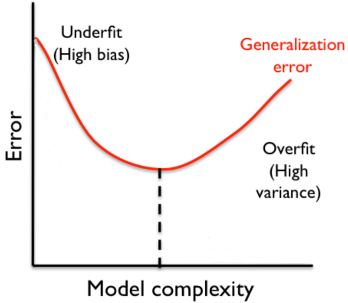
Approximation error and estimation error (Bias/Variance)

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

- 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 $(\mathbf{X}, Y)$.



- 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 $\Longleftrightarrow$ avoid overfitting and underfitting

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

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

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

Problems

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

A Statistical Point of View

Solution: For $\mathbf{X}$, estimate $Y|\mathbf{X}$ plug this estimate in the Bayes classifier: (Generalized) Linear Models, Kernel methods, k-nn, Naive Bayes, Tree, Bagging...

An Optimization Point of View

Solution: If necessary replace the loss ℓ by an upper bound ℓ' and minimize the empirical loss: SVR, SVM, Neural Network,Tree, Boosting

- 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. Generative Modeling (Bayes method)
  2. Logistic modeling (parametric method)
  3. Nearest neighbors (kernel method)

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, Y) or equivalently of X given y and 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...

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:
  $g_k(\mathbf{x}) = \ln(\mathbb{P}\{X|Y=k\}) + \ln(\mathbb{P}\left\{Y=k\right\})$

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

- 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!

Estimation

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

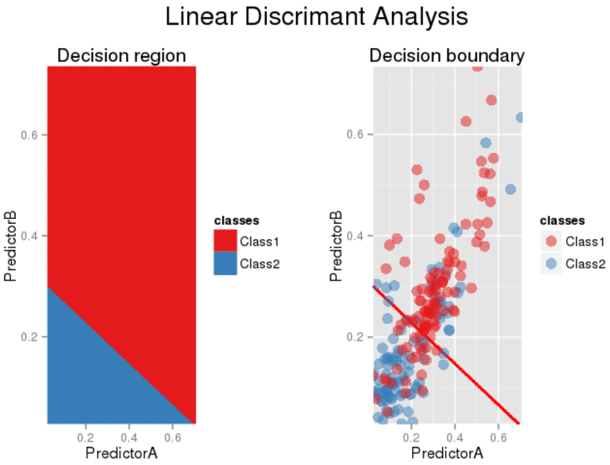
- The estimate proportion $\widehat{\mathbb{P}_k} = \frac{n_k}{n} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{\{Y_i=k\}}$
- Maximum likelihood estimate of $\widehat{\mu_k}$ and $\widehat{\Sigma_k}$ (explicit formulas)

- DA classifier

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

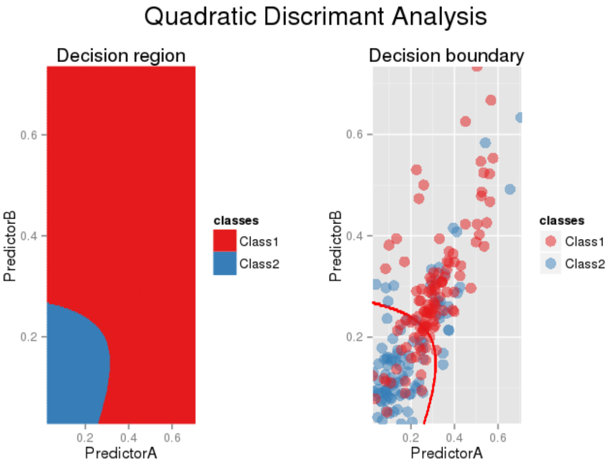
- Decision boundaries: quadratic = degree 2 polynomials.
- If one imposes $\Sigma_{-1} = \Sigma_1 = \Sigma$ then the decision boundaries is an linear hyperplan

A Statistical Point of View
Example: LDA



A Statistical Point of View

Example: QDA



Naive Bayes

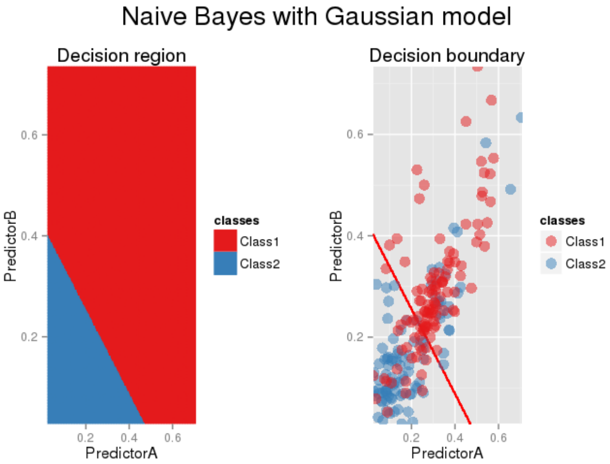
- Classical algorithm using a crude modeling for $\mathbb{P}\left\{X|Y\right\}$:
  - Feature independence assumption:

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

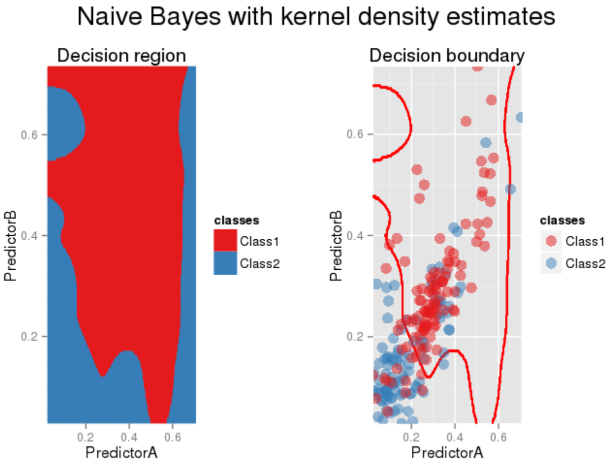
  - Simple featurewise model: binomial if binary, multinomial if finite and Gaussian if continuous
- If all features are continuous, similar to the previous Gaussian but with a diagonal covariance matrix!
- Very simple learning even in very high dimension!

A Statistical Point of View

Example: Naive Bayes



A Statistical Point of View
Example: Naive Bayes



- Direct modeling of $Y|x$.

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

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

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

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

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

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

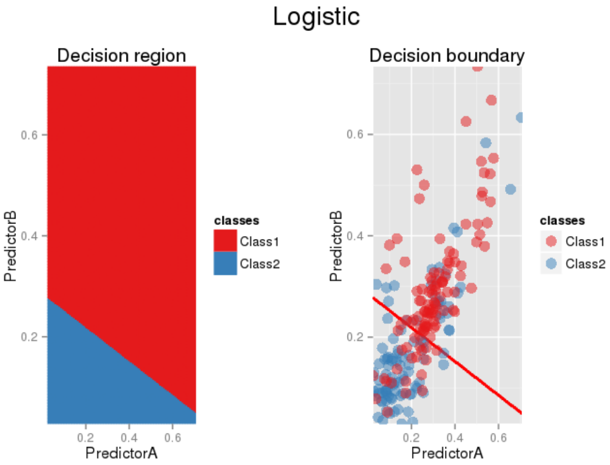
Opposite of the log-likelihood formula

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

- Convex function in β!
- **Remark:** You can also use your favorite parametric model instead of the logistic one...

