Introduction to machine learning

Michel.RIVEILL@univ-cotedazur.fr

Outline

- What is machine learning and Machine learning applications
- Materials: the data
- Methods: the machine learning pipeline
- Library and tools
- Bibliography and online ressources

What is machine learning ?

Why “Learn”?

- Machine learning is programming computers to optimize a performance criterion using example data or past experience.
- There is no need to “learn” to calculate payroll
- Learning is used when:
  - Human expertise does not exist (navigating on Mars),
  - Humans are unable to explain their expertise (speech recognition)
  - Solution changes in time (routing on a computer network)
  - Solution needs to be adapted to particular cases (user biometrics)

What We Talk About When We Talk About“Learning”

- Learning general models from a data of particular examples
- Data is cheap and abundant (data warehouses, data marts); knowledge is expensive and scarce.
- Example in retail: Customer transactions to consumer behavior:

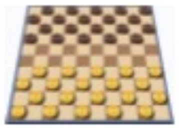
  People who bought “Da Vinci Code” also bought “The Five People You Meet in Heaven” (www.amazon.com)

- Build a model that is *a good and useful approximation* to the data.

What is Machine Learning?

- Machine Learning
  - Study of algorithms that
  - improve their performance
  - at some task
  - with experience
- Optimize a performance criterion using example data or past experience.
- Role of Statistics: Inference from a sample
- Role of Computer science: Efficient algorithms to
  - Solve the optimization problem
  - Representing and evaluating the model for inference

Machine learning

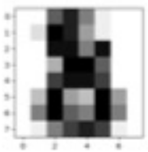
- General definition from Tom Mitchell (Carnegie Mellon 1997)
  - "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E".
- 

	Checkers learning
Tasks T	Playing checkers
Performance measure P	Percent of games won against opponents
Training experience E	Playing practice games against itself

Source: https://commons.wikimedia.org/wiki/File:International_draughts.jpg

Machine learning

- General definition from Tom Mitchell (Carnegie Mellon 1997)
  - “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".
- Ex



	Handwritten recognition
Tasks T	Recognizing and classifying handwritten words within images
Performance measure P	Percent of words correctly classified
Training experience E	A database of handwritten words with given classifications

Machine learning

- General definition from Tom Mitchell (Carnegie Mellon 1997)
    - "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".
- Exa

	Robot driving learning
Tasks T	Driving on public four-lane highways using vision sensors
Performance measure P	Average distance traveled before an error
Training experience E	A sequence of images and steering commands recorded while observing a human driver

Growth of Machine Learning

- Machine learning is preferred approach to
  - Speech recognition, Natural language processing
  - Computer vision
  - Medical outcomes analysis
  - Robot control
  - Computational biology
- This trend is accelerating
  - Improved machine learning algorithms
  - Improved data capture, networking, faster computers
  - Software too complex to write by hand
  - New sensors / IO devices
  - Demand for self-customization to user, environment
  - It turns out to be difficult to extract knowledge from human experts→failure of expert systems in the 1980’s.

Applications

- Association Analysis
- **Supervised Learning**
  - **Classification**
  - **Regression/Prediction**
- **Unsupervised Learning**
  - **Clusterisation**
  - Dimension reduction
- Reinforcement Learning

Learning Associations

- Basket analysis:
  - P (Y | X) probability that somebody who buys X also buys Y where X and Y are products/services.
  - Example: P (chips | beer) = 0.7

Market-Basket transactions

TID	*Items*
1	**Bread, Milk**
2	**Bread, Diaper, Beer, Eggs**
3	**Milk, Diaper, Beer, Coke**
4	**Bread, Milk, Diaper, Beer**
5	**Bread, Milk, Diaper, Coke**

Supervised learning

- from the data (**X**, **y**) obtain an approximation $\hat{f}(.)$ of $f(.)$
  - Such that for a new observation **x** which is not in data.
  - We can obtain a reasonable prediction $\hat{y} = \hat{f}(x)$

- Classification → predict a class
- Regression → predict a value

Regression versus classification

- Regression → predict a value

e.g. $\mathbf{X} = [\mathbf{x}_A, \mathbf{x}_W]$ and $\mathbf{y} = \begin{bmatrix} 178 \\ 173 \\ 158 \end{bmatrix}$ is height in centimeters

$\Longrightarrow$ Predict height from age and weight.

- Classification → predict a class

e.g. $\mathbf{X} = [\mathbf{x}_A, \mathbf{x}_W]$ and $\mathbf{y} = \begin{bmatrix} \text{no} \\ \text{no} \\ \text{yes} \end{bmatrix}$ says if the person is diabetic or not

$\Longrightarrow$ Predict if a person is diabetic from age and weight.

