

Machine Learning: from Theory to Practice

Statistical Learning and ML Methods

F. d'Alché-Buc and E. Le Pennec

Fall 2017

Doing Data Science

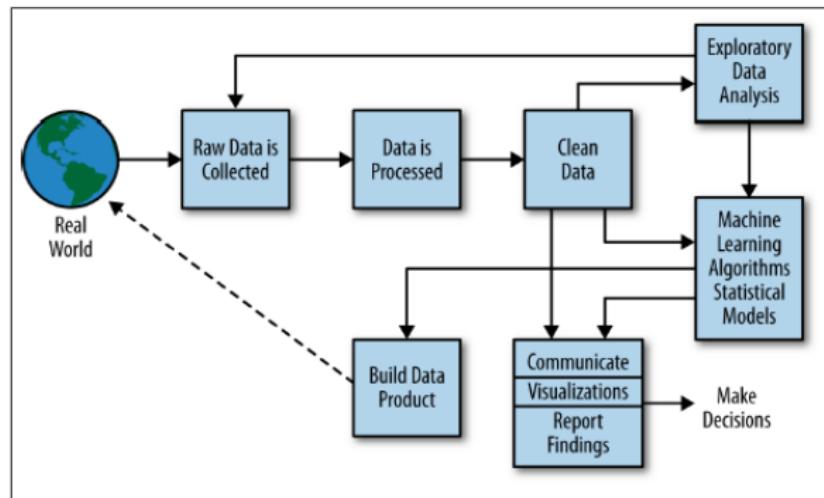
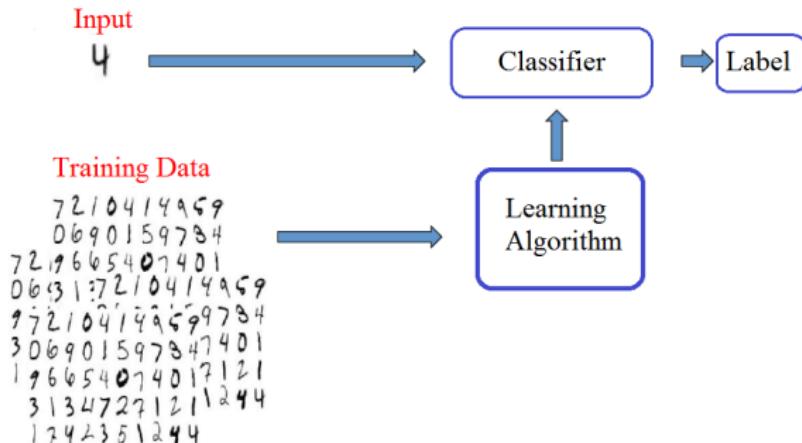


Figure 2-2. The data science process

- Doing Data Science: Straight talk from the frontline.
 - Rachel Schutt, Cathy O'Neil
 - O'Reilly

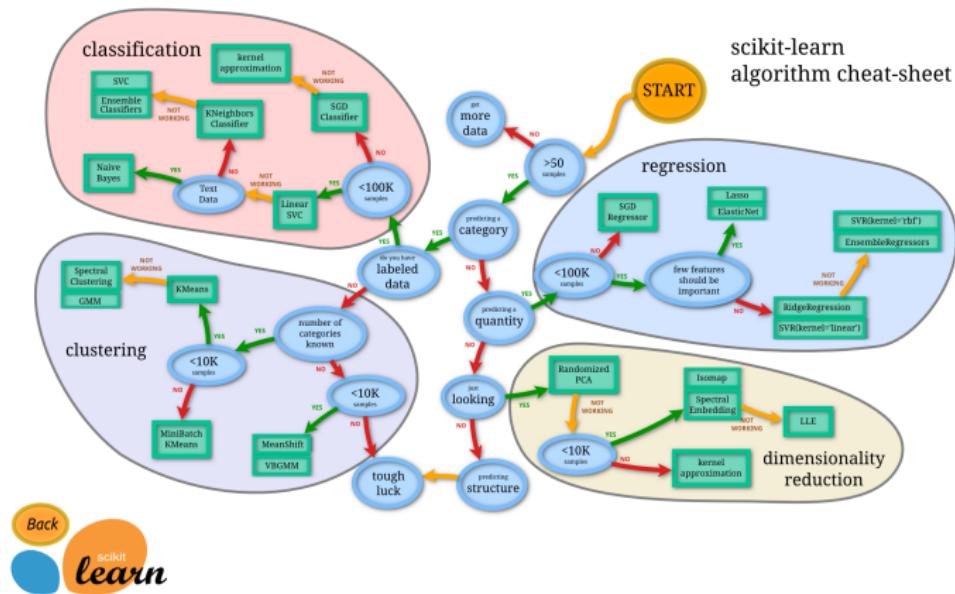
Machine Learning



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

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

Machine Learning: from Theory to Practice



- How to navigate and criticize Scikit-Learn map of ML algorithms!

Machine Learning: from Theory to Practice

Goal of the course

- to present the **fundamental principles of the classical ML methodologies** in both the **supervised** and the **unsupervised** case: linear regression, logistic regression, naive Bayes, nearest neighbor, SVM, PCA, MDS, k-means, GMM, hierarchical clustering...
- to describe some **advanced methodologies**: RKHS, trees, ensemble methods, graph, feature design...
- to explain how **those tools and their underlying principles** can be used to tackle **less classical setting** such as non tabular data (images/text) or recommendation systems.

Machine Learning: from Theory to Practice

Teaching Team

Florence d'Alché-Buc (Telecom ParisTech)

<http://perso.telecom-paristech.fr/~fdalche/>

florence.dalche@telecom-paristech.fr



Erwan Le Pennec (Polytechnique)

<http://www.cmap.polytechnique.fr/~lePennec/>

Erwan.Le-Pennec@polytechnique.edu



Organization

- Course structured in 11 sessions mixing lectures and labs.

Schedule

- **22/09:** Introduction to ML and ML Methods I (ELP)
- **19/09:** ML Methods II (ELP)
- **06/10:** Kernel and RKHS (FdA)
- **13/10:** Unsupervised Learning (ELP)
- **20/10:** Trees and Ensemble methods (ELP)
- **27/10:** Lab I
- **10/11:** ML & Graphs I (FdA)
- **17/11:** ML & Graphs II (FdA)
- **24/11:** Lab II
- **01/12:** Feature Design (ELP)
- **08/12:** Scaling ML (ELP)

Moodle and Evaluation

Moodle

- Main pedagogical tool with all the slides and labs!
- Do not forget to register if you want to validate.
- Assignment will be done through it.

Modalities

- **Lab reports** during the course period (5 pts)
 - **Written exam** in January (10 pts)
 - **Small project on a non classical ML task** due by the end of January (5 pts)
-
- **Retake** session in September.

Bibliography

-  R. Schutt and C. O'Neil (2014)
Doing Data Science: Straight talk from the frontline
O'Reilly
-  T. Hastie, R. Tibshirani and J. Friedman (2009)
The Elements of Statistical Learning
Springer Series in Statistics.
-  M. Mohri, A. Rostamizadeh and A. Talwalkar (2012)
Foundations of Machine Learning
The MIT Press.
-  Ch. Bishop (2006)
Pattern Recognition and Machine Learning
Springer Series in Information Science and Statistics
-  I. Goodfellow, Y. Bengio and A. Courville (2016)
Deep Learning / WIP
The MIT Press

Outline

1 Introduction

- Supervised Learning
- The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

- Fully Generative Modeling
- Parametric Modeling
- Non Parametric Conditional Estimation

4 A Discriminative Point of View

- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

5 Model Selection

- Models
- Feature Design
- Models, Complexity and Selection

Outline

Introduction

1 Introduction

- Supervised Learning
- The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

- Fully Generative Modeling
- Parametric Modeling
- Non Parametric Conditional Estimation

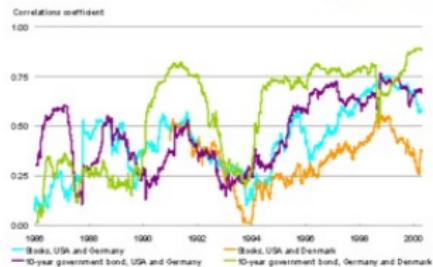
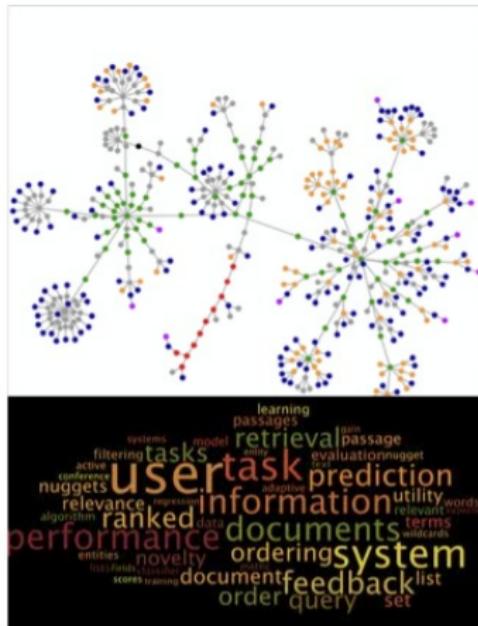
4 A Discriminative Point of View

- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

5 Model Selection

- Models
- Feature Design
- Models, Complexity and Selection

Motivation



Machine Learning everywhere !

Introduction

The collage illustrates various applications of machine learning:

- Wikipedia Network:** A complex network visualization showing edits across different Wikipedia pages, color-coded by category.
- DNA Sequence:** A diagram showing a DNA double helix with a sequence of nucleotides (G, C, A, T) and their corresponding amino acids (Ala, Arg, Asp, Asn, Cys).
- Piwik Dashboard:** A screenshot of the Piwik web analytics dashboard showing visitor statistics, actions, and referring sites for November 2008.
- Churn Prediction:** A graph illustrating features of a call network used for predicting future churners. It shows a central node with multiple connections (Strong link) and another node connected to many other nodes (Linked to many churners).

Use data to extract a prediction function

- Search engine, text-mining
- Diagnosis, Fault detection
- Business analytics
- Social network analytics
- Data-mining
- Link prediction
- Robotics

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.

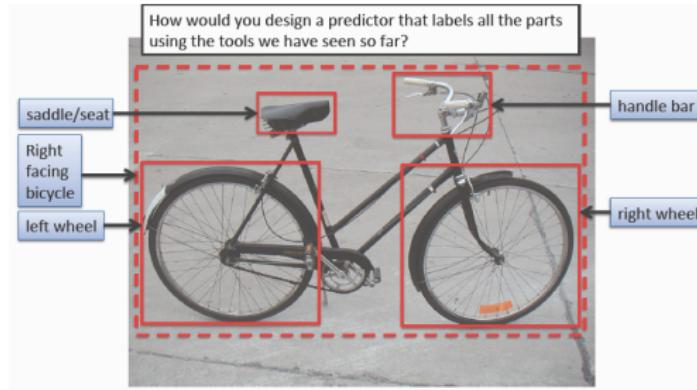
A robot endowed with a set of sensors and an online learning algorithm:



- **Task:** play football
- **Performance:** score
- **Experience:**
 - current environment and outcome,
 - past games

Object recognition in an image

Introduction



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

Online learning: *the learning algorithm keeps on interacting with the environment*

- robotics
- social networks
- cloud servers
- personalized advertising
- autonomous cars
- personalized healthcare
- security systems

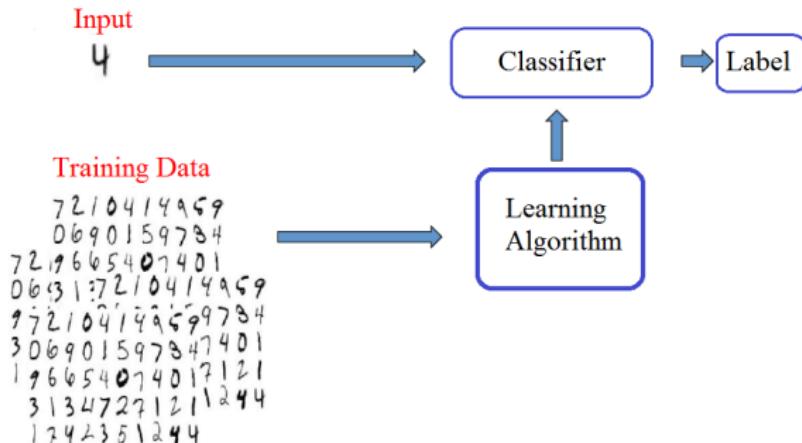
Offline or batch learning: *the learning algorithm gets a datafile and outputs some function that can be used in turn to new data*

- pattern Recognition
- diagnosis (health, plants, ...)
- link prediction in networks
- data-mining

This course → offline learning

Machine Learning

Introduction



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.

- Supervised Learning:
 - Goal: Learn a function f predicting a variable Y from an individual \mathbf{X} .
 - Data: Learning set (\mathbf{X}_i, Y_i)
- Unsupervised Learning:
 - Goal: Discover a structure within a set of individuals (\mathbf{X}_i) .
 - Data: Learning set (\mathbf{X}_i)
- First case is better posed.
- Supervised Learning (7 lectures) vs Unsupervised Learning (1 lecture)

Outline

Introduction

1 Introduction

- Supervised Learning
 - The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

- Fully Generative Modeling
- Parametric Modeling
- Non Parametric Conditional Estimation

4 A Discriminative Point of View

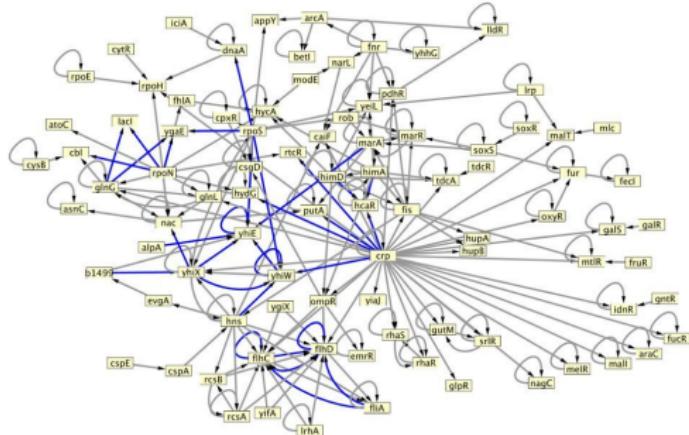
- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

5 Model Selection

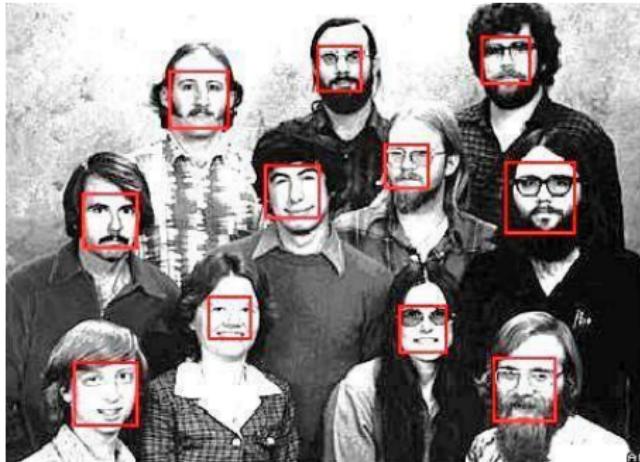
- Models
- Feature Design
- Models, Complexity and Selection

0	0	0	0	0
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4	4	4	4	4
5	5	5	5	5
6	6	6	6	6
7	7	7	7	7
8	8	8	8	8
9	9	9	9	9

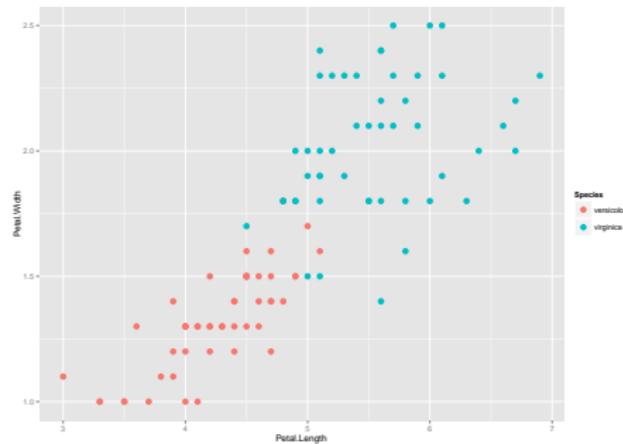
- Read a ZIP code from an envelop.
- Goal: give a number from an image.
- X = image.
- Y = corresponding number.



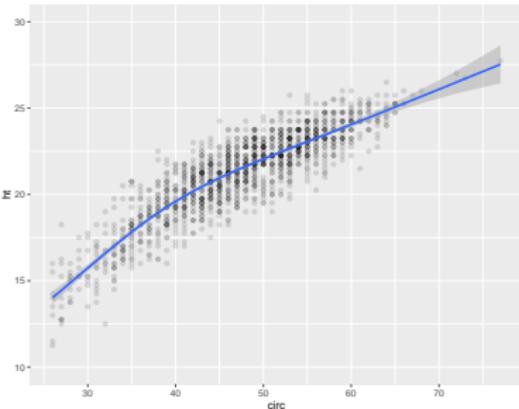
- Protein interaction network prediction.
- Goal: Predict (unknown) interactions between proteins.
- $X =$ pair of proteins.
- $Y =$ existence or no of interaction.
- Numerous similar questions in bio(informatics): genomic,...



- Goal: Detect the position of faces in an image
- Different setting?
- Reformulation as a supervised learning problem.
- X = sub window in the image
- Y = presence or no of a face...
- Lots of answer in a single image.
- Post processing required...



- Very classical dataset.
- Goal: predict the species from the shape of the petal.
- \mathbf{X} = height/width of the petal.
- \mathbf{Y} = species.



- Simple (and classical) dataset.
- Goal: predict the height from circumference.
- $\mathbf{X} = \text{circumference}$.
- $\mathbf{Y} = \text{height}$.

Outline

Introduction

1 Introduction

- Supervised Learning
- The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

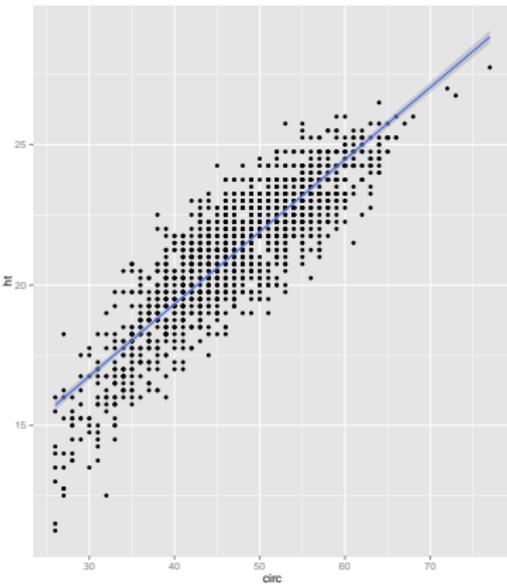
- Fully Generative Modeling
- Parametric Modeling
- Non Parametric Conditional Estimation

4 A Discriminative Point of View

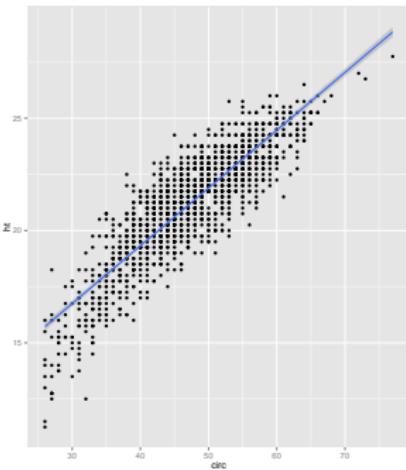
- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

5 Model Selection

- Models
- Feature Design
- Models, Complexity and Selection



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



Linear Model

- Parametric model:

$$f_{\beta}(\text{circ}) = \beta_1 + \beta_2 \text{circ}$$

- How to choose $\beta = (\beta_1, \beta_2)$?

