Introduction to Machine Learning

Carlos Cernuda,
Ekhiñe Irurozki,
Aritz Pérez

Preprocessing

Carlos Cernuda

Outline

- Introduction
- Data checking
  - Expert knowledge
  - Initial cleaning
  - Missing values
  - Outliers
- Numeric data standardization & normalization
- Numeric data discretization
  - Unsupervised
  - Supervised
- Feature selection
  - Filters
  - Wrappers
- Feature extraction
  - Principal components analysis (PCA)
- Handling unbalanced data
  - Cost sensitive approaches
  - Sampling methods

Introduction

- Which is my **purpose** when using ML with my data?

- Did I collect my data?
  - If so, did I do it right?
  - If not, can I be (almost) sure that my data was rightly collected?
  - What does "right" mean?
  - Right in terms of data representation and structure, volume, … (problem dependent)

- Is my data useful? (Reliable, representative, noisy, …)

- Once my goal and my options in term of ML algorithms (computation or memory limitations, etc.) are clear, how will I evaluate the goodness of my results and infer valid conclusions? (Not my task)

Introduction

- Summarizing:

- Know where you want to go.

- Know how to go there.

- Know (if you can) that you will eventually reach there.

Introduction

- The first step in your journey is **preprocessing** your data.
- It has been usually **neglected**.
- In real-world problems, data preprocessing takes around **80%** of your **working time**.
- It is crucial for the subsequent steps. The **bigger** your **data**, the **more important** their preprocessing process.


Welcome To Hell

Data checking

- **Expert knowledge** is always relevant, not only in preprocessing.

- Due to expert knowledge you can:
  - Know which features are the most **relevant** (from experience),
  - Know the valid ranges of numerical variables (**domain knowledge**),
  - Identify **redundancies** in the features,
  - Set **bounds** to the modeling choices.

- <u>Example</u>: NIR spectral data in viscose production => local correlation, relevant/irrelevant spectral regions, Beer-Lambert Law, limited concentrations.

Data checking

- **Data cleaning** can be done by applying expert knowledge or in a data-driven fashion.

- Get familiar with your data:
  - Look for **useless attributes** (noticeable even without expert knowledge).
  - Look for **corrupted data** (empty or constant variables).

- Just have a look and see what comes out, if the volume of your data allows it.

Missing values

- **Missing values** are unexpected "holes" in the data.

- There are three strategies for dealing with missing values:
  - **Ignore records** (rows, samples, …) where missing values are detected.
  - Treat missing values in an attribute as **legitimate values** of the attribute .
  - **Impute** a value to "fill the holes":
    - **Direct imputation**, in a supervised or unsupervised way, of a significant value, such as the mean or median (numerical attributes), the mode categorical attributes), or a suggestion by an expert.
    - More **advanced techniques** requiring some data analysis, such as k-nearest neighbours or expectation & maximization algorithm.

Outlier detection

- **Outliers** are odd values.
- Their detection can be by expert knowledge (e.g. known attribute's range) or data-driven. Let's focus on the latter.
- Outliers detection can be done in two ways:
  - Individual: Performed variable by variable. The simplest approaches consist on considering as outliers the values outside the intervals

  $[\bar{X} - 3 * S_X, \bar{X} + 3 * S_X]$ and $[Q_1 - 1.5 * IQR, Q_3 + 1.5 * IQR]$ (Box-plots, robust)
  - Collective: Using all variables. The most representative one takes into account the distribution of the data, and is based on calculating and ellipsoid around the data points, using Mahalanobis distance. The points outside it are considered as outliers.
- When detected, the record is ignored or imputed (risky).

Numeric data standardization & normalization

- When computing distances between pairs of samples, the scales of the different features is very relevant.

- Moreover, we cannot obviate the curse of dimensionality effect, i.e. in high-dimensional spaces all data is sparse. In simple words, all distances become huge.

- Therefore, if the algorithm we plan to apply after preprocessing implies distances and/or we are in a high-dimensional problem, we should transform all the features to a similar scale.