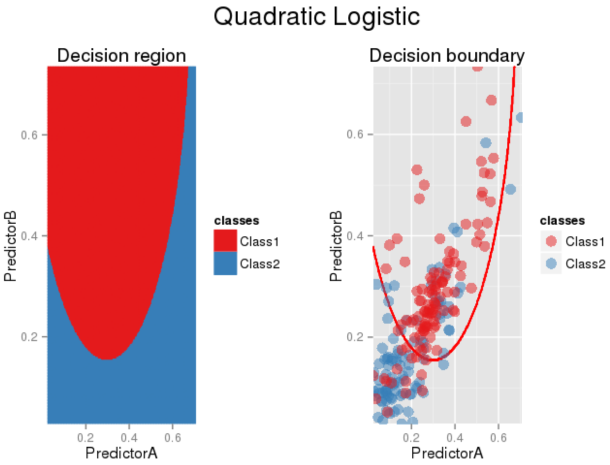
A Statistical Point of View

Example: Logistic



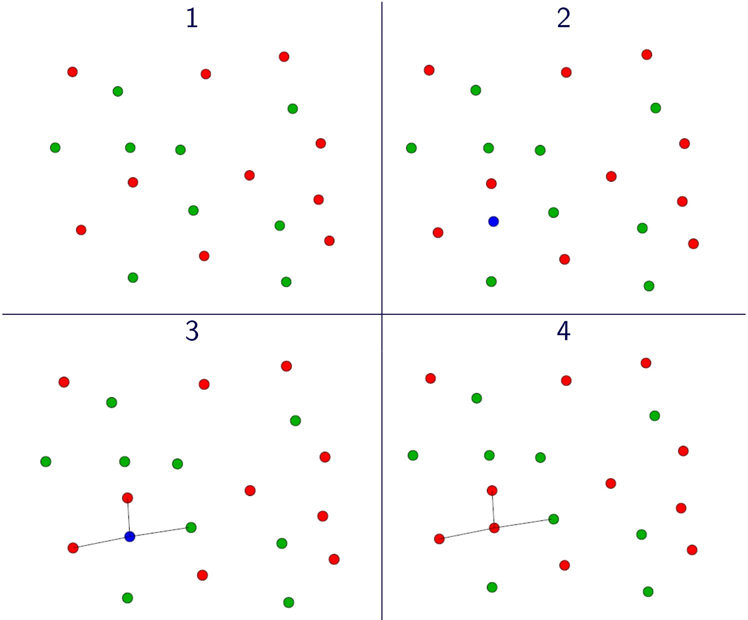
A Statistical Point of View

Example: Quadratic Logistic



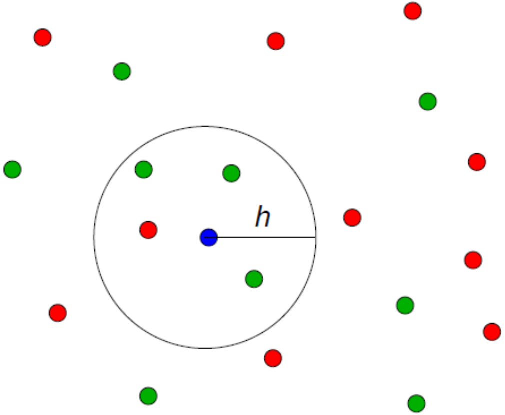
A Statistical Point of View

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



A Statistical Point of View

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



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

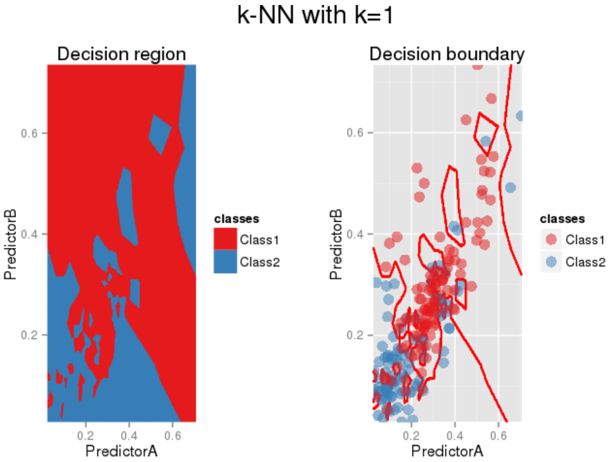
k-NN as local conditional density estimate

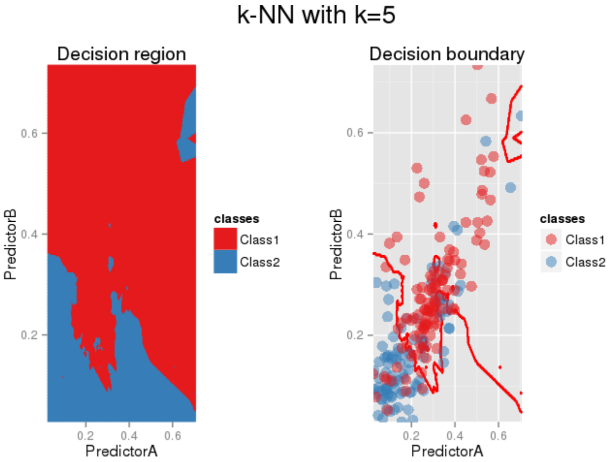
$$\widehat{p}_{+1}(\mathbf{x}) = \frac{\sum_{\mathbf{x}_i \in \mathcal{V}_{\mathbf{x}}} \mathbf{1}_{\{y_i=+1\}}}{|\mathcal{V}_{\mathbf{x}}|}$$

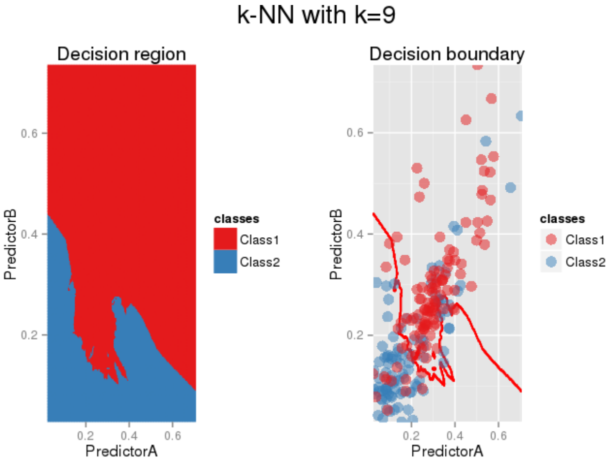
- KNN Classifier:

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

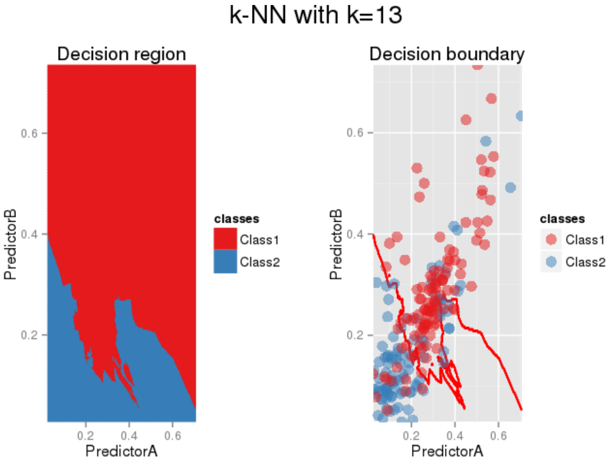
- **Remark:** You can also use your favorite kernel estimator...