Methodology

- Natural goodness criterion:

$$\begin{aligned}\sum_{i=1}^n |Y_i - f_\beta(\mathbf{X}_i)|^2 &= \sum_{i=1}^n |\text{ht}_i - f_\beta(\mathbf{circ}_i)|^2 \\ &= \sum_{i=1}^n |\text{ht}_i - (\beta_1 + \beta_2 \mathbf{circ}_i)|^2\end{aligned}$$

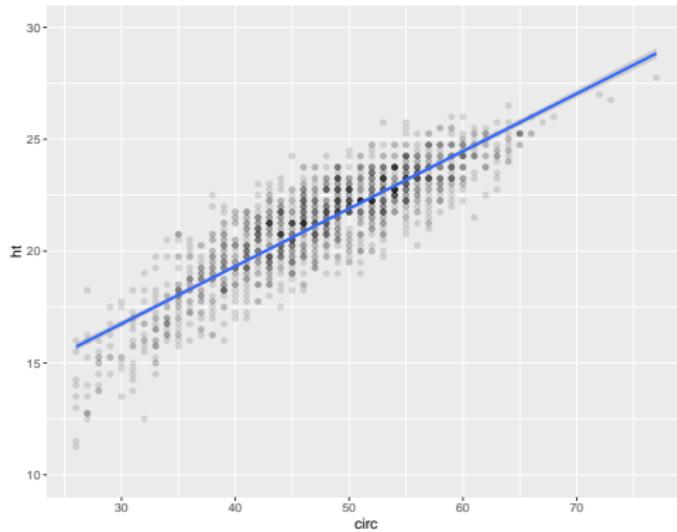
- Choice of β that minimizes this criterion!

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^2}{\operatorname{argmin}} \sum_{i=1}^n |\text{ht}_i - (\beta_1 + \beta_2 \mathbf{circ}_i)|^2$$

- Easy minimization with an explicit solution!

Prediction

Introduction



Prediction

- Linear prediction for the height:

$$\hat{ht} = \hat{f}_\beta(\text{circ}) = \hat{\beta}_1 + \hat{\beta}_2 \text{circ}$$

Linear Regression

- **Statistical model:** $(\text{circ}_i, \text{ht}_i)$ i.i.d. with the same law than a generic (circ, ht) .
- **Performance criterion:** Look for f with a small average error

$$\mathbb{E} [|\text{ht} - f(\text{circ})|^2]$$

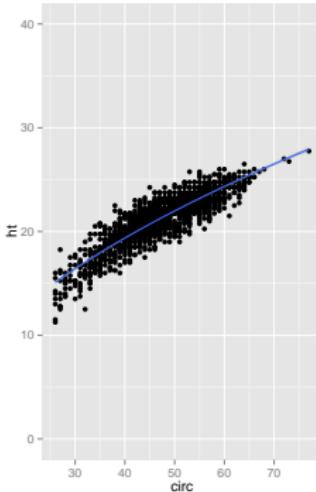
- **Empirical criterion:** Replace the unknown law by its empirical counterpart

$$\frac{1}{n} \sum_{i=1}^n |\text{ht}_i - f(\text{circ}_i)|^2$$

- **Predictor model:** As the minimum over all function is 0 (if all the circ_i are different), restrict to the linear functions $f(\text{circ}) = \beta_1 + \beta_2 \text{circ}$ to avoid over-fitting.
- **Model fitting:** Explicit formula here.
- This model can be too simple!

Polynomial Regression

Introduction

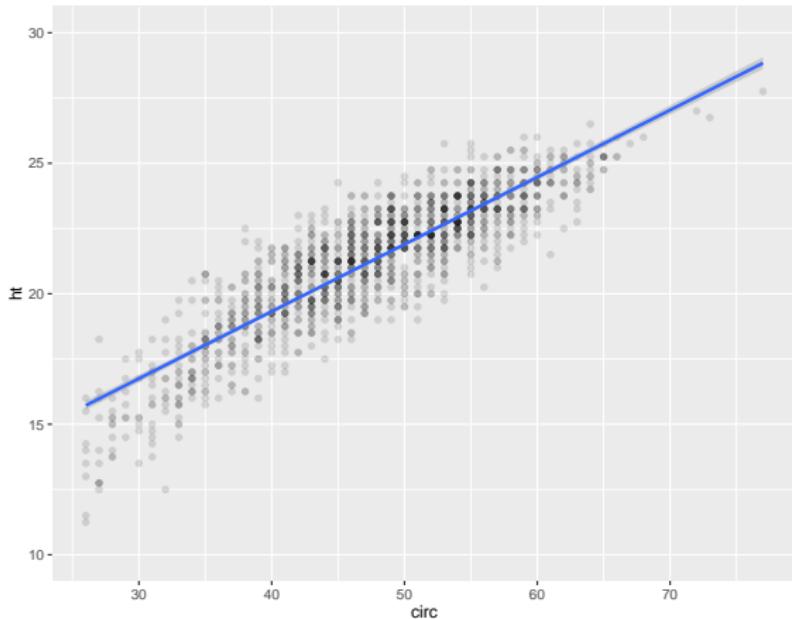


Polynomial Model

- Polynomial model: $f_{\beta}(\text{circ}) = \sum_{l=1}^p \beta_l \text{circ}^{l-1}$
- Linear in $\beta!$
- Easy least squares estimation for any degree!

Which Degree?

Introduction

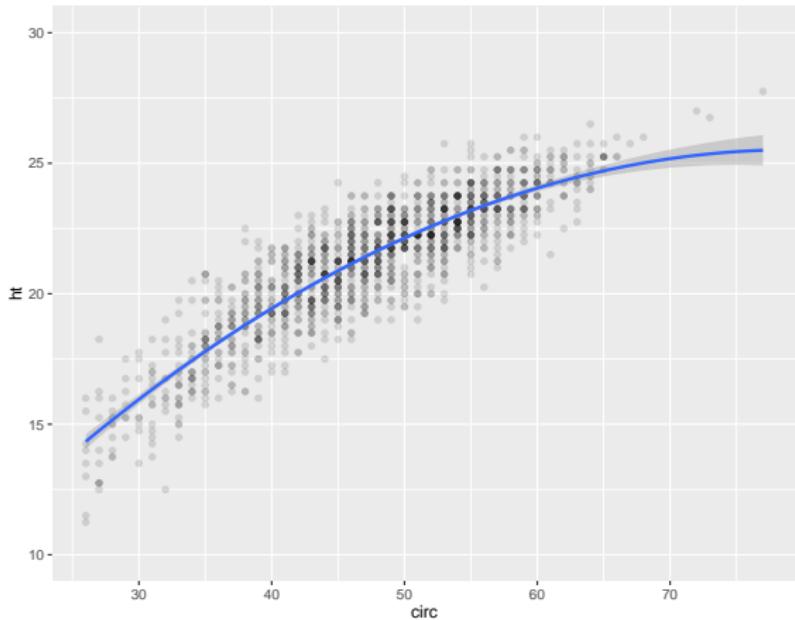


Models

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

Which Degree?

Introduction

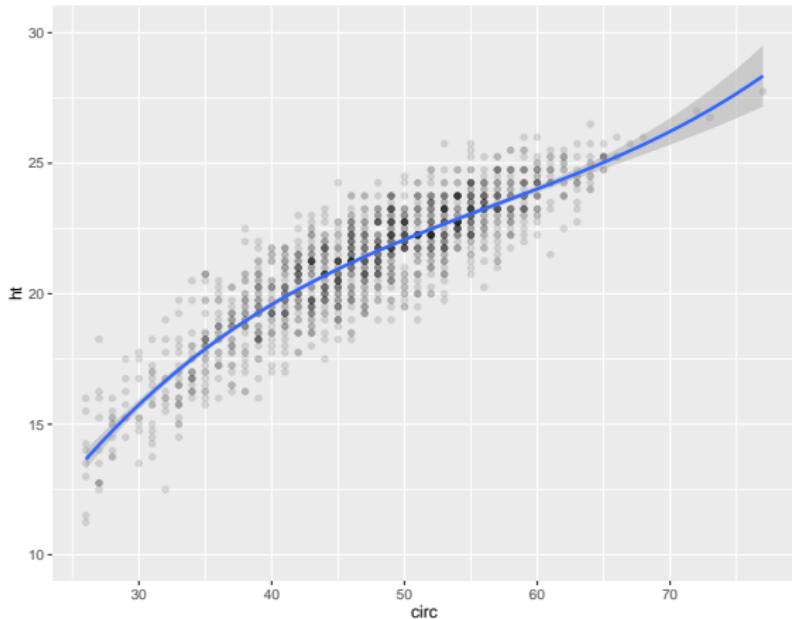


Models

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

Which Degree?

Introduction

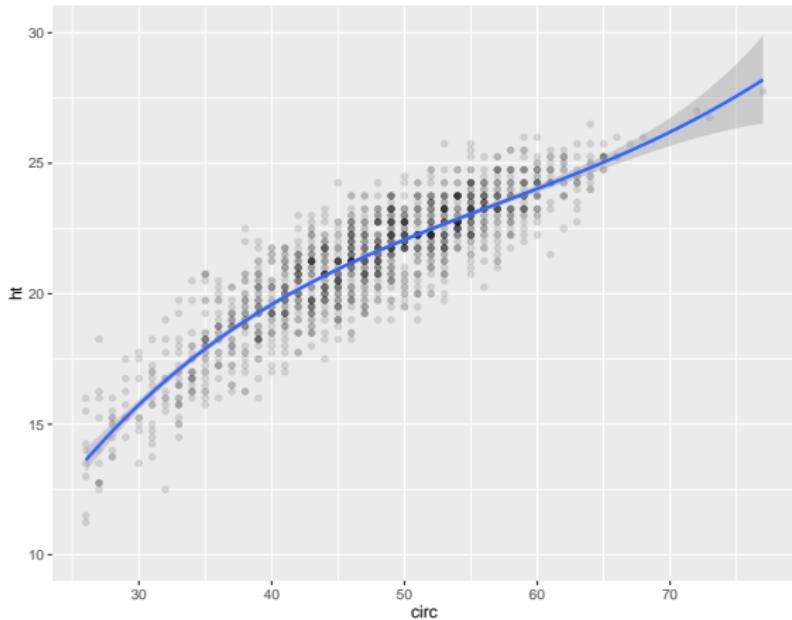


Models

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

Which Degree?

Introduction

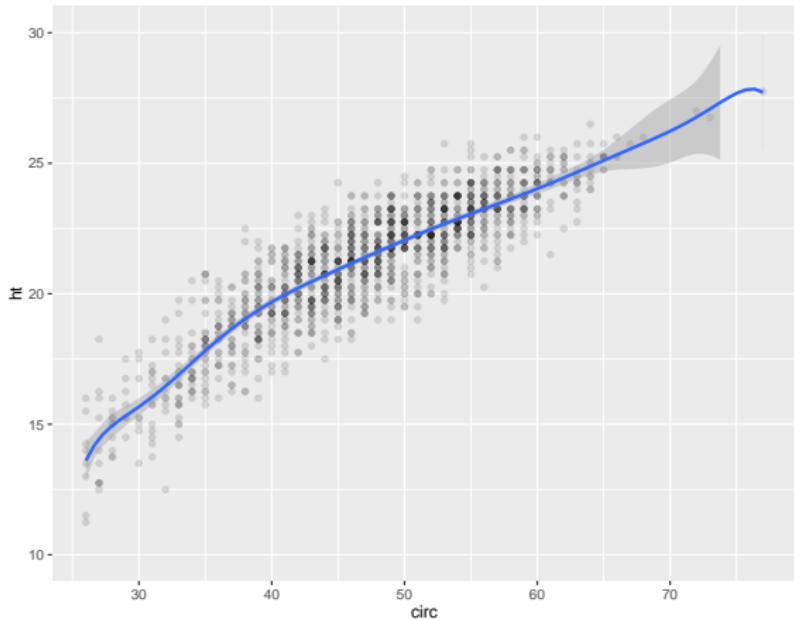


Models

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

Which Degree?

Introduction

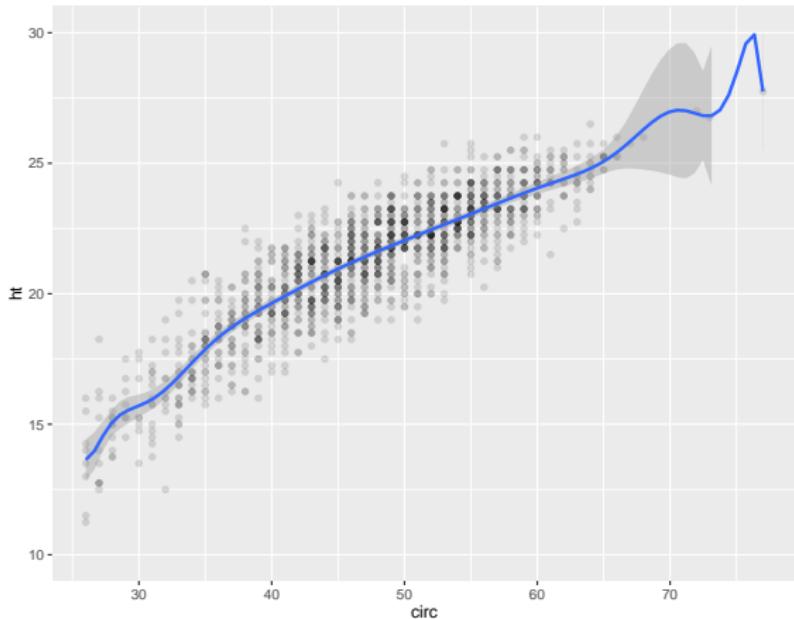


Models

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

Which Degree?

Introduction

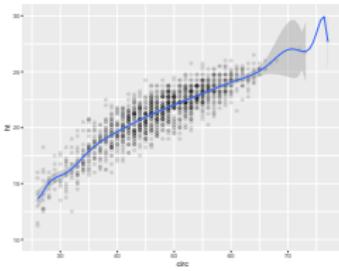
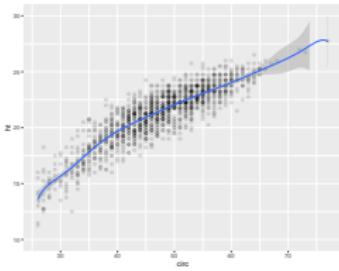
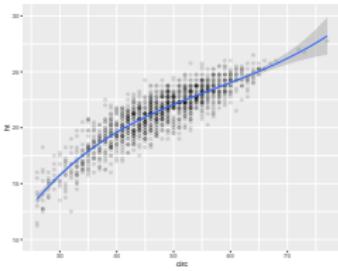
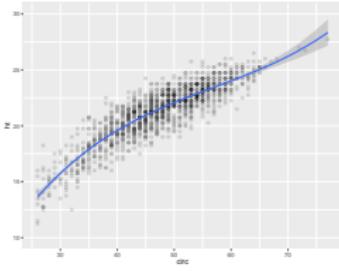
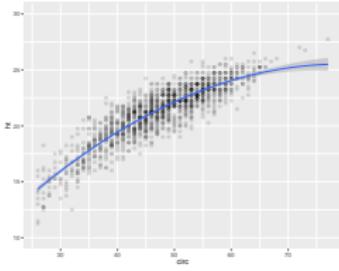
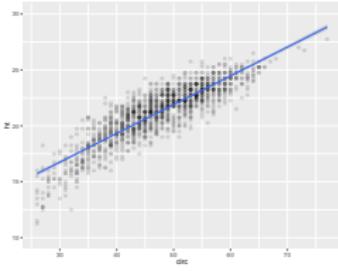


Models

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

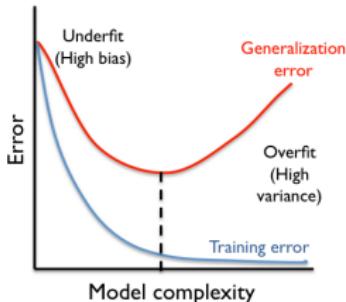
Which Degree?

Introduction



Best Degree?

- How to choose among those solution?



Error behavior

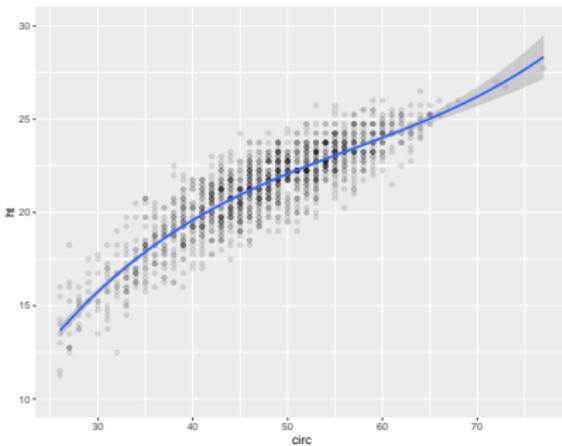
- Empirical risk (error made on the training set) decays when the complexity of the model increases.
- Quite different behavior when the error is computed on new observations (true risk / generalization error).
- Overfit for complex models: parameters learned are too specific to the learning set!
- General situation! (Think of polynomial fit...)
- Need to use an other criterion than the training error!

Two directions

- How to estimate the generalization error in a different way?
- Find a way to correct the empirical error?

Two Approaches

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



Questions

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

- 1 Introduction
 - Supervised Learning
 - The Example of Univariate Linear Regression
- 2 Statistical Supervised Learning
- 3 A Generative Point of View
 - Fully Generative Modeling
 - Parametric Modeling
 - Non Parametric Conditional Estimation
- 4 A Discriminative Point of View
 - Convexification of the Risk
 - SVM
 - Penalization
 - (Deep) Neural Networks
 - Tree Based Methods
- 5 Model Selection
 - Models
 - Feature Design
 - Models, Complexity and Selection

Supervised Learning Framework

- Input measurement $\mathbf{X} = (X^{(1)}, X^{(2)}, \dots, X^{(d)}) \in \mathcal{X}$
- Output measurement $Y \in \mathcal{Y}$.
- $(\mathbf{X}, Y) \sim \mathbf{P}$ with \mathbf{P} unknown.
- **Training data** : $\mathcal{D}_n = \{(\mathbf{X}_1, Y_1), \dots, (\mathbf{X}_n, Y_n)\}$ (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 **predictor** is a function in $\mathcal{F} = \{f : \mathcal{X} \rightarrow \mathcal{Y} \text{ measurable}\}$

Goal

- Construct a **good** predictor \hat{f} from the training data.
- Need to specify the meaning of **good**.
- Formally, classification and regression are the same problem!

Loss function

- **Loss function** : $\ell(Y, f(\mathbf{X}))$ measure how well $f(\mathbf{X})$ 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:

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

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

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

Experience, Task and Performance measure

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

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

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

Goal

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

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

Bayes Predictor (explicit solution)

- In binary classification with 0 – 1 loss:

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

- In regression with the quadratic loss

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

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

Machine Learning

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

Canonical example: Empirical Risk Minimizer

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

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

- Examples:
 - Linear regression
 - Linear discrimination with

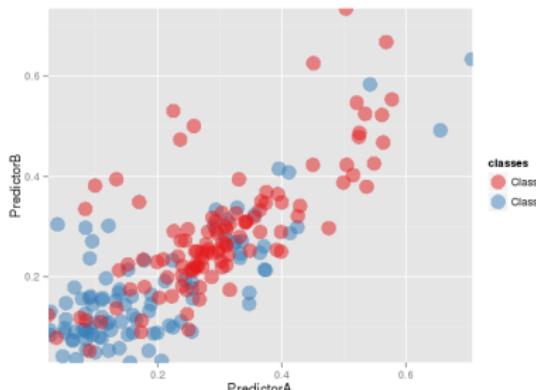
$$\mathcal{S} = \{\mathbf{x} \mapsto \operatorname{sign}\{\beta^T \mathbf{x} + \beta_0\} / \beta \in \mathbb{R}^d, \beta_0 \in \mathbb{R}\}$$

Example: TwoClass Dataset

Stat. Superv. Learn.