Uses of Supervised Learning:

Example: decision trees tools that create rules

- Prediction of future cases: Use the rule to predict the output for future inputs
- Knowledge extraction: The rule is easy to understand
- Compression: The rule is simpler than the data it explains
- Outlier detection: Exceptions that are not covered by the rule, e.g., fraud

Unsupervised Learning

- Learning “what normally happens”
- No output
- Clustering: Grouping similar instances
- Other applications: Summarization, Dimension reduction, Association Analysis
- Example applications
  - Customer segmentation in CRM
  - Image compression: Color quantization
  - Bioinformatics: Learning motifs
  - Plot data

Reinforcement Learning

- Topics:
  - Policies: what actions should an agent take in a particular situation
  - Utility estimation: how good is a state (→used by policy)
- No supervised output but delayed reward
- Credit assignment problem (what was responsible for the outcome)
- Applications:
  - Game playing
  - Robot in a maze
  - Multiple agents, partial observability, ...

ML Pipeline
From data to model

Data

- Datum
  - a characteristic or a number that may contain information about an objects, individuals, observations, populations
  - e.g. Age [years] = 31
- Data
  - multiple datum about one or multiple objects, individuals, etc
  - Machine learning use multiple variables from multiple individuals
  - We represent these data as a feature matrix
    - Rows are observation vectors (N observations or items)
    - Columns are feature vectors (M features or caracteritics
    - E.g. Age in years xA and weight xW in kilos of 3 peoples

$$\mathbf{X} = [\mathbf{x}_A, \mathbf{x}_W] = \begin{bmatrix} 31 & 68 \\ 23 & 64 \\ 32 & 58 \end{bmatrix}$$

- Without data ⇒ No machine learning!

Data

- In some cases, a feature vector **y** is supposed to depend on the other feature vectors (independent variables)
  - y is called the output
  - The data is the tuple (**X**, **y**)

- In supervised machine learning
  - We assume that there is a unknown function $f(\cdot)$
  - linking the independent part of the observation vector $\mathbf{x}_i$ to y_i
  - $y_i = \boldsymbol{f}(xi)$ or $y = \boldsymbol{f}(X)$

Data - Types: quantitative *vs.* qualitative

Quantitative

- measurable quantities - numerical
- mathematical functions can be applied (*e.g.* sum, mean)
- comparisons are possible (*e.g.* =, ≠, >, <

Qualitative

- characteristics or qualities (which type/category?)
- mathematical functions cannot be applied
- not all comparisons are possible

Data - Types: quantitative *vs.* qualitative

Quantitative

- Continuous
  - any value in an interval (age)
- Discrete
  - only a finite number of values (nb of room in a house)

Qualitative

- Nominal
  - No possible orderint
- Ordinal
  - Ordering is possible (size: XS, S, M, L, XL)

Data - Types: structred *vs.* non-structured

Structured

- column/row structured data
- easier too retrieve (SQL databases)
- *e.g.* databases, excel file, …