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

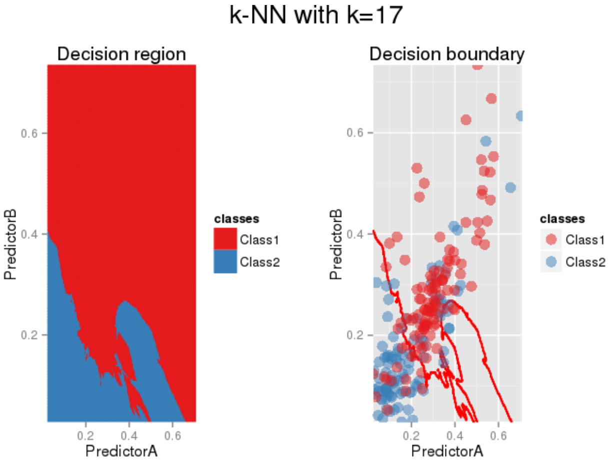

k-NN with k=5
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2


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

A Statistical Point of View
Example: KNN

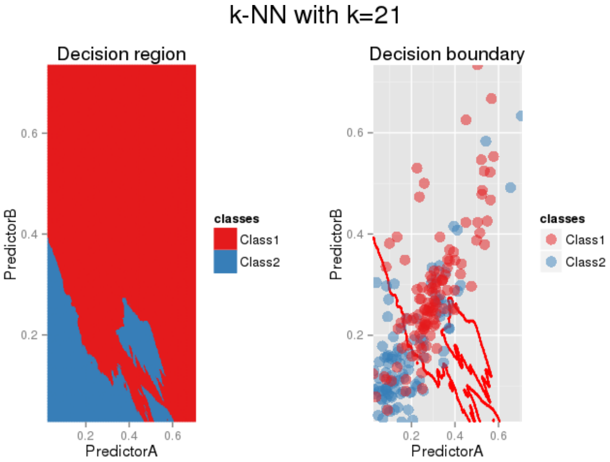


A Statistical Point of View
Example: KNN



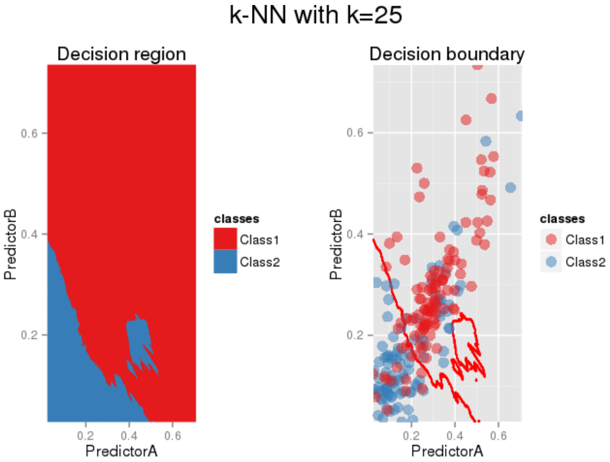
A Statistical Point of View

Example: KNN

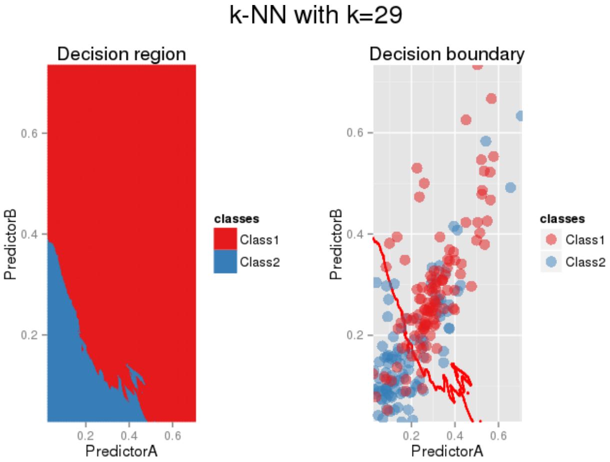


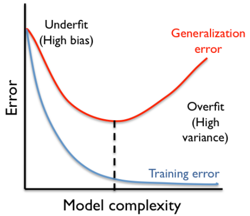
A Statistical Point of View

Example: KNN



A Statistical Point of View
Example: KNN





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).

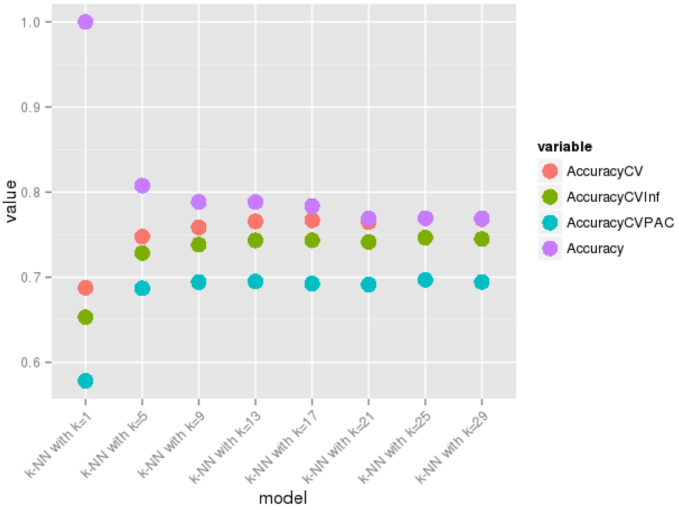
- Overfit for complex models: parameters learned are too specific to the learning set!
- General situation! (Think of polynomial fit...)
- Need to use an other criterion than the training error!

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

Cross Validation

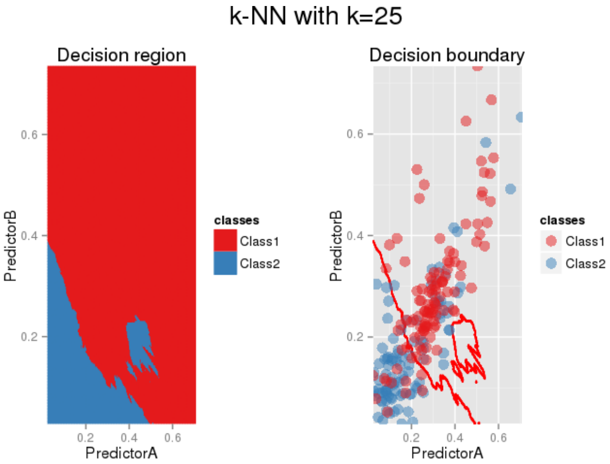
- Use $\frac{V-1}{V}n$ observations to train and $\frac{1}{V}n$ to verify!
- Validation for a learning set of size $(1 - \frac{1}{V}) \times n$ instead of n!

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


1.0
0.9
0.8
0.7
0.6
value
k-NN with k=1
k-NN with k=5
k-NN with k=9
k-NN with k=13
k-NN with k=17
k-NN with k=21
k-NN with k=25
k-NN with k=29
model
variable
AccuracyCV
AccuracyCVInf
AccuracyCVPAC
Accuracy

A Statistical Point of View

Example: KNN ($\widehat{k} = 25$ using cross-validation)



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

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

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

Problems

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

A Statistical Point of View

Solution: For $\mathbf{X}$, estimate $Y|\mathbf{X}$ plug this estimate in the Bayes classifier: (Generalized) Linear Models, Kernel methods, k-nn, Naive Bayes, Tree, Bagging...