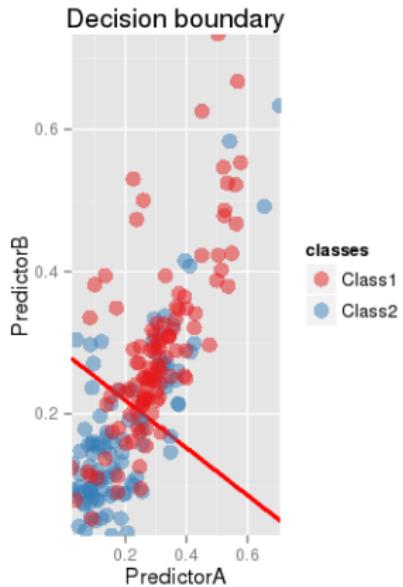
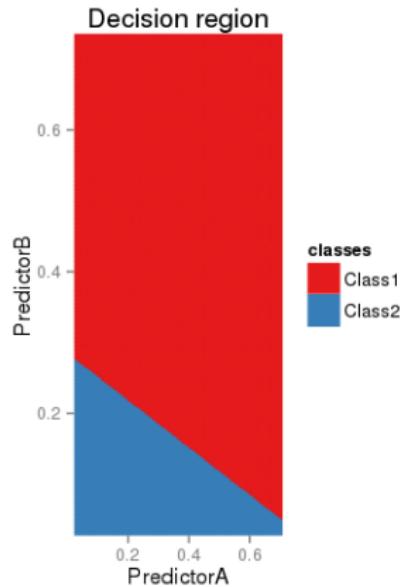
Synthetic Dataset

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



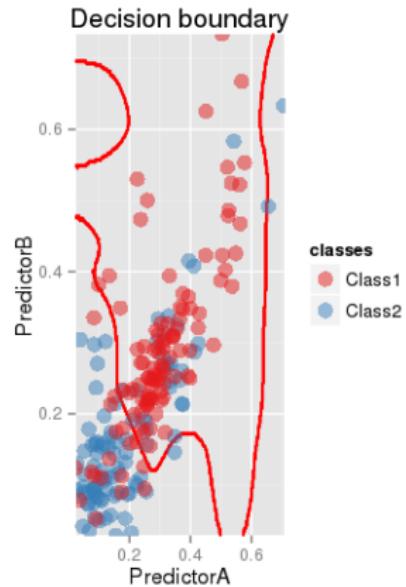
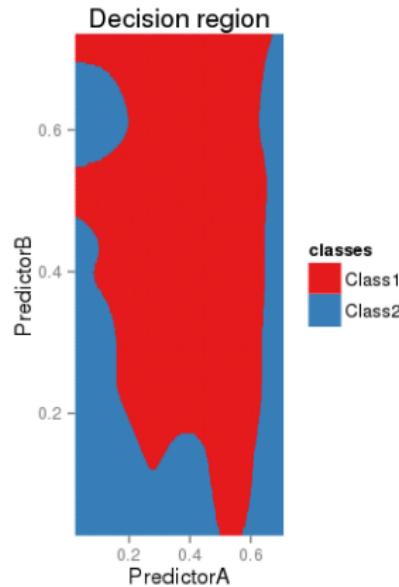
Example: Linear Discrimination

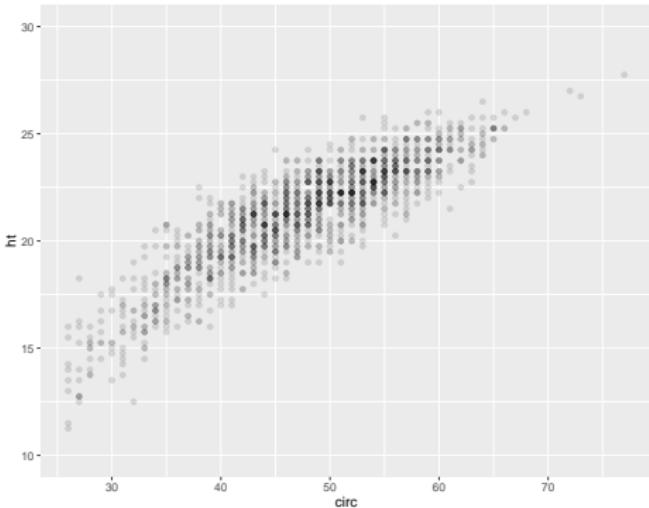
Stat. Superv. Learn.



Example: More Complex Model

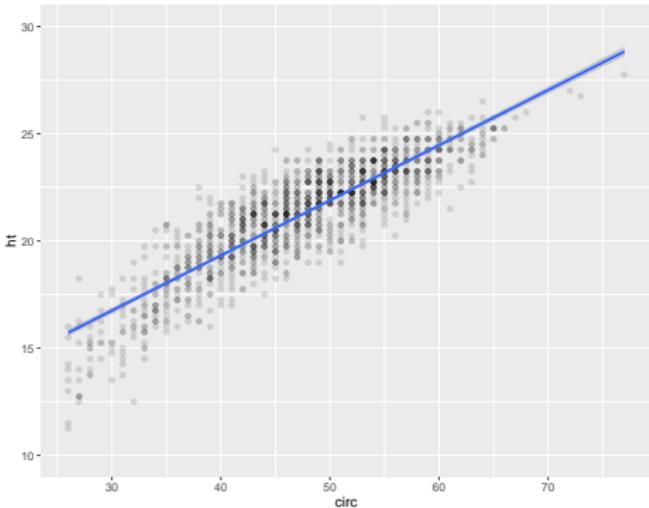
Stat. Superv. Learn.





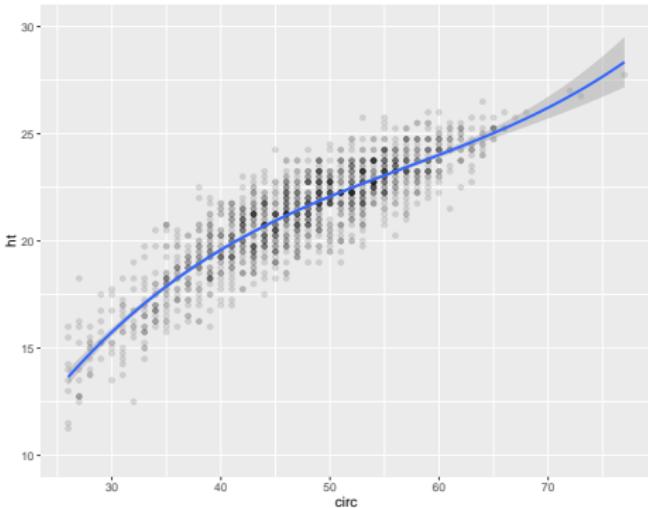
Dataset - P.A. Cornillon

- Real dataset of 1429 eucalyptus obtained by P.A. Cornillon:
 - X: circumference / Y: height
- Can we predict the height from the circumference?



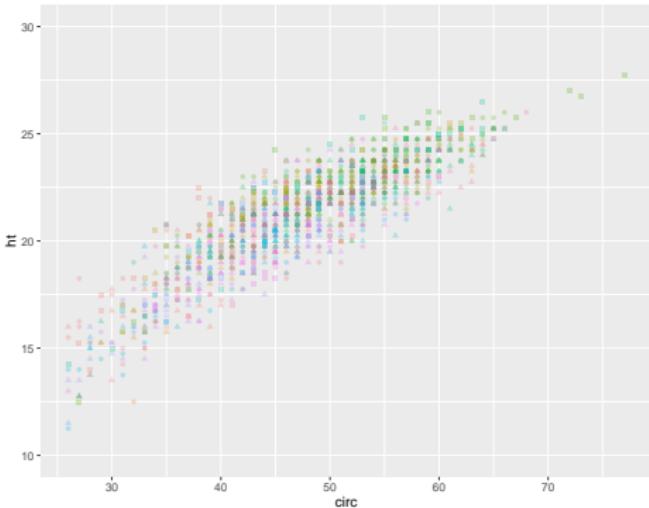
Dataset - P.A. Cornillon

- Real dataset of 1429 eucalyptus obtained by P.A. Cornillon:
 - X: circumference / Y: height
- Can we predict the height from the circumference?
 - by a **line**?



Dataset - P.A. Cornillon

- Real dataset of 1429 eucalyptus obtained by P.A. Cornillon:
 - X: circumference / Y: height
- Can we predict the height from the circumference?
 - by a **line**? by a more complex formula?



Dataset - P.A. Cornillon

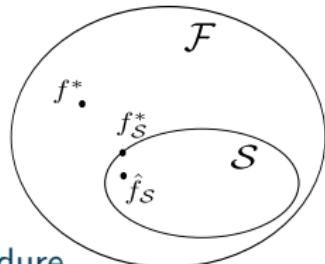
- Real dataset of 1429 eucalyptus obtained by P.A. Cornillon:
 - X: circumference, block, clone / Y: height
- Can we predict the height from the circumference?
 - by a **line**? by a more complex formula?
 - by also taking account of the block and the clone type?

Bias-Variance Dilemma

Stat. Superv. Learn.

- General setting:

- $\mathcal{F} = \{\text{measurable functions } \mathcal{X} \rightarrow \mathcal{Y}\}$
- 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} : $f_S^* = \operatorname{argmin}_{f \in \mathcal{S}} \mathcal{R}(f)$
- Estimate in \mathcal{S} : \hat{f}_S obtained with some procedure



Approximation error and estimation error (Bias/Variance)

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

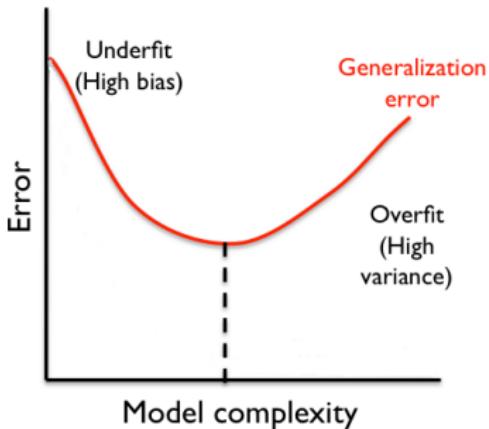
- Approx. error can be large if the model \mathcal{S} is not suitable.
- Estimation error can be large if the model is complex.

Agnostic approach

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

Under-fitting / Over-fitting Issue

Stat. Superv. Learn.



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

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

Statistical Learning Analysis

- Error decomposition:

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

- Bound on the approximation term: approximation theory.
- Probabilistic bound on the estimation term: probability theory!
- **Goal:** Agnostic bounds, i.e. bounds that do not require assumptions on \mathbf{P} ! (Statistical Learning?)
- Often need mild assumptions on \mathbf{P} ... (Nonparametric Statistics?)

Generative and Discriminative PoV

Stat. Superv. Learn.

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

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

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

Problems

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

A Generative 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...

A Discriminative 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

Outline

A Generative POV

1 Introduction

- Supervised Learning
- The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

- Fully Generative Modeling
- Parametric Modeling
- Non Parametric Conditional Estimation

4 A Discriminative Point of View

- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

5 Model Selection

- Models
- Feature Design
- Models, Complexity and Selection

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

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

Bayes Classifier (explicit solution)

- In binary classification with 0 – 1 loss:

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

- In regression with the quadratic loss

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

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

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

Plugin Bayes Classifier

- In binary classification with 0 – 1 loss:

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

- In regression with the quadratic loss

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

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

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

Three main heuristics

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

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

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

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

Theorem

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

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

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

Risk Analysis (Proof)

A Generative POV

- Let us denote $p_1(\mathbf{X}) = \mathbb{P}\{Y = 1|\mathbf{X}\}$
- Step 1: Let $\tilde{f}(\mathbf{X}) = \text{sign}(2\tilde{p}_1(\mathbf{X}) - 1)$

$$\begin{aligned}\mathbb{E}\left[\ell^{0/1}(Y, \tilde{f}(\mathbf{X}))\right] &= \mathbb{E}_{\mathbf{X}}\left[p_1(\mathbf{X})\mathbf{1}_{\tilde{f}(\mathbf{X})=-1} + (1 - p_1(\mathbf{X}))\mathbf{1}_{\tilde{f}(\mathbf{X})=1}\right] \\ &= \mathbb{E}_{\mathbf{X}}\left[(1 - p_1(\mathbf{X})) + (2p_1(\mathbf{X}) - 1)\mathbf{1}_{\tilde{f}(\mathbf{X})=-1}\right]\end{aligned}$$

- Step 2:

$$\begin{aligned}\mathbb{E}\left[\ell^{0/1}(Y, \tilde{f}(\mathbf{X}))\right] - \mathbb{E}\left[\ell^{0/1}(Y, f^*(\mathbf{X}))\right] \\ &= \mathbb{E}_{\mathbf{X}}\left[(2p_1(\mathbf{X}) - 1)(\mathbf{1}_{\tilde{f}(\mathbf{X})=-1} - \mathbf{1}_{f^*(\mathbf{X})=-1})\right]\end{aligned}$$

using the definition of $f^* = \text{sign}(2p(\mathbf{X}) - 1)$

$$= \mathbb{E}_{\mathbf{X}}\left[|2p_1(\mathbf{X}) - 1| \mathbf{1}_{f^*(\mathbf{X}) \neq \tilde{f}(\mathbf{X})}\right]$$

and using the fact that $f^*(\mathbf{X}) \neq \tilde{f}(\mathbf{X})$ implies that $\hat{p}(\mathbf{X})$ and $p(\mathbf{X})$ are not on the same side with respect to 1/2

$$\leq 2\mathbb{E}_{\mathbf{X}}[|p_1(\mathbf{X}) - \hat{p}_1(\mathbf{X})|] = \mathbb{E}_{\mathbf{X}}[\|p(\mathbf{X}) - \hat{p}(\mathbf{X})\|_1]$$

using $\|P - Q\|_1 \leq \sqrt{2\text{KL}(P, Q)}$ and Jensen

$$\leq \mathbb{E}_{\mathbf{X}}\left[\sqrt{2\text{KL}(p(\mathbf{X}), \hat{p}(\mathbf{X}))}\right] \leq (\mathbb{E}_{\mathbf{X}}[2\text{KL}(p(\mathbf{X}), \hat{p}(\mathbf{X}))])^{1/2}$$

Outline

A Generative POV

1 Introduction

- Supervised Learning
- The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

- Fully Generative Modeling
- Parametric Modeling
- Non Parametric Conditional Estimation

4 A Discriminative Point of View

- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

5 Model Selection

- Models
- Feature Design
- Models, Complexity and Selection

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

Bayes formula

- With a slight abuse of notation,

$$\begin{aligned}\mathbb{P}\{Y|\mathbf{X}\} &= \frac{\mathbb{P}\{(\mathbf{X}, Y)\}}{\mathbb{P}\{\mathbf{X}\}} \\ &= \frac{\mathbb{P}\{\mathbf{X}|Y\} \mathbb{P}\{Y\}}{\mathbb{P}\{\mathbf{X}\}}\end{aligned}$$

- **Generative Modeling:**

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

- **Rk:** Require to estimate (\mathbf{X}, Y) rather than only $Y|\mathbf{X}$!

- Great flexibility in the model design but may lead to complex computation.

- Simpler setting in classification!

Bayes formula

$$\mathbb{P}\{Y = k|\mathbf{X}\} = \frac{\mathbb{P}\{\mathbf{X}|Y = k\}\mathbb{P}\{Y = k\}}{\mathbb{P}\{\mathbf{X}\}}$$

- Binary Bayes classifier (the best solution)

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

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

Discriminant Analysis (Gaussian model)

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

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

- Discriminants functions:

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

$$g_k(\mathbf{X}) = -\frac{1}{2}(\mathbf{X} - \mu_k)^t \Sigma_k^{-1} (\mathbf{X} - \mu_k)$$

$$-\frac{d}{2} \ln(2\pi) - \frac{1}{2} \ln(|\Sigma_k|) + \ln(\mathbb{P}\{Y = k\})$$

- 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 practice, 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

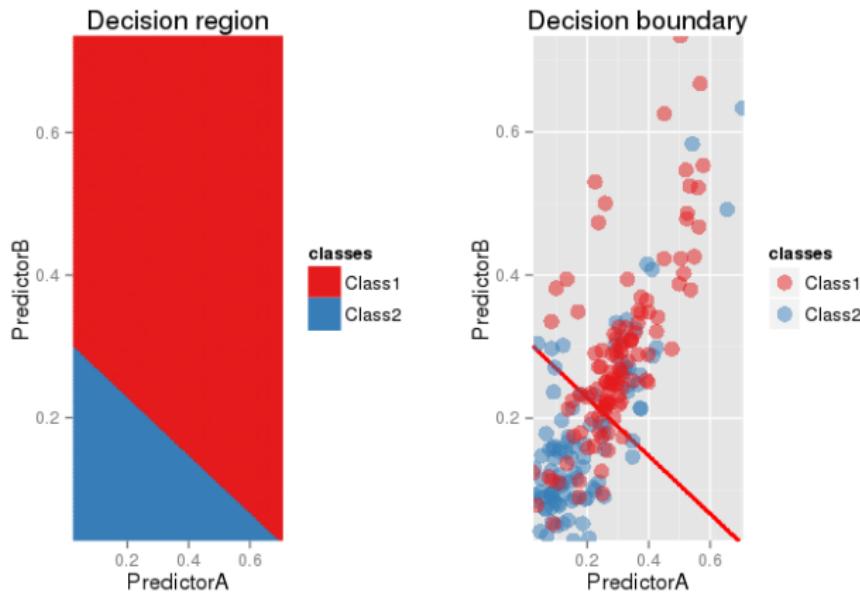
$$\widehat{f}_G(\mathbf{X}) = \begin{cases} +1 & \text{if } \widehat{g}_{+1}(\mathbf{X}) \geq \widehat{g}_{-1}(\mathbf{X}) \\ -1 & \text{otherwise} \end{cases}$$

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

Example: LDA

A Generative POV

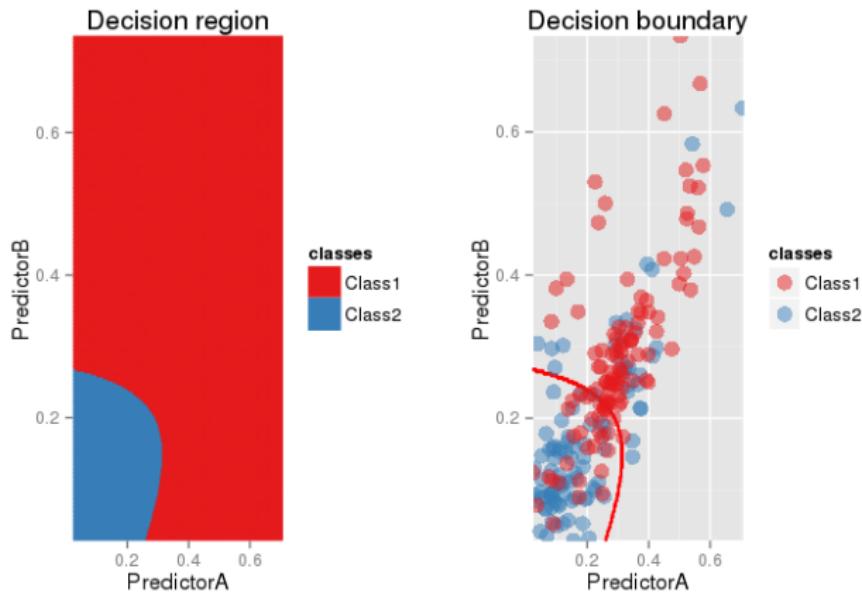
Linear Discriminant Analysis



Example: QDA

A Generative POV

Quadratic Discriminant Analysis



Naive Bayes

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

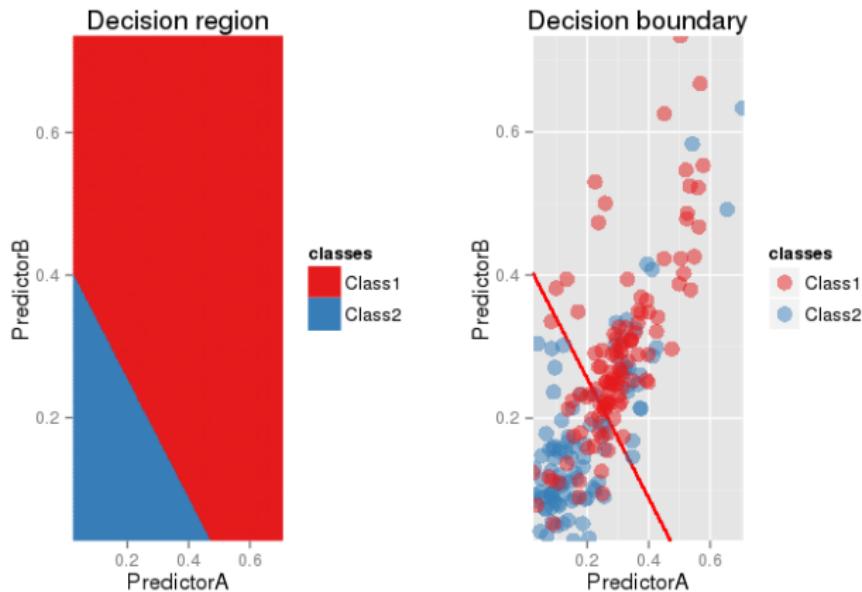
$$\mathbb{P}\{\mathbf{X}|Y\} = \prod_{i=1}^d \mathbb{P}\left\{\mathbf{x}^{(i)} \middle| Y\right\}$$