Numeric data standardization & normalization

- Some of the approaches are:
  - Mean centering: scaling to zero mean by subtracting the mean. Necessary, for instance, in approaches related to Mahalanobis distance, like PCA.
  - Standardization: scaling to zero mean and unit variance by subtracting the mean and dividing by the standard deviation.
  - Normalization: scaling to [0, 1] by subtracting the minimum and dividing by the range.
- In general, do **not** standardize or normalize your data unless you have a good reason for doing so.
- Other **transformations** from Statistics (logit, square root, power, Box-Cox, angular, etc.) are not considered because they are part of the posterior processing phase.

Discretization

- The goal of discretization is reducing the number of values of a continuous attribute by grouping them into intervals (bins). The new values are the bins.
- Some methods require discrete attributes (some naïve Bayes and Bayesian networks methods, etc.).
- Sometimes, the results are better after discretization. Some methods do it implicitly (e.g. decision trees).
- In general, the computational cost of algorithms with discrete attributes is lower than their continuous versions.

Discretization

- Types of discretization: Unsupervised vs supervised.
- **Unsupervised**. Without using the attribute of interest.
- Some require to fix the number of bins, *k*, in advance:
  - If too large the bins could have not enough data to be significant.
  - If too little, we might loose the informative power of the attribute.
  - It is tricky to find out the right value.
- Examples:
  - **Equal-width**. Divide the range into *k* equal-width ranges. It could cause unbalance (e.g. the salary in a company).
  - **Equal-frequency**. Divide the range into *k* ranges with the same frequency. Better than equal-width in terms of clumping, but could produce odd artifacts with frequent or special values.

Discretization

- Types of discretization: Unsupervised vs supervised.
- **Supervised**. Using the attribute of interest.
- In general, they have the advantage of **not having to fix *k* in advance**.
- Example:
  - **Fayyad & Irani's MDLP algorithm**. It is a recursive algorithm to detect the best cut-points for the bins using mutual information between the attribute to be discretized and the attribute of interest.
- As a positive **side effect**, the methods in which k is not fixed in advance can be used as part of the **feature selection** step (still to come) to **detect irrelevant features**, when $k = 1$.

Feature selection

- With N variables, there are 2^N **possible feature subsets**. Exploring all options is not frequently feasible.

- **Feature selection** consists on choosing a subset of the original features according to certain criteria to be optimized (**optimization problem**).

- Depending on the criteria and the way to optimize it we can get two types of outputs:
  - Just the subset of selected features.
  - A measurement of the **importance** of the features. This allows us to rank the features. Then we could get a subset by fixing a threshold for a cut-point of the importance.

Feature selection

- There are three types:
  - Filter methods: based on **intrinsic properties** of the data. The most widely used are **statistical** properties, such as correlation or mutual information. Not algorithm dependent (universal).
  - Wrapper methods: using some **learning task** in the selection. Unless the computational cost is too high, we would use the same learning task chosen for the subsequent processing. Usually much more **computationally expensive** than filters.
  - Embedded methods: included inside the algorithm (outside preprocessing).
- Examples of filters: **Correlation-based feature selection** (different definitions of correlation), **InfoGain** (mutual information), **ReliefF** (distance to the nearest sample from the same class and from a different class), **simmetrical uncertainty** (entropy of the sample and the class).

Feature extraction

- **Feature extraction** is the action of defining new features from all or part of the original ones.
- Despite expert knowledge could be also used, we focus on data-driven approaches.
- When applying feature extraction, there is usually a price to pay in terms of **loss of interpretability** because of the lack of meaning of the artificial features.
- There are both **supervised** and **unsupervised** methods.
- We will have a deeper look to the most famous unsupervised (and linear) method: **principal component analysis** (PCA).

