Principles of Machine Learning: Session 1

Methusalem Colloquium Mini-Course

S. Nõmm

[1]Department of Software Science, Tallinn University of Technology

06.12.2022

Introduction

- Preliminaries
- Overview
- General work flow
- Similarity (Metric) function
- Unsupervised learning
- Anomaly detection
- Supervised learning

What is Machine Learning?

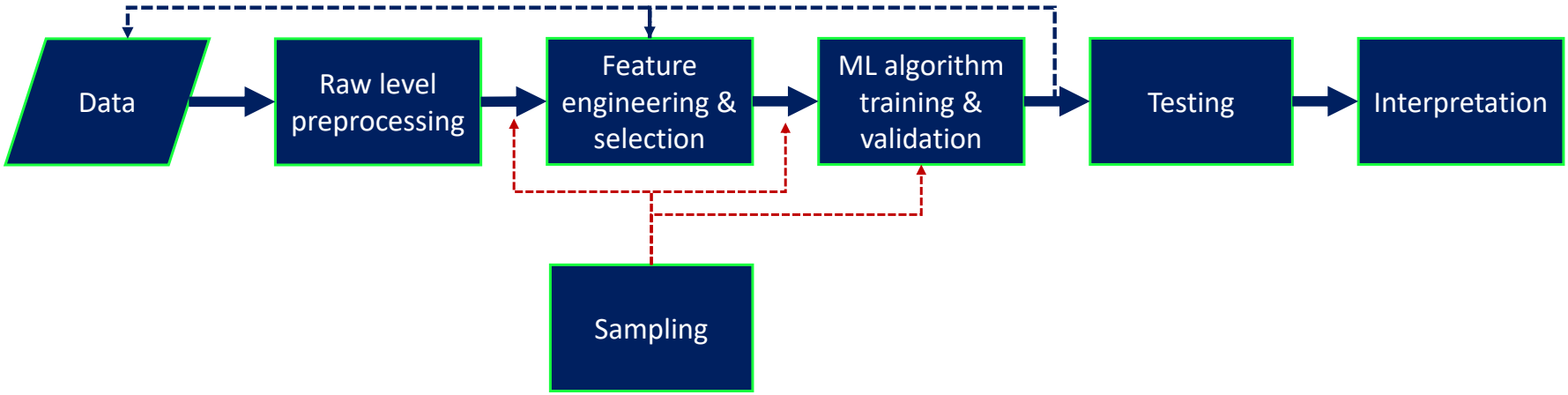
Arthur Samuel, 1959, defined machine learning as: Field of study that gives computers the ability to learn without being explicitly programmed.

Tom Mitchell, 1997, has defined machine learning as: A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Differences between learning and pure optimization

- Machine learning usually acts indirectly. Usually performance measure P is defined, it may be intractable.
- We are trying to to improve P by reducing different cost function $J(\Theta)$.

General workflow



Distance function

Distance function is one of most fundamental notions in Machine learning and Data mining. Formally defined in pure mathematics as metric function. It provides a measure of similarity or distance between two elements.

Definition

A function $S : X \times X \to \mathbb{R}$ is called a metric if for any elements x, y and z of X the following conditions are satisfied.

1. Non-negativity or separation axiom

$$S(x, y) \geq 0$$

2. Identity of indiscernible, or coincidence axiom

$$S(x, y) = 0 \Leftrightarrow x = y$$

3. Symmetry

$$S(x, y) = S(y, x)$$

4. Subadditivity or triangle inequality

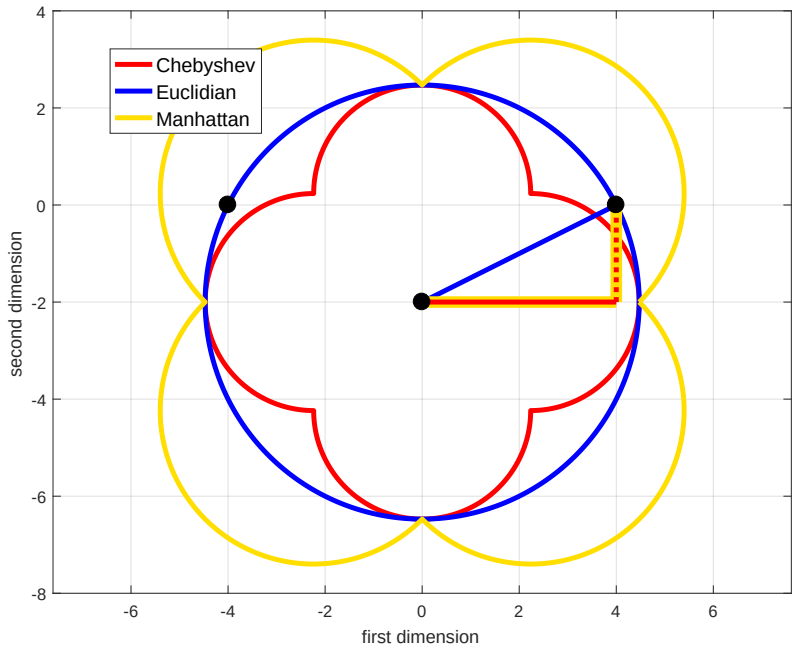
$$S(x, z) \leq S(x, y) + S(y, z)$$

Distance functions: Minkowsky

$$S(x,y) = \left(\sum_{i=1}^{d} \mid x_i - y_i \mid^p\right)^{1/p}$$

- $p < 1$ triangle inequality is violated; therefore, for values of p smaller than one, the equation above is not a distance function.
- $p = 1$ case of Manhattan distance.
- $p = 2$ case of Euclidian distance.
- $p \to \infty$ case of Chebyshev distance.

