# DATA

In a dataset each attribute is a type, and it tells us what properties of the attribute are reflected in the values used to measure it. Knowing the type is very important because it tells us properties of the measured value and allows us to avoid foolish actions such as computing the average value of, for example postal code. We define 4 types of attributes:

| | ATTRIBUTE TYPE | DESCRIPTION | EXAMPLES | OPERATIONS |
|---|---|---|---|---|
| CATEGORICAL (QUALITATIVE) | NOMINAL | The values of a nominal attribute are just different names; i.e., nominal values provide only enough information to distinguish one object from another (=, ≠). | Area Code | mode |
| | | | Churn | entropy |
| | | | State | contingency |
| | | | eye color | |
| | | | gender | |
| | ORDINAL | The values of an ordinal attribute provide enough information to order objects (<, >). | {bad, good, excellent} | median |
| | | | grades | percentiles |
| | | | street numbers | rank correlation |
| | | | | run tests |
| | | | | sign tests |
| NUMERIC (QUANTITATIVE) | INTERVAL | For interval attributes, the difference between values are meaningful, i.e., a unit of measurements exists (+, -). | calendar dates | mean |
| | | | temperature in Celsius or Fahrenheit | standard deviation |
| | | | | Pearson's correlation |
| | | | | t and F tests |
| | | | | |
| | RATIO | For ratio attributes, both differences and ratios are meaningful, (*, /). | Day Mins | geometric mean |
| | | | Eve Mins | harmonic mean |
| | | | monetary quantities | percentiles |
| | | | length | variation |
| | | | electrical current | |

We can distinguish the attributes initially just based on the number of values they can take:

- Discrete: They can have a finite or countably infinite number of values and they can be:
  1. Categorical
  2. Numeric
  3. Binary
- Continuous: They have values which are real numbers, for example : Temperature, weight, height etc.

If an attribute is qualitative we can compute it's mode or the frequency of data.
If it is quantitative we can consider calculating the quantiles where a quantile of order 0<q<1 is the value in the sorted data such as q*100% of the values are on it's left, median is the quantile of order 1/2. We can also compute the mean, if the mean is very sensible to the outliers we use median as a more robust estimate of the middle of a set of values or we can use a trimmed mean, which is a mean excluding the outliers (to be exact we exclude sorted values above a certain percentage of value at the beginning and at the end). To measure the dispersion of the values we can also use the range which is the difference between the maximum value and the minimum value, but range can be misleading when values are concentrated in a narrow band of values so we use the variance instead, where the variance is calculated as $var=\frac{1}{n}\sum_{i=1}^{n}(x_i-x_{medio})^2$ e definiamo $\sigma=\sqrt{var}$ , or we can calculate the Absolute Average Deviation as $AAD=\frac{1}{n}\sum_{i=1}^{n}|x_i-x_{medio}|$ or the Median Absolute Deviation $MAD=median(|x_1-x_{medio}|,\dots|x_n-x_{medio}|)$

Since the variance depends on mean and it's sensitive to outliers so we define the InterQuartile Range (IQR) as the difference between 75% quantile and 25% quantile as a more robust estimate of the spread.

The Variance-Covariance Matrix is $COV(X,Y)=\frac{1}{n-1}\sum_{i=1}^{n}(x_i-x_{medio})(y_i-y_{medio})$ is a square, simmetric matrix. We define the Linear Correlation Coefficient $corr(X,Y)=\frac{COV(X,Y)}{\sqrt{var(X)}\sqrt{var(Y)}}$, this measure of association does not depend on the variance of each attributes, -1<corr(X,Y)<1 and bigger the value bigger the linear relationship between them.

We can also visualize them using histograms, dividing the value in bins and showing the number of records that fall into each bin. The bins can have different width and can be also an interval for quantitative values. For the Qualitative attribute when they are too much it's better to aggregate them. Another type of plot is the Box and Whiskers (Box Plot), which is applied to quantitative attributes only.

## Missing Replacements

The values can be missing for different reasons such as:

- It was not measurable
- It was not considered relevant when collecting the data
- It wasn't saved well
- The saving devices faulted
- Incosistent values with respect to other attributes

It is one of the first problems to solve before starting any data mining study. How can it be solved?

1. Record Removal: Not always the best choise assume 1% percent of the data is missing for every attribute and we have 100 attributes that than the probability of no missing data is $(0.99)^{100} = 0.366$!
2. Manual Imputation: It can be very time consuming.
3. Global Constant: such as -1 for age, to indicate that this value is missing.
4. Mode Replacement: Simply substitute the missing values with the mode.
5. Mean Replacement: The same as above except we use mean.
6. Conditional Mean Replacement: We calculate the mean only of those values which have the same value as the missing value one in another attribute.
7. Most Probable: We think/assume a linear correlation between one or more attributes and calculate there possible coefficient (excluding obviously the missing values) and use it to guess the missing data. We can use even more complex model not only the linear correlation, so we can manage the case of qualitative attributes also.

## Preprocessing

Data Preprocessing consists in a number of different strategies and tecniques that are interrelated in complex ways, the preprocessing is devoted to make the data more suitable for data mining.
The topics are:

1. Aggregation: It consists of combining two or more records into a single object, using a function to determine the new object, for example for qualitative attributes such as churn we can take the frequency of the elements.
   - Advantages:
     1. Smaller Data Sets so we can use more time consuming algorithms.
     2. Change of Scale/Scope to see a high level of view instead of a low-level view.
     3. Reduce Variance, usually the aggregated records are more stable, for the qualitative attributes it would be reduced entropy.
   - Disadvantes: Lose a lot of interesting details in the data.
2. Sampling: If the data we receive is very big we can decide to sample it, in order to use the most poerful and computationally demanding data mining models and algorithms. The sampling is a complex work, the key for an effective sampling is using a sample that works the same as the main dataset, ie it has the same property, such as mean, as the original

dataset, in this case we say that the sample is representative. To make a sampling we need to choose 2 things: Sample size and sampling tecnique. Many tecnique exists, we use bascially the following:

- ✔ Simple Random Sampling:
  - Without Replacement: each selected record is removed from the data set
  - With Replacement: each selected record is not removed form the data set

  When comparing the sample is small compared to the original one then two two methods are considered the same since their result are not much different. This method may not be the best method qualitative attribute with highly differente frequency of data, in such case the random sample can fail to represent the type of values with less frequency.
- ✔ Stratified Sampling: Is a better sampler but also more costly, if the sample is big enough, with not so different frequency in the qualitative data, random sampling can be better and faster. The stratified sampling is divided in two category:
  - Equal Number: An equal number of values is taken from the dataset regardless of their frequency.
  - Proportional: The proportions of the values in the sample is as near as possible to the original dataset.

Once decided the sampling tecnique we need to choose the sample size: If the sample is large we increase the probability of having a rapresentative sample, but elimate the advantage of sampling since the data set is still big.

At the same time if the sample is too small, we might miss the patterns or erroneous pattern can be detected. So detemining the proper sample size can be a very difficult task.

3. Dimensionality Reduction: In may cases the data sets can have a lot of attributed, for example if we create a word count data set with a lot of words we can have tens of thousand of attributes. Reducing the number of tttributed can have different advantages
   - ◆ Many data mining algorithms work better with lower attributes, since a lot of noise in the data is reduced due to irrelevant attributes.
   - ◆ Interpretability is increased since it depends on lesser attributes.
   - ◆ Graphical representation of the data is facilitated.
   - ◆ Amount of memory and the time used by algorithms are reduced.

   Many data analysis becomes significantly harder as the dimension of the data increases, and as it increases also the data becomes increasingly sparse thus occupying more space.

   Another tecnique is derived from the linear algebra, it consists in projecting the fata from a higher-dimensional space into a lower-dimensional space, it is called Principal Component Analysis (PCA), which finds new attribute (called Principal Component) that:
   - ✔ are a Linear Combination of the original attributes.
   - ✔ Are othogonal to each other.
   - ✔ Capture the maximun amount of variation in the data.

   We are usually asked to specify the number of components to retain or he percentage of variation we want to explain. A similar tecnique to PCA is Singular Value Decomposition(SVD).
4. Feature Subset Selection
5. Feature Creation
6. Discretization and Binarization:
   - ✔ Binarization: It may be useful to transform continuos and discrete attributes into one or more binary attributes, this process is called binarization. If we transform n discrete attribute, firstly we map them to first integers then we will need $s=\lceil \log_2(k) \rceil$ binary digits to represent them. If the atribute was ordinal we will need to maintain the order. If we cannot assume any correlation between them we need a binary atribute to represent each attribute in this case we need n binary digits to represent it.
   - ✔ Discetization: The best discretization depends on the algorithm being used and on the other attributes being considered. Furthermore the dicretization is considered in

isolation: Firsly we sort out the attribute, now we need to select where to locate split points and how many categories we need. The discretization can be:

- Unsupervised: does not exploit any information except the values of the continuos attribute to be discretized and can be divided in two catagories:
  1. Equal Width Unsupervised Discetization: User select the number k of intervals and the interval created will have the same width, so basically we will have k intervals with same width.
  2. Equal Frequency Unsupervised Discetization: The user decides the number of intervals k, we create intervals with more or less the same frequency (in case we cannot pefectly divide the number of attributes equally), so we will have k intervals with aproximately the same number of elements.
- Supervised: Exploits additional information to discretize the continuos attribute. In this the method places the split points in such a way that some measure of purity, computed exploiting the class attribute, is maximized.
  - The most common measure is the Entropy for the i-th interval as:

$$e_i = -\sum_{k=1}^{K} p_{ki} \log_2(p_{ki})$$

    where K is number of classes for the class attribute and $p_{ki}$ represents the probability of class k associated with the i-th interval. In particular if the i-th interval contains only records of a given class then $e_i = 0$ which is the maximum purity, and if it contains equally often all classes then $e_i$ is maximun which is the minimum purity.
    We define the overall entropy as:

$$E = \sum_{i=1}^{n} w_i e_i$$

    where $w_i = m_i / m$ and $m_i$ is the number of records in the i-th interval, m is the number of records and n is the number of intervals. We find the split point such as the overall entropy E is minimized equivalently the purity is maximized.

