Data Preparation

Data Preprocessing is a technique that is used to convert the raw data into a clean data set. It is necessary because of the presence of unformatted real world data, which may not be supported by the desired ML/DL algorithm.

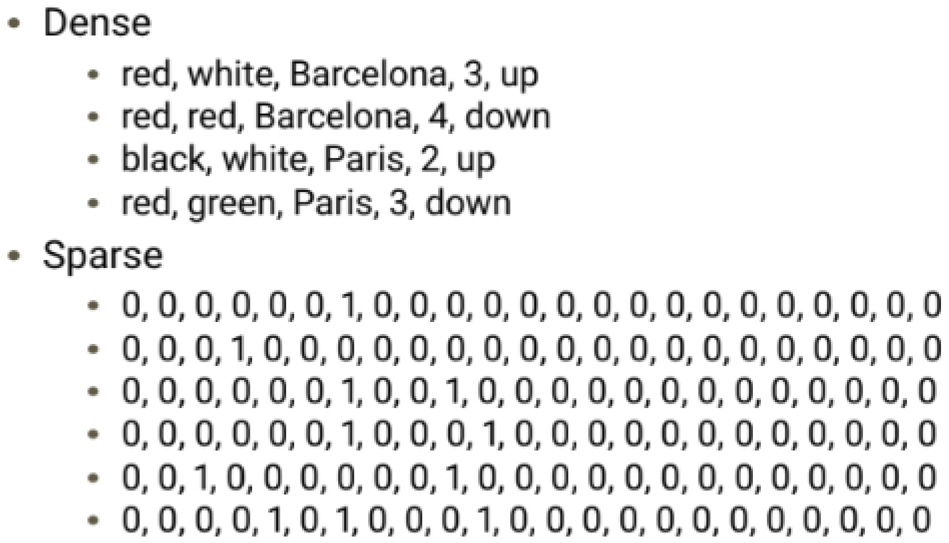
In other words, whenever the data is gathered from different sources, it is collected in raw format which is not feasible for the analysis. Certain steps are executed to convert the data into a small clean data set, among which **Feature extraction**, **Data Cleaning**, **Feature selection and transformation**.

Definitions

Before proceeding to data preprocessing, some definitions are in need.

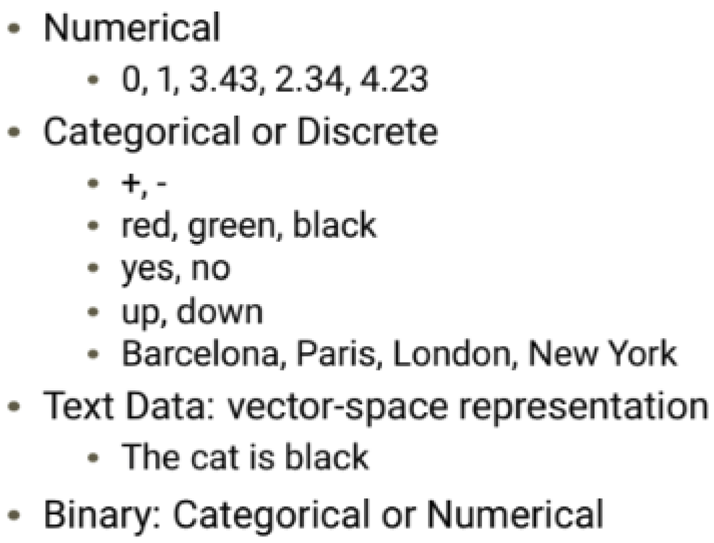
An **instance**, or **example** or **feature-vector**, is a transaction, an entry of a table. It is composed by different fields, called **features** or **dimensions** or **attributes**.

An instance can be **dense**, if the number of 0s or NULL values is low, or **sparse**, if most of the features are non significant for that instance.



- Dense
  - red, white, Barcelona, 3, up
  - red, red, Barcelona, 4, down
  - black, white, Paris, 2, up
  - red, green, Paris, 3, down
- Sparse
  - 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
  - 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
  - 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
  - 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
  - 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
  - 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

A feature can also have different types:



- Numerical
  - 0, 1, 3.43, 2.34, 4.23
- Categorical or Discrete
  - +, -
  - red, green, black
  - yes, no
  - up, down
  - Barcelona, Paris, London, New York
- Text Data: vector-space representation
  - The cat is black
- Binary: Categorical or Numerical

According to attribute/instance type, different kinds or relationships (**patterns**) can be extracted, with different algorithms.

With feature analysis, **classification** algorithms can predict the value of a discrete attribute, while **regression** algorithm can predict the value of a numeric attribute.

With instance analysis, **clustering** algorithms can determine a subset of rows with similar feature values, while **outlier detection** can determine **noise**, which means rows that are very different from the other rows.