Distance function: Examples 1



Euclidean distance

$$S(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Manhattan distance

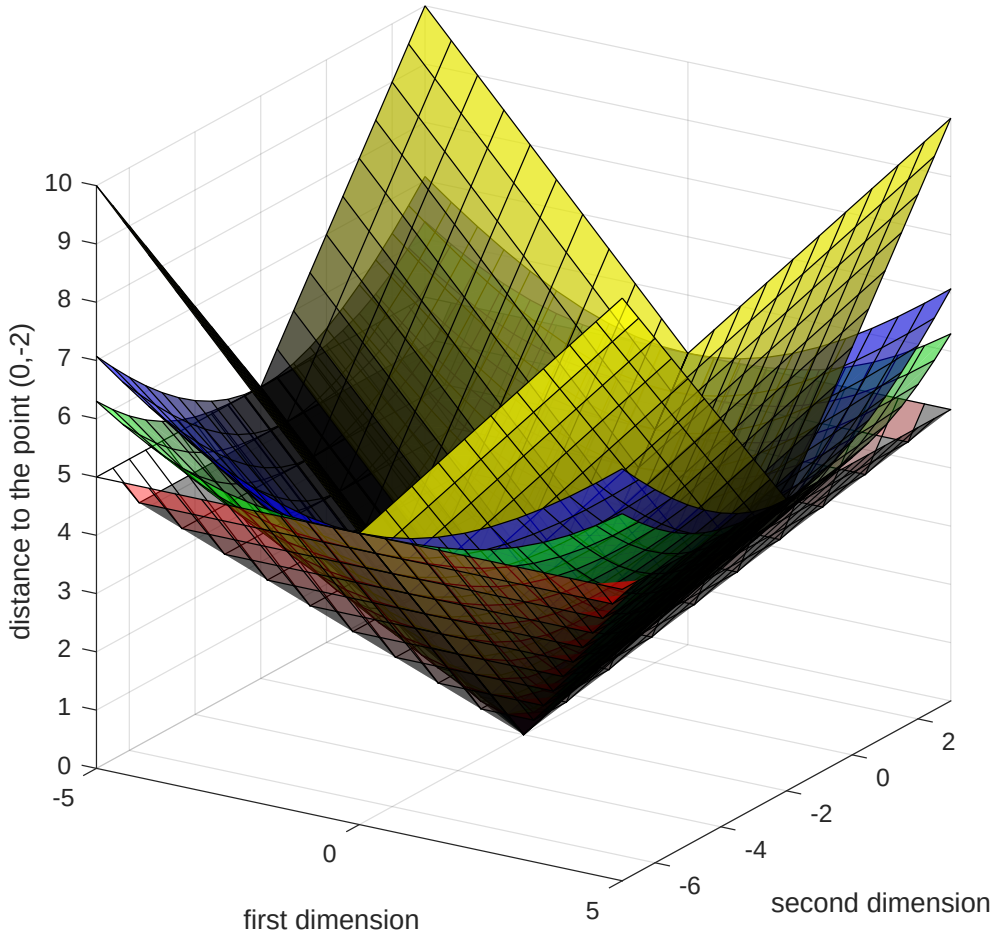
$$S(x,y) = \sum_{i=1}^{n} \mid x_i - y_i \mid$$

Chebyshev distance

$$S(x,y) = \max_i\big(\mid x_i - y_i \mid\big)$$

Distance function: Examples 2

3D representation of the Minkovski distances for different values of parameter p. $p = 1$ - yellow surface, Manhattan; $p = 2$ - blue surface, Euclidean,; $p = 3$ - green surface; $p \to \infty$ - red surface, Chebyshev.