  In case of the categorical attributes it can sometimes have too many values, if it is ordinal we can use similar tecniques to those for the continuos attributes, if they are nominal the we need other approches:

  - We create a smaller set in which this nominal attribute can take place, for example if we had an attribute with a lot of cities we could create a smaller set using the states, but we need to exploit the domain knowledge and generate a new attribute. But not always the domain knowledge is available. In this case empirical approches, such as grouping values toghether only if such grouping results in imporoved classification performance or achieves some other objective.

7. Variable Transformation: it refers to a trasformation that is applied to all the values of a variable, there are different types of transformations:
   - ✔ Simple function: a simple mathematical function is applied to each value, such as log, sqrt, sin, cos, exp etc. but need two think if the order need to be maintain (use crescent function in this case such as log, exp and don't use sin, cos, $x^2$) and if the trasformation applies to all values, especially negative and 0(we can't use log with 0).
   - ✔ Normalization and Standardization: transform entire set of values to have a particular property. Let X with mean $\mu$ and standard deviation $\sigma$, then $Z = \frac{X - \mu}{\sigma}$ has mean 0 and standard deviation 1, it's the standardized of X. While the normalization is to project maximum value and minimum value to different number usuall 0 and 1, maintaining the order. It can be very useful, for example in the case where we need to compute the sum of different continuos attribute, this avoid one or few attributes taking

large value to dominate the new attribute sum. In the case of standardization, it depends on estimators like mean and standard deviation so is strongly affected by outliers so is often modified, in this case we can replace the mean with median and the standard deviation can be replaced by Absolute Average Deviation (AAD).

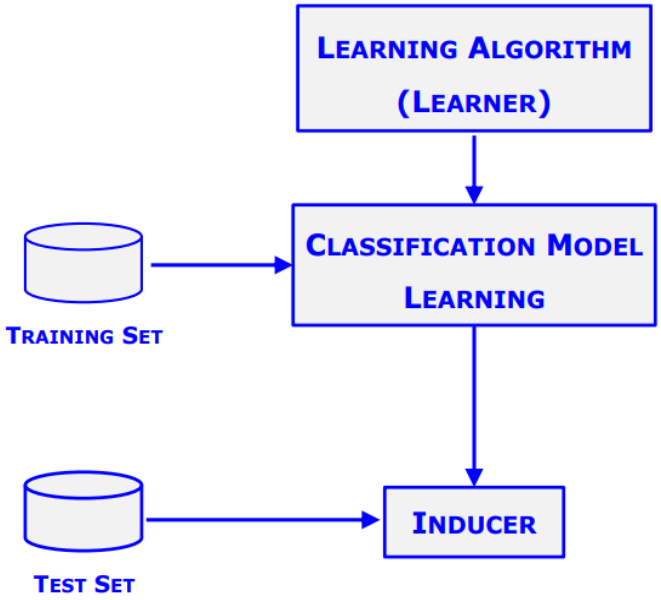
# Classification Modeling

We develop a classification model to predict an attribute, right now we will focus in the case this attribute is binary. We start with some terminology:
The attribute we need to predict is called class or target or output attribute, while the other attributes are called explanatory or input attributes. So we have the following schema:


A classification model solves the classification task/problem. It can be useful for the following purposes:

- Descriptive Modeling: can serve as explanatory tool to distinguish between objects of different classe.
- Predictive Modeling; predicts the class of an unkown record. It can be treated as a Black Box that automatically assigns a class label when presented with the attribute set of an unkown records.

We build a classification model form a data set using classification tecniques. We divide our data set into two: Training set and Test set.The **training set** consists of records whose class labels are known, and is used to build (learn and validate) a classification model and is later applied to the test set. The **test set** consists of records whose class label are assumed to be unkown.
We take the training set and use it to learn the classification model, the classification model learning takes place using the Learning Algorithm (Learner). The output of this learning is the Inducer, it is an instance of the classification model. The Inducer is queried to predict the values of class attribute for the test set records. The following schema is the one described above:




The problem is how do we measure the performance of the classification model?

Trivially, a first approch for the performance evaluation is based on the counts of the test record correctly and incorrectly predicted by the inducer. It is tabulated in the Confusion Matrix:

| | | Inducer Prediction (IP) | |
|---|---|---|---|
| | | -1 | +1 |
| Actual Class (AC) | -1 | TN | FP |
| | +1 | FN | TP |

Where TN – True Negative, FN – False Negative, TP – True Positive and FP -False Positive.
Using this matrix we can define a preformance metric defined as following :

$$Accuracy = \frac{TN + TP}{TN + FN + FP + TP}$$

and a complementary view is provided by using the error, defined as follows:

$$Error = \frac{FN + FP}{TN + FN + FP + TP}$$

It can be easily noted that Error = 1 – Accuracy, so they are basically the same, in the sense that they don't give any new information. The Confution Matrix contains the information needed to evaluate the performance of the classification model.

# Classification Performance Evaluation

Point estimate of the classification model's accuracy is not good enough to be confident that the classification model will provide reliable prediction when queried on unseen records, due to the risk of overfitting/underfitting.
The errors commited by a classification model are divided into two groups:

- **Training Error**: the number of records of the training set which are misclassified.
- **Generalization Error**: expected error on previously unseen records (Test Set)

A good classification model must not only fit the training set but, it must also accurately classify records it has never seen before (example the test set). So a good classification model must have low Training and Generalization error. In the case we have that the model fits the training set too well but has poor Generalization error we have the **model overfitting** fenomena. The opposite behavior when we have the model has poor training set error but has a higher generalization error we have the **model underfitting** fenomena.
The classification models are compared in terms of:

- Accuracy: It measures the capability of the classification model to give reliable prediction on new records and allows to select the instance of the classification model which likely provide the best prediction performance on new records.
  Given $D_T$ training set with t records and $D_{TS}$ test set with v records such that they have no element in common and let $D = D_{TS} \cup D_T$ and m = t + v, a good indicator of the accuracy achieved by a classifier is represented by the percentage of $D_{TS}$ which are correctly classified. Let $y_i$ be the class value associated with the instance $\underline{x_i}$ in $D_{TS}$ and $f(\underline{x_i})$ be the prediction then we define the loss function as:

$$L(y_i, f(\underline{x_i})) = \begin{cases} 0 & \text{se } y_i = f(\underline{x_i}) \\ 1 & \text{se } y_i \neq f(\underline{x_i}) \end{cases}$$

Then the accuracy can be computed as $acc(D_{TS}) = 1 - \frac{1}{v}\sum_{i=1}^{v} L(y_i, f(\underline{x_i}))$ and the error is

$$err(D_{TS}) = 1 - acc(D_{TS}) = \frac{1}{v}\sum_{i=1}^{v} L(y_i, f(\underline{x_i}))$$

- Speed: Classification algorithm differs from : Learning Time and Space Memory, selecting the type of classification depends on these two aspects. If a classifier requires high learning time or great memory, it can learnt using a sample, even though, we accept not to exploit all the possible information in spite of learning a given type of classification model that we think will ensure good performance.
- Robustness: A classification model can be robust or not with respect to:
  - Outliers
  - Missing Data
  - Variation of training and test data set
- Scalability: We say a classfiers is scalable if it is capable of learning from huge amount of data, this property is connected to the learning speed.
- Interpretability: It is relevant to extract rules to be simple and clearly understood by the domain expert for the problem in study.