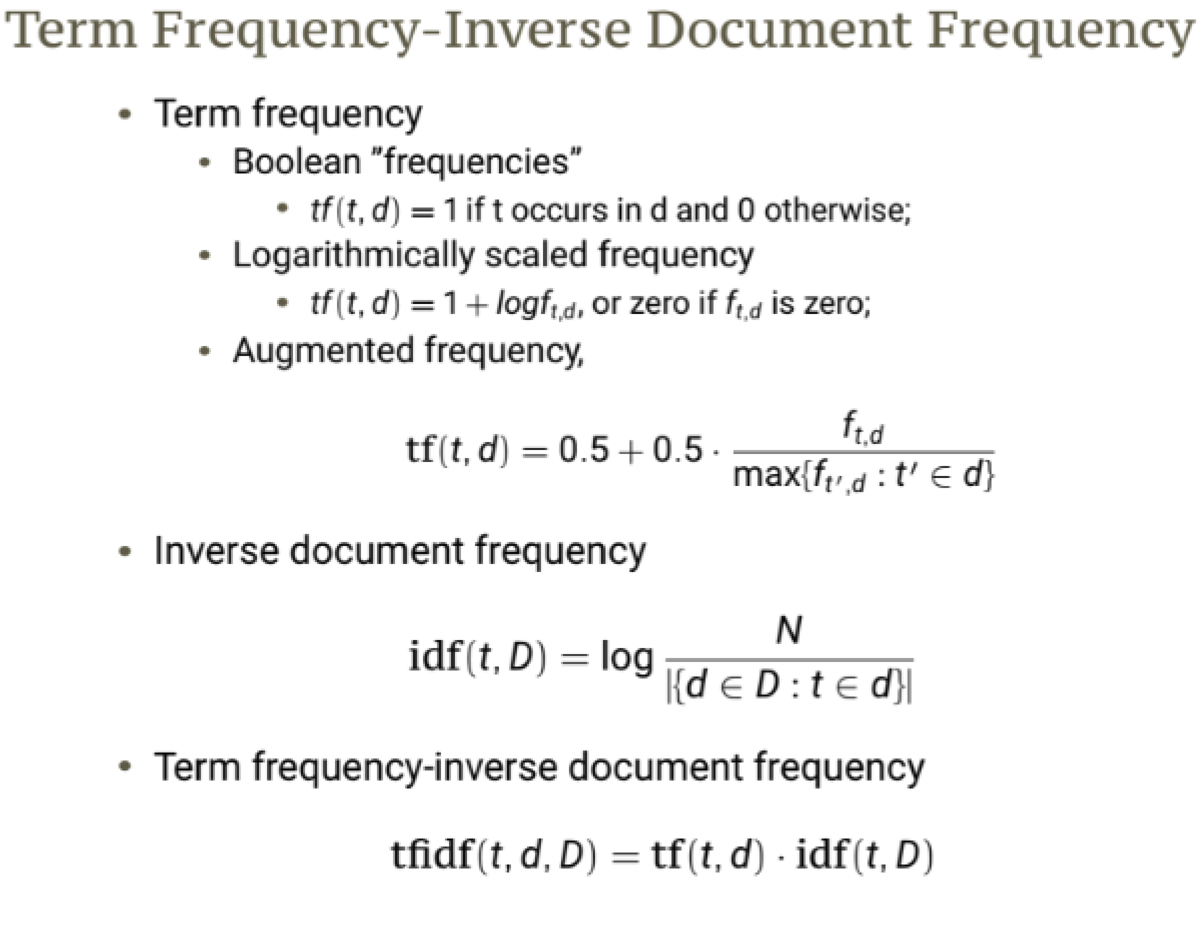
Feature extraction

Different sources generate different kinds of data, structured or not, from which features and therefore information have to be extracted:

- sensor data - wavelets or Fourier transform
- image data - histogram or visual words
- web logs - multidimensional data
- network traffic - procotol-specific, bytes transferred
- text data - documents, tweets, multidimensional data

Each of those has its own way to be converted to a more useful datatype, according to the type of feature to input into the classifier:

- numeric to discrete {equi-width ranges, equi-log ranges, equi-depth ranges}
- discrete to numeric (binarization)
- text to numeric {stop word removal, **tf-idf**, multidimensional data}
- time series to discrete sequence data (SAX: equi-depth discretization after window-based averaging)
- time series to numeric data {discrete wavelet transform, discrete Fourier transform}