- 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

A Generative POV

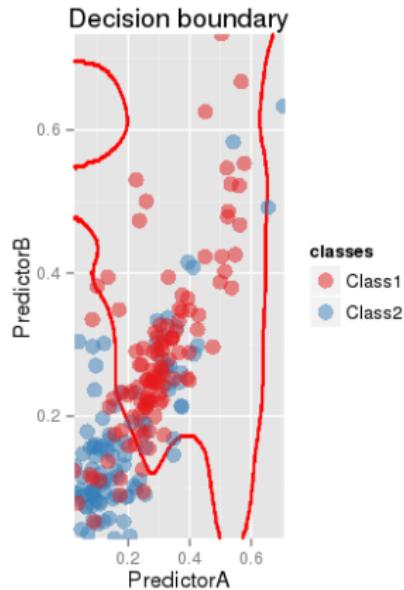
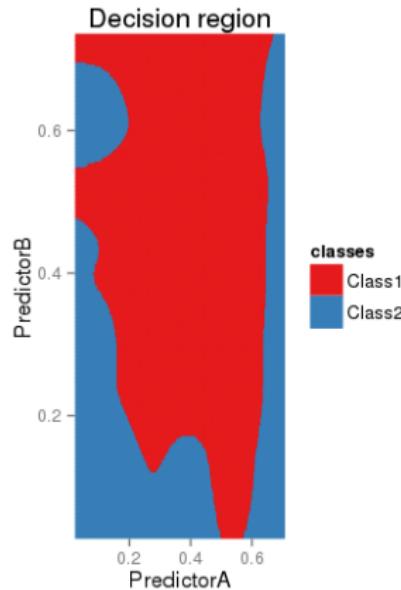
Naive Bayes with Gaussian model



Example: Naive Bayes

A Generative POV

Naive Bayes with kernel density estimates



- Other models of the world!

Bayesian Approach

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

Graphical Models

- Markov type models on Graphs

Gaussian Processes

- Multivariate Gaussian models
- ...

Outline

A Generative POV

1 Introduction

- Supervised Learning
- The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

- Fully Generative Modeling
- **Parametric Modeling**
- Non Parametric Conditional Estimation

4 A Discriminative Point of View

- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

5 Model Selection

- Models
- Feature Design
- Models, Complexity and Selection

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

Maximum Likelihood Approach

- Classical choice for θ :

$$\hat{\theta} = \operatorname{argmin}_{\theta} - \sum_{i=1}^n \log \mathbb{P}_\theta \{ Y_i | \mathbf{X}_i \}$$

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

$$\mathbb{E} [\text{KL}(Y|\mathbf{X}, \mathbb{P}_\theta \{ Y|\mathbf{X} \})]$$

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

Linear Models

- Classical choice: $\theta = (\theta', \varphi)$

$$\mathbb{P}_\theta \{Y|\mathbf{X}\} = \mathbb{P}_{\mathbf{X}^t \beta, \varphi} \{Y\}$$

- Very strong assumption!

- Classical examples:

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

Plugin Linear Discrimination

- Model $\mathbb{P}\{Y = +1|\mathbf{X}\}$ by $h(\beta^T \mathbf{X} + \beta_0)$ with h non decreasing.
- $h(\beta^T \mathbf{X} + \beta_0) > 1/2 \Leftrightarrow \beta^T \mathbf{X} + \beta_0 - h^{-1}(1/2) > 0$
- Linear Classifier: $\text{sign}(\beta^T \mathbf{X} + \beta_0 - h^{-1}(1/2))$

Plugin Linear Classifier Estimation

- Classical choice for h :

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

$$h(t) = F_{\mathcal{N}}(t) \quad \text{probit}$$

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

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

Probabilistic Model

- By construction, $Y|\mathbf{X}$ follows $\mathcal{B}(\mathbb{P}\{Y = +1|\mathbf{X}\})$
- Approximation of $Y|\mathbf{X}$ by $\mathcal{B}(h(\beta^T \mathbf{X} + \beta_0))$
- *Natural* probabilistic choice for β : β minimizing the distance between $\mathcal{B}(h(\mathbf{X}^t \beta))$ and $\mathcal{B}(\mathbb{P}\{Y = 1|\mathbf{X}\})$.

KL Distance

- *Natural* distance: Kullback-Leibler divergence

$$\begin{aligned} & \text{KL}(\mathcal{B}(\mathbb{P}\{Y = 1|\mathbf{X}\}), \mathcal{B}(h(\mathbf{X}^t \beta))) \\ &= \mathbb{E}_{\mathbf{X}} [\text{KL}(\mathcal{B}(\mathbb{P}\{Y = 1|\mathbf{X}\}), \mathcal{B}(h(\mathbf{X}^t \beta)))] \\ &= \mathbb{E}_{\mathbf{X}} \left[\mathbb{P}\{Y = 1|\mathbf{X}\} \log \frac{\mathbb{P}\{Y = 1|\mathbf{X}\}}{h(\mathbf{X}^t \beta)} \right. \\ &\quad \left. + (1 - \mathbb{P}\{Y = 1|\mathbf{X}\}) \log \frac{1 - \mathbb{P}\{Y = 1|\mathbf{X}\}}{1 - h(\mathbf{X}^t \beta)} \right] \end{aligned}$$

log-likelihood

- KL:

$$\text{KL}(\mathcal{B}(\mathbb{P}\{Y = 1|\mathbf{X}\}), \mathcal{B}(h(\mathbf{X}^t \beta)))$$

$$\begin{aligned} &= \mathbb{E}_{\mathbf{X}} \left[\mathbb{P}\{Y = 1|\mathbf{X}\} \log \frac{\mathbb{P}\{Y = 1|\mathbf{X}\}}{h(\mathbf{X}^t \beta)} \right. \\ &\quad \left. + (1 - \mathbb{P}\{Y = 1|\mathbf{X}\}) \log \frac{1 - \mathbb{P}\{Y = 1|\mathbf{X}\}}{1 - h(\mathbf{X}^t \beta)} \right] \end{aligned}$$

$$\begin{aligned} &= \mathbb{E}_{\mathbf{X}} [-\mathbb{P}\{Y = 1|\mathbf{X}\} \log(h(\mathbf{X}^t \beta)) \\ &\quad - (1 - \mathbb{P}\{Y = 1|\mathbf{X}\}) \log(1 - h(\mathbf{X}^t \beta))] + C_{\mathbf{X}, Y} \end{aligned}$$

- Empirical counterpart = opposite of the log-likelihood:

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

- Minimization of possible if h is regular...

Logistic Regression and Odd

- Logistic model: $h(t) = \frac{e^t}{1+e^t}$ (most *natural* choice...)
- The Bernoulli law $\mathcal{B}(h(t))$ satisfies then

$$\frac{\mathbb{P}\{Y=1\}}{\mathbb{P}\{Y=-1\}} = e^t \Leftrightarrow \log \frac{\mathbb{P}\{Y=1\}}{\mathbb{P}\{Y=-1\}} = t$$

- Interpretation in term of odd.
- Logistic model: linear model on the logarithm of the odd.

Associated Classifier

- Plugin strategy:

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

Likelihood Rewriting

- Opposite of the log-likelihood:

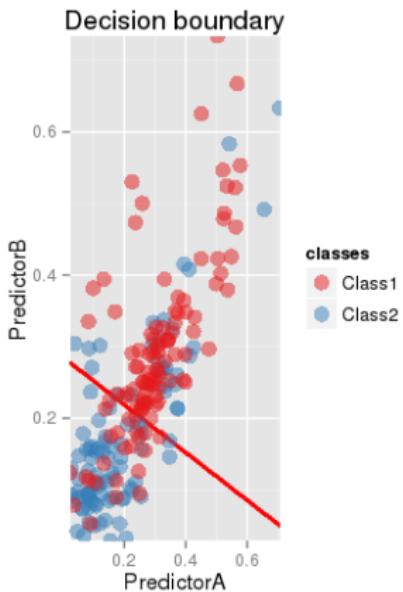
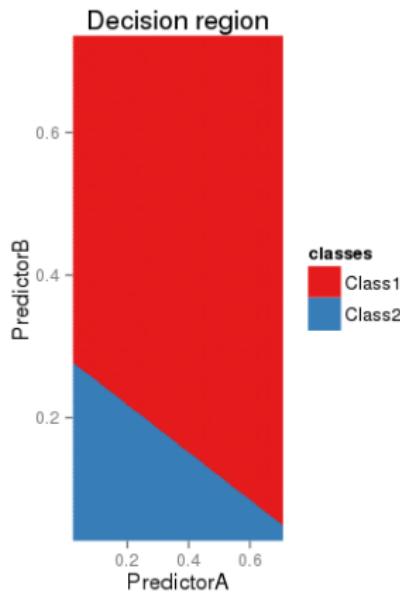
$$\begin{aligned} & -\frac{1}{n} \sum_{i=1}^n (\mathbf{1}_{Y_i=1} \log(h(\mathbf{x}_i^t \beta)) + \mathbf{1}_{Y_i=-1} \log(1 - h(\mathbf{x}_i^t \beta))) \\ & = -\frac{1}{n} \sum_{i=1}^n \left(\mathbf{1}_{Y_i=1} \log \frac{e^{\mathbf{x}_i^t \beta}}{1 + e^{\mathbf{x}_i^t \beta}} + \mathbf{1}_{Y_i=-1} \log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

- Convex and smooth function of β
- Easy optimization.

Example: Logistic

A Generative POV

Logistic



Transformed Representation

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

$$f_{\beta}(\mathbf{X}) = \Phi(\mathbf{X})^t \beta$$

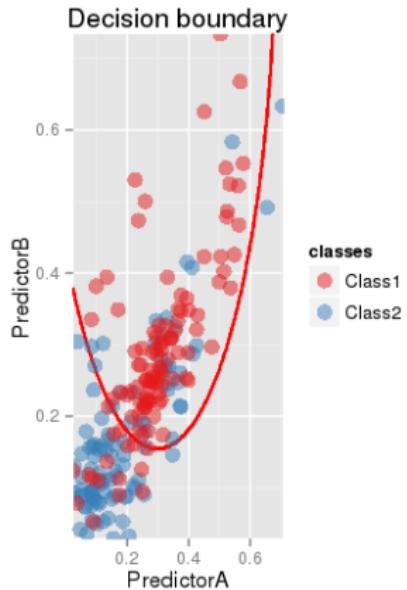
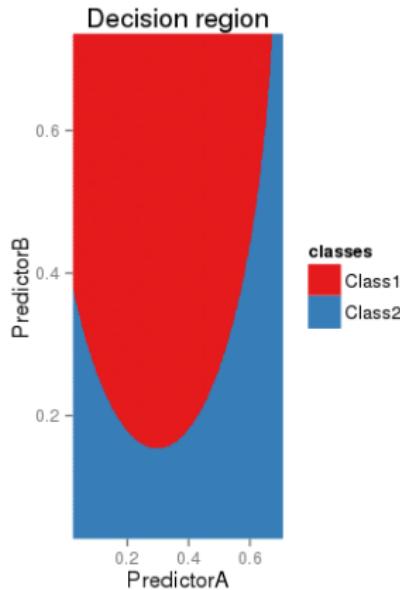
Feature Design

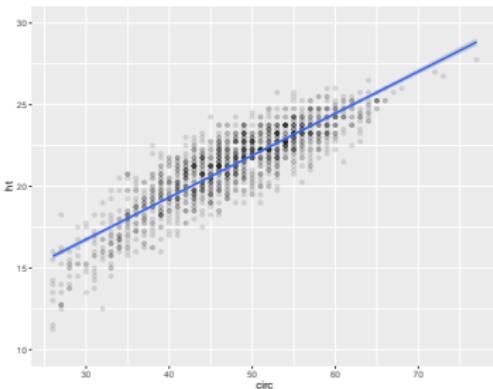
- Art of choosing Φ .
- Examples:
 - Renormalization, (domain specific) transform
 - Basis decomposition
 - Interaction between different variables...

Example: Quadratic Logistic

A Generative POV

Quadratic Logistic





Gaussian Linear Model

- **Model:** $Y|\mathbf{X} \sim \mathcal{N}(\mathbf{X}^t \boldsymbol{\beta}, \sigma^2)$ plus independence
- Probably the most classical model of all time!
- Maximum Likelihood with explicit formulas for the two parameters.
- In regression, estimation of $\mathbb{E}[Y|\mathbf{X}]$ is sufficient: other/no model for the noise possible.

Generalized Linear Model

- Model entirely characterized by its mean (up to a scalar nuisance parameter) ($v(\mathbb{E}_\theta[Y]) = \theta$ with v invertible).
- Exponential family: Probability law family P_θ such that the density can be written

$$f(y, \theta, \varphi) = e^{\frac{y\theta - v(\theta)}{\varphi} + w(y, \varphi)}$$

where φ is a nuisance parameter and w a function independent of θ .

- Examples:
 - Gaussian: $f(y, \theta, \varphi) = e^{\frac{y\theta - \theta^2/2}{\varphi} + \frac{y^2/2}} \varphi^{-1}$
 - Bernoulli: $f(y, \theta) = e^{y\theta - \ln(1+e^\theta)} (\theta = \ln p/(1-p))$
 - Poisson: $f(y, \theta) = e^{(y\theta - e^\theta) + \ln(y!)} (\theta = \ln \lambda)$
- Linear Conditional model: $Y|\mathbf{X} \sim P_{\mathbf{X}^t \beta}$
- ML fit of the parameters

Outline

A Generative POV

1 Introduction

- Supervised Learning
- The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

- Fully Generative Modeling
- Parametric Modeling
- Non Parametric Conditional Estimation**

4 A Discriminative Point of View

- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

5 Model Selection

- Models
- Feature Design
- Models, Complexity and Selection

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

Non Parametric Conditional Estimation

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

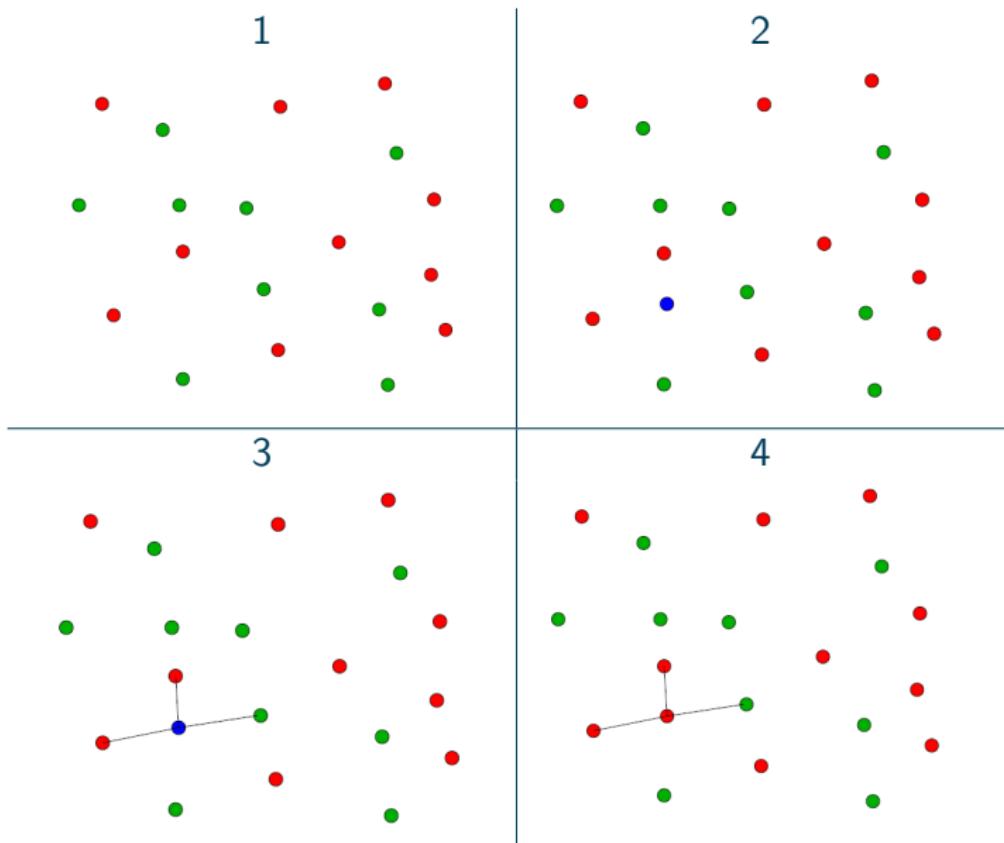
- **Idea:** The behavior of $Y|\mathbf{X}$ is locally *constant* or simple!

Kernel

- Choose a kernel K (think of a weighted neighborhood).
 - For each $\tilde{\mathbf{X}}$, compute a simple localized estimate of $Y|\mathbf{X}$
 - Use this local estimate to take the decision
-
- In regression, estimation of $\mathbb{E}[Y|\mathbf{X}]$ is sufficient.