Given $D_T$ training set with t records $D_{TS}$ test set with v records such that they have no element in common and let $D = D_{TS} \cup D_T$ and m = t + v, as before and let the acc( $D_{TS}$ ) same as before.
**Holdout** partitions the dataset D in two subsets through a simple random sampling (usually 2/3 for the training set and 1/3 for test set). In holdout we limit the data used to learn and the data to estimate the reliability. We can see that the estimate depends on the coise of the test set, thus we can under/over estimate the accuracy. A robust measure can be obtained through Iterated Holdout and Cross Validation.
The **Iterated Holdout** consists in iterating R times the holdout method. For each iteration r we extract random sample $D_T$ consisting of t records and we have $D_{TSr} = D - D_{Tr}$ and we iterate it R times calculating for every step $acc(D_{TSr})$ and defining accuracy as: $acc = \frac{1}{R}\sum_{r=1}^{R} acc(D_{TSr})$
In this way the accuracy is a little less biased since it depends less on the selected test set. So, iterated holdout significantly improves on holdout. However iterated holdout does not allow the number of times a given record is contained in the training set and int the test set, this could result in a strong bias wth specific reference to cases with data set contains mainly dominant records such i.e. outliers. This problem is solved by the cross validation.
A **K-folds Cross Validation** ensures that each record of the dataset D is included into the training set the same number of times and exactly one time in the test set.
We divide D into K disjoint subsets with almost a constant number of records. At the k-th iteration we have $D_{Tk} = \{D_1, \ldots, D_{k-1}, D_{k+1}, \ldots, D_K\} \wedge D_{TSk} = D_k$ we train the model on $D_{Tk}$ and use $D_{TSk}$ as the test set. For the accuracy we simply do the mean of accuracies: $acc = \frac{1}{K}\sum_{k=1}^{K} acc(D_{TSk})$
Typical values for K= 3, 5, 10. In case of scarse data we can use **Leave One Out Cross Validation(LOOCV)** obtained assuming each record as a partition of the dataset, the K=number of elements of D, is is an extreme case.
In K-fold Cross Validation, we usually ask that each partition of the dataset contains the same proportion for possible values of the class attribute, if the class proportions are strongly different, we adopt the stratified sampling, it can be of any kind proportional, equal probability.
We would like to compare the classifiers, but we cannot use the accuracy based on the test set only. For example if from an inducer A we had an accuracy of 0.85 with test set consisting of 30 records and from inducer B we had accuracy of 0.75 with test set consisting of 5000 records, which one is the better one? We need to estimate the confidence interval of the accuracy for both inducer, and

explain the difference in accuracy as a result of variation in the composition of the test sets using the testing of the statistical significance of the observed deviation.
Le the predicting of the value of the class attribute for a test record be a binomial experiment.
Given a test set $D_N$ consisting of N records and let:
X number of records correctly predicted by an inducer, and p, true but unknown, it's accuracy.
Then X has binomial distribution with mean = N*p and variance = N*p*(1-p).
We define the empirical accuracy as acc=X/N → this is still an aleatory variable with binomial distribution since it derives from one, and has mean = p and variance = p*(1-p)/N. The binomial distribution can be used to estimate the confidence interval for acc, approximating it with a normal distribution when N>>1. In this case the confidence interval with significance $\alpha$ is computed as:

$$P\left(-Z_{1-\frac{\alpha}{2}} < \frac{acc - p}{\sqrt{p \cdot (1-p)/N}} < Z_{1-\frac{\alpha}{2}}\right) = 1 - \alpha$$

Rearranging the above inequality, we compute the confidence interval with significance $\alpha$ for p, which is the unknown classification model's accuracy:

$$\left(\frac{acc + \frac{Z^2_{1-\frac{\alpha}{2}}}{2 \cdot N} - Z_{1-\frac{\alpha}{2}} \cdot \sqrt{\frac{acc}{N} - \frac{acc^2}{N} + \frac{Z^2_{1-\frac{\alpha}{2}}}{4 \cdot N^2}}}{\left(1 + \frac{Z^2_{1-\frac{\alpha}{2}}}{N}\right)}, \frac{acc + \frac{Z^2_{1-\frac{\alpha}{2}}}{2 \cdot N} + Z_{1-\frac{\alpha}{2}} \cdot \sqrt{\frac{acc}{N} - \frac{acc^2}{N} + \frac{Z^2_{1-\frac{\alpha}{2}}}{4 \cdot N^2}}}{\left(1 + \frac{Z^2_{1-\frac{\alpha}{2}}}{N}\right)}\right)$$

It can be non symmetrical since we are approximating the binomial with a normal distribution. We observe that the interval does not depend on p (obviously) but it depends on the N number of records of test set (the larger the N the more precise and symmetrical is the interval) and the empirical accuracy acc.
Let M1 and M2 be two classifiers:
We evaluate M1 on test set D1 containing n1 records, and it achieves error rate e1
We evaluate M2 on test set D2 containing n2 records, and it achieves error rate e2
We assume that D1 and D2 are two independent sets. We would like to see if the difference between e1 and e2 is statistically significant. So suppose n1 and n2 >>1 such as e1 and e2 can be approximated using normal distribution. And define d = e1 – e2, with mean=dt and variance = $\sigma_d^2$ the variance can be approximated with $\widehat{\sigma_d^2} = e_1(1-e_1)/n_1 + e_2(1-e_2)/n_2$ so the confidence interval for the true difference dt is $(d - z_{1-\alpha/2} * \widehat{\sigma}_d, d + z_{1-\alpha/2} * \widehat{\sigma}_d)$ now we have 3 cases:

- $(d - z_{1-\alpha/2} * \widehat{\sigma}_d, d + z_{1-\alpha/2} * \widehat{\sigma}_d) \ni 0$ It means that the difference d is not statistically significant al level $\alpha$, so the to model M1 and M2are not different in terms of error at level $\alpha$
- $d + z_{1-\alpha/2} * \widehat{\sigma}_d < 0$ in this case M1 is better than M2 at level $\alpha/2$ (since e1 < e2)
- $d - z_{1-\alpha/2} * \widehat{\sigma}_d > 0$ in this case M2 is better than M1 at level $\alpha/2$ (since e1 > e2)

Consider M1 and M2 to be compared using a K-folds cross validation (KF-CV), partitioning the dataset D into K disjoint subset with almost a constant number of records, in each step we calculated M1k and M2k inducer for the model M1 and M2 respectively at the k-th iteration. Each M1k and M2k are tested on the same partition, let e1k and e2k the error rate of M1k and M2k. Define dk = e1k – e2k, if K is sufficiently large then dk is normally distributed with:
mean = $d_t^{CV}$ and standard deviation = $\sigma^{CV}$, we approximate the overall variance through:

$$\frac{\widehat{\sigma^2_{d^{CV}}} = \sum_{k=1}^{K} (d_k - \bar{d})^2}{K(K-1)}$$ where $\bar{d} = \frac{1}{K}\sum_{k=1}^{K} d_k$, here for the confidence interval we use the student distribution finding the interval to be: $\left(\bar{d} - t^{K-1}_{1-\alpha/2} \cdot \bar{\sigma}_{d^{CV}}, \bar{d} + t^{K-1}_{1-\alpha/2} \cdot \bar{\sigma}_{d^{CV}}\right)$

where $d_{1-\alpha/2}^{K-1}$ is the quantile associated with confidence $1-\alpha$ and K -1 degrees of freedom, we use the confidence interval to establish statistical difference between M1 and M2 and to do so we proceed same as above, if 0 spans in such interval then M1 and M2 are not statistically different and so on.

# Classification Imbalance Problem

Suppose in a data set with churners, there are only 15% of churners, you create a model that classifies every user as a non churner (like a Zero-R model), in this case we have an accuracy of 85% but such model is useless and so is the measure of the performance of the model, since accuracy treats every class as equally important, but usually the rare class is considered more interesting than the majority class, we will denote the rare class as the positive class and the majority class as negative class. We take in account the confusion matrix

| | | Inducer Prediction (IP) | |
|---|---|---|---|
| | | -1 | +1 |
| Actual Class (AC) | -1 | TN | FP |
| | +1 | FN | TP |

We define the following values:

a) TNR, True Negative Rate or Specificity: TNR = TN/(TN+FP), it is the fraction of negative records correctly predicted;
b) TPR, True Positive Rate or Sensitivity; TPR = TP/(TP+FN), it is the fraction of positive records correctly predicted;
c) FPR, False Positive Rate, FPR = FP/(TN+FP), it is the fraction of negative records predicted as a positive class;
d) FNR, False Negative Rate, FNR = FN/(TP+FN), it is the fraction of positive records predicted as a negative class.