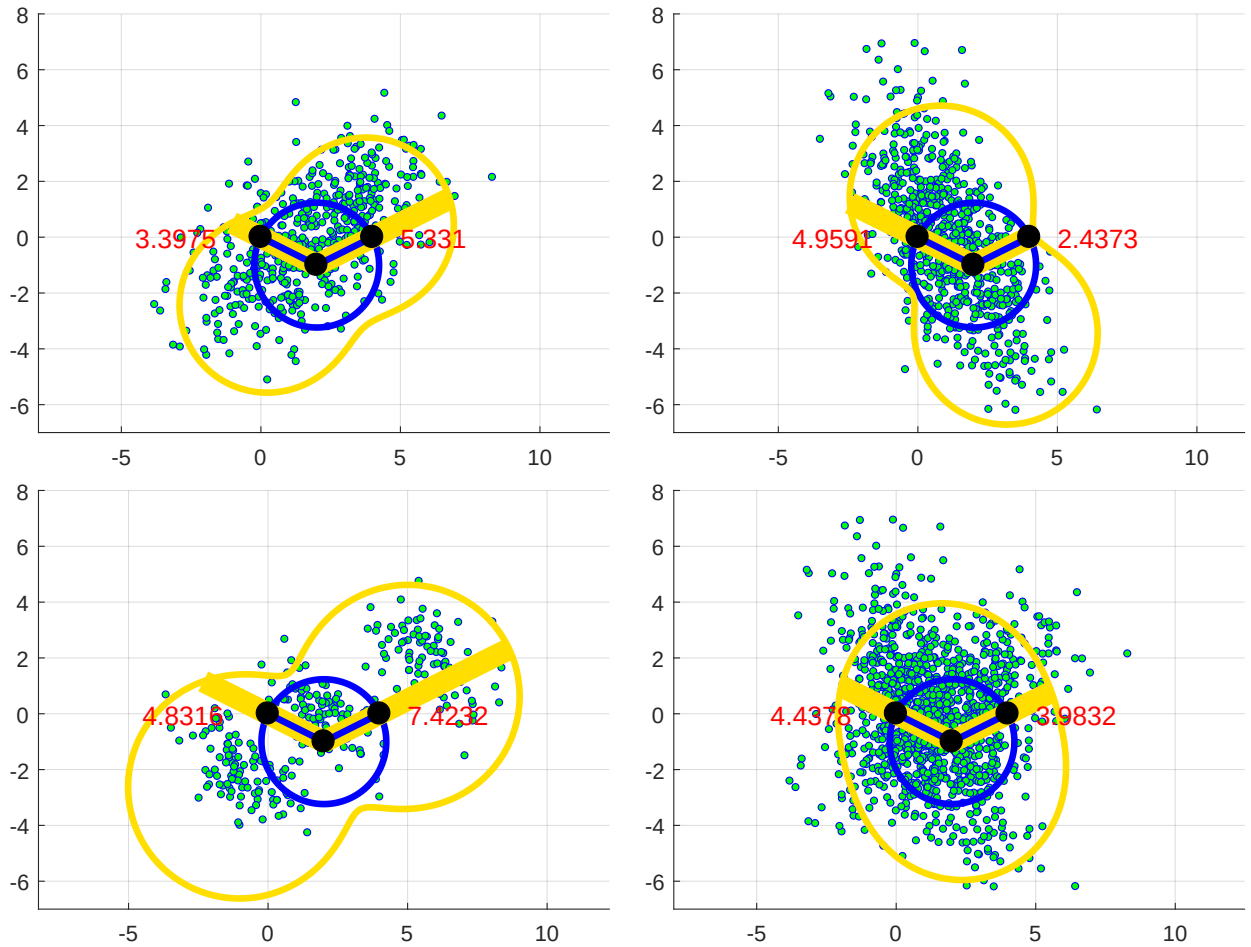


Distance function: Examples 3

Mahalanobis distance

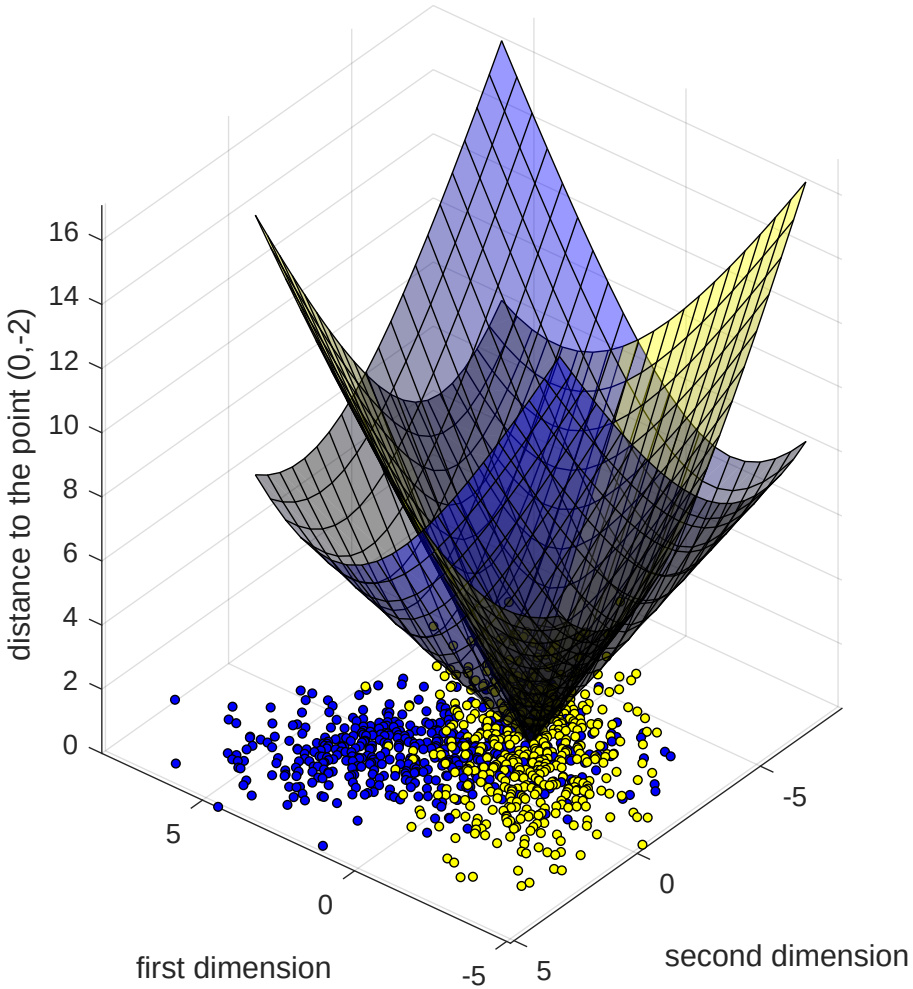
$$S(x,y) = \sqrt{(x-y)^T C^{-1}(x-y)}$$

where C is the covariance matrix. Takes into account impact of data distribution.



Distance function: Examples 4

- Impact of the rotation of the underlying data set.



Distance function: Examples 5

- Canberra distance

$$S(x,y) = \sum_{i=1}^{n} \frac{|x_i - y_i|}{|x_i| + |y_i|}$$

  weighted version of Manhattan distance.

- Cosine distance Cosine similarity is the measure of the angle between two vectors

$$S_c(x,y) = \frac{x \cdot y}{\|x\|\|y\|}$$

  Usually used in high-dimensional positive spaces, ranges from -1 to 1. The cosine distance is defined as follows

$$S_C(x,y) = 1 - S_c(x,y)$$

Distance function: Examples 6

- Levenshtein or SED distance. SED - minimal number of single -charter edits required to change one string into another. Edit operations are as follows:
  - insertions
  - deletions