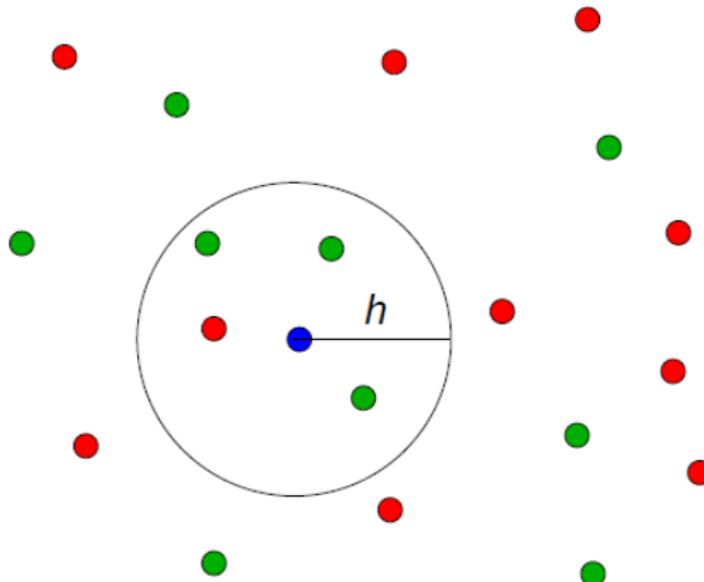
Example: k Nearest-Neighbors ($k = 3$)

A Generative POV



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

A Generative POV



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

k -NN as local conditional density estimate

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

- KNN Classifier:

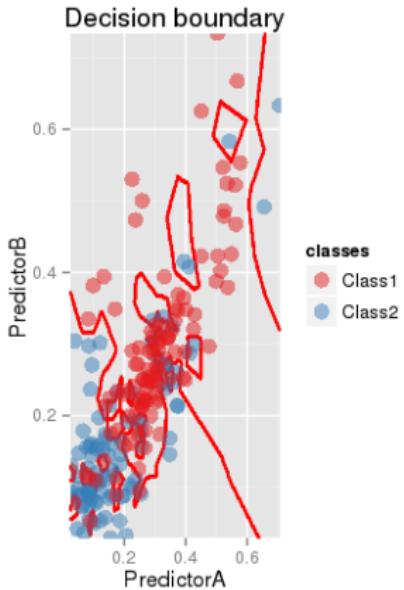
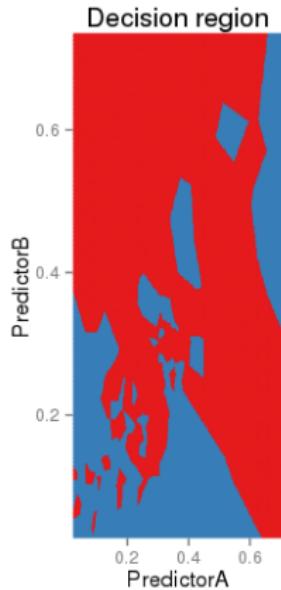
$$\widehat{f}_{KNN}(\mathbf{X}) = \begin{cases} +1 & \text{if } \mathbb{P}\{\widehat{Y=1}|\mathbf{X}\} \geq \mathbb{P}\{\widehat{Y=-1}|\mathbf{X}\} \\ -1 & \text{otherwise} \end{cases}$$

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

Example: KNN

A Generative POV

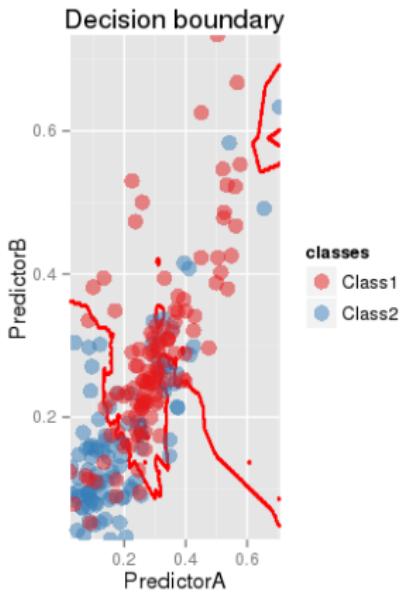
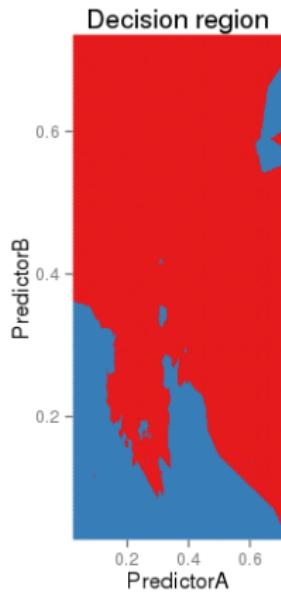
k-NN with k=1



Example: KNN

A Generative POV

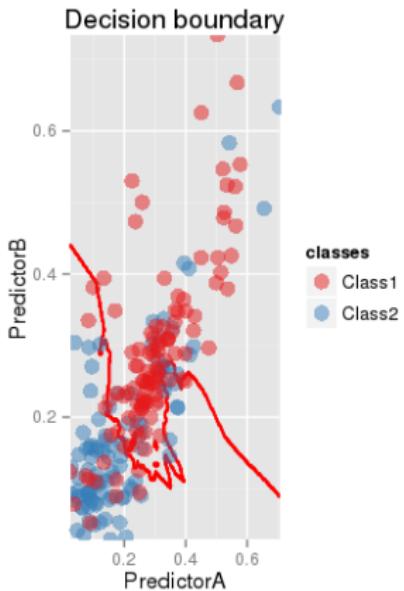
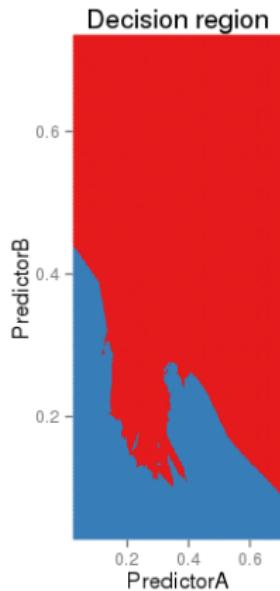
k-NN with k=5



Example: KNN

A Generative POV

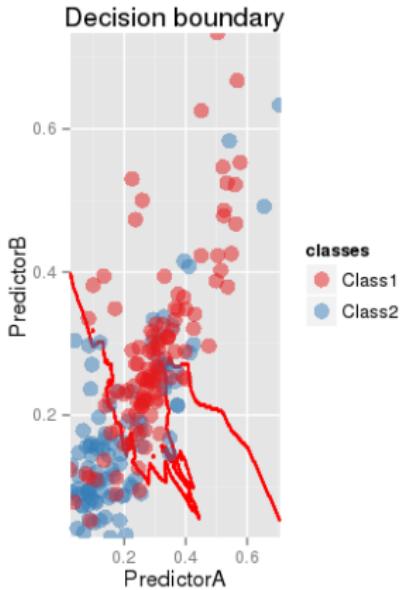
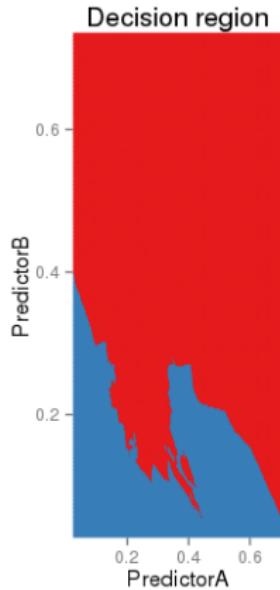
k-NN with k=9



Example: KNN

A Generative POV

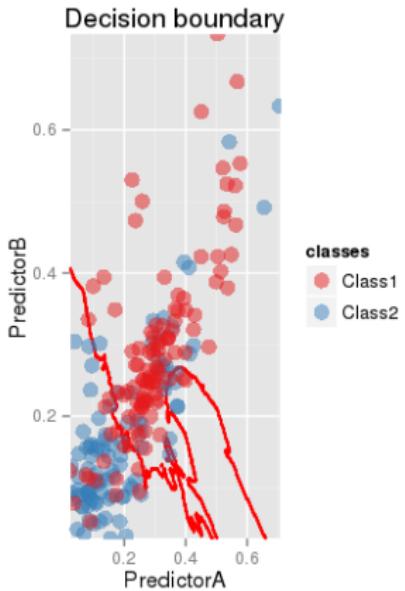
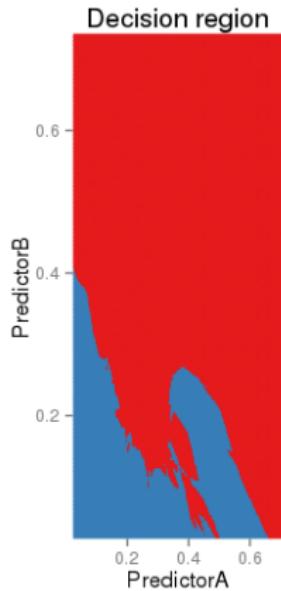
k-NN with k=13



Example: KNN

A Generative POV

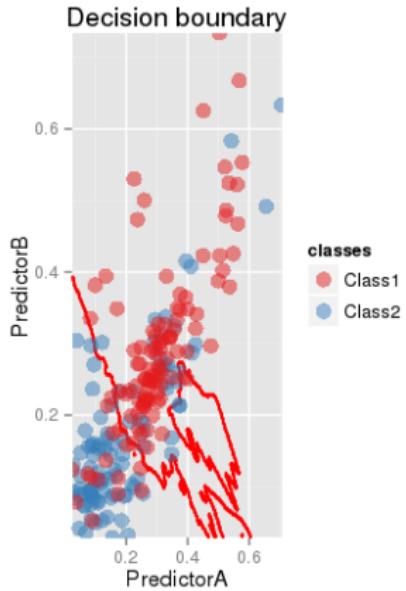
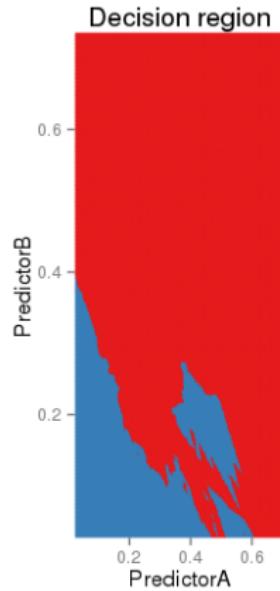
k-NN with k=17



Example: KNN

A Generative POV

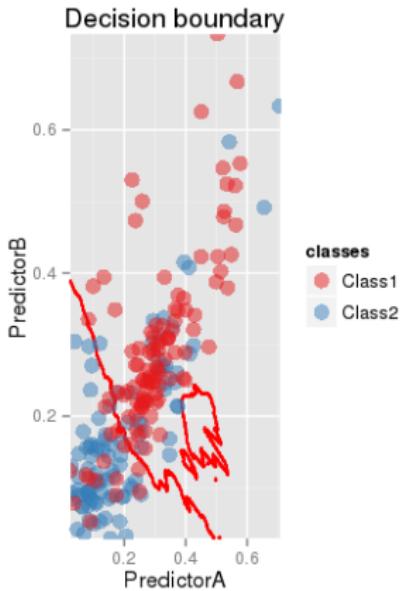
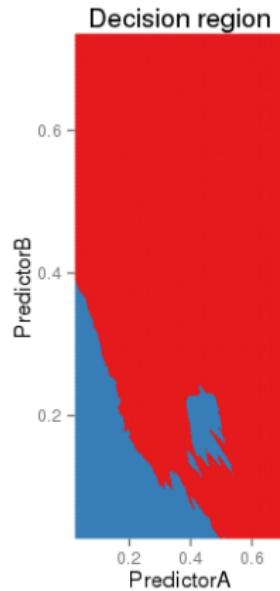
k-NN with k=21



Example: KNN

A Generative POV

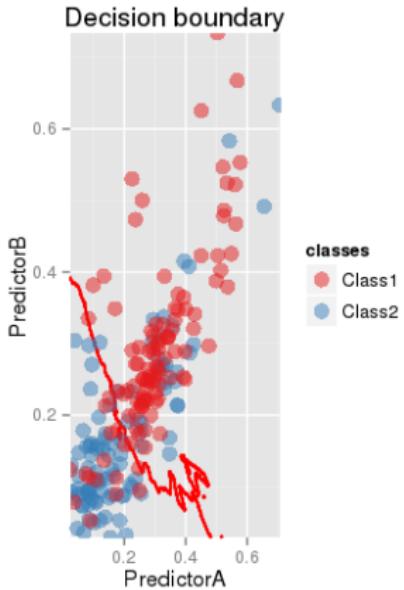
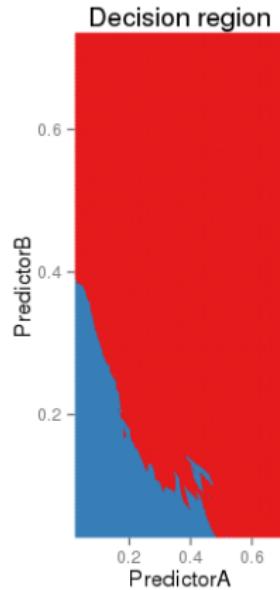
k-NN with k=25



Example: KNN

A Generative POV

k-NN with k=29



A naive idea

- $\mathbb{E}[Y|\mathbf{X}]$ can be approximated by a local average:

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

where $\mathcal{B}(\mathbf{X})$ is a neighborhood of \mathbf{X} .

- Heuristic:

- On the one hand, if $\mathbf{X} \rightarrow \mathbb{E}[Y|\mathbf{X}]$ is regular then

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

- On the other hand,

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

Neighborhood and Size

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

Bandwidth

- Heuristic:
 - A large bandwidth ensures that the average is taken on many samples and thus the variance is small...
 - A small bandwidth is thus that the approximation $\mathbb{E}[Y|\mathbf{X}] \simeq \mathbb{E}[Y|\mathbf{X}' \in \mathcal{N}(\mathbf{X})]$ is more accurate.

Weighted Local Average

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

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

Kernel

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

Density Estimation

- How to estimate the density p of \mathbf{X} with respect to the Lebesgue measure from an i.i.d. sample $(\mathbf{X}_1, \dots, \mathbf{X}_n)$.
- Parametric approach: assume that the density has a known parametrized shape and estimate those parameters.
- Nonparametric approach: do not assume that the density has a known parametrized shape and
 - Approximate it by a parametric one, whose parameters can be estimated
 - Estimate directly the density
- Important nonparametric statistic topic!

Kernel Density Estimation (Parzen)

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

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

- If $K = \frac{1}{Z_h} \mathbf{1}_{\|\mathbf{t}\| \leq h}$ this can be easily interpreted in term of local empirical density of samples!
- General K corresponds to the same smoothing idea we have used before.
- We will often use $K_h(t) = \frac{1}{h^d} K(t/h)$ and let

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

Bandwidth choice

- A small h leads to a small bias but a large variance...
- A large h leads to a small variance but a large bias...
- Theoretical analysis possible!

Nadaraya-Watson Heuristic

- Provided all the densities exist

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

- Replace the unknown densities by their estimates:

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

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

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

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

Nadaraya-Watson

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

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

- Same local weighted average estimator!

Bandwidth Choice

- Bandwidth h of K allows to balance between bias and variance.
- Similar but more complex theoretical analysis of the error is possible.
- Same conclusion: the smoother the densities the easier the estimation but the optimal bandwidth depends on the unknown regularity!

Another Point of View on Kernel

- Nadaraya-Watson estimator:

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

- Can be viewed as a minimizer of

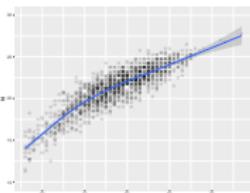
$$\sum_{i=1}^n |Y_i - \beta|^2 K_h(\mathbf{X} - \mathbf{X}_i)$$

- Local regression of order 0!

Local Linear Model

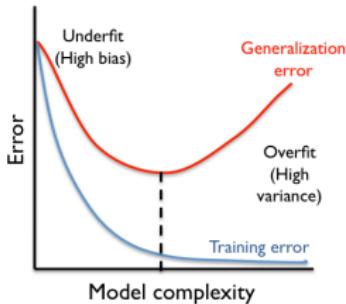
- Estimate $\mathbb{E}[Y|\mathbf{X}]$ by $\hat{f}(\mathbf{X}) = \langle \hat{\beta}(\mathbf{X}), \phi(\mathbf{X}) \rangle$ where ϕ is any function of \mathbf{X} and $\hat{\beta}(\mathbf{X})$ is the minimizer of

$$\sum_{i=1}^n |Y_i - \langle \beta, \phi(\mathbf{X}_i) \rangle|^2 K_h(\mathbf{X} - \mathbf{X}_i).$$



1D Nonparametric Regression

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



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!

Two Approaches

- **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}

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

and choose the model with the smallest penalized risk.

Which loss to use?

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



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

Cross Validation

- Use $(1 - \epsilon)n$ observations to train and ϵn to verify!
- Validation for a learning set of size $(1 - \epsilon) \times n$ instead of n !
- Unstable error estimate if ϵn is too small ?
- Most classical variations:
 - Hold Out,
 - Leave One Out,
 - V -fold cross validation.

Principle

- Split the dataset \mathcal{D} in 2 sets $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$ of size $n(1 - \epsilon)$ and $n\epsilon$.
- Learn \hat{f}^{HO} from the subset $\mathcal{D}_{\text{train}}$.
- Compute the empirical error on the subset $\mathcal{D}_{\text{test}}$:

$$\mathcal{R}_n^{HO}(\hat{f}^{HO}) = \frac{1}{n\epsilon} \sum_{(\mathbf{x}_i, Y_i) \in \mathcal{D}_{\text{test}}} |Y_i - \hat{f}^{HO}(\mathbf{x}_i)|^2$$

Model Selection by Cross Validation

- Compute $\mathcal{R}_n^{HO}(\hat{f}_S^{HO})$ for all possible models \mathcal{S} ,
 - Select the model with the smallest CV error,
 - Reestimate the \hat{f}_S with all the data.
-
- Does not take into account the variability of $\mathcal{R}_n^{HO}(\hat{f}^{HO})$ (\hat{f} and error estimate)
 - Biased toward simpler model as the estimation does not use



Principle

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

$$\mathcal{R}_n^{-v}(\hat{f}^{-v}) = \frac{1}{n_v} \sum_{(\mathbf{x}_i, Y_i) \in \mathcal{D}_v} |Y_i - \hat{f}^{-v}(\mathbf{x}_i)|^2$$

- Compute the average empirical error:

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

- Leave One Out : $V = n$.

Analysis (when n is a multiple of V)

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

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

$$\mathbb{V} [\mathcal{R}_n^{CV}(\hat{f})] = \frac{1}{V} \mathbb{V} [\mathcal{R}_n^{-v}(\hat{f}^{-v})]$$

$$+ (1 - \frac{1}{V}) \text{Cov} [\mathcal{R}_n^{-v}(\hat{f}^{-v}), \mathcal{R}_n^{-v'}(\hat{f}^{-v'})]$$

- Average risk for a sample of size $(1 - \frac{1}{V})n$.
- Variance term much more complex to analyze!
- Fine analysis shows that the larger V the better...
- Accuracy/Speed tradeoff: $V = 5$ or $V = 10$!

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

Gaussian Interval (Quasi independence assumption)

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

PAC approach (Small risk estimation error assumption)

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



- **Very simple idea:** use a second learning/verification set to compute a verification error.
- Sufficient to remove the dependency issue!

Cross Validation

- Use $(1 - \epsilon)n$ observations to train and ϵn to verify!
- Validation for a learning set of size $(1 - \epsilon) \times n$ instead of n !
- Unstable error estimate if ϵn is too small ?
- Most classical variations:
 - Leave One Out,
 - V -fold cross validation.



Principle

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

$$R_n^{-v}(\hat{f}^{-v}) = \frac{1}{n_v} \sum_{(\mathbf{x}_i, Y_i) \in \mathcal{D}_v} \ell(Y_i, \hat{f}^{-v}(\mathbf{x}_i))$$

- Compute the average empirical error:

$$R_n^{CV}(\hat{f}) = \frac{1}{V} \sum_{v=1}^V R_n^{-v}(\hat{f}^{-v})$$

Analysis (when n is a multiple of V)

- The $R_n^{-v}(\hat{f}^{-v})$ are identically distributed variable but are not independent!
- Consequence:

$$\mathbb{E} [R_n^{CV}(\hat{f})] = \mathbb{E} [R_n^{-v}(\hat{f}^{-v})]$$

$$\mathbb{V} [R_n^{CV}(\hat{f})] = \frac{1}{V} \mathbb{V} [R_n^{-v}(\hat{f}^{-v})]$$

$$+ (1 - \frac{1}{V}) \text{Cov} [R_n^{-v}(\hat{f}^{-v}), R_n^{-v'}(\hat{f}^{-v'})]$$

- Average risk for a sample of size $(1 - \frac{1}{V})n$.
- Variance term much more complex to analyse!
- Fine analysis shows that the larger V the better...
- Accuracy/Speed tradeoff: $V = 5$ or $V = 10$!