An Optimization Point of View

Solution: If necessary replace the loss ℓ by an upper bound ℓ' and minimize the empirical loss: SVR, SVM, Neural Network,Tree, Boosting

- The best solution f^* is the one minimizing

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

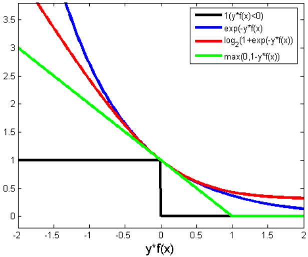
Empirical Risk Minimization

- 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

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

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

An Optimization Point of View
Classification Loss and Convexification



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

Classical convexification

- Logistic loss: $\ell(y, f(x)) = \log(1 + e^{-yf(x)})$ (Logistic / NN)
- Hinge loss: $\ell(y, f(x)) = (1 - yf(x))_+$ (SVM)
- Exponential loss: $\ell(y, f(x)) = e^{-yf(x)}$ (Boosting...)

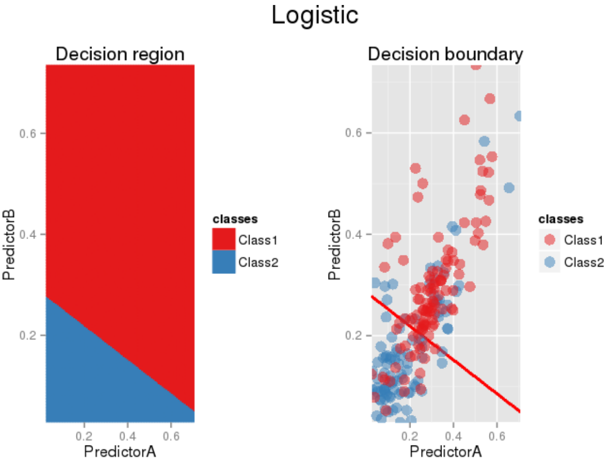
- Ideal solution:

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

Logistic regression

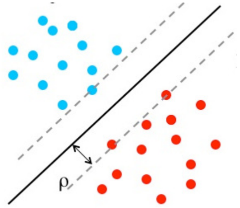
- Use $f(x) = \langle \beta, x \rangle + b$.
- Use the logistic loss $\ell(y, f) = \log_2(1 + e^{-yf})$, i.e. the -log-likelihood.

- Different vision than the statistician but same algorithm!


Logistic
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

An Optimization Point of View

Ideal Separable Case



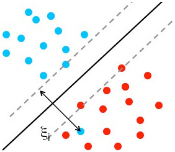
- Linear classifier: $\text{sign}(\langle \beta, x \rangle + b)$
- Separable case: $\exists(\beta, b), \forall i, y_i(\langle \beta, x \rangle + b) > 0$!

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 $\|\beta\|^2$.

An Optimization Point of View

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:

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

- Non convex minimization.

SVM: better convex relaxation!

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

- Convex relaxation:

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

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

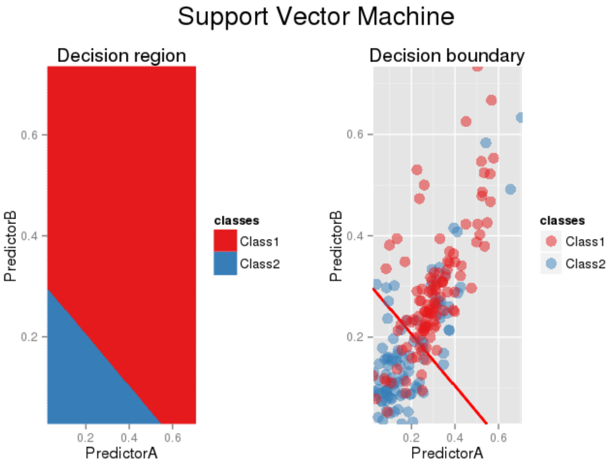
- **Prop:** $\ell^{0/1}(y_i, \operatorname{sign}(\langle \beta, x\rangle + b)) \leq \max(1 - y_i(\langle \beta, x\rangle + b), 0)$

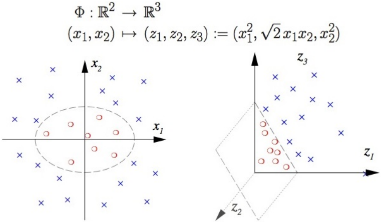
Penalized convex relaxation (Tikhonov!)

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

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

An Optimization Point of View
SVM



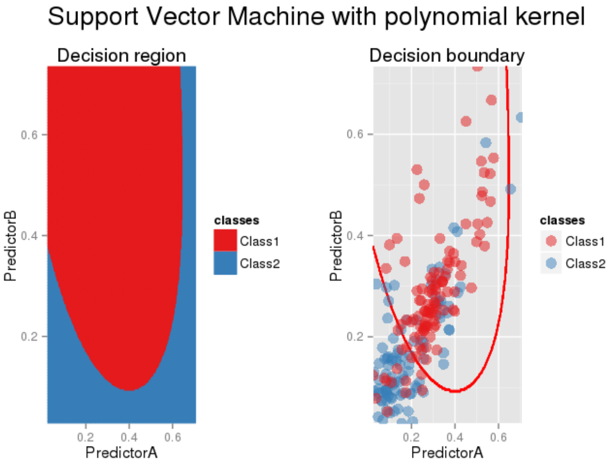


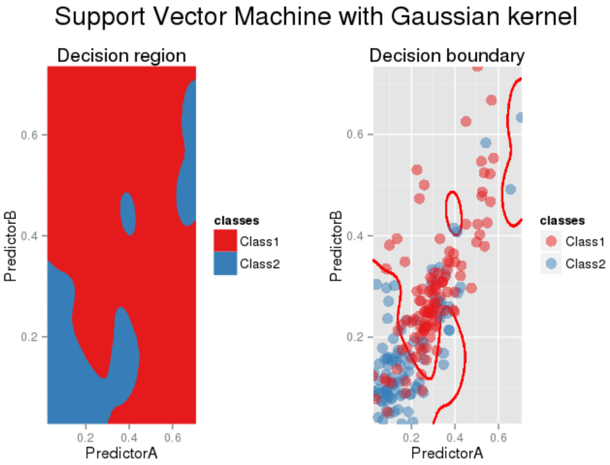
- Non linear separation: just replace x by a non linear $\Phi(x)$...

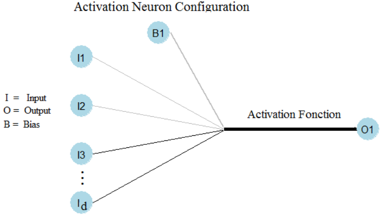
Kernel trick

- 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...


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


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

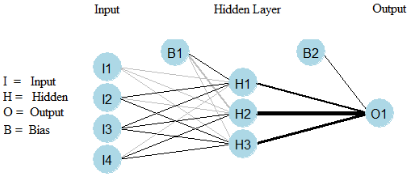


Artificial neuron

- Structure:
  - Mix inputs with a weighted sum,
  - Apply a (non linear) activation 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.