  - substitutions
- SED(delta, delata)$=1$ delete "a" or SED(kitten,sitting)$=3$: substitute "k" with "s",substitute "e" with "i", insert "g".
- Specialized similarity measures Distance and similarity functions applicable to the graphs, temporal data etc.

Unsupervised learning

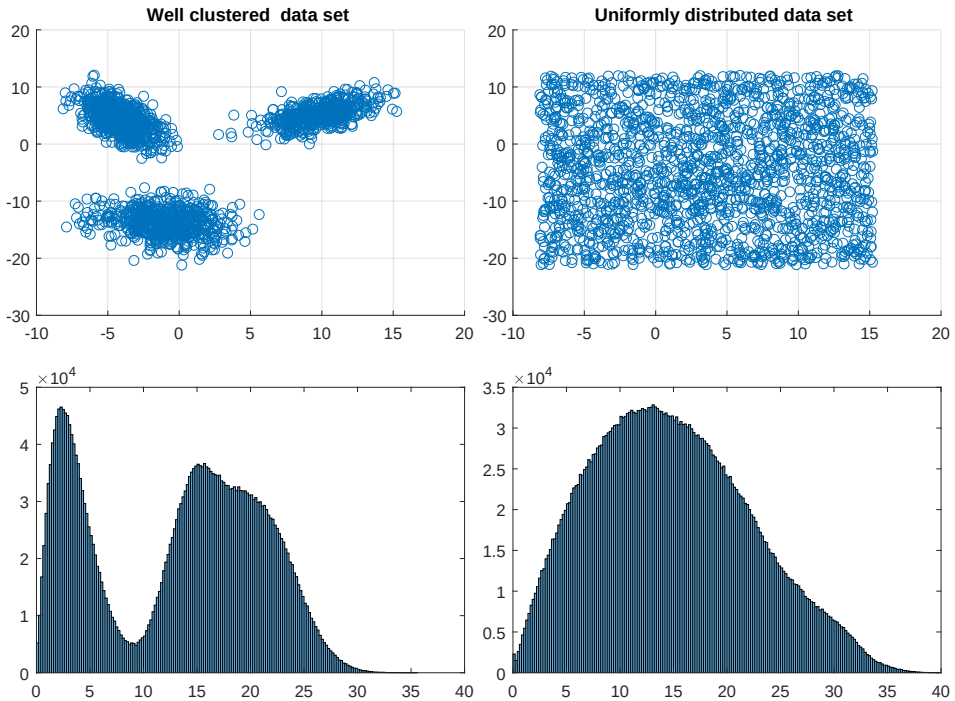
Unsupervised machine learning techniques are used to discover the structure of a given data set.

Unsupervised learning: Feature selection

- Filter Methods: Use Entropy or Hopkins Statistics to decide set of features leads best clustering behaviour. Filter methods may be applied on the stage of preprocessing.
- Warper models: clustering algorithm is used to evaluate the quality of subset of features.

Unsupervised learning: Feature selection

- Underlying idea s that features with uniformly distributed values carry less information compared to those distributed non uniformly.
- Distance distributions of well-clustered sets should be different from those uniformly distributed.