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

A fast LOO formula for the linear regression

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

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

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

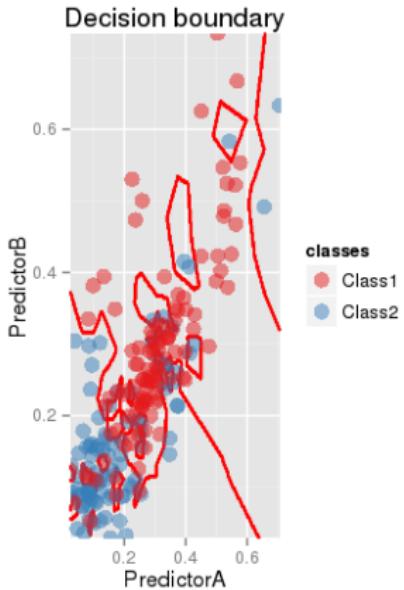
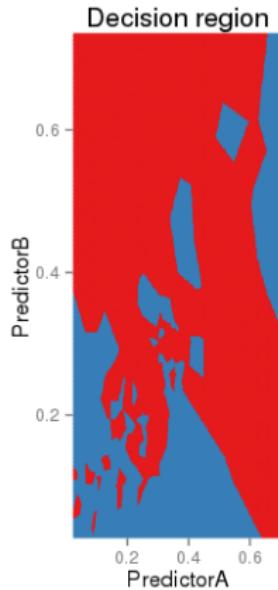
- Proof based on linear algebra!
- Leads to a fast formula for LOO:

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

Example: KNN

A Generative POV

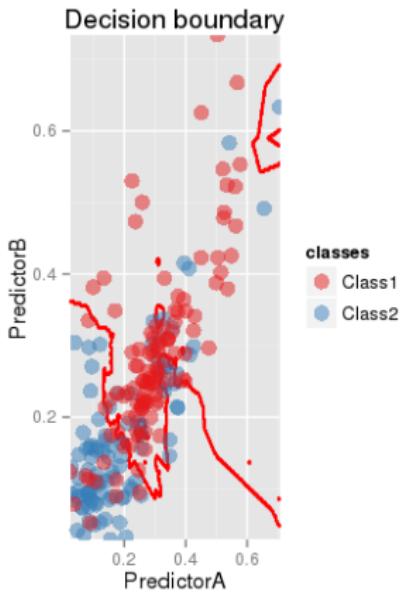
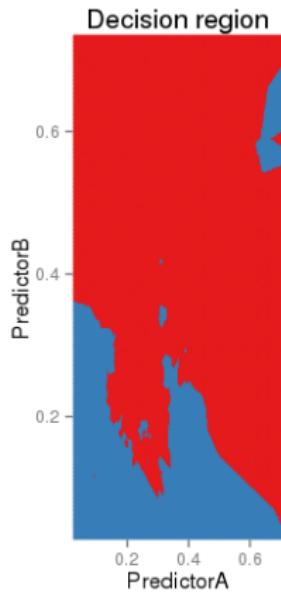
k-NN with k=1



Example: KNN

A Generative POV

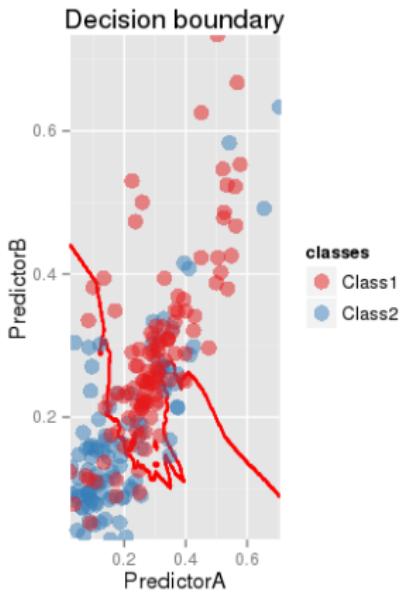
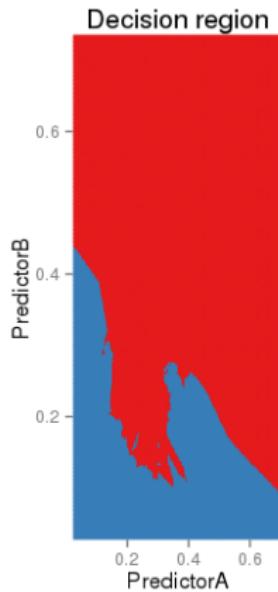
k-NN with k=5



Example: KNN

A Generative POV

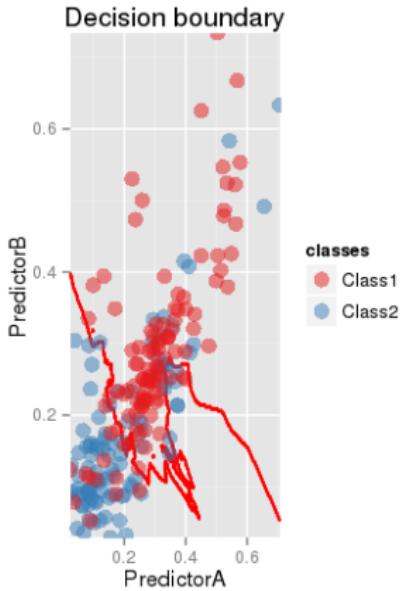
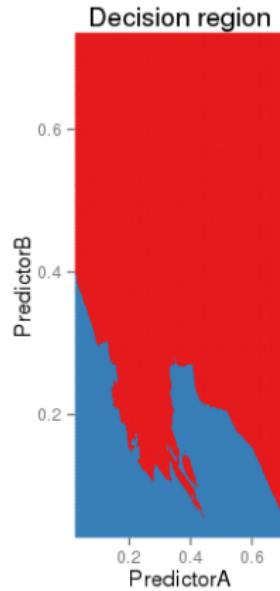
k-NN with k=9



Example: KNN

A Generative POV

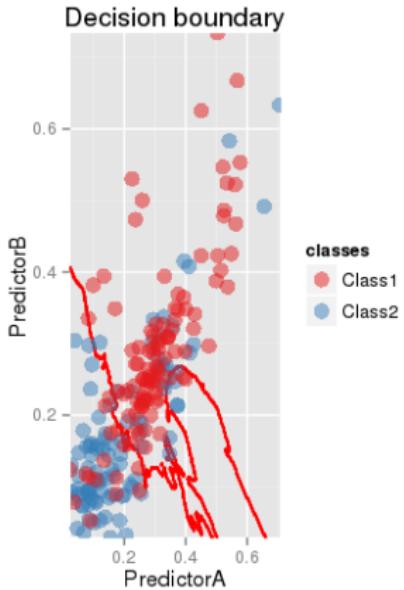
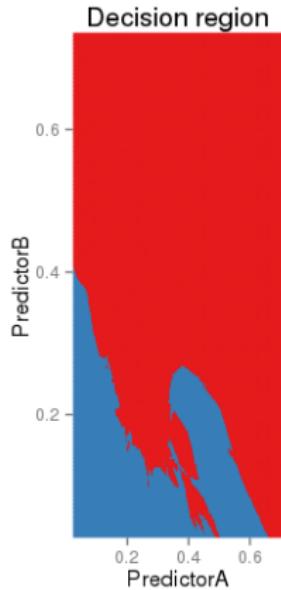
k-NN with k=13



Example: KNN

A Generative POV

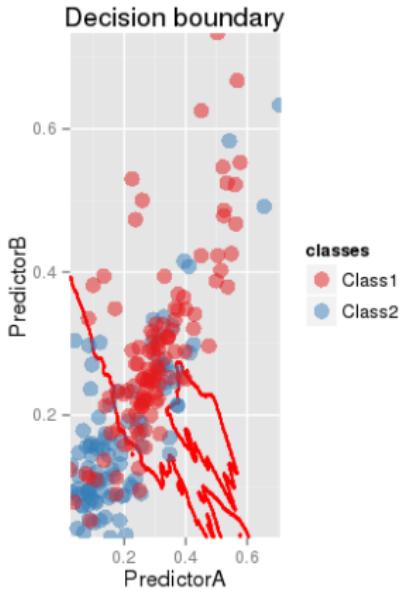
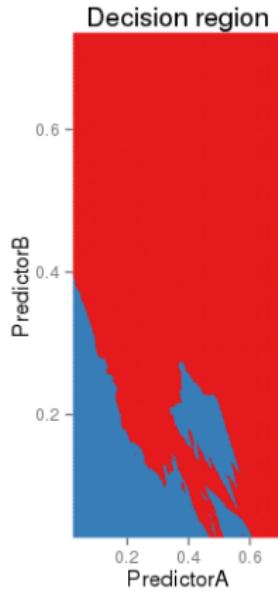
k-NN with k=17



Example: KNN

A Generative POV

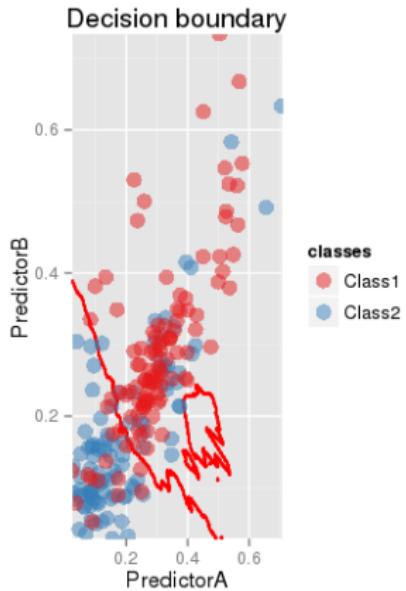
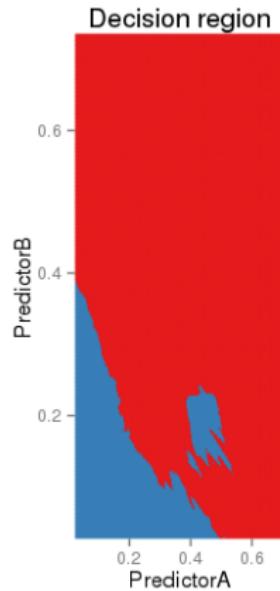
k-NN with k=21



Example: KNN

A Generative POV

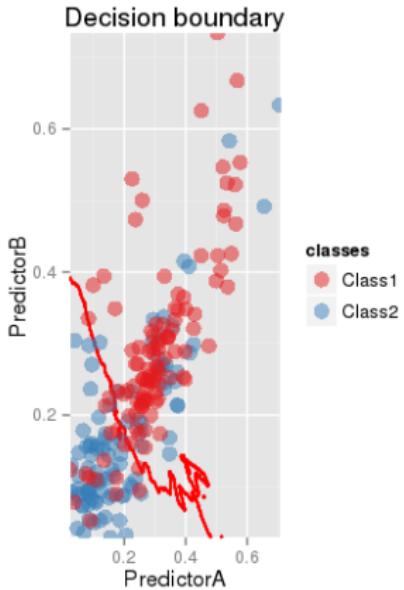
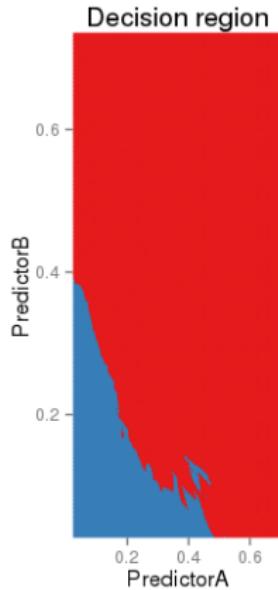
k-NN with k=25



Example: KNN

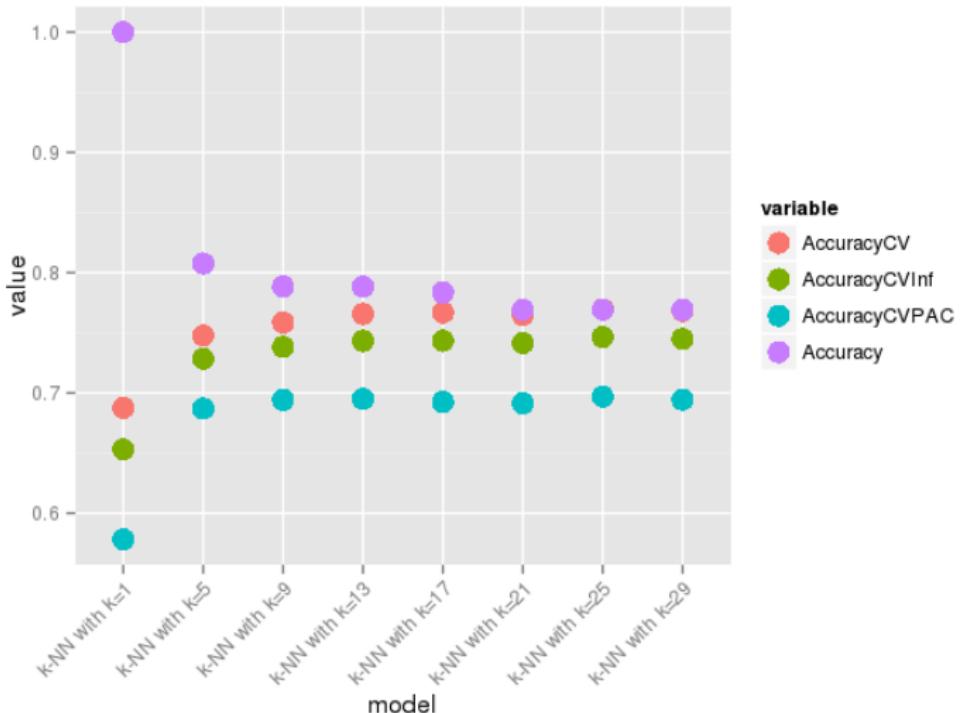
A Generative POV

k-NN with k=29



Cross Validation

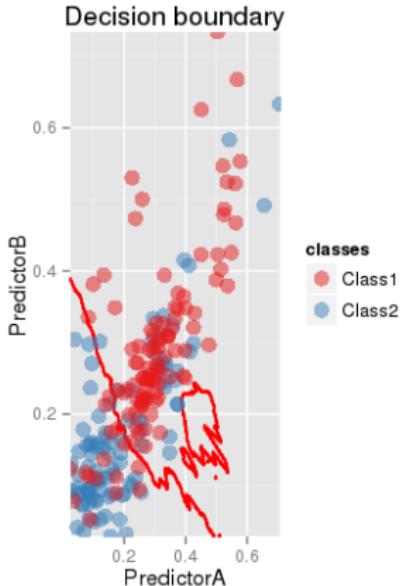
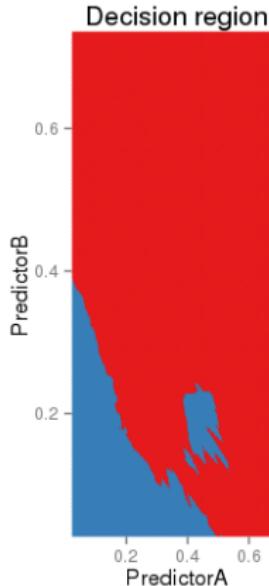
A Generative POV



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

A Generative POV

k-NN with k=25



Outline

A Discriminative PoV

- 1 Introduction
 - Supervised Learning
 - The Example of Univariate Linear Regression
- 2 Statistical Supervised Learning
- 3 A Generative Point of View
 - Fully Generative Modeling
 - Parametric Modeling
 - Non Parametric Conditional Estimation
- 4 A Discriminative Point of View
 - Convexification of the Risk
 - SVM
 - Penalization
 - (Deep) Neural Networks
 - Tree Based Methods
- 5 Model Selection
 - Models
 - Feature Design
 - Models, Complexity and Selection

Statistical and Optimization Framework

A Discriminative PoV

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

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

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

Problems

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

A Generative 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...

A Discriminative 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

Outline

A Discriminative PoV

1 Introduction

- Supervised Learning
- The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

- Fully Generative Modeling
- Parametric Modeling
- Non Parametric Conditional Estimation

4 A Discriminative Point of View

- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

5 Model Selection

- Models
- Feature Design
- Models, Complexity and Selection

- The best solution f^* is the one minimizing

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

Empirical Risk Minimization

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

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

- Intractable for the $\ell^{0/1}$ loss!

Risk Convexification

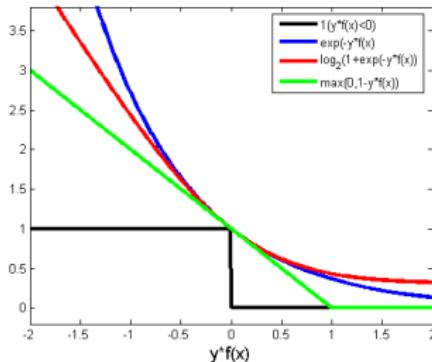
- Replace the loss $\ell(Y, f_\theta(\mathbf{X}))$ by a convex upperbound $\ell'(Y, f_\theta(\mathbf{X}))$ (surrogate loss).
- Minimize the average of the surrogate empirical loss

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

- Use $\hat{f} = \text{sign}(\tilde{f})$
- Much easier optimization.

Instantiation

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

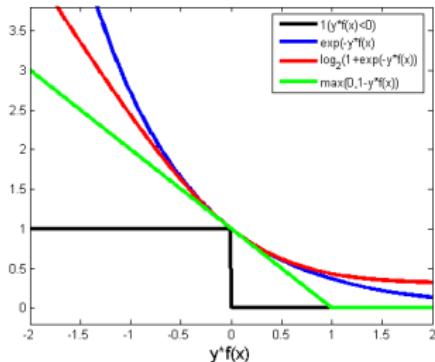


Convexification

- Replace the loss $\ell^{0/1}(Y, f(\mathbf{X}))$ by
$$\ell'(Y, f(\mathbf{X})) = l(Y_i f(\mathbf{X}))$$
with l a convex function.
- Further mild assumption:** l is decreasing, differentiable at 0 and $l'(0) < 0$.

Classification Loss and Convexification

A Discriminative PoV



Classical convexification

- Logistic loss: $\ell'(Y, f(\mathbf{X})) = \log(1 + e^{-Yf(\mathbf{X})})$ (Logistic / NN)
- Hinge loss: $\ell'(Y, f(\mathbf{X})) = (1 - Yf(\mathbf{X}))_+$ (SVM)
- Exponential loss: $\ell'(Y, f(\mathbf{X})) = e^{-Yf(\mathbf{X})}$ (Boosting...)

The Target is the Bayes Classifier

- The minimizer of

$$\mathbb{E} [\ell'(Y, f(\mathbf{X}))] = \mathbb{E} [I(Yf(\mathbf{X}))]$$

is the Bayes classifier $f^* = \text{sign}(2\eta(\mathbf{X}) - 1)$

Control of the Excess Risk

- It exists a convex function Ψ such that

$$\begin{aligned}\Psi \left(\mathbb{E} [\ell^{0/1}(Y, \text{sign}(f(\mathbf{X})))] - \mathbb{E} [\ell^{0/1}(Y, f^*(\mathbf{X}))] \right) \\ \leq \mathbb{E} [\ell'(Y, f(\mathbf{X}))] - \mathbb{E} [\ell'(Y, f^*(\mathbf{X}))]\end{aligned}$$

- Theoretical guarantee!