Non strutured

- image, text, video
- harder too retrieve
- *e.g.* emails, radiography

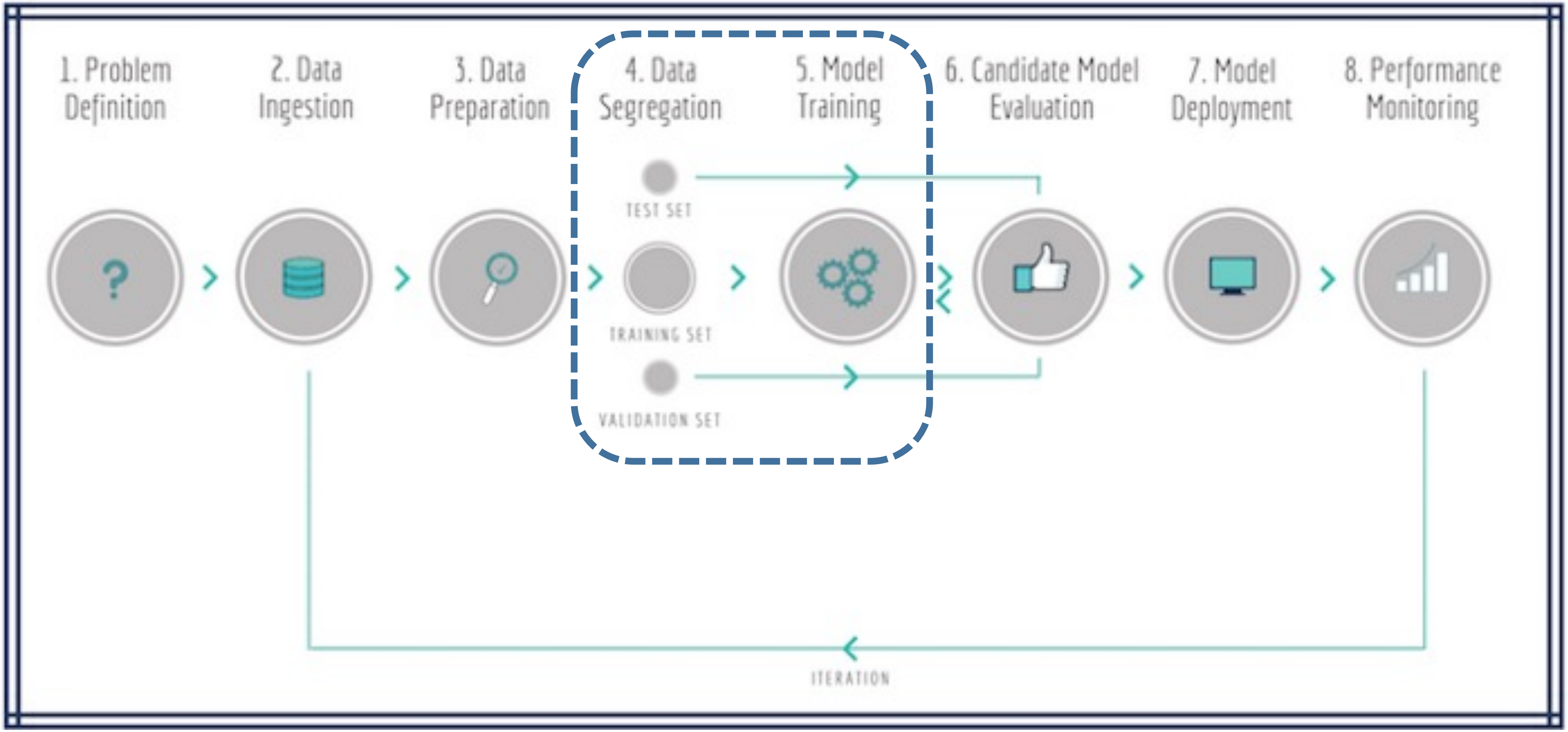
Semi Structured

- XML, JSON, CSV, logs
- easier too retrieve (NOSQL databases)
- *e.g.* Twitter data

Taxonomy of models and examples of algorithm

		Nature of $\hat{y}$	
		Continuous	Finite
Presence of y	Yes	**Regression** • Linear regression • Decision trees and forest • Support vector regression (SVR)	**Classification** • Logistic regression • Naive Bayes • Decision trees and forest • Support vector classification (SVC)
	No	**Dimension reduction** • Principal component analysis (PCA)	**Clustering** • K-means • Hierarchical clustering

Machine learning pipeline



Software / Tools

Software and languages

- **Specialized software platforms** - visual interfaces, easy to use but specialized
  - Weka, Orange, Knime
- **Numerical computing environments** - harder to use, but more General
  - Matlab (proprietary), Scilab, Octave
- **Programming languages** - hardest to use, but most general and professional
  - Python, R, Julia, Java, Scala

Python for the labworks

- We will use Python 3, which is distributed with Anaconda
  - Doc.: https://docs.python.org/3.6/tutorial/
- We will also use Jupyter Notebook: web application allowing to create and share documents with code, text, figures and equations.
  - Doc.: https://jupyter-notebook.readthedocs.io/en/ latest/notebook.html#the-jupyter-notebook
- You can download Anaconda (Jupyter is included) from

  ⇒ https://www.anaconda.com/download/
- Or use Google Colab

  ➔ https://colab.research.google.com

Google Colab

- Presentation
  - https://www.youtube.com/watch?v=inN8seMm7UI
- Colaboratory, often shortened to "Colab", allows you to write and execute Python code in your browser. It offers the following advantages:
  - No configuration required
  - Free access to GPUs
  - Easy sharing
- **This is the choice we make in this course.**

Python for the labworks : useful libraries

- Numpy: support for vectors, matrices and multi-dimensional arrays along with high-level mathematical functions to operate on these arrays
  - Doc.: https: //docs.scipy.org/doc/numpy/user/quickstart.html
- Scipy: library for scientific and technical computing. It contains modules for optimization, linear algebra, integration, interpolation, signal/image processing and other tasks common in science and engineering
  - Doc.: https://docs.scipy.org/doc/scipy/reference/ tutorial/index.html

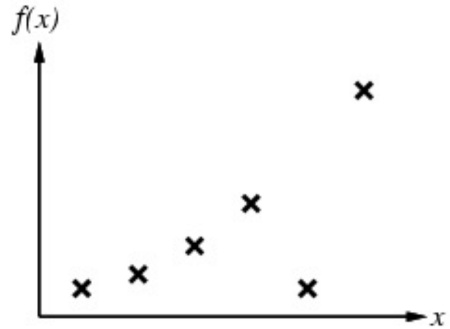
Python for the labworks : useful libraries