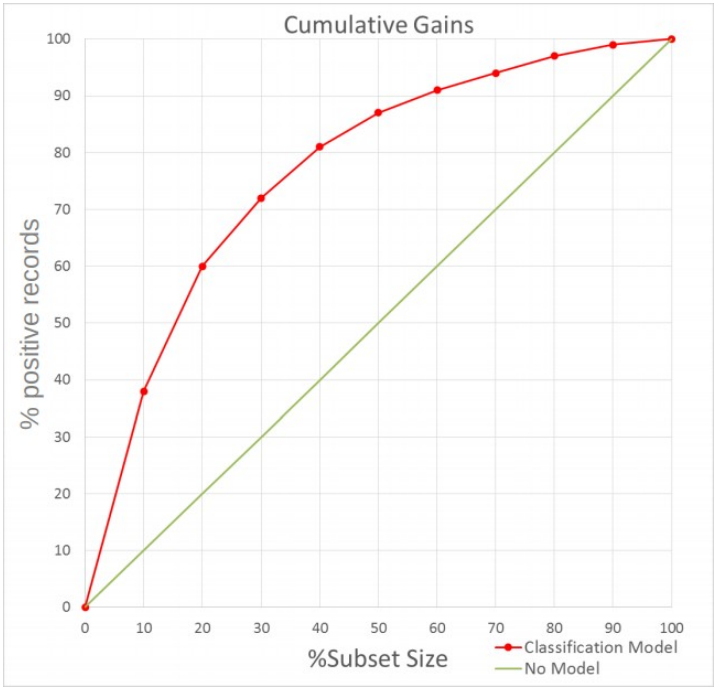
We define also the **Recall** and **Precision** as follows:
Precision, p = TP/(TP+FP), it represents the number of records actually positive of those classified as such, the higher the precision the lower the number of false positive errors committed.
Recall, r = TP/(TP+FN), it represents the number of positive records correctly predicted. Larger recall means dew positive records misclassified as negative class, r = TPR. It is often possible to construct models that maximize one of the two metrics between p and r but not the other, so we define another metric which summarizes both metrics together: $F_1 = \frac{2*r*p}{r+p}$ it is the harmonic mean between recall and precision, higher value of $F_1$ Measure ensures that both recall and precision are high. We can generalize the $F_1$ Measure with: $F_\beta = \frac{(\beta^2+1)*r*p}{r+\beta^2 p}$ , in this case we have that if $\beta = 0 \, then \, F_\beta = p$ and if $\beta = \infty \, then \, F_\beta = r$ .
Suppose that we want to give more weight when we misclassify a churner than when we misclassify a non churner, in this case we introduce a cost matrix:
And we define the cost as:
Cost = C--*TN + C-+*FP + C+-*FN + C++*TP
Oss: If C-+ = C+- ie the matrix is symmetrical the the cost is proportional to the accuracy.
If we have the cost matrix we want to build a model such as the cost is the lowest possible but it should have a good accuracy as well. But the costs are rarely known so we ponder various different scenarios.

| | | PREDICTED CLASS | |
|---|---|---|---|
| | | -1 | +1 |
| ACTUAL CLASS | -1 | $C_{--}$ | $C_{-+}$ |
| | +1 | $C_{+-}$ | $C_{++}$ |

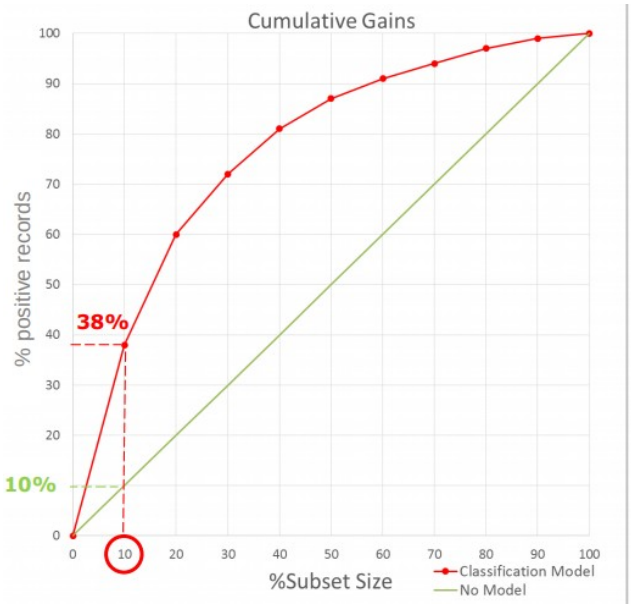
We take a subset of the data set such that we are able to classify 60% of all the positive class, while if we were to do a random sampling we would have 37.5%, we define this increase as the **lift factor** yielded by the classification model (M) with respect to the random sampling as the ratio between 60 and 37.5, so the positive class predicted by M and random sampling respectively.
What happens when I change the classification model and the parameters of the subset, how can I say which model is more profitable? To know the profitability we need to know the costs involved, so we follow another path: Given a classification model M, which outputs the probabilities for the positive class, we find the subset of the dataset that have:

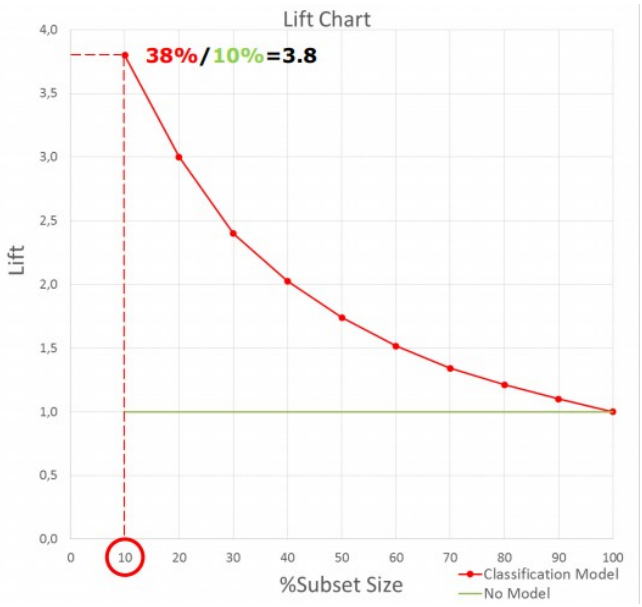
- High proportion of positive records
- Higher that the data set itself

We sort the dataset in descending order of probability of being a positive class, we take the subset starting at the top and the compute the lift factor. We plot a graph with % positive records found on the y axis and the % of the subset size on the x axis, this plot is called the **cumulative gain**:

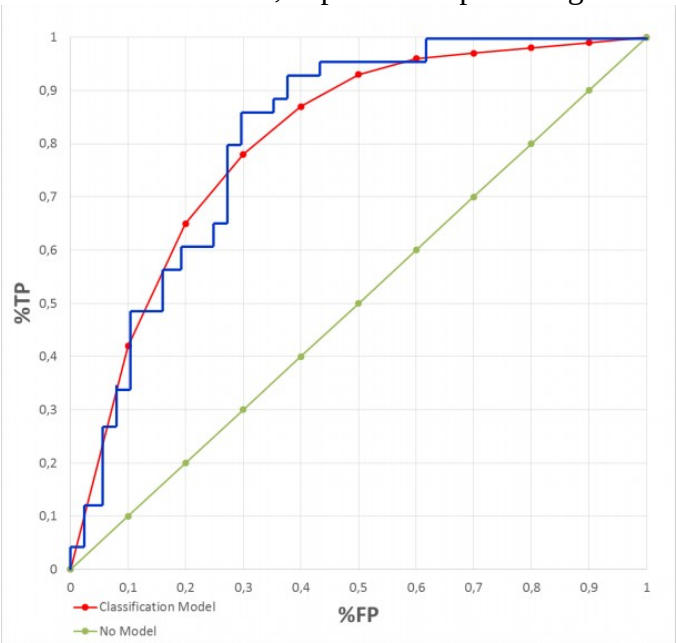



In this plot the ratio between the green ordinate and red ordinate, with the same %Subset Size is the lift factor. We can also plot the **lift chat** calculating that ratio:



The Lift charts are related to a graphical technique for evaluating the models known as **Receiver Operating Characteristic Curve (ROC Curve)**.
ROC Curve plots on the y axis the number of positive records included in the subset, expressed as the total number of positive record (TPR, %TP) and on the x axis the number of negative records included in the subset, expressed as percentage of the total number of negative records (FPR, %FP):




Normally The ROC curve is like the blue line because of the dependence on the particular subset, it can be reduced applying cross validation, this way it gets near it's theoretical form, the red one.
Now we can classify a model by studying the ROC curves, the upper curve is the better model, usually there is no global better model, in fact, two ROC curves can cross each other so that before a certain %FP one model is better and after that point the other one is better.

# Feature Selection

We wish to reduce the number of attributes we are using to improve the classifier's interpretability. To do that we need to discover which attributes are redundant or irrelevant.
There are different approaches:

- Brute Force: apply all the possible subsets of the available attributes as the input attribute for the classification model. The only problem is that the number of subset given n attributes are $2^n$;
- Embedded: attributes selection occurs as a byproduct of classifier learning (BN, DT, …);
- Filter: attributes are selected before learning the classifier;
- Wrapper: the classifier is used to find the optimal subset of the available attributes.

The difference between filter and wrapper is that the filter uses an objective function to determine the useful attributes and after selecting them learns a classifier model, while the wrapper learns the model every cycle with the attributes while selecting the most useful ones. It is obvious that a wrapper is slower than a filter.
A filter can be:

- Univariate:
  - Choose an association measure between candidate attributes and class attribute, for example gain ration, mutual information.
  - Sort the candidate attributes according to the association measure.
  - Select the R best candidates as input attribute for the classification learning.
  - It is likely that irrelevant attributes are correctly identified while it may not correctly identify the redundant attributes.
- Multivariate:
  - Jointly identify irrelevant and redundant attributes.
  - A good subset of attributes contains attributes that are strongly associated with the class attribute but uncorrelated among them.
  - Uses symmetric uncertainty measures or correlation based measure.

The univariate filters can be:

- parametric: includes t-test, ANOVA, Mutual Information
- Non parametric test: includes Mann-Whitney, Kruskal-Wallis and Permutation test

The multivariate filters mainly includes Correlation Feature Selection, Relief and Blanket Set.