- By definition,

$$\mathbb{E}[I(Yf)|\mathbf{X}] = \eta(\mathbf{X})I(f) + (1 - \eta(\mathbf{X}))I(-f)$$

- Let $H(f, \eta) = \eta I(f) + (1 - \eta)I(-f)$, the optimal value for \tilde{f} satisfies $\delta H(\tilde{f}, \eta) = -\eta \delta I(\tilde{f}) + (1 - \eta) \delta I(-\tilde{f}) \ni 0$.
- With a slight abuse of notation, we denote by $\delta I(\tilde{f})$ and $\delta I(-\tilde{f})$ the two subgradients such that

$$\eta \delta I(\tilde{f}) - (1 - \eta) \delta I(-\tilde{f}) = 0$$

- Now we discuss the sign of \tilde{f} :
 - If $\tilde{f} > 0$, $\delta I(-\tilde{f}) < \delta I(\tilde{f})$ and thus $\eta > (1 - \eta)$, i.e. $2\eta - 1 > 0$.
 - Conversely, if $\tilde{f} < 0$ then $2\eta - 1 < 0$
- Thus $\text{sign}(\tilde{f}) = \text{sign}(2\eta - 1)$ i.e. the minimizer of $\mathbb{E}[I(yf)|\mathbf{X}]$ is $f^*(\mathbf{X}) = \text{sign}(2\eta(\mathbf{X}) - 1)$

- We define $H(\eta) = \inf_f H(f, \eta) = \inf_f (\eta I(f) + (1 - \eta)I(-f))$. By construction, H is a concave function satisfying $H(1/2 + x) = H(1/2 - x)$.
- Furthermore, one verify that if we consider the minimum over the *wrong sign classifiers*, $\inf_{f, f(2\eta-1) < 0} H(f, \eta) = I(0)$.
- Indeed,

$$\begin{aligned}
 & \inf_{f, f(2\eta-1) < 0} H(f, \eta) \\
 &= \inf_{f, f(2\eta-1) < 0} (\eta I(f) + (1 - \eta)I(-f)) \\
 &\geq \inf_{f, f(2\eta-1) < 0} (\eta(I(0) + I'(0)f) + (1 - \eta)(I(0) - I'(0)f)) \\
 &\geq I(0) + \inf_{f, f(2\eta-1) < 0} I'(0)f(2\eta - 1) = I(0)
 \end{aligned}$$

- Furthermore,

$$\begin{aligned}
 \mathbb{E} [\ell'(Y, f(\mathbf{X}))] &= \mathbb{E}_{\mathbf{X}} [H(f, \eta(\mathbf{X}))] \\
 \mathbb{E} [\ell'(Y, f^*(\mathbf{X}))] &= \mathbb{E}_{\mathbf{X}} [H(\eta(\mathbf{X}))]
 \end{aligned}$$

- We define then

$$\Psi(\theta) = I(0) - H\left(\frac{1+\theta}{2}\right)$$

which is thus a convex function satisfying $\Psi(0) = 0$ and $\Psi(\theta) > 0$ for $\theta > 0$.

- Recall that

$$\begin{aligned} & \mathbb{E} [\ell^{0/1}(Y, \text{sign}(f(\mathbf{X})))] - \mathbb{E} [\ell^{0/1}(Y, f^*(\mathbf{X}))] \\ &= \mathbb{E}_{\mathbf{X}} [|2\eta(\mathbf{X}) - 1| \mathbf{1}_{f^*(\mathbf{X}) \neq \text{sign}(f(\mathbf{X}))}] \end{aligned}$$

- Using Jensen inequality, we derive

$$\begin{aligned} & \Psi \left(\mathbb{E} [\ell^{0/1}(Y, \text{sign}(f(\mathbf{X})))] - \mathbb{E} [\ell^{0/1}(Y, f^*(\mathbf{X}))] \right) \\ &\leq \mathbb{E}_{\mathbf{X}} \left[\Psi \left(|2\eta(\mathbf{X}) - 1| \mathbf{1}_{f^*(\mathbf{X}) \neq \text{sign}(f(\mathbf{X}))} \right) \right] \end{aligned}$$

- Using $\Psi(0) = 0$ and the symmetry of H ,

$$\begin{aligned} & \Psi \left(\mathbb{E} [\ell^{0/1}(Y, \text{sign}(f(\mathbf{X})))] - \mathbb{E} [\ell^{0/1}(Y, f^*(\mathbf{X}))] \right) \\ &\leq \mathbb{E}_{\mathbf{X}} \left[\left(I(0) - H \left(\left(\frac{1 + |2\eta(\mathbf{X}) - 1|}{2} \right) \right) \right) \mathbf{1}_{f^*(\mathbf{X}) \neq \text{sign}(f(\mathbf{X}))} \right] \\ &\leq \mathbb{E}_{\mathbf{X}} \left[(I(0) - H(\eta(\mathbf{X}))) \mathbf{1}_{f^*(\mathbf{X}) \neq \text{sign}(f(\mathbf{X}))} \right] \\ &\leq \mathbb{E}_{\mathbf{X}} \left[(I(0) - H(\eta(\mathbf{X}))) \mathbf{1}_{f(\mathbf{X})(2\eta(\mathbf{X}) - 1) < 0} \right] \end{aligned}$$

- Using the property of the wrong sign classifiers

$$\begin{aligned} & \Psi \left(\mathbb{E} \left[\ell^{0/1}(Y, \text{sign}(f(\mathbf{X}))) \right] - \mathbb{E} \left[\ell^{0/1}(Y, f^*(\mathbf{X})) \right] \right) \\ & \leq \mathbb{E}_{\mathbf{X}} \left[(H(f, \eta(\mathbf{X})) - H(f^*, \eta(\mathbf{X}))) \mathbf{1}_{f(\mathbf{X})(2\eta(\mathbf{X})-1)<0} \right] \\ & \leq \mathbb{E}_{\mathbf{X}} [(H(f, \eta(\mathbf{X})) - H(f^*, \eta(\mathbf{X})))] \\ & \leq \mathbb{E} [\ell'(Y, f(\mathbf{X}))] - \mathbb{E} [\ell'(Y, f^*(\mathbf{X}))] \end{aligned}$$

- Ideal solution:

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

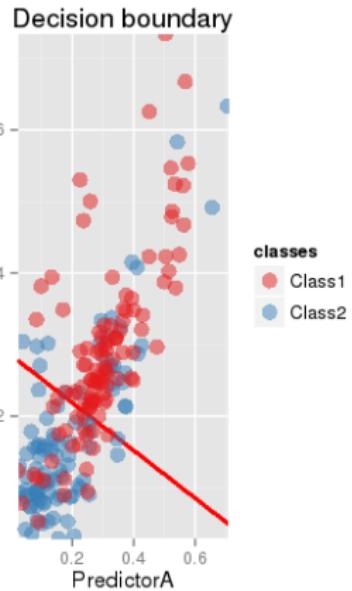
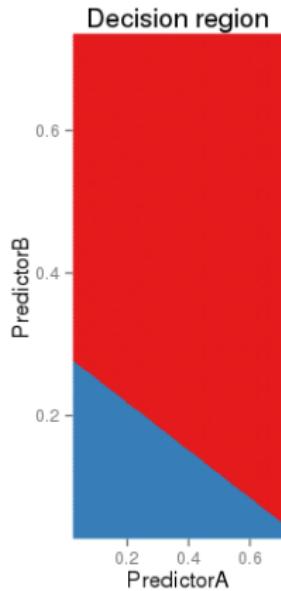
Logistic regression

- Use $f(\mathbf{X}) = \mathbf{X}^t \beta + b$.
- 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

A Discriminative PoV

Logistic



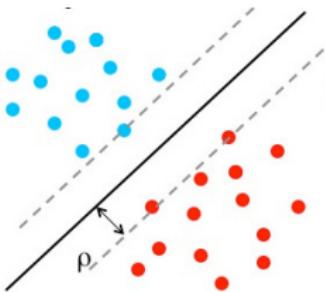
Outline

A Discriminative PoV

- 1 Introduction
 - Supervised Learning
 - The Example of Univariate Linear Regression
- 2 Statistical Supervised Learning
- 3 A Generative Point of View
 - Fully Generative Modeling
 - Parametric Modeling
 - Non Parametric Conditional Estimation
- 4 A Discriminative Point of View
 - Convexification of the Risk
 - SVM
 - Penalization
 - (Deep) Neural Networks
 - Tree Based Methods
- 5 Model Selection
 - Models
 - Feature Design
 - Models, Complexity and Selection

Ideal Separable Case

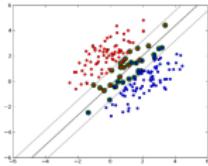
A Discriminative PoV



- Linear classifier: $\text{sign}(\mathbf{X}^t \beta + b)$
- Separable case: $\exists(\beta, b), \forall i, Y_i(\mathbf{X}_i^t \beta + b) > 0!$

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

- Strict separation: $\exists(\beta, b), \forall i, Y_i(\mathbf{X}_i^t \beta + b) \geq 1$
- Maximize the distance between $\mathbf{X}^t \beta + b = 1$ and $\mathbf{X}^t \beta + b = -1$.
- Equivalent to the minimization of $\|\beta\|^2$.



- What about the non separable case?
- Relax the assumption that $\forall i, Y_i(\mathbf{X}_i^t \beta + b) \geq 1$.
- Naive attempt:

$$\operatorname{argmin} \|\beta\|^2 + C \frac{1}{n} \sum_{i=1}^n \mathbf{1}_{Y_i(\mathbf{X}_i^t \beta + b) \leq 1}$$

- Non convex minimization.

SVM: better convex relaxation!

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

SVM as a Penalized Convex Relaxation

A Discriminative PoV

- Convex relaxation:

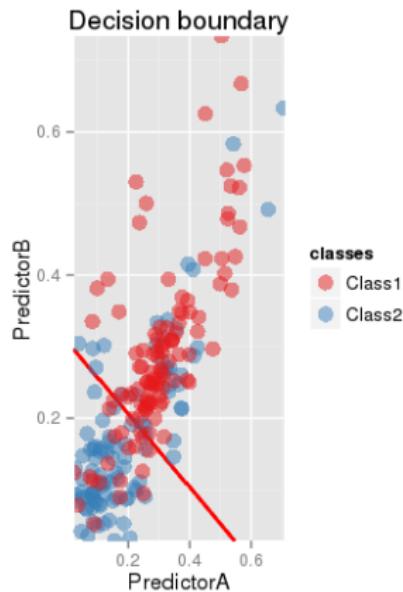
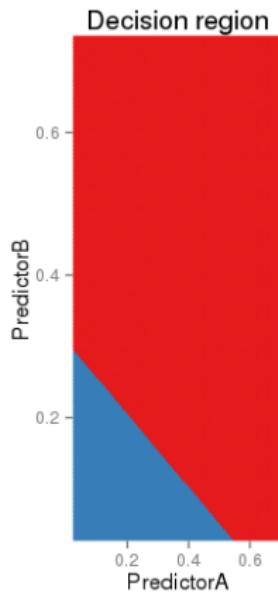
$$\begin{aligned} & \operatorname{argmin} \|\beta\|^2 + C \frac{1}{n} \sum_{i=1}^n \max(1 - Y_i(\mathbf{X}_i^t \beta + b), 0) \\ &= \operatorname{argmin} \frac{1}{n} \sum_{i=1}^n \max(1 - Y_i(\mathbf{X}_i^t \beta + b), 0) + \frac{1}{C} \|\beta\|^2 \end{aligned}$$

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

Penalized convex relaxation (Tikhonov!)

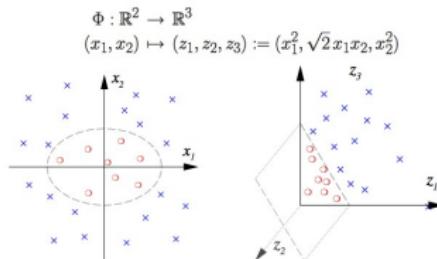
$$\begin{aligned} & \frac{1}{n} \sum_{i=1}^n \ell^{0/1}(Y_i, \operatorname{sign}(\mathbf{X}_i^t \beta + b)) \\ & \leq \frac{1}{n} \sum_{i=1}^n \max(1 - Y_i(\mathbf{X}_i^t \beta + b), 0) + \frac{1}{C} \|\beta\|^2 \end{aligned}$$

Support Vector Machine



The Kernel Trick

A Discriminative PoV

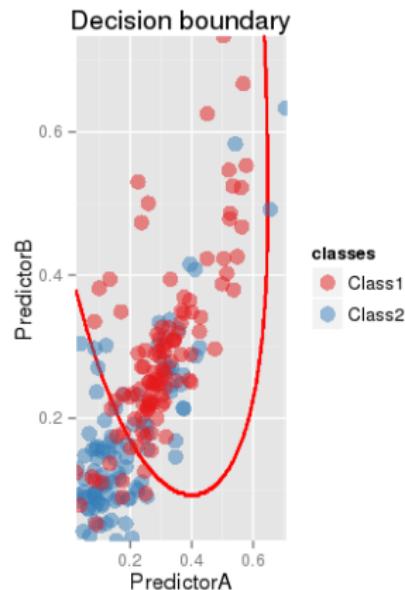
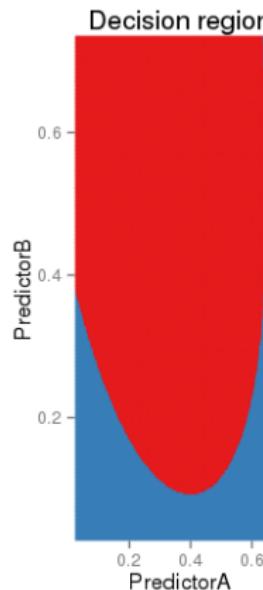


- Non linear separation: just replace x by a non linear $\Phi(x)\dots$

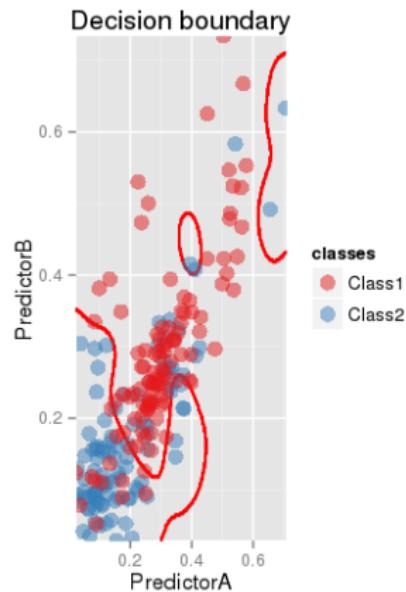
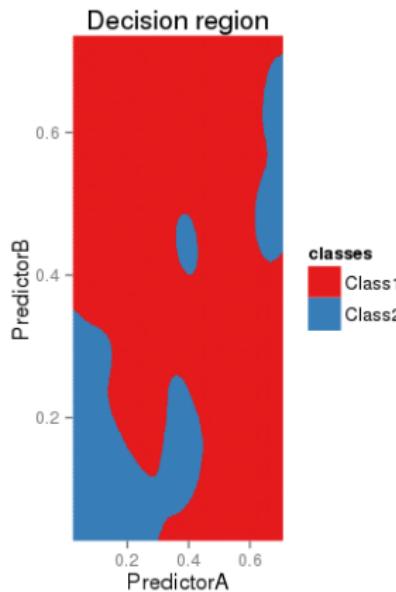
Kernel trick

- Computing $k(x, x') = \langle \Phi(x), \Phi(x') \rangle$ may be easier than computing $\Phi(x)$, $\Phi(x')$ and then the scalar product!
- Φ can be specified through its definite positive kernel k .
- Examples: Polynomial kernel $k(x, x') = (1 + \langle x, x' \rangle)^d$, Gaussian kernel $k(x, x') = e^{-\|x-x'\|^2/2}, \dots$
- RKHS setting!
- Can be used in (logistic) regression and more...

Support Vector Machine with polynomial kernel



Support Vector Machine with Gaussian kernel



Outline

A Discriminative PoV

- 1 Introduction
 - Supervised Learning
 - The Example of Univariate Linear Regression
- 2 Statistical Supervised Learning
- 3 A Generative Point of View
 - Fully Generative Modeling
 - Parametric Modeling
 - Non Parametric Conditional Estimation
- 4 A Discriminative Point of View
 - Convexification of the Risk
 - SVM
 - **Penalization**
 - (Deep) Neural Networks
 - Tree Based Methods
- 5 Model Selection
 - Models
 - Feature Design
 - Models, Complexity and Selection

Two Approaches

- **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}

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

and choose the model with the smallest penalized risk.

Which loss to use?

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

Penalization as a risk correction

- The empirical loss computed on an estimator selected in a family according to the data is biased!
 - Optimistic estimation of the risk...
 - Estimate an upper bound of this optimism for a given family, called the penalty.
 - Add it to the empirical loss
-
- One can also think of the penalty as a way to force the use of *simple* models...
 - **Rk:** Interpretability with both the statistical and the optimization point of view.

Penalized Loss

- Minimization of

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

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

Penalties

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

Instantiation

- Penalized Loss for Linear Model
- Structural Risk Minimization

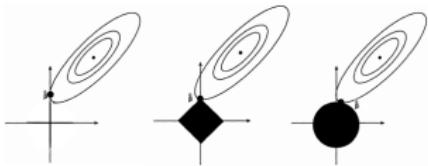
- **Setting:** Gen. linear model = prediction of Y by $h(\mathbf{X}^t \beta)$.

Model coefficients

- Model entirely specified by β .
- Coefficientwise:
 - $\beta_i = 0$ means that the i th covariate is not used.
 - $\beta_i \sim 0$ means that the i th covariate has a *low* influence...
- If some covariates are useless, better use a simpler model...

Submodels

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



Sparsity

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

Norm Constraint and Sparsity

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

Constrained Optimization

- Choose a constant C .
- Compute β as

$$\operatorname{argmin}_{\beta \in \mathbb{R}^d, \|\beta\|_p \leq C} \frac{1}{n} \sum_{i=1}^n \ell(Y_i, h(\mathbf{X}_i^t \beta))$$

Lagrangian Reformulation

- Choose λ and compute β as

$$\operatorname{argmin}_{\beta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^n \ell(Y_i, h(\mathbf{X}_i^t \beta)) + \lambda \|\beta\|_p^{p'}$$

with $p' = p$ except if $p = 0$ where $p' = 1$.

- Easier calibration...

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

Penalized Linear Model

- Minimization of

$$\operatorname{argmin}_{\beta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^n \ell(Y_i, h(\beta^t \mathbf{X}_i)) + \text{pen}(\beta)$$

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

- Variable selection if β is sparse.

Classical Penalties

- AIC: $\text{pen}(\beta) = \lambda \|\beta\|_0$ (non convex / sparsity)
- Ridge: $\text{pen}(\beta) = \lambda \|\beta\|_2^2$ (convex / no sparsity)
- Lasso: $\text{pen}(\beta) = \lambda \|\beta\|_1$ (convex / sparsity)
- Elastic net: $\text{pen}(\beta) = \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2$ (convex / sparsity)
- Easy optimization if pen (and the loss) is convex...
- Need to specify λ !**

Classical Examples

- Penalized Least Squares
- Penalized Logistic Regression
- Penalized Maximum Likelihood
- SVM
- Tree pruning
- Sometimes used even if the parametrization is not linear...

Penalized $\ell^{0/1}$ loss (Structural Risk Minimization)

- Minimization of

$$\operatorname{argmin}_{f_m, m \in \mathcal{M}, f_m \in \mathcal{S}_m} \frac{1}{n} \sum_{i=1}^n \ell^{0/1}(Y_i, f_m(\mathbf{X}_i)) + \text{pen}(m)$$

where $\text{pen}(m)$ is a complexity driven penalty...

- No easy optimization here!

Classical Penalties

- Finite class: $\text{pen}(m) = \lambda \sqrt{\frac{\log |\mathcal{M}|}{n}}$
- Finite VC Dimension: $\text{pen}(m) = \lambda \sqrt{\frac{d_{VC}(\mathcal{S}_m) \log\left(\frac{en}{d_{VC}(\mathcal{S}_m)}\right)}{n}}$

- Need to specify λ !**

Penalized convexified ℓ loss

- Minimization of

$$\operatorname{argmin}_{f_m, m \in \mathcal{M}, f_m \in \mathcal{S}_m} \frac{1}{n} \sum_{i=1}^n \ell(Y_i, f_m(\mathbf{X}_i)) + \text{pen}(m)$$

where $\text{pen}(m)$ is a complexity driven penalty...

- No easy optimization here!
- Reuse the previous $\text{pen}(m)$!**
- Need to specify λ !**
- SVM case:
 - $d_{VC} \sim \|\beta\|^2$ which advocates for a penalty in $\lambda \|\beta\| \dots$
 - A penalty in $\lambda' \|\beta\|^2$ is more convenient numerically and there is a correspondence between the two problems...

Practical Selection Methodology

- Choose a penalty shape $\widetilde{\text{pen}}$.
- Compute a CV error for a penalty $\lambda \widetilde{\text{pen}}$ for all $\lambda \in \Lambda$.
- Determine $\widehat{\lambda}$ the λ minimizing the CV error.
- Compute the final model with a penalty $\widehat{\lambda} \widetilde{\text{pen}}$.