- Pandas: high-level building block for doing practical, real world data analysis in Python.
  - Doc.: https://pandas.pydata.org/docs/user_guide/index.html
- Matplotlib: plotting library, it provides a large number of plotting options, 2D line graphs, bar graphs, scatterplots, 3D surfaces, contour plots, images, polar charts and pie charts.
  - Doc.: https://matplotlib.org/tutorials/index.html
- **Scikit-learn**: machine learning library featuring algorithms for regression, classification, dimensionality reduction and clustering.
  - Doc.: http: //scikit-learn.org/stable/tutorial/index.html

First model: linear regression
(just to do some first experimentation)

Linear regression

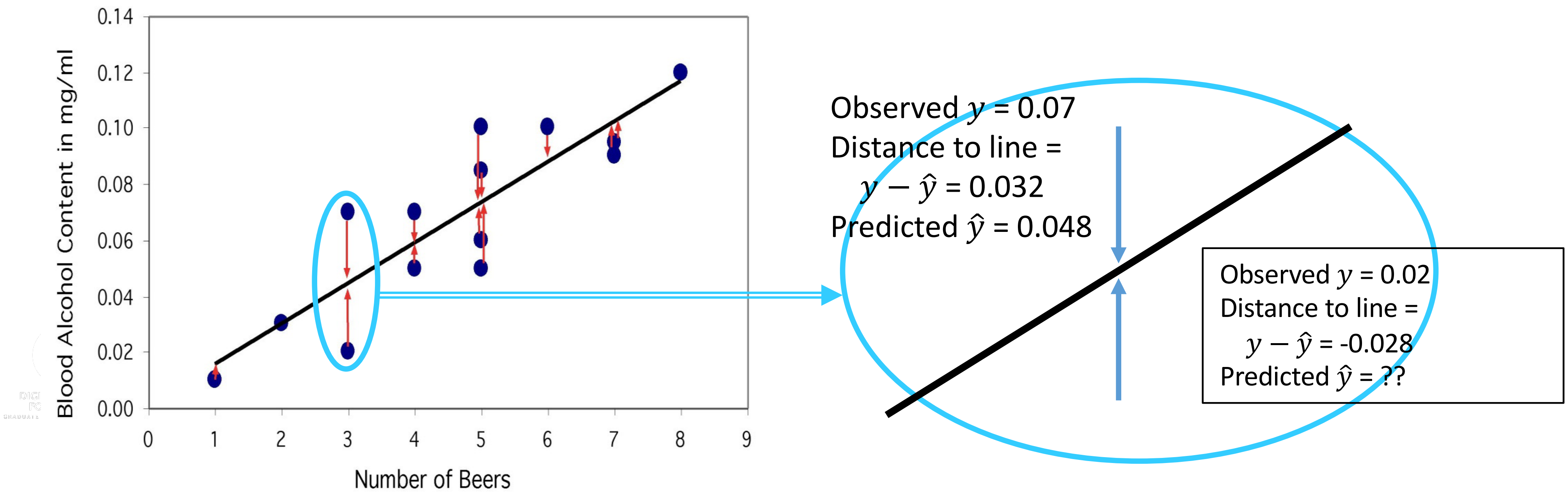
- When do we use simple linear regression?
  - Input feature $X = [x_1, x_2, \ldots, xn]$
  - Output feature $y = [y_1, y_2, \ldots, yn]$ with continuous amplitude
- Linear regression model
  - The prediction function $\widehat{f_{ß}}(X)$ is specified as
    - $\hat{y}$ = $ß_1 * \mathrm{x} + ß_0$ → curve fitting
    - $ß_1$ = slope, $ß_0$ = intercept



    - Or in more dimention : $\hat{y} = [1\ x^{(1)}\ x^{(2)} \ldots x^{(N)}]ß = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \ldots \\ \hat{y}_N \end{bmatrix} = \begin{bmatrix} 1 & x_1^{(1)} & \ldots. & x_1^{(k)} \\ 1 & x_2^{(1)} & \ldots. & x_2^{(k)} \\ & & \ldots & \\ 1 & x_N^{(1)} & \ldots. & x_N^{(k)} \end{bmatrix} \begin{bmatrix} ß_0 \\ ß_1 \\ \ldots \\ ß_k \end{bmatrix}$
    - N items / k features