Unsupervised learning: Feature selection

- Entropy

$$E = -\sum_{i=1}^{m} \left[p_i \log(p_i) + (1 - p_i) \log(1 - p_i) \right].$$

where p_i is the proportion of points in the region i, m - total number of regions. Large values of E indicate poor clustering behaviour.

- Hopkins statistics. Let $\mathcal{D}$ be the data set to investigate and $\mathcal{R}$ is a representative sample of $\mathcal{D}$, of power r. $\mathcal{S}$ is a synthetic data set of r data points randomly generated from the same domain. Let $\alpha_1, \ldots \alpha_r$ be the distances of each point of $\mathcal{R}$ to the nearest neighbour in $\mathcal{D}$ and $\beta_1, \ldots \beta_r$ are the distances of each point of $\mathcal{S}$ to the nearest neighbour in $\mathcal{D}$. The Hopkins statistic is defined as follows:

$$H = \frac{\sum_{i=1}^{r} \beta_i}{\sum_{i=1}^{r} (\alpha_i + \beta_i)}.$$

Higher values of H indicate highly clustered data.

Unsupervised learning: Clustering

Definition

Clustering is a process of grouping elements of the given data set into groups with respect to chosen similarity criteria.

This definition requires one to determine the following "parameters" either in process of learning or before it.

- Similarity criteria (distance or metric function).
- Algorithm and its goodness criteria.
- Validation.

Hyperparameter is the parameter which value is not determined during the learning.
As a result of clustering each element is assigned label describing which cluster it belongs to.

Taxonomy of clustering techniques

Most common clustering techniques may be classified as follows:

- **Representative based techniques:** k-means, k-medians, k-medoids, etc. Each cluster has a representative which is either the element of the data set or an element from the same space as all other elements of the dataset. Shape of the clusters is affected by the choice of distance function. Number of clusters is usually a hyperparameter.
- **Hierarchical clustering techniques:** Agglomerative and Divisive techniques. Not always relies on the distance function. Different levels of clustering granularity provide different provide different application specific insides.
- **Grid and Density based techniques:** Relies on the local density of the data points. Well suited for the clusters of irregular shapes.
- **Probabilistic algorithms:** EM and EM-like algorithms.

K - means

K - means is one of the most popular algorithms belongs to the class of iterative descent methods.

- It is intended for the quantitative variables.
- Squared Euclidean distance as dissimilarity measure.
- The idea is to assign close points to the same cluster. Minimize natural loss ("energy") function.

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} N_k \sum_{C(i)=k} |x_i - \bar{x}_k|^2.$$

where $\bar{x}_k$ is the mean vector associated with the kth cluster (centroid). $N_k = \sum_{i=1}^{N} I(C(i) = k)$.

- Iterative descent algorithm is used to achieve this goal.

Representative based clustering

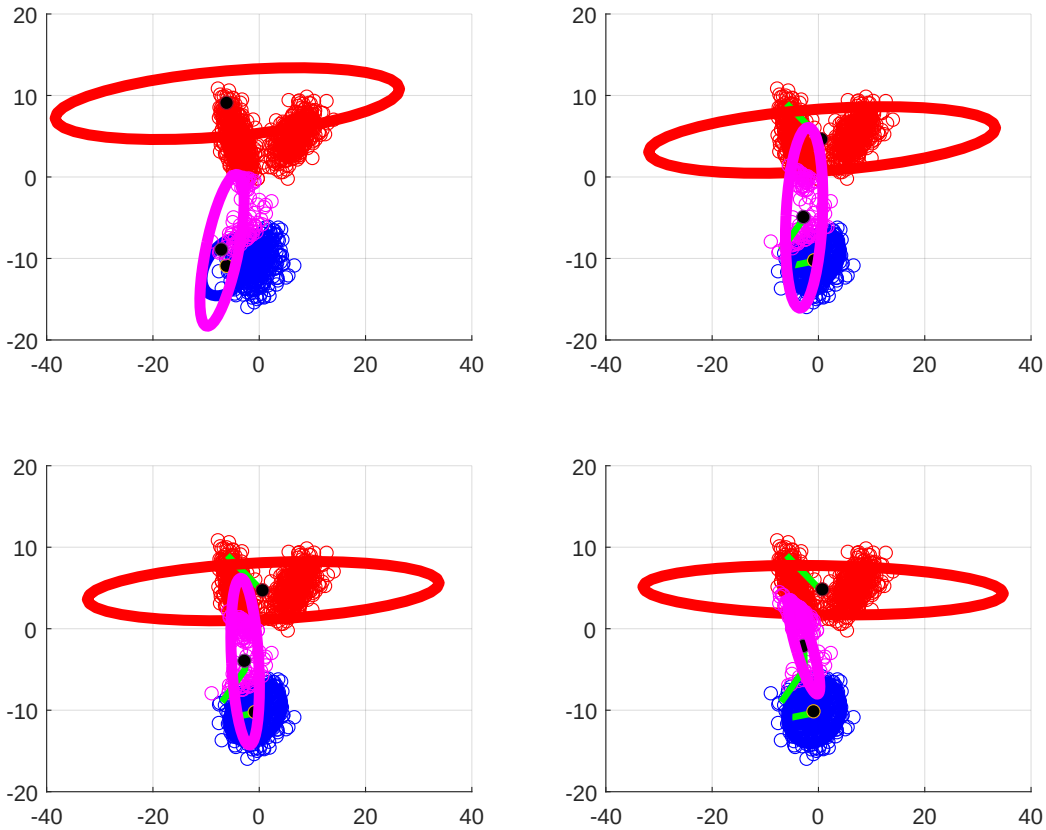
K-means:

- Hyperparameters: K - number of desired clusters, distance function.
- Intialize: generate K random points from the same limits as initial dataset. These points are referred as centroid.
- Repeat:
  - For each point assign the label of closest centroid.
  - For each label recompute centroid as the mean of all points with given label.
- Until converge.
- Report labels of each point.

Other representative based techniques differ only by the way representative is find.

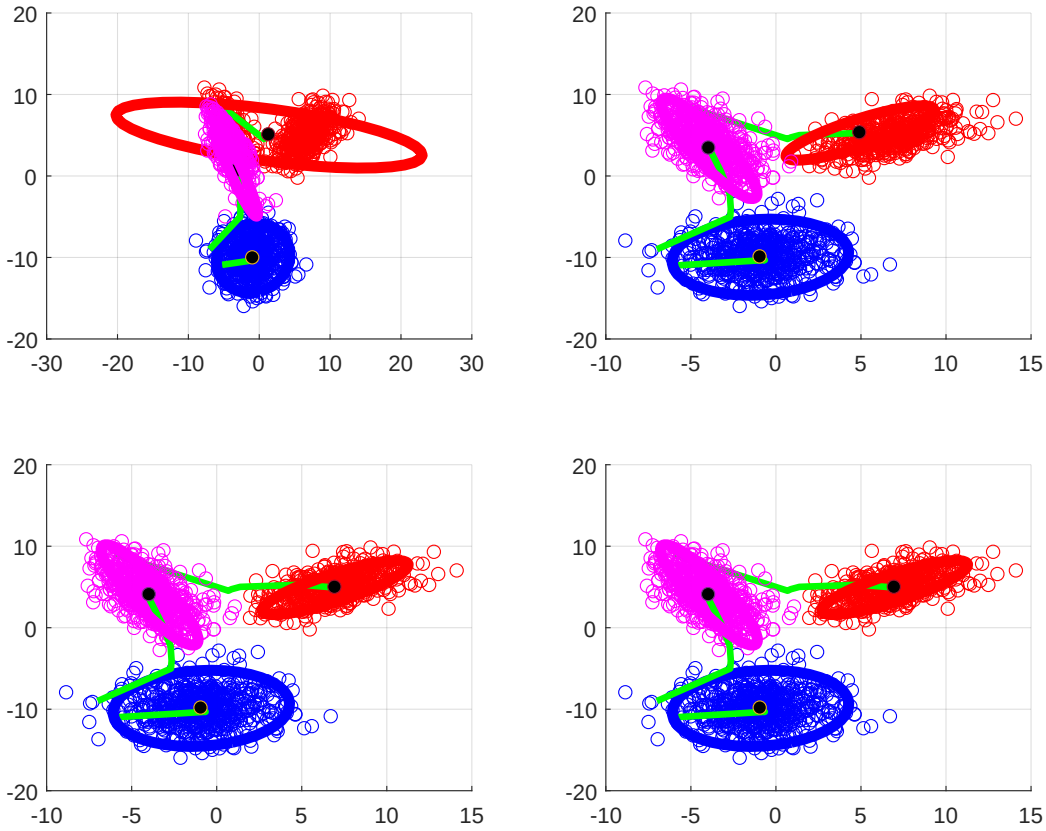
K-means clustering example

Steps 1 - 4



K-means clustering example

Steps 5 - 8



Gaussian Mixture Model

Mixture of Gaussians

$$p(x_i|\theta) = \sum_{k=1}^{K} \tau_k \mathcal{N}(x_i|\mu_k, \Sigma_k).$$

where τ_k are the mixing weights, μ_k are the means and Σ_k are the covariance matrices for each base distribution of the mixture.

EM-algorithm

Let us consider K - means from the probabilistic point of view.

- (E-step) Each data point pf the set $\mathcal{D}$ has a probability belonging to cluster j, which is proportional to the scaled and exponentiated Euclidean distance to each representative Y_j. In the case of k-means algorithm, this is done on a 'hard' way, by choosing the smallest Euclidean distance to the cluster representative.
- (M-step) The center Y_j is the weighted mean over all the data points where the weight is defined by the probability of assignment to cluster j. The 'hard' version of this is used in k-means where each data point is either assigned to a cluster or not.

EM-algorithm

Assumption: the data was generated from a mixture of k distributions with probability distributions $\mathcal{G}_1 \ldots \mathcal{G}_k$. Each distribution $\mathcal{G}_i$ represents a cluster and is also referred to as a mixture component.