Why not using only CV?

- **If** the penalized likelihood minimization is easy, much cheaper to compute the CV error for all $\lambda \in \Lambda$ than for all possible estimators...
- CV performs best when the set of candidates is not too big (or is structured...)

Outline

A Discriminative PoV

1 Introduction

- Supervised Learning
- The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

- Fully Generative Modeling
- Parametric Modeling
- Non Parametric Conditional Estimation

4 A Discriminative Point of View

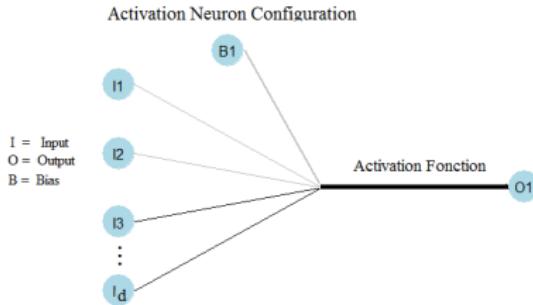
- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

5 Model Selection

- Models
- Feature Design
- Models, Complexity and Selection

Artificial Neuron and Logistic Regression

A Discriminative PoV

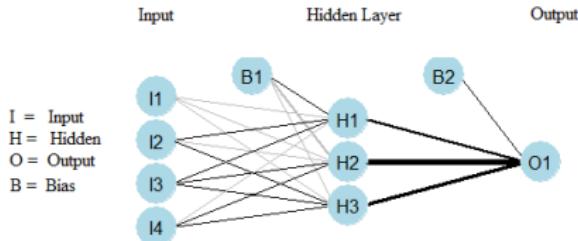


Artificial neuron

- 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.



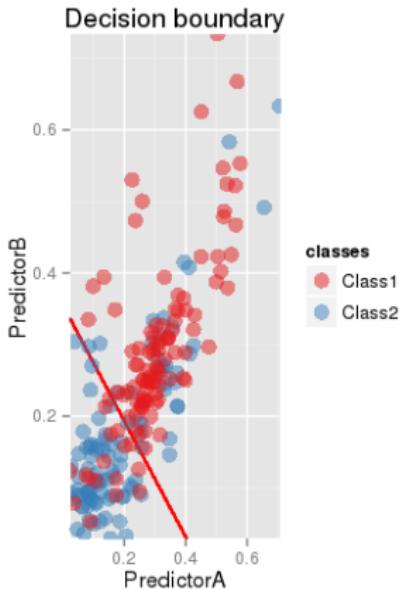
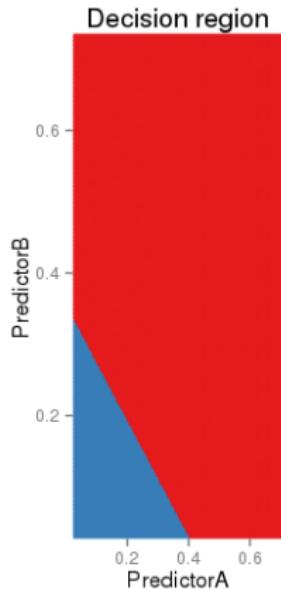
Neural network structure

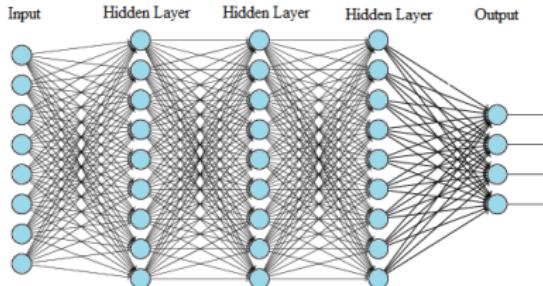
- Cascade of artificial neurons organized in layers
- Thresholding decision only at the output layer
- Most classical case use logistic neurons and the -log-likelihood as the criterion to minimize.
- Classical (stochastic) gradient descent algorithm with a clever implementation (Back propagation)
- Non convex and thus may be trapped in local minima.

Neural network

A Discriminative PoV

Neural Network





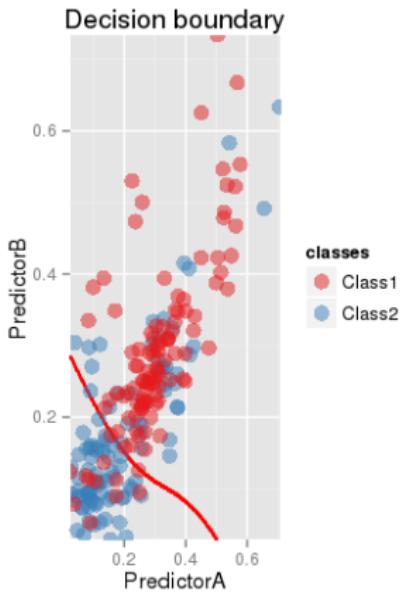
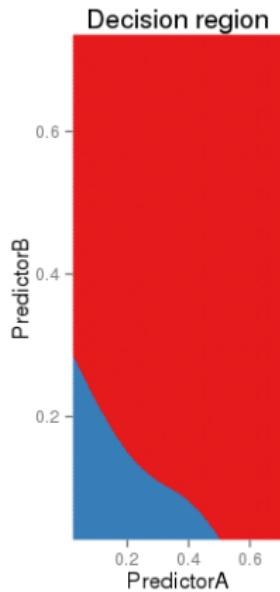
Deep Neural Network structure

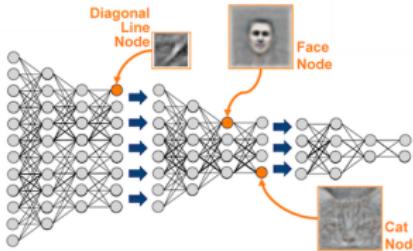
- Deep cascade of layers!
- No conceptual novelty...
- But a lot of details that enabled to obtain a good solution: clever initialization, better activation function, weight regularization, accelerated stochastic gradient descent, early stopping...
- Use of GPU...
- Very impressive results!

Deep Neural Network

A Discriminative PoV

H2O NN





Family of Machine Learning algorithm combining:

- a (deep) multilayered structure,
- a clever optimization including initialization and regularization.
- Examples: Deep Neural Network, Deep (Restricted) Boltzman Machine, Stacked Encoder, Recursive Neural Network...
- Transfer learning: use as initialization a pretrained deep structure.
- Appears to be very efficient but lack of theoretical fundation!

Outline

A Discriminative PoV

1 Introduction

- Supervised Learning
- The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

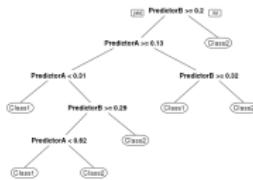
- Fully Generative Modeling
- Parametric Modeling
- Non Parametric Conditional Estimation

4 A Discriminative Point of View

- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

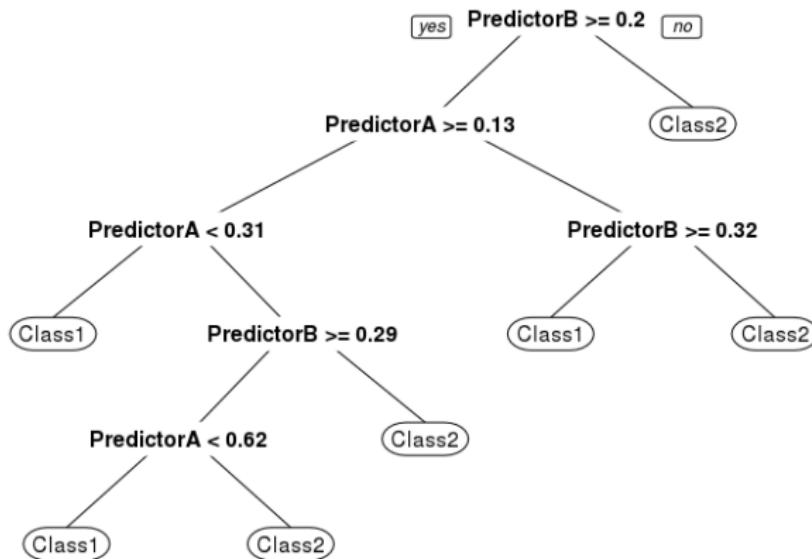
5 Model Selection

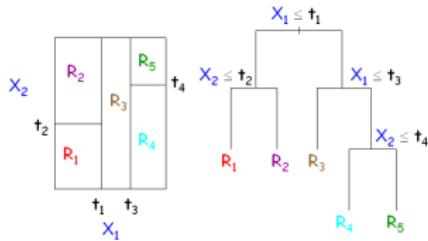
- Models
- Feature Design
- Models, Complexity and Selection



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)





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

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

- CART: Gini index (classification)

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

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

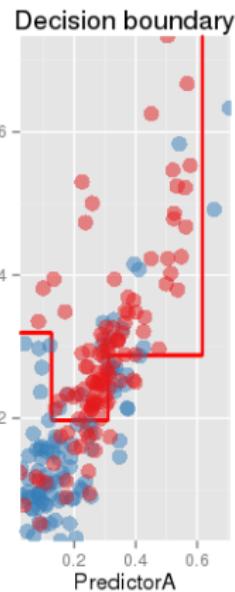
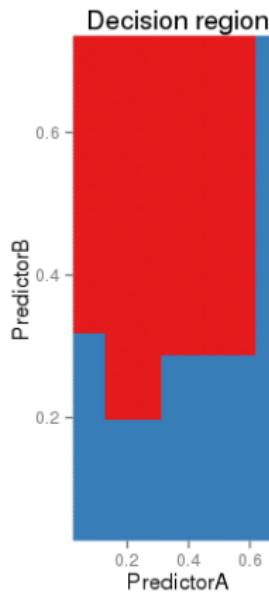
$$C(R, \bar{R}) = \sum_{x_i \in R} H(R) + \sum_{x_i \in \bar{R}} H(\bar{R})$$

- 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!
 - Additional pruning often use.

CART



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

Parallel construction

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

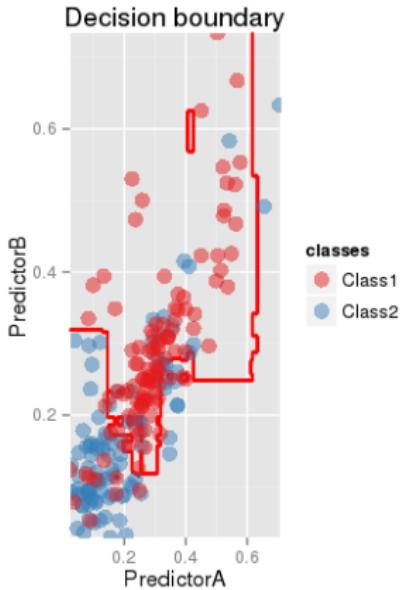
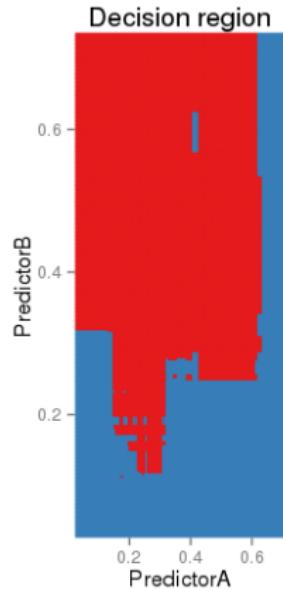
Sequential construction

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

Ensemble methods

A Discriminative PoV

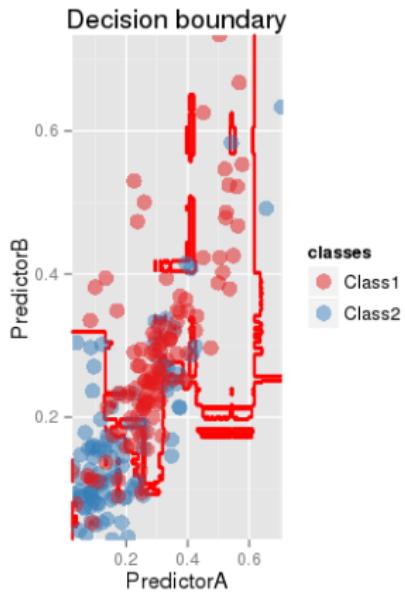
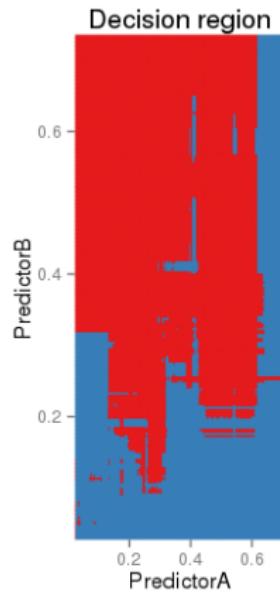
Bagging



Ensemble methods

A Discriminative PoV

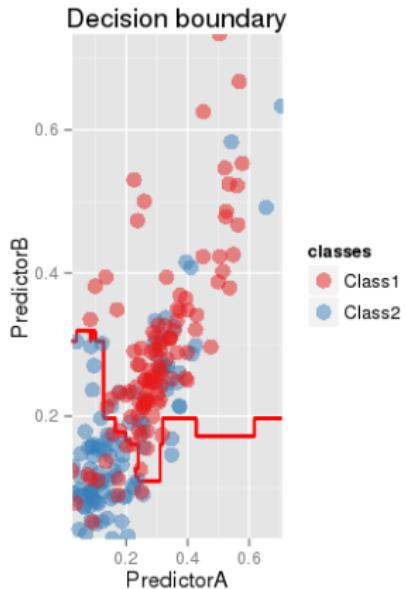
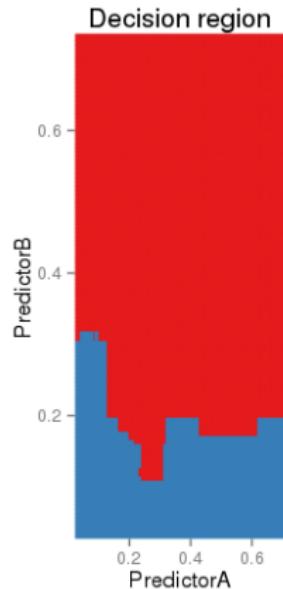
Random Forest



Ensemble methods

A Discriminative PoV

AdaBoost



Outline

Model Selection

1 Introduction

- Supervised Learning
- The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

- Fully Generative Modeling
- Parametric Modeling
- Non Parametric Conditional Estimation

4 A Discriminative Point of View

- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

5 Model Selection

- Models
- Feature Design
- Models, Complexity and Selection

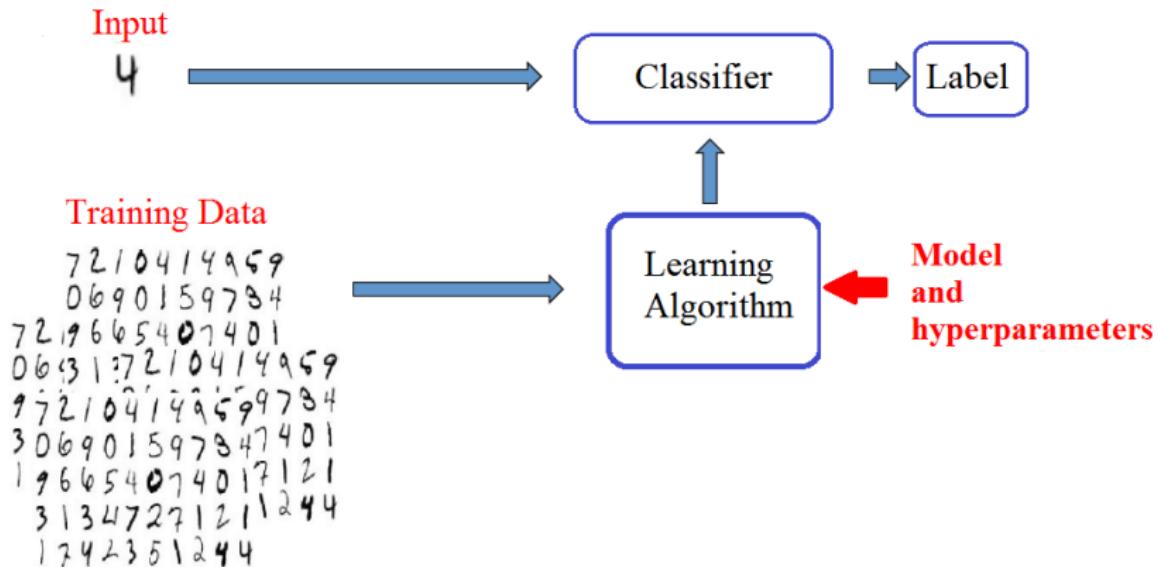
Outline

Model Selection

- 1 Introduction
 - Supervised Learning
 - The Example of Univariate Linear Regression
- 2 Statistical Supervised Learning
- 3 A Generative Point of View
 - Fully Generative Modeling
 - Parametric Modeling
 - Non Parametric Conditional Estimation
- 4 A Discriminative Point of View
 - Convexification of the Risk
 - SVM
 - Penalization
 - (Deep) Neural Networks
 - Tree Based Methods
- 5 Model Selection
 - Models
 - Feature Design
 - Models, Complexity and Selection

Model and Hyperparameters

Model Selection



- Ideal solution:

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

Logistic

- Model $Y|\mathbf{X}$ 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:

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

Generative Modeling

- Estimate $\mathbf{X}|Y$ with a density estimator as well as $\mathbb{P}\{Y\}$
 - Deduce using the Bayes formula an estimate $Y|\mathbf{X}$.
 - Plug the estimate in the Bayes classifier.
-
- Model hyperparameters:
 - Features
 - Generative model

- Ideal solution:

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

Kernel methods

- Estimate $Y|\mathbf{X}$ with a kernel conditional density estimator.
- Plug the estimate in the Bayes classifier.
- Model hyperparameters:
 - Features
 - Bandwidth and kernel

- Ideal solution:

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

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:

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

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:

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

NN

- Neuron: $\mathbf{X} \mapsto \sigma(\mathbf{X}^t \beta + b)$
 - 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:

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

Single tree

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

Bagging/Random Forest

- Averaging of several predictors (statistical point of view)

Boosting

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

Outline

Model Selection

- 1 Introduction
 - Supervised Learning
 - The Example of Univariate Linear Regression
- 2 Statistical Supervised Learning
- 3 A Generative Point of View
 - Fully Generative Modeling
 - Parametric Modeling
 - Non Parametric Conditional Estimation
- 4 A Discriminative Point of View
 - Convexification of the Risk
 - SVM
 - Penalization
 - (Deep) Neural Networks
 - Tree Based Methods
- 5 Model Selection
 - Models
 - Feature Design
 - Models, Complexity and Selection

Transformed Representation

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

$$f_{\beta}(\mathbf{X}) = \Phi(\mathbf{X})^t \boldsymbol{\beta}$$

Feature Design

- Art of choosing Φ .
- Examples:
 - Renormalization, (domain specific) transform
 - Basis decomposition
 - Interaction between different variables...
- Need to select a good transformation.

Outline

Model Selection

1 Introduction

- Supervised Learning
- The Example of Univariate Linear Regression

2 Statistical Supervised Learning

3 A Generative Point of View

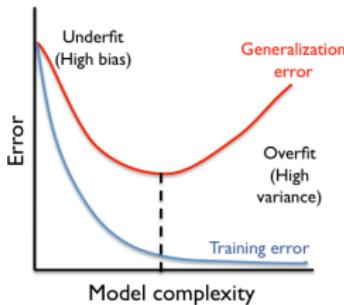
- Fully Generative Modeling
- Parametric Modeling
- Non Parametric Conditional Estimation

4 A Discriminative Point of View

- Convexification of the Risk
- SVM
- Penalization
- (Deep) Neural Networks
- Tree Based Methods

5 Model Selection

- Models
- Feature Design
- Models, Complexity and Selection



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!

Two Approaches

- **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}

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

and choose the model with the smallest penalized risk.

Which loss to use?

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

- 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...