Term Frequency-Inverse Document Frequency

- Term frequency
  - Boolean "frequencies"
    - $tf(t,d) = 1$ if t occurs in d and 0 otherwise;
  - Logarithmically scaled frequency
    - $tf(t,d) = 1 + log f_{t,d}$, or zero if $f_{t,d}$ is zero;
  - Augmented frequency,

$$\mathrm{tf}(t,d) = 0.5 + 0.5 \cdot \frac{f_{t,d}}{\max\{f_{t',d} : t' \in d\}}$$

- Inverse document frequency

$$\mathrm{idf}(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

- Term frequency-inverse document frequency

$$\mathrm{tfidf}(t,d,D) = \mathrm{tf}(t,d) \cdot \mathrm{idf}(t,D)$$

Data Cleaning

Real world data is mostly composed of:

- Inaccurate data (**missing data**) - There are many reasons for missing data such as data is not continuously collected, a mistake in data entry, technical problems with biometrics and much more.
- The presence of **noisy data** (erroneous data and outliers) - The reasons for the existence of noisy data could be a technological problem of gadget that gathers data, a human mistake during data entry and much more.
- **Inconsistent data** - The presence of inconsistencies are due to the reasons such that existence of duplication within data, human data entry, containing mistakes in codes or names, i.e., violation of data constraints and much more.

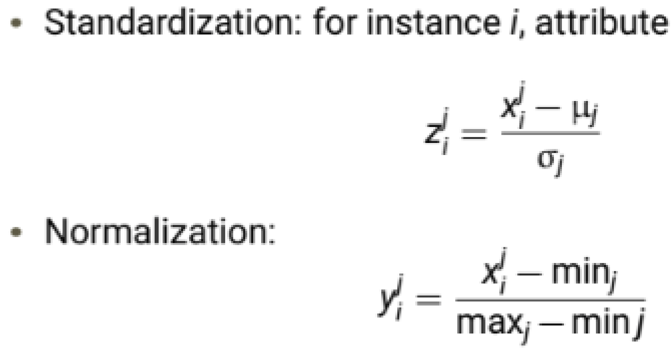
These inconsistencies are to be taken care of before moving on to the analysis phase.

Missing data implies incomplete data: the option are to ignore those entries, to estimate the missing values, or use algorithms that can handle missing values.

Noise can be easily dealt with thanks to **clustering** algorithms or *binning*.

Incorrect entries are more tricky to handle, due to their very different nature: there can be duplicates, inconsistencies, mistakes, etc. Generally, domain knowledge can give a better understanding of these errors and how to deal with them. Alternatively, data-centric methods can be used, which deal with inconsistencies.

Data belonging to the same domani but with different unit of measure (such as currency for example) should be **standardized** or **normalized**:



- Standardization: for instance *i*, attribute *j*:

$$z_i^j = \frac{x_i^j - \mu_j}{\sigma_j}$$

- Normalization:

$$y_i^j = \frac{x_i^j - \min_j}{\max_j - \min j}$$

Feature Selection and Transformation

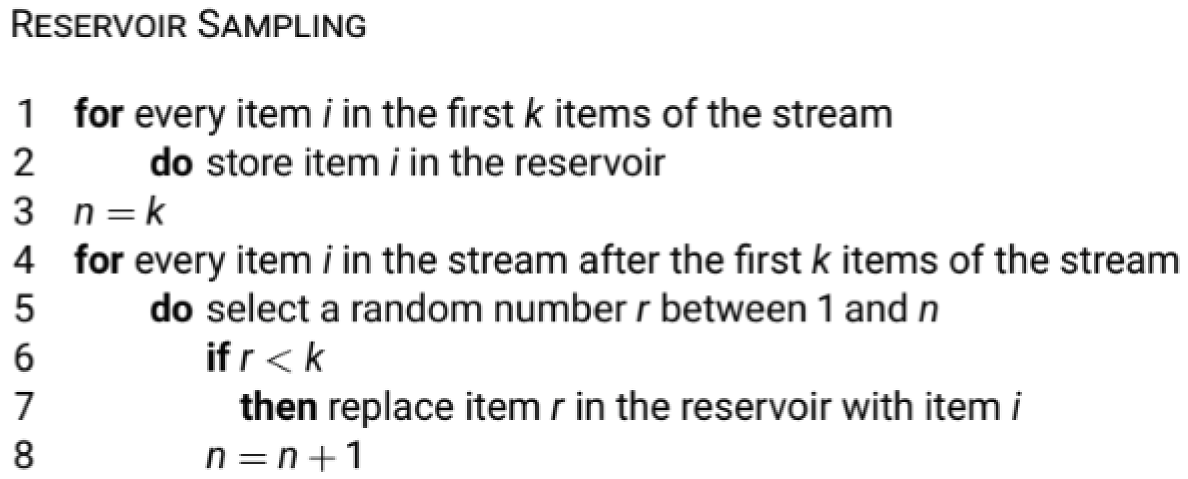
Once the data is clean and ready for analysis, it may include many features that do not concern our analysis. The process of **feature selection** selects a subset of relevant features for use in the model construction. The goals are four:

- simplification of models
- shorter training time
- avoid the *curse of dimensionality*
- reduce *overfitting* (reduction of *variance*)

Sampling the data is one of the possible approaches to reduce the dataset. It consists of taking just a portion of the original dataset, but with all of its feature. It can be compared to SQL *where* clause, filtering a set of instances.