Feature extraction

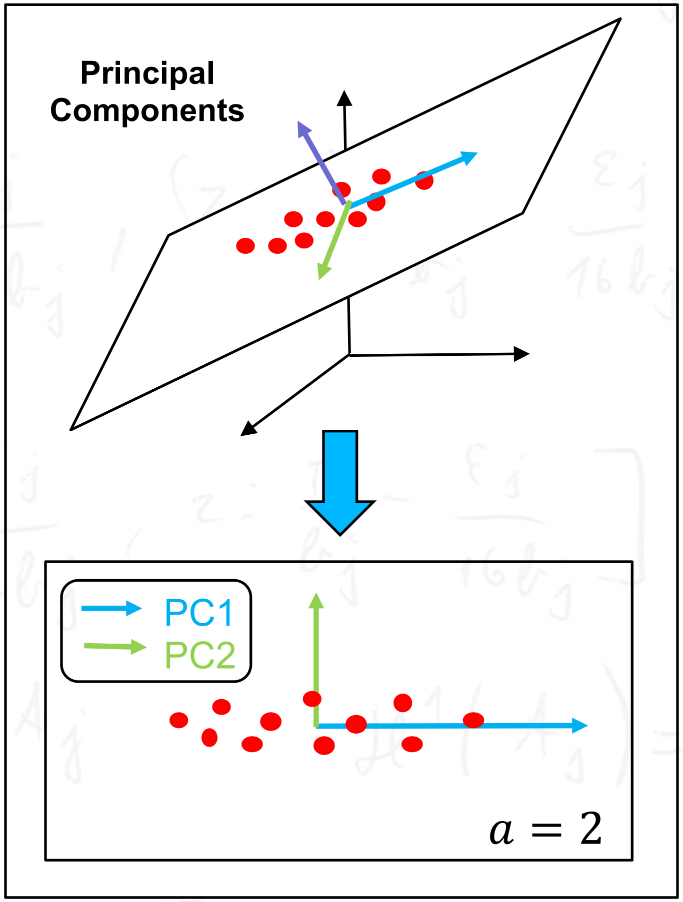
- **Principal Components Analysis**

Principal component analysis

- Based on the decomposition given by:

$$X = T_a P_a^T + E_a \qquad a \equiv \#PCs$$

- The PCs (LVs) are selected by maximizing scores variance
- $P \equiv$ Loadings => Coefficients of the linear combination that defines the PCs
- $T \equiv$ Scores => Coordinates of the data in the new axes
- $E \equiv$ Residuals (null if $a = N$)



Handling unbalanced data

- **Unbalanced data** in classification happens when the class of interest has very low frequency.
- Example: Predicting if certain part of a machine is faulty or not, based on sensor data (fault detection problem).
- Cost-sensitive methods: penalize the errors in the minority class in order to guide the training towards a better representation of that class.
- Example: If the proportion of minority class samples is p, a usual approach is to consider an error in the prediction of the majority class $(1-p)/p$ times more relevant than an error in the prediction of the minority class.

Handling unbalanced data

- Sampling-based methods:
  - Oversampling: add artificially samples of the minority class by copying (randomly or not) or creating instances.
  - Undersampling: Remove samples (randomly or not) of the majority class.
  - Combined: Both oversampling and undersampling.

- Pros & Cons:
  - Classes get equilibrated, promoting a better classification trade-off.
  - Undersampling removes samples that could contain relevant discriminant information.
  - Oversampling incorporates artificial/redundant information that could drive models (depending on their characteristics) towards non-realistic conclusions.