- (E-Step) Given the current value of the parameters in Θ, estimate the posterior probability $P(\mathcal{G}_i|X_j, \Theta)$ of the component $\mathcal{G}_i$ having been selected in the generative process, given that we have observed data point X_j . The quantity $P(\mathcal{G}_i|X_j, \Theta)$ is also the soft cluster assignment probability that we are trying to estimate. This step is executed for each data point X_j and mixture component G_i.
- (M-Step) Given the current probabilities of assignments of data points to clusters, use the maximum likelihood approach to determine the values of all the parameters in Θ that maximize the log-likelihood fit on the basis of current assignments.

- Presence of latent variables makes complicated to compute ML estimates. Introduce negative log likelihood function.

$$NLL(\theta) = -\frac{1}{N} \log p(\mathcal{D}|\theta).$$

- Let x be the observed variables and z_i be the hidden or missing variables. The goal is to maximize the log likelihood of the observed data.

$$\ell(\theta) = \sum_{i=1}^{N} \log p(x_i|\theta) = \sum_{i=1}^{N} \log \Big[p(x_i, z_i|\theta) \Big].$$

- Complete data log likelihood could not be computed because z_i is unknown.

$$\ell_C(\theta) = \sum_{i=1}^{N} \log p(x_i, z_i|\theta).$$

- Expected complete data log likelihood

$$\begin{aligned} Q(\theta, \theta^{t-1}) &= \mathbb{E}[\ell_c(\theta)|\mathcal{D}, \theta^{t-1}] \\ &= \sum_i \sum_k r_{i,k} \log \tau_k + \sum_i \sum_k r_{r,k} \log p(x_i|\theta_k). \end{aligned}$$

EM for GMM

- E step:

$$r_{i,k} = \frac{\tau_k p(x_i|\theta_k^{(t-1)})}{\sum_{k'} \tau_{k'} p\left(x_i|\theta_{k'}^{(t-1)}\right)}$$

- M step: Optimize Q with respect to the θ and τ.

$$\begin{aligned}
\tau_k &= \frac{\sum_i r_{i,k}}{N} \\
\mu_k &= \frac{\sum_i r_{i,k} x_i}{r_k} \\
\Sigma_k &= \frac{\sum_i r_{i,k}(x_i - \mu_k)(x_i - \mu_k)^T}{r_k} = \frac{\Sigma_i r_{i,k} x_i x_i^T}{r_k} - \mu_k \mu_k^T
\end{aligned}$$

Validation

- **Sum of square distances to centroids.** (SSQ) This criteria is suited for K-means since it minimizes the loss function.
- **Intracluster to intercluster distance ratio.** Sample r points from the data set. Let P be the set of pairs that belong to the same cluster and Q the set of remaining pairs.

$$II = \frac{\sum_{(x_i,x_j)\in P} S(x_i,x_j)/|P|}{\sum_{(x_i,x_j)\in Q} S(x_i,x_j)/|Q|}$$

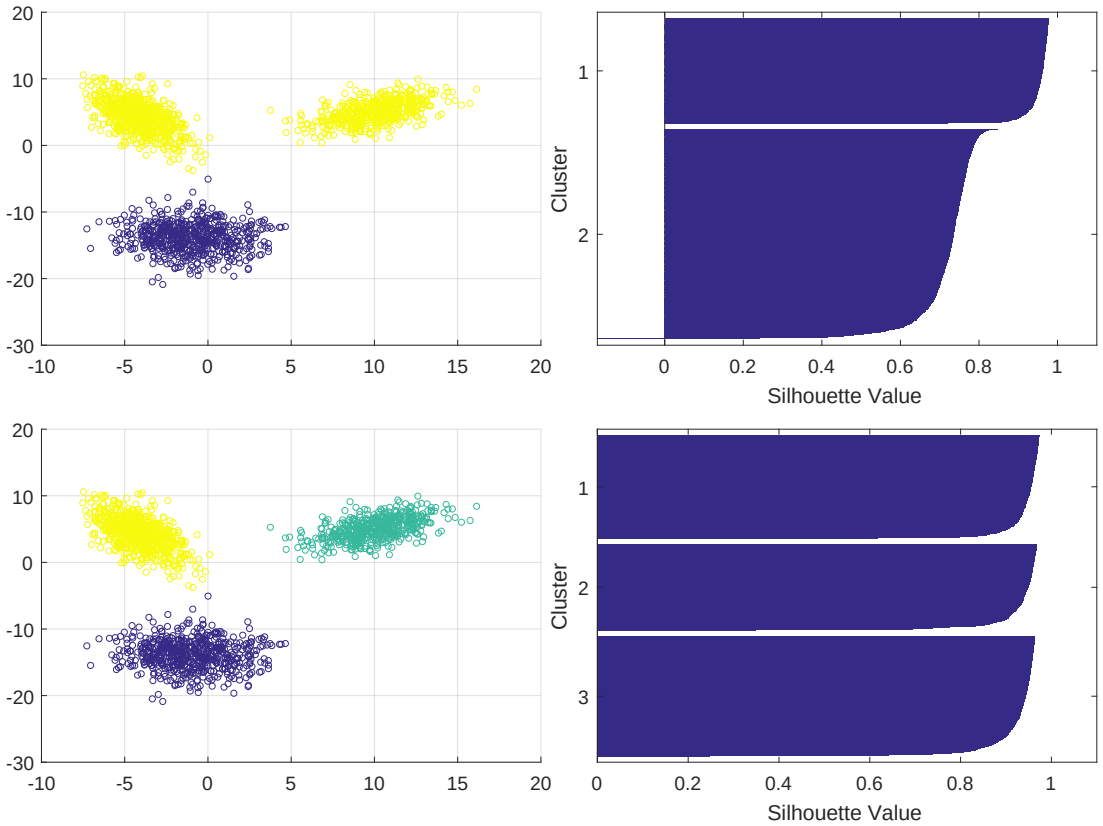
  Small values of the ratio indicate better clustering behaviour.
- **Silhouette coefficient**