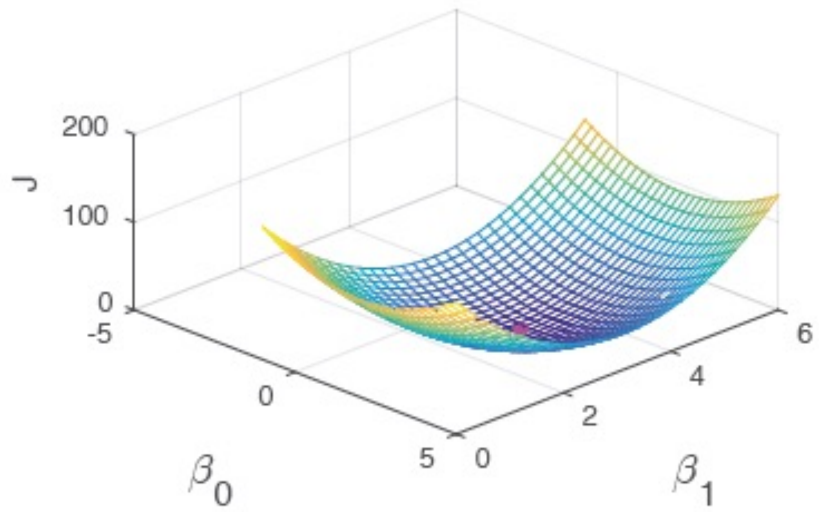
Linear regression

- The **least-squares regression line** is the unique line such that the sum of the **vertical distances** between the data points and the line is zero, and the sum of the squared vertical distances is the smallest possible.
  - $\hat{y}$ = Xß



How to solve the problem (i.e. find ß) ?

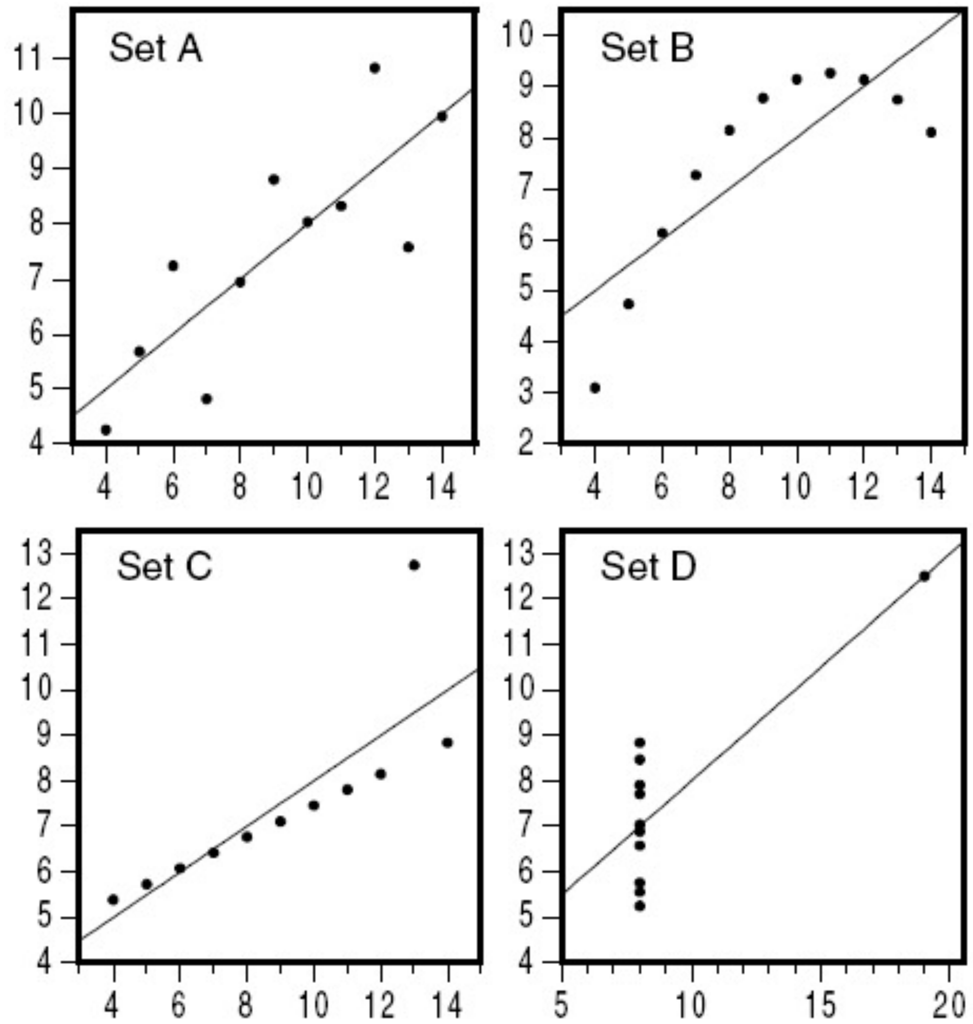
- Find values of ß such that $\hat{f}_{ß}$ gives $\hat{y}$ close to available y
  - Close ? You have to define it.
    - For example minimize of the square error
    - $J(ß) = \frac{1}{N}\sum(y_i - \hat{y}_i)^2 = \sum(y_i - x_i ß)^2$
- J is a **cost function**
  - J is a function of ß, since X, y are fixed
  - The cost function is convex
    → There is only one global minimum



Depending on the data, we can have several scenarios

Moderate linear association; regression OK.