Now we present the advantage and disadvantage of the univariate and multivariate form:

| | Advantages | Disadvantages |
|---|---|---|
| **Uni-Variate** | speed<br>scalability<br>independent on the classifier | ignore that attributes can be dependent<br>ignore interactions with the classifier |
| **Multi-Variate** | model dependency between attributes<br>independent on the classifier<br>computational cost compares favorably to wrapper | slower than Uni-Variate techniques<br>less scalable than Uni-Variate Techniques<br>ignore interactions with the classifier |

The main Advantages of the feature selections are:

i. Reducing of the cost of data collection;
ii. Reduction of the inference time, time required by the classifier to predict the value of the class attribute;
iii. Increased interpretability;
iv. Increased Accuracy.

The main Motivations are:

i. to avoid the overfitting phenomena;
ii. to develop a faster and cost-efficient classifier;
iii. to improve the understanding of the data generating process.

How does the features selection algorithm works? We take the available attributes, choose a search strategy (brute, filter, wrapper), take a subset of attributes, evaluate them and if a certain criteria is not met we return to the search strategy. Once the selected criteria is met we use the selected attributes a go to a validation procedure.

The **Feature creation** from the original dataset, can capture the relevant information much more effectively. Furthermore the number of attributes can be smaller that the original ones allowing to reap the benefits of dimensionality reduction. Some of the methodologies are:

- Feature Extraction: consists in creating of new features set from the original raw data, with this a much broader set of classification models can be applied, the feature extraction is highly domain specific.
- Mapping the data in a new space: thanks to this procedure a totally different view of the data can reveal interesting features which increase understanding and classification performance.
- Feature Construction: the available attributes contain information that is not suitable for some classification algorithms, one or more new features constructed out of the original attributes can be more useful than the original features.

Some ML models allow hyper-parameter, for example in the NN we had the regularization parameter $\lambda$, how can we select the proper value of the hyper-parameter. We can't use the training set otherwise it would be biased since we use it to learn the model, i.e., we already used it's information. The value of $\lambda$ is selected to optimize some performance measure, in order to avoid the optimistic view of the performance in the training set i.e. we could we overfitting the model. Usually we introduce a third division of the dataset, including the validation set. If the dataset is small it may not be possible to apply it. The hyper parameter is optimized using the validation set, and then the test set can provide an unbiased estimation of the performance, we can also use it with the Hold-out, iterated Hold-out and cross validation.

In case of the Filter (both multivariate and univariate), feature selection is performed using the Train/Test set scheme: Relevance and/or redundancy are estimated using the training set, once selected we use the training set to learn a classifier and estimate is with a test set.
In case of a Wrapper, feature selection is performed using Train/Validation/Test scheme: Relevance and/or redundancy are estimated using the train and validation set.
Then we use the validation set to optimize the performance measure when different attributes are used by the classification model. Selecting the optimal subset of attributes is the same as selecting the regularization parameter $\lambda$ in the NN.
Once selected the Train and Validation set are merged together to learn the classifier, usually the same used to do the feature selection. The we estimate is using the test set.

# Clustering

Given a dataset we want to partition our rows into meaningful groups to understand them, for this we use cluster analysis.
The utility of cluster analysis:

- Understanding: understanding the classes, or conceptual meaningful groups of the objects that share common characteristics, it plays an important role in how people analyze and describe the world. Some examples are the hierarchical classification of the living beings, segmenting customers for marketing activity.
- Utility: provides an abstraction from individual data objects to the cluster in which these data object reside. More precisely, the goal is to find the most representative cluster prototypes. Secondary goals can be: summarization(for the dimesionality reduction), compression(to each cluster prototype is associated an integer) and efficiently finding nearest neighbors.

The goals are:

- The objects within a group be similar to one another and different from the objects of the other groups.
- The greater the similarity within a group and the greater the difference between groups the better or more distinct the clustering.

Yet the definition of cluster is imprecise and it depends on the nature of the data and the desired result.
Clustering can be:

- Partitional vs Hierarchical
  - A partitional clustering is a division of the set of data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset.
  - Hierarchical clustering permits clusters to have sub-clusters, in this case the clustering is a set of clusters that are organized as a tree.
- Exclusive vs Overlapping vs Fuzzy
  - The exclusive clustering assign each data object to a single cluster.
  - Overlapping clustering allows a data object to be placed in more that one cluster.
  - Fuzzy clustering allows a data object to belong to each cluster with a membership weight between 0 and 1.
- Complete vs Partial
  - A complete clustering assigns every data object to a cluster.
  - The partial clustering is not constrained to assign every data object to a cluster. The main motivation is that some data objects may not belong to well-defined groups(for example noises and/or outliers).

Different notions of a cluster:

- Well Separated Cluster: Given a cluster, each object in the cluster is closer to every other object in the same cluster that to any object not in the cluster. In this case we can use a threshold to specify that all objects in a cluster must be sufficiently close to one another.

Idealistic definition of cluster, satisfied only when the data contains natural clusters that are quite far from each other, and it does not need to be globular, can have any shape.

- Prototype-Based Cluster: Given a cluster, each object is close to the prototype that defines the cluster than to prototype of any other cluster. For continuous attribute, the prototype is often the centroid(the average of all values in the cluster), when the centroid is not representative then we use medoid(most representative object of the cluster). It's usually globular.
- Density-Based Cluster: A custer is a dense region of objects that is surrounded by a region of low density. Used when the cluster are irregular or intertwined, and when noise and outliers are present.
- Graph-Based Cluster: If the data is represented as a graph, nodes are the objects and links represent the connection among object, then a cluster can be defined as a connected component. Tends to be globular.

How does the process works?

Firstly we start we the feature selection to insure the retention of the meaning of the original attributes. Now the Feature Extraction is capable of producing feature that could be of better use in covering the data structure. However, it may generate features that are difficult to interpret.
Ideal features should be of use in distinguishing patterns belonging to different clusters, immune to noise and easy to obtain and interpret.
After we determine the proximity measure and construct a criterion function.
Once the proximity measure is determined, clustering can be formulated as an optimization problem with a specific objective function.
Given a clustering algorithm on a data set, it can always produce a partition whether or not there really exists a particular structure in the data. Different clustering approaches usually lead to different clusters of data, even for the same algorithm if we change a parameter or the presentation order of input, it may affect the final result.
Therefore, we need an effective evaluation standard and criteria to have a degree of confidence for our result, regardless of the algorithm that we choose.
Last step is the to provide users with meaningful insights from the original data, so that they can develop a clear understanding of the data and therefore effectively solve the problem encountered.
The clustering is helps suggesting hypotheses, for another experiment or analysis, a set of cluster is not itself a finished result but only a possible outline.

Cluster analysis is rooted into concepts of similarity and dissimilarity. In many cases once the similarities and dissimilarities have been computed, the available data set is not needed anymore.
The **similarity** between two objects is a numerical measure of degree to which the two objects are alike. The more two objects are alike the higher the similarity, it is usually non negative and between 0 (no similarity) and 1 (complete similarity).
On the other hand the **dissimilarity** is the measure to which two objects are different. It might take values also on [0,Inf) or [0,1] it depends on our choice. Proximity is defined to refer either similarity or dissimilarity .Transformations are often applied to convert similarities into dissimilarity, we introduce **Proximity** which is usually defined on [0,1], in such way to use a scale in which proximity indicated the fraction of similarity (or dissimilarity) between two records.
Generally $s' = \frac{s - s_{min}}{s_{max} - s_{min}}$ is the transformation used to convert similarities (or dissimilarities) on the interval [0,1], if it takes value on [0,Inf) then we need a non-linear transformation such as $d' = \frac{d}{1+d}$ , at this point if they falls in the interval [0,1] we can define d = 1 - s (or s = 1 – d), another approach could saying that d = -s (or s = -d).

Here are a few examples of possible proximity:

| ATTRIBUTE TYPE | | DISSIMILARITY | SIMILARITY |
|---|---|---|---|
| CATEGORICAL (QUALITATIVE) | NOMINAL | $d = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{otherwise} \end{cases}$ | $s = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases}$ |
| | ORDINAL | $d = \frac{\lvert x - y \rvert}{n - 1}$ | $s = 1 - d$ |
| NUMERIC (QUANTITATIVE) | INTERVAL | $d = \lvert x - y \rvert$ | $s = -d \quad s = exp(-d)$ <br> $s = \frac{1}{1 + d}$ |
| | RATIO | | $s = 1 - \frac{d - min\_d}{max\_d - min\_d}$ |

In the case of a nominal attribute it is a little bit more complicated, for example what would mean for eye colors of a person to be similar. Also the order should be taken in account in cases of the quality of a product which could take value in {poor, fair, OK, good, wonderful}. In the last case we could associate an integer to each value, poor =1, fair = 2, …, but we are assuming that each value is equidistant. While for the numeric value the natural choice is the distance as a measure of dissimilarity.