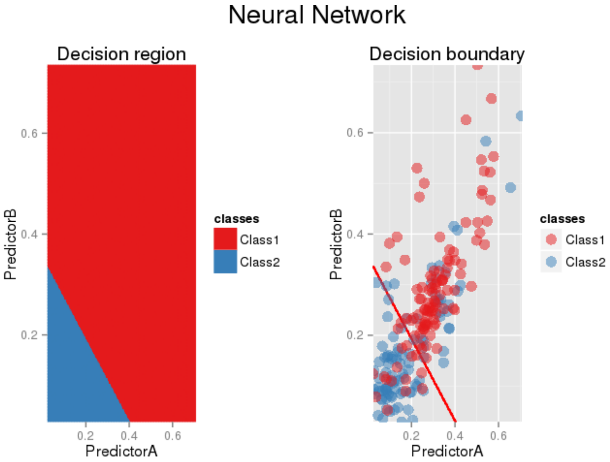
Neural network structure

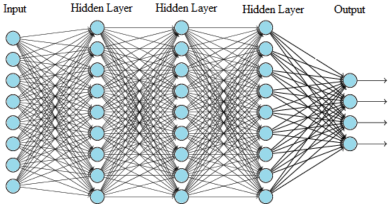
- Cascade of artificial neurons organized in layers
- Thresholding decision only at the output layer

- 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.

An Optimization Point of View

Neural network

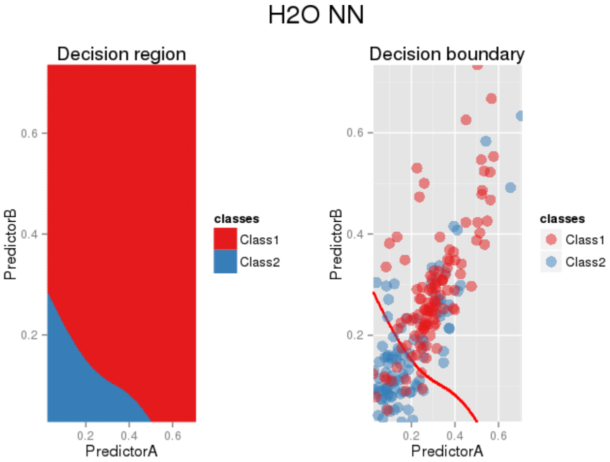


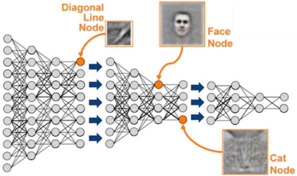


Deep Neural Network structure

- 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!

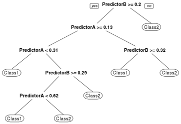

H2O NN
Decision region
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2
Decision boundary
PredictorB
0.6
0.4
0.2
0.2
0.4
0.6
PredictorA
classes
Class1
Class2



Family of Machine Learning algorithm combining:

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

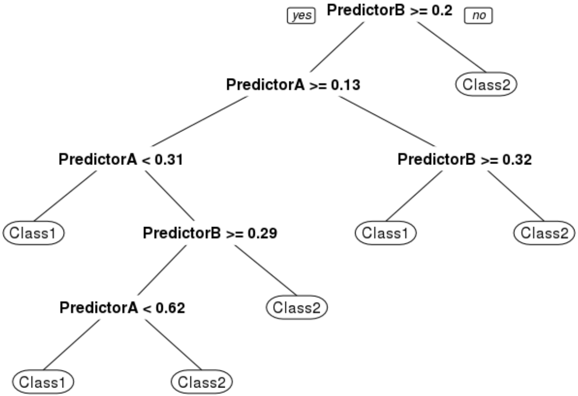
- Examples: Deep Neural Network, Deep (Restricted) Boltzman Machine, Stacked Encoder...
- Appears to be very efficient but lack of theoretical fundation!

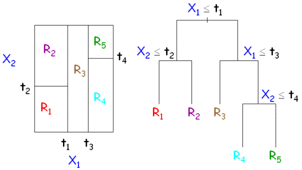


Tree principle

- 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)


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



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...

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...)

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!

- 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.

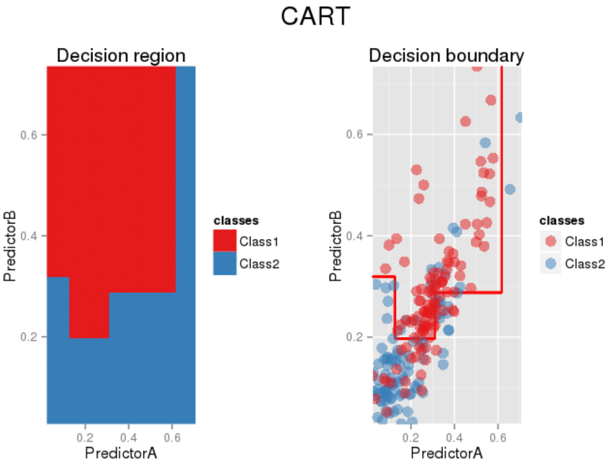
Key idea

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

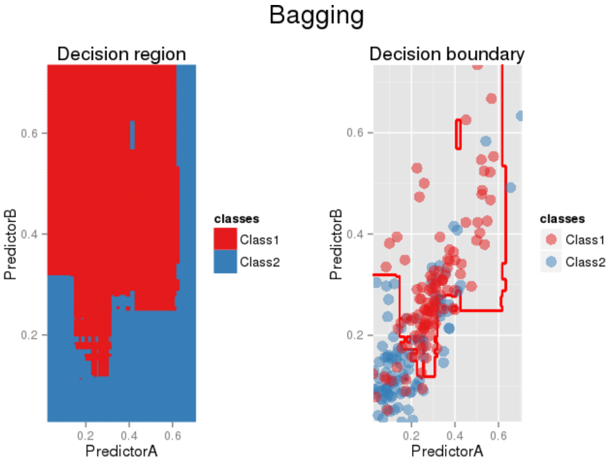
- Lack of robustness for single trees.
- How to combine trees?

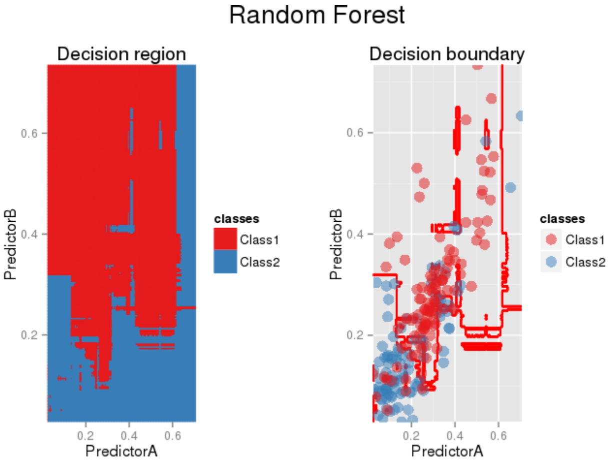
Parallel construction

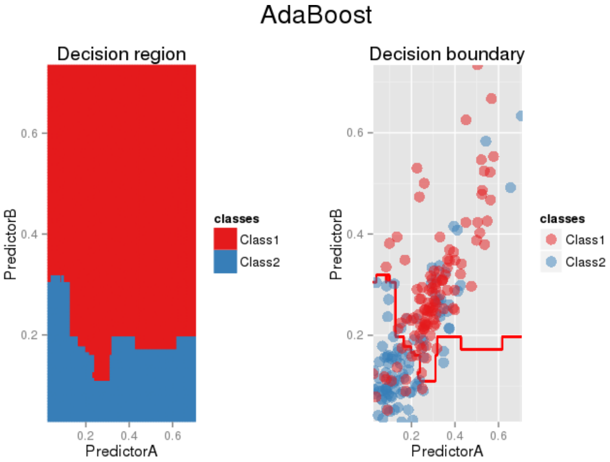
- Construct several trees from bootstrapped samples and average the responses (bagging)
- Add more randomness in the tree construction (random forests)