Obvious nonlinear relationship; regression inappropriate.



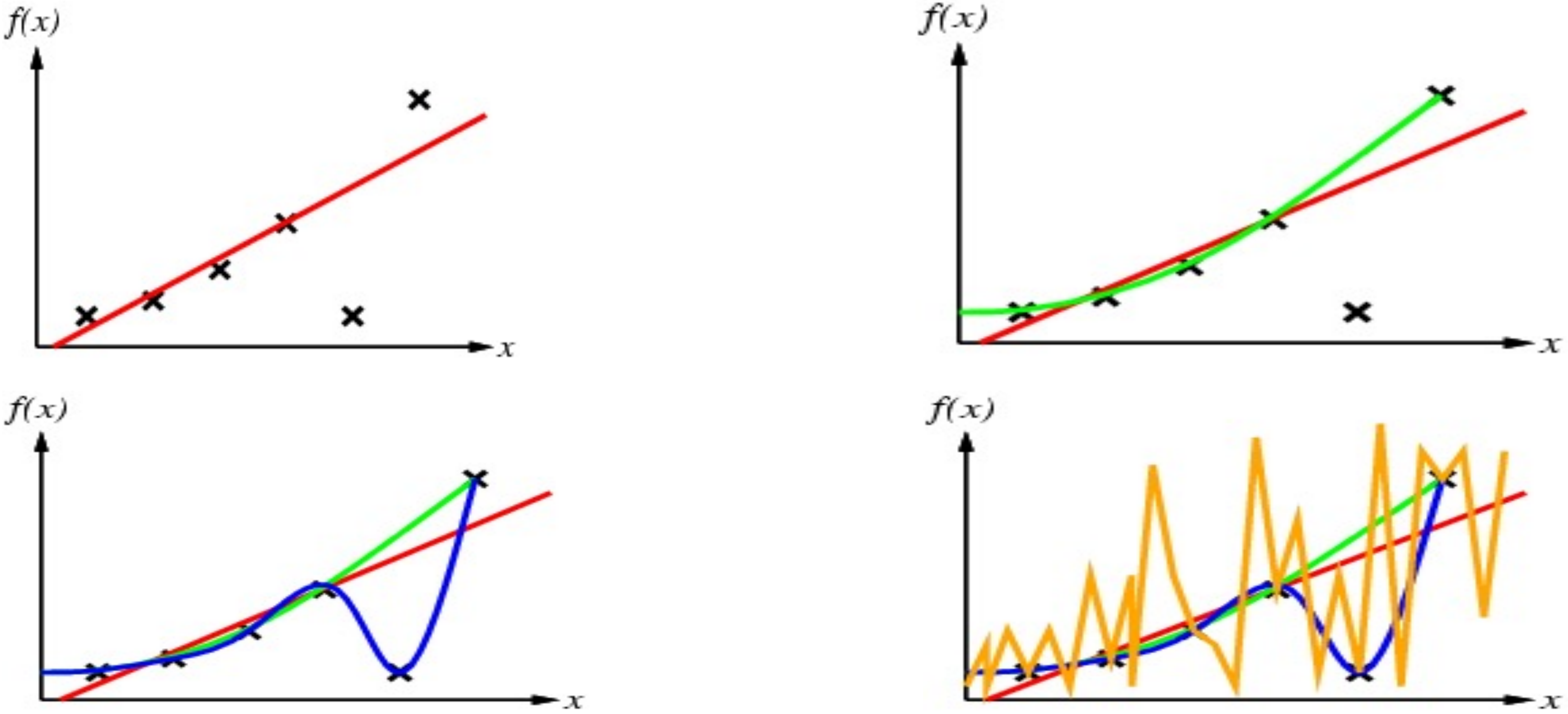
One extreme outlier, requiring further examination.