Handling unbalanced data

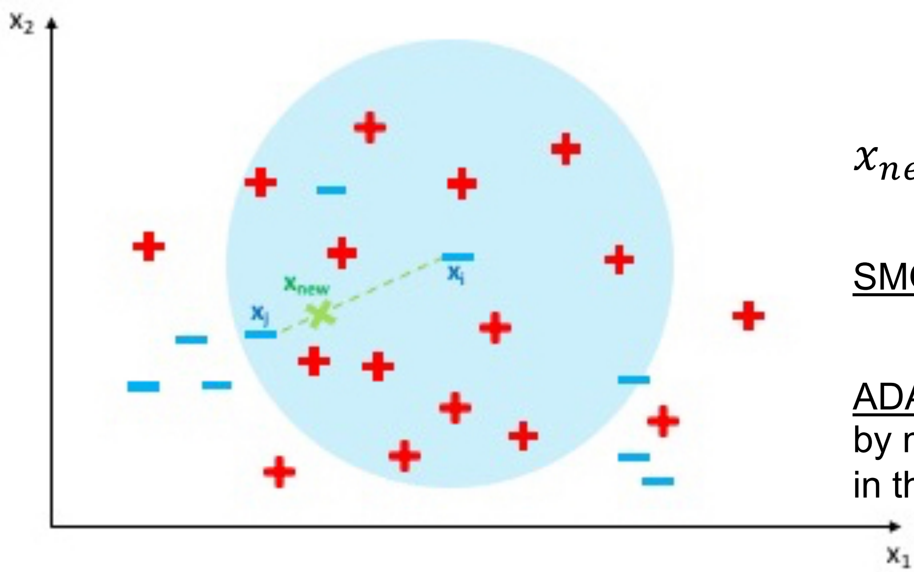
- Oversampling methods

❖ Selection methods:

- ❑ *Random oversampling*. With or without replacement, pure or guided,…

❖ Generation methods:

- ❑ *Synthetic Minority Oversampling TEchnique* (SMOTE).
- ❑ *ADAptive SYNthetic sampling method* (ADASYN).



Randomly selected from the neighborhood of x_i

$$x_{new} = \lambda x_i + (1-\lambda)x_j$$

<u>SMOTE</u>: Neighborhood by KNN where K is pre-fixed

<u>ADASYN</u>: Neighborhood by K-NN where K is estimated by means of the density of the minority class samples in the K'-NN (K' pre-fixed and "big")

Handling unbalanced data

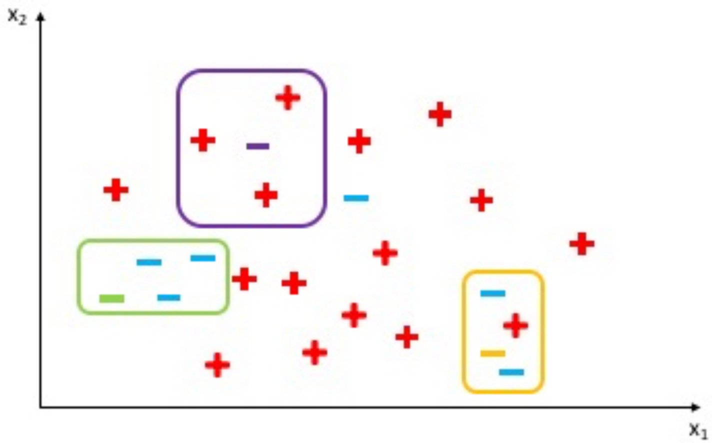
- Oversampling methods

❖ Generation methods:

- ❑ SMOTE
- ❑ ADASYN

Problem: Both suffer in presence of extreme values (outliers or not)
Solution: Design variants where they are not that relevant.

❖ Borderline variants of SMOTE



The minority class samples are classified as:
- Safe (all K-NNs from minority class)
- Noisy (all not from minority class)
- In danger (at least half from minority class)

SMOTE variants (only in danger points considered):

- Borderline-1 SMOTE: The randomly selected x_j is from minority class
- Borderline-2 SMOTE: The randomly selected x_j could be from any class
- SVM SMOTE: The new sample x_{new} is generated using support vectors.

Handling unbalanced data

- Undersampling methods

❖ Generation methods:

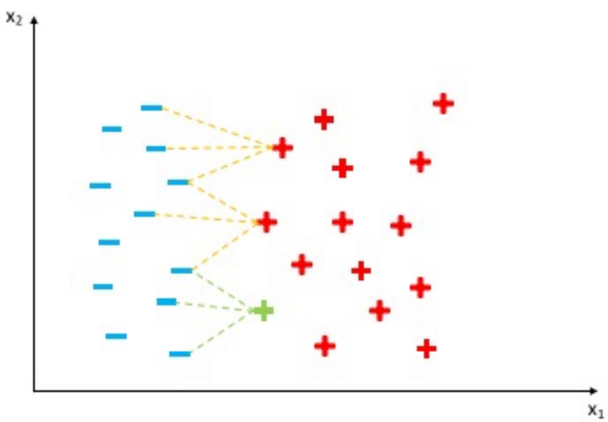
- Cluster centroids undersampling. New majority class samples are the centroids of the clusters obtained by *clustering using representatives* (CURE) algorithm.