Sequential construction

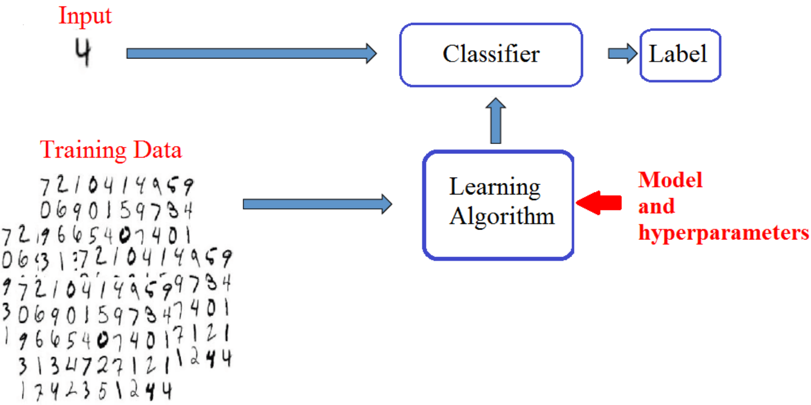
- Construct a sequence of trees by reweighting sequentially the samples according to their difficulties (AdaBoost)
- Reinterpretation as a stagewise additive model (Boosting)


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


Random Forest
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2


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

Model Selection
Model and Hyperparameters



- Ideal solution:

$$f^*(x) = \arg\max \mathbb{P}\{Y|x\}$$

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...

- Ideal solution:

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

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

- Ideal solution:

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

Kernel methods

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

- Model hyperparameters:
  - Features
  - Bandwidth and kernel

- 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.

- 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

- 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)

- Ideal solution:

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

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)

Models

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

- Key to obtain good performance!

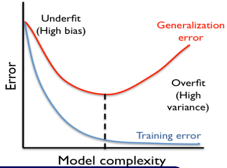
Approximation error and estimation error (Bias/Variance)

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

- 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!

- Empirical error biased toward complex models!



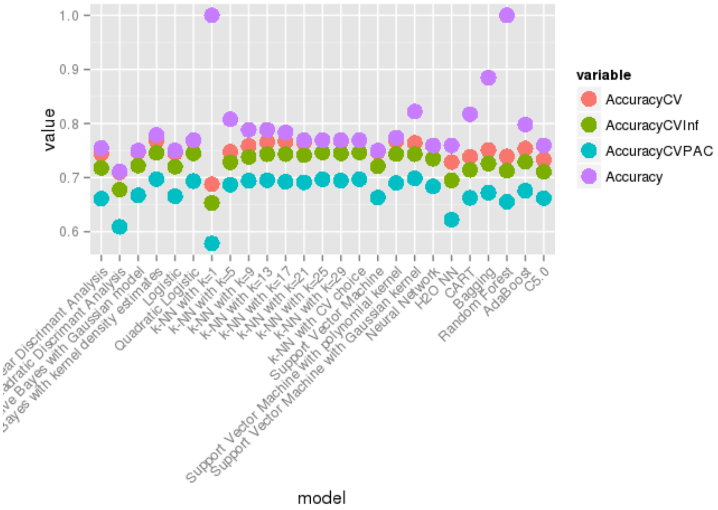
Selection criterion

- **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 $\mathcal{S}$

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

and choose the model with the smallest penalized risk.

Model Selection
Cross Validation



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

Mixture

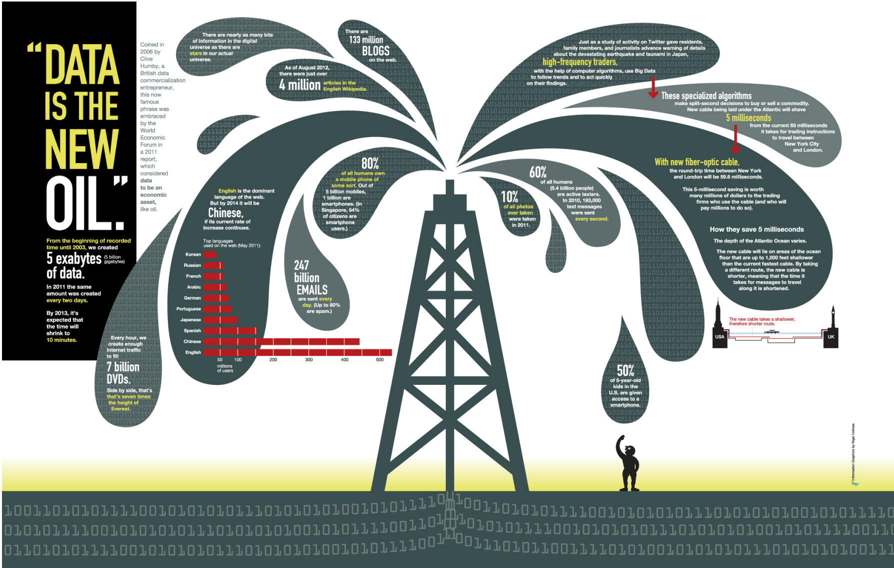
- Model averaging
- Data dependent model averaging (learn mixture weights)

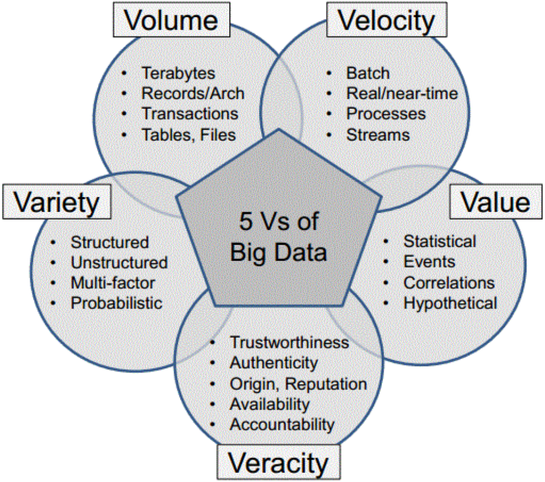
Stagewise

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

Data Science and Big Data

Data is the new Oil!



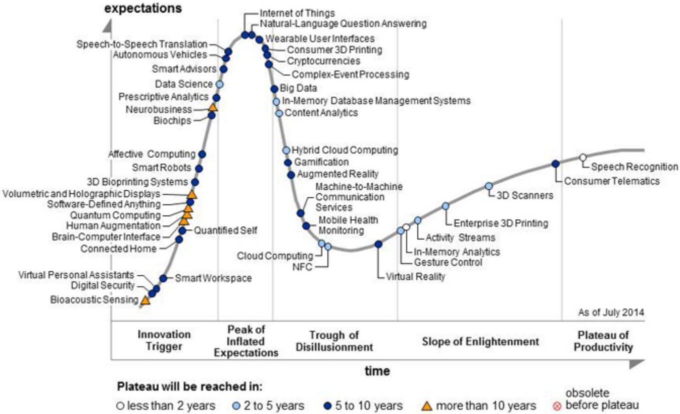

Volume
Terabytes
Records/Arch
Transactions
Tables, Files
Velocity
Batch
Real/near-time
Processes
Streams
Variety
Structured
Unstructured
Multi-factor
Probabilistic
5 Vs of
Big Data
Value
Statistical
Events
Correlations
Hypothetical
Trustworthiness
Authenticity
Origin, Reputation
Availability
Accountability
Veracity

Data Science and Big Data

Lots of Words!