Only two values for *x;* a redesign is due here…

How do evaluate the model

- We need an absolute criteria → a **metrics**
- For exemple in regression problem we can choose between:
  - Mean absolute error (MAE) : MAE = $\frac{1}{N}\sum|yi\ -\hat{y}_i|$
    - The higher it is, the worse the model is
    - Difficult interpretation from its value
  - Mean squared error (MSE) : $MSE = \frac{1}{N}\sum(y_i\ -\hat{y}_i)^2$
    - Gives more weight to major mistakes
    - Note: MSE is measured in square units of y
  - Root mean squared error (RMSE): $RMSE = \sqrt{MSE}$

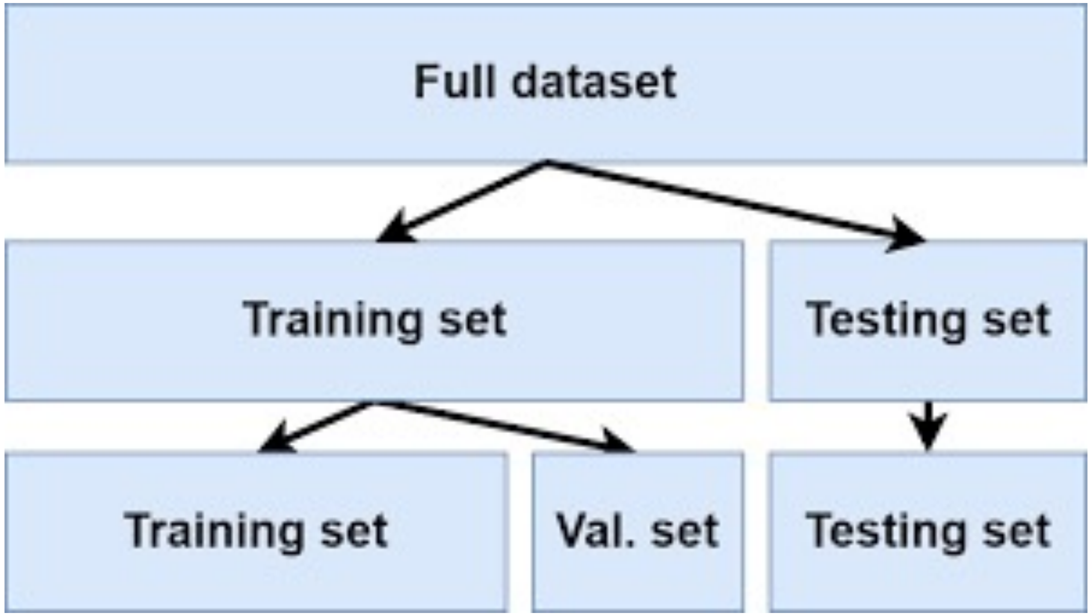
Adjustment of the model to minimize the error

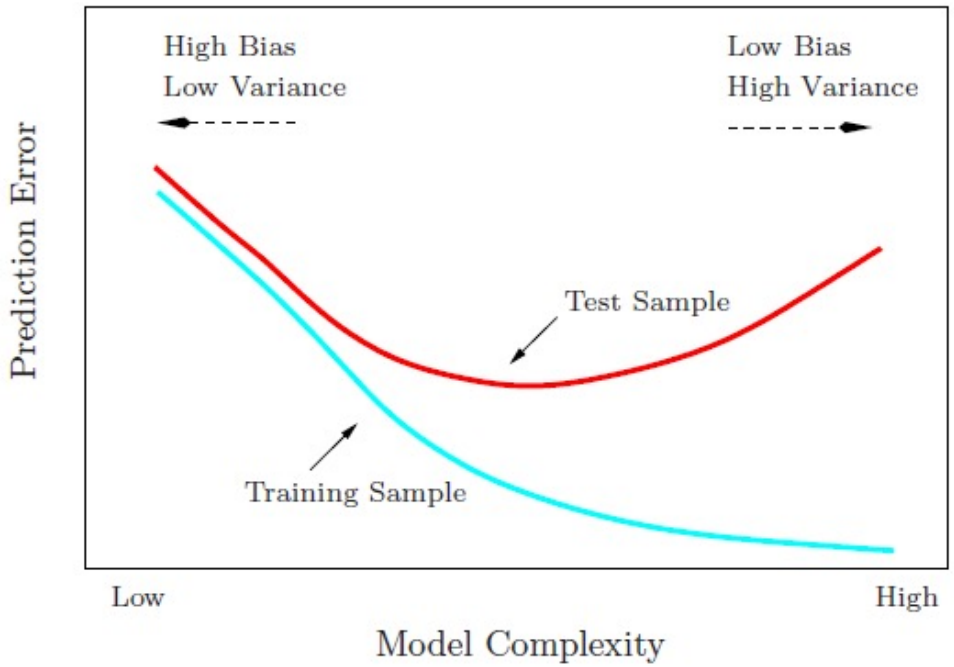


Model generalization

- Reminder: we want to build a model with data that we have in order to use it on future data
  - the model must be able to generalize i.e. correctly predict data that are not in the training data.
- The simple memorization of learning examples is a coherent hypothesis that does not generalize.
- Occam's Razor: Finding a simple hypothesis guarantees generalization.

Training Error vs Test Error





Loss function / Metrics for regression

- To build a model we generally need
  - a **cost function** that we will try to minimize
  - and a **metric** that allows us to evaluate the model
- For the regression we already have a first vision (not complete)
  - Loss
    - MAE (mean absolute error), MSE (mean squared error)
  - Metrics
    - The same
- But what are the metrics for classification?