Some proximity measures:

- Minkowski Distances, distances are dissimilarities with certain property, in this case we assume all attributes to be numeric. Usually the similarity does not satisfy the triangle inequality but does satisfy the symmetry(s(x,y)=1 iff x=y) and non negativity.
  - Manhattan Distance: $d(x,y)=\sum_{k=1}^{n}|x_k-y_k|$
  - Euclidean Distance: $d(x,y)=\sqrt{\sum_{k=1}^{n}(x_k-y_k)^2}$
  - Supremum Distance: $d(x,y)=\lim_{r\to\infty}\sqrt[r]{\sum_{k=1}^{n}(|x_k-y_k|)^r}$
- Simple Matching Coefficient: $SMC(x,y)=\frac{|matching\,attributes|}{|attributes|}$

  Let the following:
  f00 = #attributes which are equal to 0 both for x and y
  f10 = #attributes which are equal to 1 for x and 0 for y
  f01 = #attributes which are equal to 0 for x and 1 for y
  f11 = #attributes which are equal to 1 both for x and y
  so the SMC = (f11+f00)/(f11 + f10 + f01 + f00) from here we deduct that: SMC takes value on [0,1] and SMC = 0 when x!=y completely while is  = 1 when x=y.
  This measure counts both presences and absences equally. It should be used when the attributes are symmetric binary(the value 0 and 1 are equally valuable).
- Jaccard Coefficient: $J(x,y)=\frac{|matching\ presences|}{|attributes\ except\ 00\ matches|}=\frac{f_{11}}{f_{11}+f_{10}+f_{01}}$

This is to be used when all attributes are asymmetric binary. For example we associate the value 1 which means the item was purchased while 0 means it was not purchased, we need not to know the items that were not purchased.

While the following attribute x = (0, 0, 0, 0, 1, 0, 0, 0, 0) would be similar to
y = (0, 0, 0, 0, 1, 0, 0, 1, 0) in SMC,
In the case of x = (1, 1, 1, 1, 0, 1, 1, 1, 0)
y = (1, 1, 1, 1, 1, 1, 1, 1, 0), in the case of of SMC we will have the same result as the one above.
It is clear that in the example of products in a supermarket the second one would be similar but the first two clients have nothing in common, in this case Jaccard Coefficient would be way more better.

An extension of the Jaccard Coefficient is the Tanimoto Coefficient (Extended Jaccard Coefficient):

$EJ(x,y)=\frac{x*y}{(\|x\|)^2+(\|y\|)^2-x*y}$ , it reduces to Jaccard coefficient in case of binary attributes.

- Cosine Similarity: $\cos(x,y)=\frac{x*y}{\|x\|*\|y\|}$ it take value on [-1,1], is equal to 1 when x is the same as y and is equal to 0 when x and y share no terms, it ignore the 00 values like the Jaccard Coefficient, but is able to handle non binary attributes. It is useful for comparing sparse records, it is widely used in Information Retrieval where documents represented as a vectors of counts must be compared. To be used when all attributes are numeric.
- Correlation: $corr(x,y)=\frac{cov(x,y)}{stdv(x)*stdv(y)}$ takes values in [-1,1] is 0 when x and y are uncorrelated and equal to 1 in absolute value when x and y are perfectly correlated.

There are several issues related to proximity measures:

- How to handle the case in which attributes have different scales and/or are correlated
  If they have different scale, unless we didn't proceed with normalization, one might dominate the other distance in the euclidean distance.
  While if they are correlated and have different range of values, and the distribution is approximately gaussian, the Mahalanobis Distance is used:
  $Mahal(x,y)=(x-y)\sum^{-1}(x-y)^T$ where $\sum^{-1}$ *is* the inverse of the variance-covariance matrix of x and y.
  Mahal(x,y) takes value on [0,Inf] it is equal to 0 when x and y are the same while more x and y are different, greater the value of Mahal.
- How to calculate proximity between records composed of different types of attributes
  - assume you have n attributes, for each attribute k, compute the similarity $s_k(x,y)$ such as it takes value on [0,1]
  - Define a indicator variable $\delta_k$ for the attribute k as follows:

$$\delta_k=\begin{cases}0 \text{ if the } k^{th} \text{ attribute is asymmetric and both records have value 0,}\\ \quad \text{or at least one of the records has a missing value for the } k^{th} \text{ attribute}\\ \\ 1 \text{ otherwise}\end{cases}$$

  - compute the overall similarity as $similarity(x,y)=\frac{\sum_{k=1}^{n}\delta_k s_k(x,y)}{\sum_{k=1}^{n}\delta_k}$
- How to handle proximity calculation when attributes have different weight, i.e., when not all attributes contribute equally to the proximity of records.

If the sum of weight $w_k$ sum to 1, then we can modify the formula above to:

$$similarity(x,y)=\frac{\sum_{k=1}^{n} w_k \delta_k s_k(x,y)}{\sum_{k=1}^{n} \delta_k}$$

- Also the Minkowski distance can be modified as follows: $d(x,y)=\sqrt[r]{\sum_{k=1}^{n} w_k(|x_k-y_k|)^r}$

Selecting the right measure the following observation are useful:

- Dense and Continuous Data → metric distance measures are often used
- Sparse Data, Asymmetric Binary → similarity measures that ignore 00 matches cosine, Jaccard and Extended Jaccard.

# Clustering Evaluation

In a Cluster analysis study the following issues must be addressed:

- Which similarity or proximity measure must be used.
- Choose between Exclusive or Overlapping or Fuzzy clustering.
- Implement Complete or Partial clustering.

We have that from external advisor that the ideal number of groups in clustering should be between $K_{min}$ and $K_{max}$, so we try different cluster algorithms, with different parameters or the type of linking for agglomerative hierarchical clustering or Eps and MinPts for DBSCAN.
In the supervised classification, many measures(accuracy, recall) and method (cross validation, ….) are available to evaluate and compare the models that are well-accepted.
We don't have a well-developed cluster evaluation method or measure. One of the main problems are that a clustering algorithm is designed to find clusters, even if that data set has no natural cluster structure.
The most important issues for cluster validation are:

- Determine the clustering tendency of a set of data, i.e., distinguishing whether non-random structure actually exists in the data.
- Determine the correct number of clusters

To address the issues of cluster validation we use evaluation measures or indices, which are classified into the following three types:

- External or Supervised: measures the extent to which the clustering structure discovered by a clustering algorithm matches some external structure.
- Internal or Unsupervised: measures the goodness of a clustering structure without respect to external information. The are divided in two more categories:
  - Cohesion Measures (compactness, tightness), which determine how closely related the two objects in a cluster are
  - Separation Measures (isolation) , which determine how distinct or well-separated a cluster is from the other clusters.
- Relative: compares different clustering or clusters. It can be supervised or unsupervised. Such measures are not a separate type of cluster evaluation measure, but are instead a specific use of such measures.

**External or Supervised indices:**

Let $P = \{P_1, \ldots, P_r\}$ be a partition of m objects into r categories and let $C = \{C_1, \ldots, C_k\}$ be the partition obtained with a clustering algorithm into k clusters.
In the case of Supervised or External Indices compare P to C, we have 4 cases:

1) x and y belong to the same cluster of C and same category of P. number of pair = (a)
2) x and y belong to the same cluster of C but different category of P. (b)
3) x and y belong to different clusters of C but to the same category of P. (c)
4) x and y belong to different clusters of C and to different category of P. (d)

We have that total number of pairs amount to m*(m-1)/2 = a + b + c + d.
The different indices for the external validity are:

- RAND : R = (a+b)/M $R \in [0,1]$
- Jaccard : J = a / (a+b+c) $J \in [0,1]$
- Fowlkes and Mallows $FM = \sqrt{\frac{a}{a+b} \times \frac{a}{a+c}}$ $FM \in [0,1]$
- Γ Statistics $\Gamma = \frac{M \times a - (a+b) \times (a+c)}{\sqrt{(a+b) \times (a+c)(M-a-b) \times (M-a-c)}}$ $\Gamma \in [-1,1]$

The larger the values, the more similar are C and P, the better the quality of the clusters applying different clustering algorithm.

**Internal or Unsupervised indices**:

Many internal measures of indices of cluster validity for partitional clustering are based on the notion of cohesion or separation. Given a set of k clusters : $C_1$, …. ,$C_k$ we define the overall validity as : $\sum_{i=1}^{k} w_i \, validity(C_i)$ , where the validity function can be cohesion, separation or any combination of them. The weights $w_i$ will vary on the clustering validity measure. In some cases they are set to be 1 or are the cardinality of the corresponding set, while in other cases they reflect a more complicated property, such as the square root of the cohesion. When validity = cohesion higher values are better while in case validity = separation lower values are better.

For graph based clusters, the cohesion of a cluster can be defined as the sum of the weights of the links in the proximity graph that connects points within the cluster.
$cohesion(C_i) = \sum_{x,y \in C_i} proximity(x,y) = \sum_{x,y \in C_i} similarity(x,y)$ , when considering the attribute space, we have that similarity is inversely proportional to dissimilarity/distance, therefore cohesion and similarity are maximized when dissimilarity/distance is minimized.

For graph based clusters, the separation of a cluster can be defined as the sum of the weights of the links from point in one cluster to the points in the other cluster.
$separation(C_i, C_j) = \sum_{x \in C_i, y \in C_j} proximity(x,y) = \sum_{x \in C_i, y \in C_j} similarity(x,y)$ , also in this case we have that when considering the attribute space, we have that similarity is inversely proportional to dissimilarity/distance, therefore cohesion and similarity are minimized when dissimilarity/distance is maximized.