❖ Selection methods: Two types named **controlled** and **cleaning** techniques

- **Controlled techniques**. The reduction of the majority class is controlled because we decide how many samples will be removed.
  - *Random undersampling*. Pure or guided,…
  - *NearMiss*. Majority class samples are selected using certain heuristic rules. Three main variants:
    - ✓ NearMiss-1
    - ✓ NearMiss-2
    - ✓ NearMiss-3

Handling unbalanced data

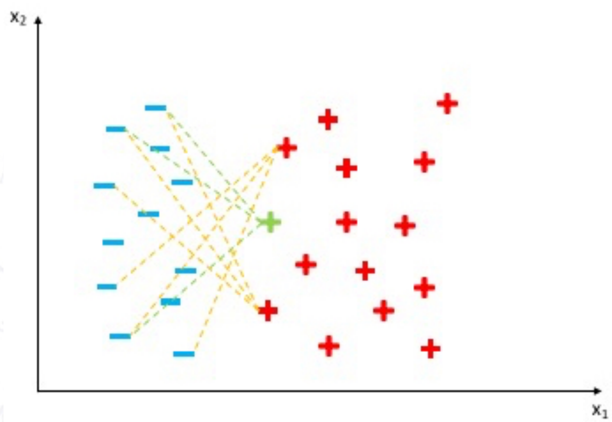
NearMiss-1



The heuristic rule is the minimum average distance to the N closest minority class samples.

Here N = 3 and the selected sample is depicted in green.

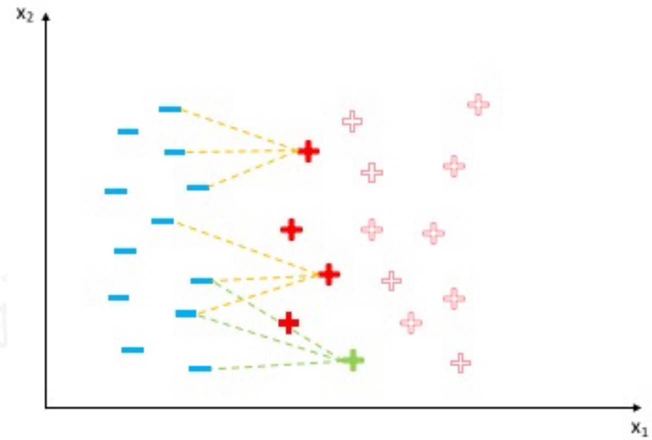
NearMiss-2



The heuristic rule is the minimum average distance to the N farthest minority class samples.

Here N = 3 and the selected sample is depicted in green.

NearMiss-3



The heuristic rule has two steps: First, keep the M nearest majority class neighbors for each minority class sample (dark red). Second, select the majority class sample with maximum average distance to its N minority class nearest neighbors.

Here M = 5, N = 3 and the selected sample is depicted in green.

Handling unbalanced data

- **Cleaning techniques**. The final reduction is not known in advance because we identify and delete dispensable majority class samples.

  - *Tomek's links*. Pairs of points from different classes that are mutual nearest neighbors, i.e. $(x, y)\ Tomek's\ link \iff \forall z, d(x, z) \geq d(x, y) \wedge d(y, z) \geq d(x, y)$. The action could be to remove both or only the majority class sample (y).

  - *Edited Nearest Neighbors* (ENN). Majority class samples that ***does not agree enough*** with their neighborhood are removed. Different agreement criteria edit the neighborhoods in different ways.

  - *Instance hardness*. The hardness of a sample is quantified by the difficulty of predicting its class right (usually employing cross validation on certain simple classification algorithm). Then a threshold is employed as a cutpoint in order to decide which majority class samples will be removed.

- Combined sampling methods
  - Random oversampling + random undersampling
  - SMOTE + Tomek's links
  - SMOTE + ENN