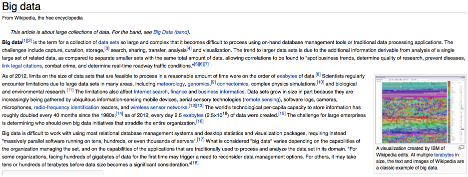
Data Science and Big Data

Don't Believe the Hype?



- Data Science and Big Data: Much more than a hype!

Data Science and Big Data
Wikipedia



Big data

From Wikipedia, the free encyclopedia

This article is about large collections of data. For the band, see Big Data (band).

Big data[1][2] is the term for a collection of data sets so large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications. The challenges include capture, curation, storage,[3] search, sharing, transfer, analysis[4] and visualization. The trend to larger data sets is due to the additional information derivable from analysis of a single large set of related data, as compared to separate smaller sets with the same total amount of data, allowing correlations to be found to "spot business trends, determine quality of research, prevent diseases, link legal citations, combat crime, and determine real-time roadway traffic conditions."[5][6][7]

As of 2012, limits on the size of data sets that are feasible to process in a reasonable amount of time were on the order of exabytes of data.[8] Scientists regularly encounter limitations due to large data sets in many areas, including meteorology, genomics,[9] connectomics, complex physics simulations,[10] and biological and environmental research.[11] The limitations also affect Internet search, finance and business informatics. Data sets grow in size in part because they are increasingly being gathered by ubiquitous information-sensing mobile devices, aerial sensory technologies (remote sensing), software logs, cameras, microphones, radio-frequency identification readers, and wireless sensor networks.[12][13] The world's technological per-capita capacity to store information has roughly doubled every 40 months since the 1980s;[14] as of 2012, every day 2.5 exabytes (2.5×10^{18}) of data were created.[15] The challenge for large enterprises is determining who should own big data initiatives that straddle the entire organization.[16]

Big data is difficult to work with using most relational database management systems and desktop statistics and visualization packages, requiring instead "massively parallel software running on tens, hundreds, or even thousands of servers".[17] What is considered "big data" varies depending on the capabilities of the organization managing the set, and on the capabilities of the applications that are traditionally used to process and analyze the data set in its domain. "For some organizations, facing hundreds of gigabytes of data for the first time may trigger a need to reconsider data management options. For others, it may take tens or hundreds of terabytes before data size becomes a significant consideration."[18]

A visualization created by IBM of Wikipedia edits. At multiple terabytes in size, the text and images of Wikipedia are a classic example of big data.

- **Big data** is an all-encompassing term for any collection of data sets so large and complex that it becomes difficult to process using traditional data processing applications.
- **Data science** is the study of the generalizable extraction of knowledge from data, yet the key word is science.
- **Statistics** is the study of the collection, analysis, interpretation, presentation and organization of data.
- **Machine Learning** explores the construction and the study of algorithms that can learn from and make predictions on data.

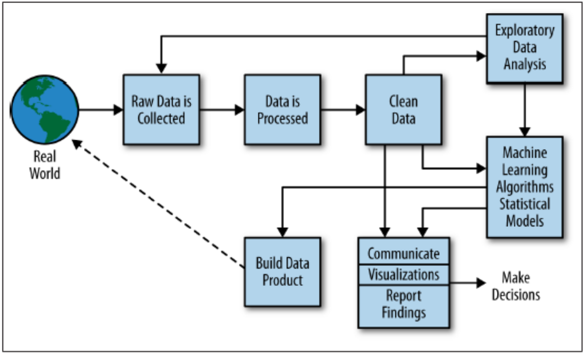


Figure 2-2. The data science process

Doing Data Science: Straight talk from the frontline

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

Data everywhere

- Huge volume,
- Huge variety...

Affordable computation units

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

- Growing academic and industrial interest!

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()
}
```

Data Science and Big Data

Big Data is (quite) Easy

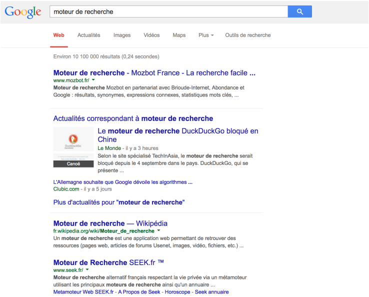
Example of *off the shelves* solution

```
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!