For the prototype-based clusters, the cohesion of a cluster can be defined as the sum of the proximities with respect to the prototype (centroid or medoid we indicate it with $c_i$) of the cluster.
$cohesion(C_i) = \sum_{x \in C_i} proximity(x, c_i) = \sum_{x \in C_i} similarity(x, c_i)$

For the prototype based clusters, the separation between two clusters can be measured by the proximity of the two clusters prototypes.
$separation(C_i, C_j) = proximity(c_i, c_j) = similarity(c_i, c_j)$ , another measure could be:
$separation(C_i) = proximity(c_i, c) = similarity(c_i, c)$ , where c is the overall prototype.

Now given the validity function we have that $overall\ validity = \sum_{i=1}^{k} w_i \, validity(C_i)$ , however the weight $w_i$ need to be fixed, given $m_i$ the number of elements of a cluster, usually we use $1/m_i$ for the graph-based cohesion, 1 for the prototype-based cohesion and $m_i$ for prototype separation.
Potentially any measure of cluster validity can be used as an objective function for a clustering algorithm and vice versa.

Many of these measures of cluster validity can be used to evaluate individual clusters or objects, and we could rank individual clusters according to their specific value of cluster validity. A cluster that has high value of cohesion can be considered better than one that has lower value. And use this information to improve the quality of the cluster analysis process. For example if a cluster is not very cohesive the split it into several clusters and if two clusters are relatively cohesive but not well separated then merge the into a single cluster.
We can also evaluate the objects within a cluster in terms of their contribution to the overall cohesion or separation of the cluster, in particular the object that contribute more to the overall cohesion or separation of a cluster are those near the interior of the clusters while the objects that contribute less to the overall cohesion or separation of a cluster are near the edge of the cluster.
A measure to evaluate the contribution of a record to the clustering solution is the **Silhouette Coefficient,** which is a cluster evaluation measure which exploits the concepts of the interior and edge of a cluster to evaluate data points, cluster and the entire set of clusters. It combine the cohesion and separation and for the i-th record is defined as $s_i = \frac{b_i - a_i}{max(a_i, b_i)}$ , where $a_i$ is the average distance of the i-th record to all other objects in its cluster, while $b_i$ is the minimum of the average distances of the i-th object to all the objects in each given cluster different from the cluster to which the i-th object belongs to. We have that $s_i \in [-1,1]$ and the negative coefficient means that the average distance to points in its cluster is greater than the minimum average distance to points in another cluster ($a_i > b_i$), but we want the coefficient to be positive ($a_i < b_i$), and for $a_i$ to be as close to 0 as possible since $s_i = 1$ when $a_i=0$.
We can compute the average silhouette coefficient of a cluster by taking the average of all $s_i$ in the cluster, define an overall measure of goodness of a clustering bu computing the average silhouette coefficient of all points of all dataset.
The silhouette coefficient is defined for partitional clustering while for the hierarchical clustering a different measure is used : Cophenetic Correlation Coefficient. It measures the degree of similarity between the Proximity matrix P and the Cophenetic Matrix Q whose elements record the proximity level where pairs of data points are grouped in the same cluster for the first time. It's value lie in [-1,1] and the more the value is closer to 1 the more significant is the similarity between P and Q the better the fit of hierarchy to the data. However, for average linkage, even large values of the cophenetic correlation coefficient cannot assure sufficient similarity between the two matrices.

The Validity paradigm: We note that both external and internal criteria are closely related to statistical methods and hypothesis test. The Validity paradigm, for a clustering structure, is based on the null hypothesis: There is no structure on the dataset.

1) Then the validity paradigm starts by choosing the clustering structure and validation type: external or internal.
2) The next step is to determine a validation index.
3) Define a null hypothesis of no structure: There are mainly 3 commonly used null hypothesis:
   i. Random Position Hypothesis, all the location of the m data points in some specific region of a n-dimensional space are equally likely → For ratio Data
   ii. Random Graph Hypothesis, all [m * m] rank order proximity matrices are equally likely → Used for ordinal proximities between pairs of data objects
   iii. Random Label Hypothesis, all permutation of the label on m data points are equally likely → For all Data Types.
4) Establish the baseline distribution under the null hypothesis achieved using monte carlo analysis and bootstrapping.
5) Calculate the index associated to the specific clustering solution we have developed.
6) Test the hypothesis of no structure, we compare this value to the baseline of the value or more precisely to a specific quantile of the empirical baseline distribution.

You hope to reject the null hypothesis which would mean that there is some structure in the developed clustering.

## Relative Indices:

Internal and external indices require statistical testing, which could become computationally intensive. Relative criteria eliminate this requirement and concentrate on the comparision of the clustering results generated by different clustering algorithm or the same algorithm with different input parameters.

The fundamental problem of cluster validity is the true number k of the clusters.

The main indices are:

- Calinski And Harabasz: The value of k is the corresponding to the maximum.
- Dunn: The value of k is the corresponding to the maximum, a large value suggests the presence of compact and well separated clusters.
- Davies-Bouldin: The value of k is the corresponding to the minimum.

For the probabilistic mixture model-based clustering the main indices are:

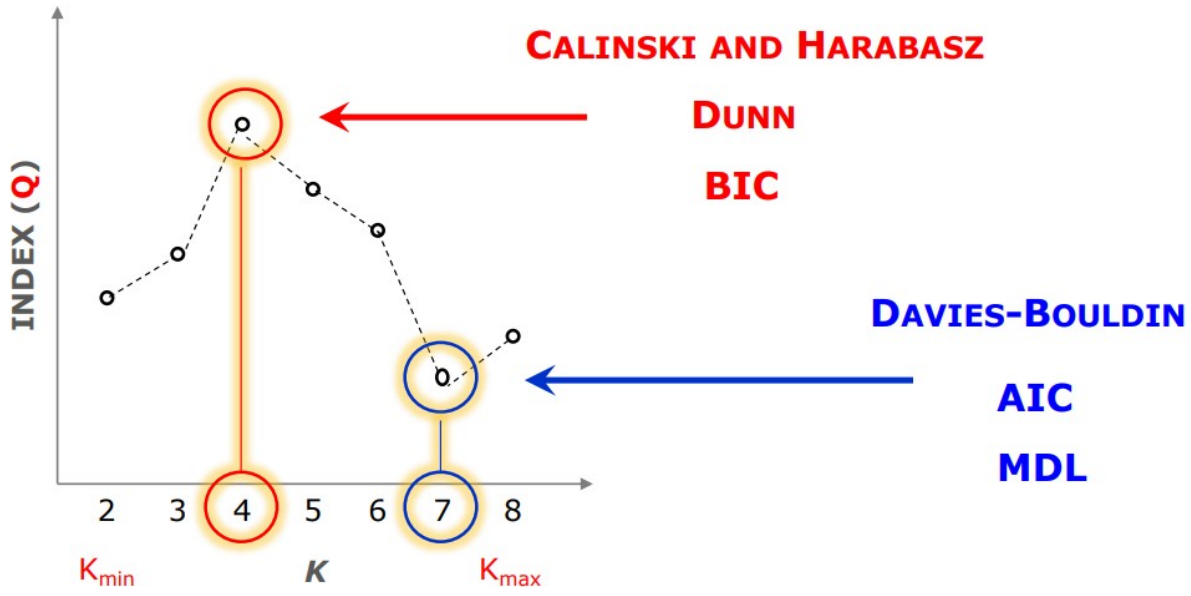
- Akaike Information Criterion (AIC): The value of k is the corresponding to the minimum.
- Minimum Description Length (MDL): The value of k is the corresponding to the minimum.
- Bayesian Information Criterion (BIC): The value of k is the corresponding to the maximum.

Given $K_{min}$ and $K_{max}$ the number of clusters is obtained by running a clustering algorithm several times (r), the algorithm is as follows.

**GENERATE SEQUENCE OF CLUSTERING STRUCTURES**

1. Choose a clustering algorithm and a validity index
2. **FOR** $K$=$K_{min}$ to $K_{max}$ **DO**
3. **FOR** $i$=1 to $r$ **DO**
4. run the clustering algorithm with $K$ and use parameter values different from the previous running
5. compute the value $q$ of the validity index and set $q(i) = q$
6. **END FOR**
7. Choose the best value $\mathbf{q}^*$ of the validity index $\{q_1, \ldots, q_r\}$
8. set $Q(K) = \mathbf{q}^*$
9. **END FOR**

The clustering structures are then evaluated based on the computer index, and the optimal clustering solution is determined by choosing the best value of the index. In the case of hierarchical clustering structures, the indices are also known as stopping rules, which tell where the best level is in order to cut the dendogram. Here is an example of how to choose the optimal K given $K_{min}$=2 and $K_{max}$=8:



# Association Analysis

The decision about the products positioning is a very difficult task, in the case of the mini-market we need to be the information about the customer behavior, and to to this we need a dataset containing the transactions and a list of all products. Usually the file is such that the transaction file has all the transaction with the first column as the ID and the second column contain the product ID of all the products that were bought, and the product file contains it's ID and detail like price, name.
Association analysis is used to discover interesting and useful relationships hidden in the large amount of data. The uncovered relationship can be represented in form of association rules or sets or Frequent Items. For example the we might find the association rule: {Product1} → {Product2} in this case Product1 is associated with Product2, which means that there is strong association between the sales of Product1 and Product2, in this context we say that the Product1 is the antecedent of association rule while Product2 is the consequent of the association rule.
We transform the given data structure in a different data structure:
The first Column is associated with the Transaction ID and each other column is associated to a each Product ID/name (we binarize the data), and for each transaction we put 1 in the column of product if it was bought (it was in the basket of transaction dataset) and 0 otherwise. For example:

| Transaction ID | swiss cheese | cherry coke | bio coke | Peppers | scrambled egg | Pomegranate | strawberries |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |

Let $I=\{i_1,\ldots,i_k\}$ be the set of all items in a market basket data and $T=\{t_1,\ldots,t_n\}$ be the set of all transaction, so we have k items and n transactions. A collection of zero or more items is termed an itemset, if an itemset contains k-terms, it is called a k-itemset. The null or empty set is an itemset that contains no items. The transaction width is defined as the number of items present in a transaction.
The transact $t_j$ is said to contain the itemset X, if X is a subset of $t_j$, for example given the transaction: |212 | Biocoke, swiss cheese, strawberries| we have that it contains itemset {Biocoke,swiss cheese} but does not contain the itemset {Biocoke, cheddar}.
An important property of an itemset is its support count, which refers to the number of transactions that contain a particular itemset, we define the support count for itemset X like :
$\sigma(X) = |\{t_i| \quad X \subset t_i, t_i \in T \quad \}|$ , number of transaction $t_i$ of T that contains the itemset X.
An association rule is an implication expression of the form: X → Y where X and Y are disjoint. X is the antecedent and Y is the consequent.
The strength of an association rule can be measured in terms of its support and confidence:
**Support** detemines how often a rule is applicable to a given data set and defined as:

$$s\{X \rightarrow Y\} = \frac{\sigma(X \cup Y)}{N}$$

While the **Confidence** determines how frequently items in Y appear in transactions that contain X:

$$c\{X \rightarrow Y\} = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

A rule with low support

- is such that it may occur simply by chance
- it is uninteresting from a business perspective, it may be not profitable to promote items that customers seldom but together (some exception however exist)

The support is often used to eliminate uninteresting rules and it shares an interesting property which can be exploited for the efficient discovery of association rules.
The confidence measures the reliability of the inference made by a rule:

- the higher the confidence the more likely is for Y to be present in transactions that contain X
- it estimates the conditional probability of Y given X

However the confidence does not have to be interpreted as causation, it just suggests a strong co-occurrence of antecedent X and consequent Y.

The association rule mining problem can be defined as follows:
Given a set of transaction T, find all the rules having support >= minsup and confidence>= minconf, where minsup and minconf are the corresponding support and confidence tresholds.
A brute-force approach for mining association rules is to compute the support and confidence for every possible rule. In this case the total number of possible rules extracted from a data set that contains k different items is $R = 3^d - 2^{d+1}+1$, which grows pretty fast and is very computationally demanding.
A common strategy adopted in many association rule mining algorithms is to decompose the problem into two major sub-tasks:

- Frequent Itemset Generation, whose objective is to find all the itemsets that satisfy the minsup treshold. There itemsets are alled **Frequent Itemsets.**
- Rule Generation, whose objective is to extract all the high-confidence rules from the Frequent Itemsets found in the previous step. There rules are called **Strong Rules**.

The computational requirement for Frequent Itemsets generation are generally more expensive that those of Rule Generation.
It is important to establish a set of well accepted criteria for evaluating the quality of association patterns. We use two sets of criteria:

- Statistical Arguments: In this case patterns involving a set of mutually independent items and patterns covering very few transactions, we may capture spurious relationship in the data so uninteresting to us. We can eliminate them by applying an objective interestingness mesure, which uses statistics derived from data to determine whether a pattern in interesting, for example using support, confidence, correlation.
- Subjective Arguments: A pattern is considered subjectively uninteresting unless it reveals unexpected information about the data or provides useful knowledge that can lead to profitable actions. For example the fact that a customer buys butter and bread despite having high support and confidence is considered uninteresting, because the relationship represented by the rule may seem rather obvious, while buying beer and pampers is unexpected and may suggest cross-selling opportunity for retailers thus is considered interesting. However incorporating subjective knowledge into pattern evaluation is a difficult task because it requires a considerable amount of prior information from the domain experts.

An objective interestingness measure is:

- a data-driven approach for evaluating the quality of association patterns
- domain-independent and requires minimal input from the users (usually threshold only)
- computed based on the frequency cunts tabulated in a contingency table:

| | B | not B | |
|---|---|---|---|
| A | $f_{11}$ | $f_{10}$ | $f_{1+}$ |
| not A | $f_{01}$ | $f_{00}$ | $f_{0+}$ |
| | $f_{+1}$ | $f_{+0}$ | N |

$\mathbf{f_{1+}}$ = support count for A

$\mathbf{f_{+1}}$ = support count for B

A = item A is present in the transaction

not A = item A is not present in the transaction

Judging the interestingness of association rule using confidence can be misleading, since the confidence ignores the support of the itemset in the rile consequent. We introduce some evaluation measures:

- Interest Factor: addresses the problem of confidence, i.e., ignoring the support of the itemset appearing in the rule consequent, to introduce the interest factor we start from the Lift, Lift is defined as $Lift = \frac{c(A \rightarrow B)}{s(B)}$ , the ratio of the rule confidence over the support of the

consequent. In case if binary attributes Lift is equivalent to another objective measure called Interest Factor: $I(A,B)=\frac{s(A,B)}{s(A)s(B)}=N\frac{f_{11}}{f_{1+}f_{-1}}$ where s is the support function σ. It compares the frequency of a pattern against a baseline frequency computed under the statistical independence assumption. We have that I(A,B) is:

- = 1 if A and B are independent
- > 1 If A and B are positively associated
- < 1 if A and B are negatively associated

Interest factor is not used in text mining to evaluate association between pairs of words because it can be extremely misleading.

- Correlation Analysis: analyzes relationship between a pair of attributes. Continuous attributes are analyzed using Pearson's correlation coefficient. Correlation for binary attributes is measured using the ϕ-coefficient, defined as: $\phi=\frac{f_{11}f_{00}-f_{01}f_{10}}{\sqrt{f_{1+}f_{+1}f_{0+}f_{+0}}} \quad \in[-1,1]$ . If the items are statistically independent then it's value is 0. It is more suitable for analyzing symmetric binary attributes, where present and absent of value is equally relevant.
- IS measure: allows to handle asymmetric binary attributes, defined as: $IS(A,B)=\sqrt{I(A,B)s(A,B)}=\frac{s(A,B)}{\sqrt{s(A)s(B)}}$ , where I is the interest factor, and is larger when the interest factor and support of the pattern are large. It can be used in text mining to evaluate association between pair of words. When A and B are independent we have: $IS(A,B)=\frac{s(A,B)}{\sqrt{s(A)s(B)}}=\frac{s(A)s(B)}{\sqrt{s(A)s(B)}}=\sqrt{s(A)s(B)}$ , it shares similar problem as the confidence measure, the value of the measure can be quite large, even for uncorrelated and negatively correlated patterns.

There are other alternative measures proposed for analyzing relationships between pairs of binary variables, there measures can be divided into 2 categories:

- Symmetric: A measure M is symmetric if M(A → B) = M (B → A), in this case we have that I(A,B)=s(A,B)/(s(A)s(B)) = s(B,A)/(s(A)s(B))=I(B,A) so it is also symmetric. The symmetric measures are generally used for evaluating itemsets.
- Asymmetric: A measure is asymmetric if M(A → B) ≠ M(B → A), an example is confidence because c(A → B) = σ(A ∪ B)/σ(A) ≠ σ(A ∪ B)/σ(B) = c(B → A). The asymmetric measure is more suitable for analyzing association rules.

Some examples of Symmetric measures are: Correlation (ϕ), Odds ratio, Kappa, Interest (I), Cosine(IS), Jaccard, Collective strenght.
Some examples of Asymmetric measures are: Gini Index, Mutual Information, Certainty factor, Added value, J-measure, Goodman-Kruskal.
A wide variety of objective measures is available, it is reasonable to question whether the measures can produce similar ordering results when applied to a set of association patterns. It is also important to understand what their differences are in order to determine which measure is more suitable for analyzing certain types of patterns. For example the Correlation (ϕ), Odds ratio, Kappa and collective strength may not be suitable for analyzing asymmetric binary attributes. While Interest(I),Cosine(IS) and Jaccard are preferred for asymmetric binary attributes. The symmetric measures Cosine (IS) and Jaccard are particularly used for document analysis and market basket analysis but in this case are the Correlation (ϕ), Odds ratio and Interest(I) must not be used.
All the measures presented so far are defined for pairs of binary attributes. However, many of them, such as support, are applicable to larger-sized itemsets. Furthermore, Interest Factor, IS and the Jaccard coefficient, can be extended to more than two attributes using the frequency tables tabulated in a multi-dimensional Contingency Table (Hyper Table), in this case we have:

$$I(A,B,C)=N^{3-1}\frac{f_{111}}{f_{1++}f_{+1+}f_{++1}}$$