Static data have many ways to be sampled, such as *sampling with/without replacement*, *biased sampling*, *stratified sampling*. Data streams instead are to be handled with the *Reservoir Sampling* method:

> Given a data stream, choose *k* items with the same probability, storing only *k* elements in memory.



RESERVOIR SAMPLING

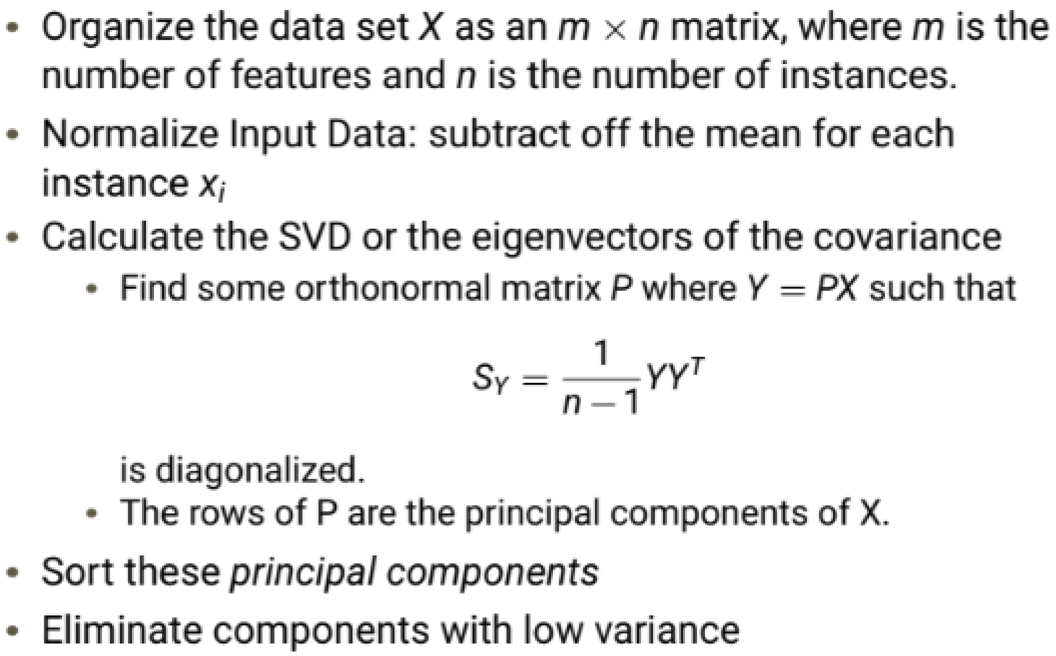
```
for every item $i$ in the first $k$ items of the stream
      do store item $i$ in the reservoir
$n = k$
for every item $i$ in the stream after the first $k$ items of the stream
      do select a random number $r$ between 1 and $n$
         if $r < k$
            then replace item $r$ in the reservoir with item $i$
         $n = n + 1$
```

An alternative approach is the **feature subset selection**. It consists of selecting some of the subsets of an instance, but for every instance of the starting dataset. It can be compared to the SQL *select* operator.

There are different types of feature subset selection: *supervised f.s.*, *unsupervised f.s.*, *biased sampling*, *stratified sampling*.

A last approach could be looking at data from a different point of view, in order to **reduce the dimensionality**: it's the main concept behind some algorithms, such as *PCA (Principal Component Analysis)*, *SVD (Single Value Decomposition)*, *LSA (Latent Semantic Analysis)*.

Among those, **PCA** is probably the most known. It computes the most meaningful basis to re-express a noisy, confused dataset. By computing a new basis, it could be possible to filter out the noise and reveal hidden dynamics. Just as if data were to be viewed from another perspective. A PCA algorithm follows these steps:



- Organize the data set X as an $m \times n$ matrix, where m is the number of features and n is the number of instances.
- Normalize Input Data: subtract off the mean for each instance x_i
- Calculate the SVD or the eigenvectors of the covariance
    - Find some orthonormal matrix P where $Y = PX$ such that

      $$S_Y = \frac{1}{n-1} YY^T$$

      is diagonalized.
    - The rows of P are the principal components of X.
- Sort these *principal components*
- Eliminate components with low variance