Loss function / Metrics for classification

- Loss : for example Binary Cross Entropy
  - $J(ß) = -\frac{1}{N}\sum y_i \log(\widehat{y}_i) + (1 - yi)\log(1 - \widehat{y_i})$
- Metrics
  - Many are based on confusion matrix
  - ACCuracy $= \frac{TN+TP}{N}$
  - Recall $= \frac{TP}{FN+TP}$
  - Precision $= \frac{TP}{FP+TP}$

True value	0	TN	FP
	1	FN	TP
		0	1
		Predicted	

Data Cleaning and Preparation
(It's just an introduction

Missing Data

- Missing data is a fact and usually quite common in many data analysis applications.
- Unfortunately many models do not know how to "work" with missing data.
- Several strategies are possible:
  - Delete examples that contain missing data.
  - Give a default value to the missing data (min, max, average, median, etc)

Duplicates and Outliers

- A duplicate = an observation present more than once
  - Often related to a problem during data acquisition
- An outlier = a value or observation that is "distant" from other observations
  - May be due to the inherent variability of the observed phenomenon
  - Or indicate an experimental error
- If we can detect them correctly (it is not always easy),
  - it is better to eliminate them
  - They disturb the generalization of the model, especially if they are errors

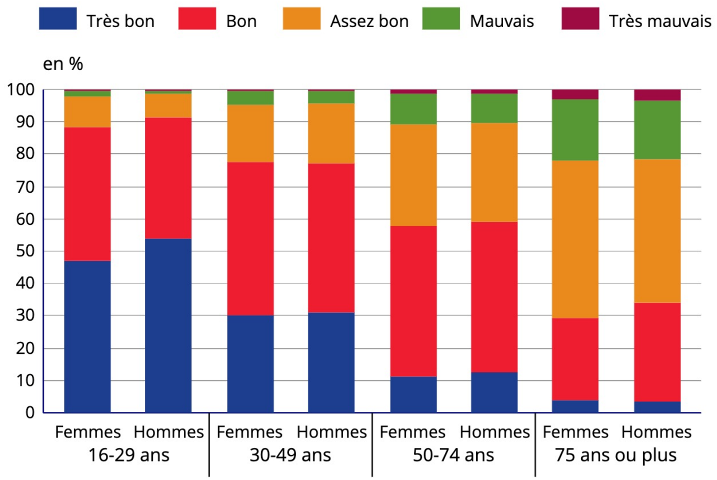
Data normalization for continuous data

- For all continuous data, **as soon as a model uses the notion of distance** between examples, it is necessary to normalize the data.

- Many strategies are possible:
  - Rescaling $= \dfrac{X - Xmin}{X_{max} - Xmin}$ bring all values into the range [0,1]
  - Standard $= \dfrac{X - \mu}{\sigma}$ where μ is the mean and σ is the standard deviation
  - Robust = removes the median and scales the data according to the quantile range

Transform continuous data to categorical one

- Equal width (or distance) binning
  - Consiste à diviser la plage de la variable en k intervalles de largeur égale.
  - Width of the interval $= \dfrac{\boldsymbol{Xmax - Xmin}}{\boldsymbol{k}}$
- Equal depth (or frequency) binning
  - Division of the range [X_{min}, X_{max}] into intervals that contain an equal number of points
- Domaine binning



Categorical data encoding

- Many models do not know how to use categories, so it is necessary to transform categories into numbers or vectors
- Ordinal encoding
  - Transform categorical features as an integer array → imposes an ordinal relationship
  - i.e. [XS, S, M, L, XL] → [1, 2, 3, 4, 5]
- One Hot Encoding
  - Transform categorical features as a vector array → no relationship
  - Each bit represents a possible category. If the variable cannot belong to multiple categories at once, then only one bit in the group can be "on."
  - i.e. [M, F] → [[1,0], [1,0]]

Resources: Datasets

- UCI Repository: http://www.ics.uci.edu/~mlearn/MLRepository.html
- UCI KDD Archive: http://kdd.ics.uci.edu/summary.data.application.html
- Statlib: http://lib.stat.cmu.edu/
- Delve: http://www.cs.utoronto.ca/~delve/
- Kaggle: https://www.kaggle.com

Resources: Bibliography

- Some free pdf available
- Introduction to Machine Learning:
  - https://ai.stanford.edu/~nilsson/MLBOOK.pdf
- Mathematics for Machine Learning:
  - https://mml-book.github.io/book/mml-book.pdf
- Practical Machine Learning with Python:
  - pdf available
- Hands-On Machine Learning with Scikit-Learn and TensorFlow
  - pdf available