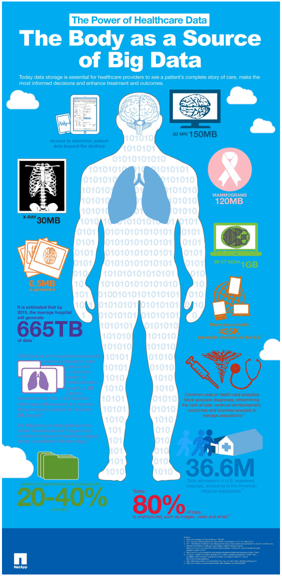
Data Science and Big Data

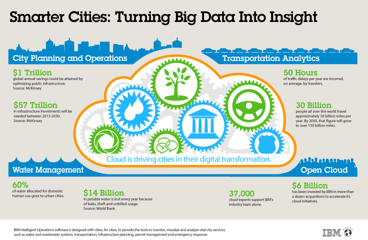
Web and Marketing



Data Science and Big Data

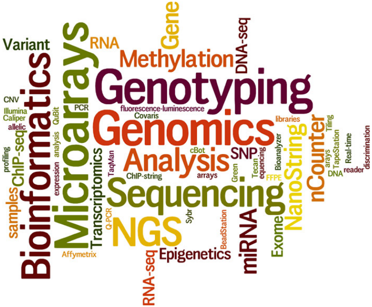
Industry and Society

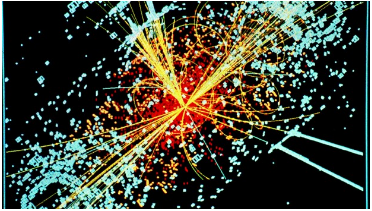




Data Science and Big Data

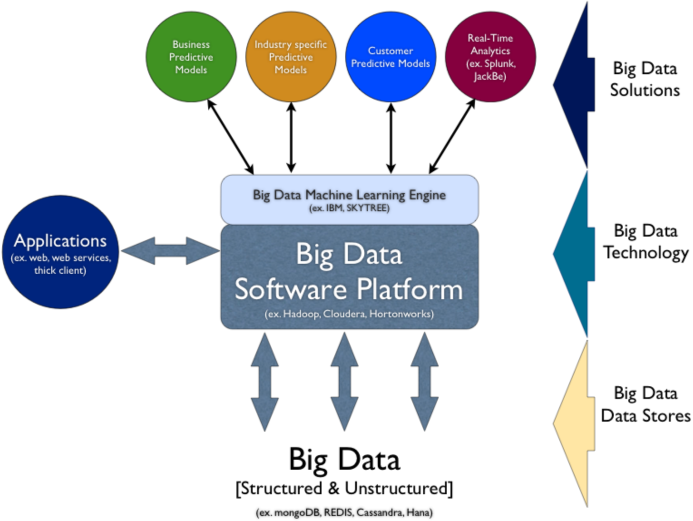
Science





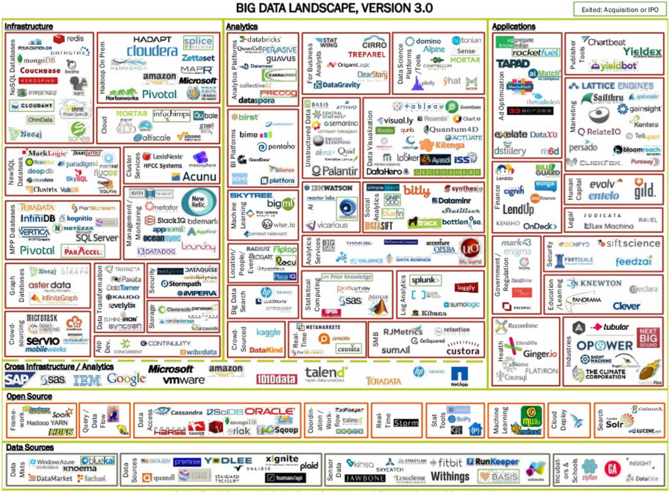
Data Science and Big Data

A Complex Ecosystem!



Data Science and Big Data

A Complex Ecosystem!



© Matt Turck (@mattturck), Sutian Dong (@sutiandong) & FirstMark Capital (@firstmarkcap)

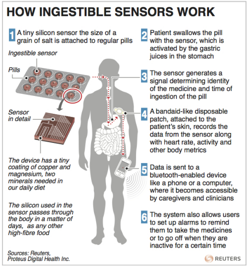
- Applied math **AND** Computer science
- Huge importance of domain specific knowledge: physics, signal processing, biology, health, marketing...

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
- Software(s)...

Data Science and Big Data

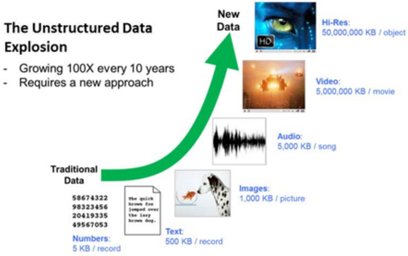
Data acquisition

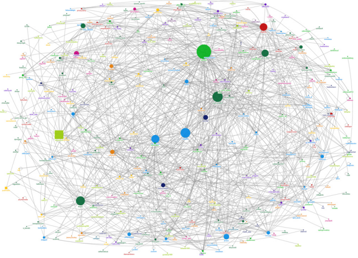


Some challenges

- 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?

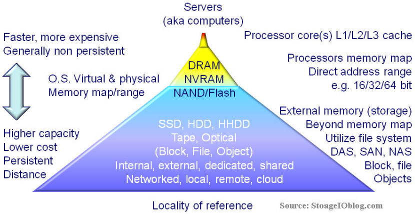
Data Science and Big Data
Unstructured Data

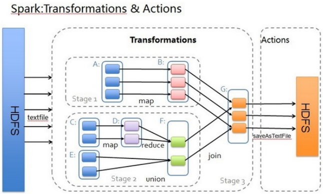




Some challenges

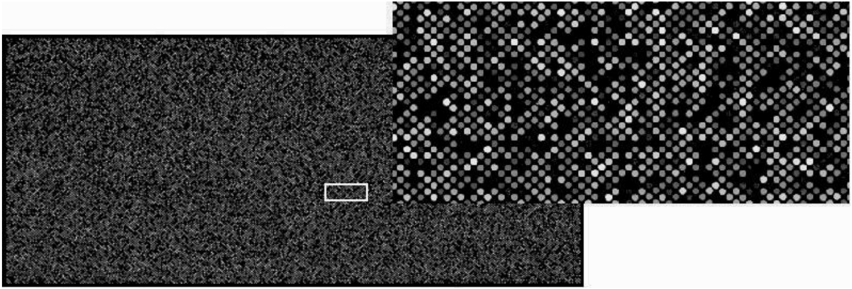
- How to store efficiently the data?
- How to describe (model) them to be able to process them?
- How to combine data of different nature?
- How to learn dynamics?





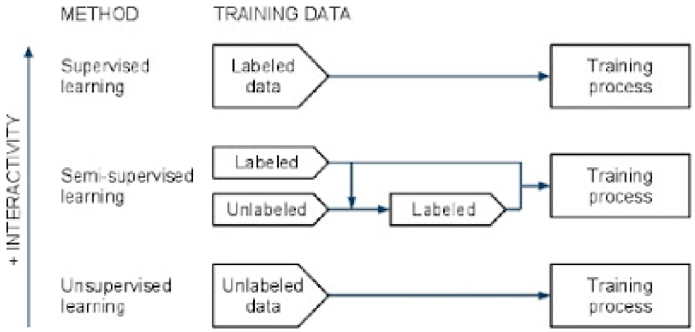
Some challenges

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



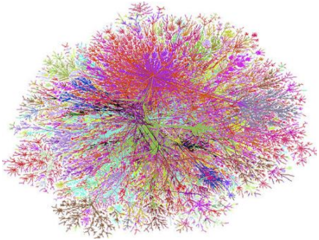
Some challenges

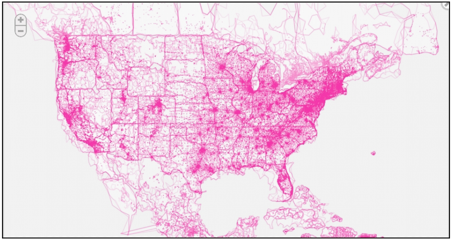
- How to describe (model) the data?
- How to reduce the data dimensionality?
- How to select/mix models?



Some challenges

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

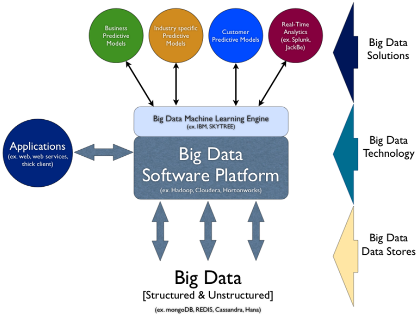




Some challenges

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

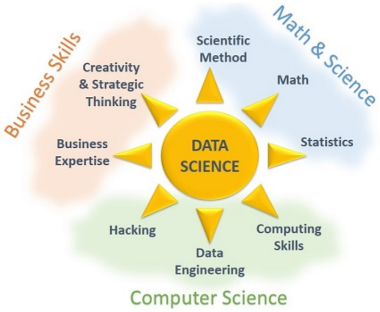
Data Science and Big Data
Software(s)

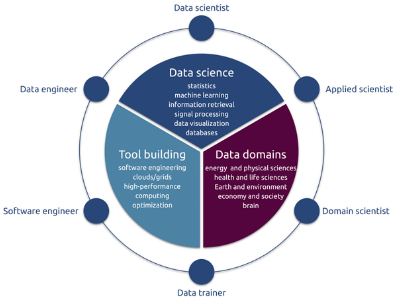


Some challenges

- How to construct a consistent ecosystem?
- How to construct interoperable systems?

Data Science and Big Data
Data Scientists!





Challenges

- No one masters all the skills!
- Importance of teams.
- Training...

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