$$s(i) = \frac{D^{\text{out}}_{\text{min}_i} - D^{\text{in}}_{\text{avg}_i}}{\max\{D^{\text{out}}_{\text{min}_i}, D^{\text{in}}_{\text{avg}_i}\}}$$

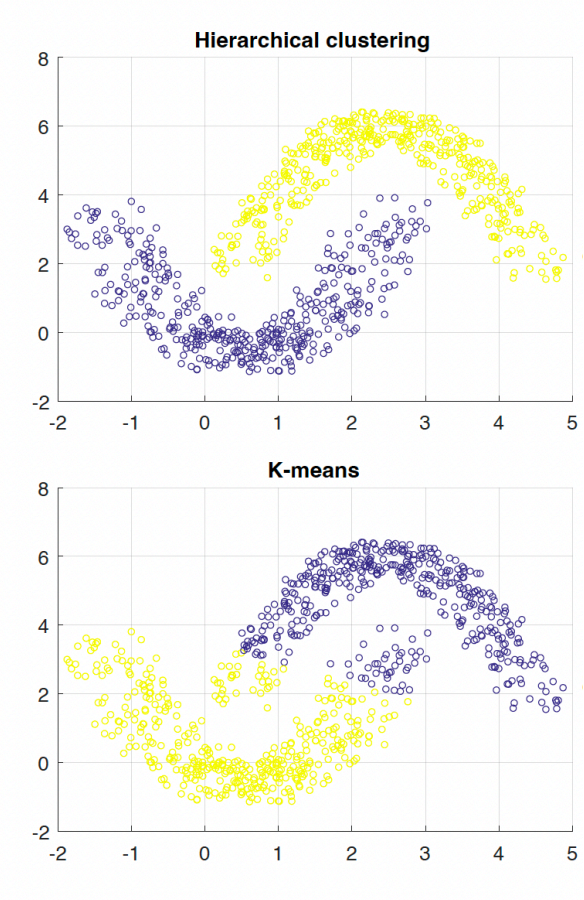
  where $D^{\text{in}}_{\text{avg}_\text{i}}$ is the average distance of point x_i to points within the cluster it belong to. Compute average distance of point x_i to the points of each cluster. Let $D^{\text{out}}_{\text{min}_i}$ is the minimum of these average distances. $s(i) \in (-1, 1)$. Overall coefficient is the average of the individual points coefficients. Large positive values indicate highly separated clusters.

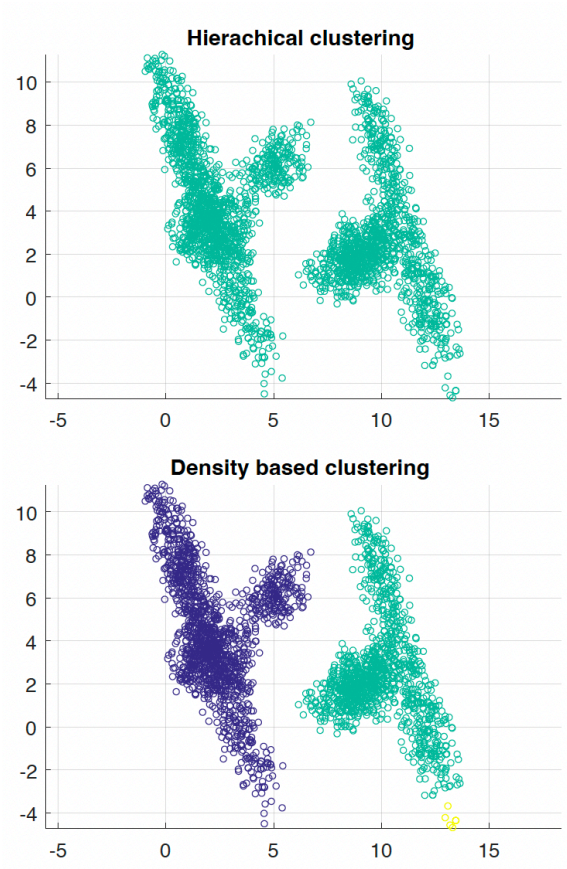
Silhouette coefficient

- Considered to be most popular criteria for clustering validation.
- Silhouette plot is the graphic representation of the silhouette coefficient.
- Overall silhouette coefficient may be used to determine number of clusters.



Limitations



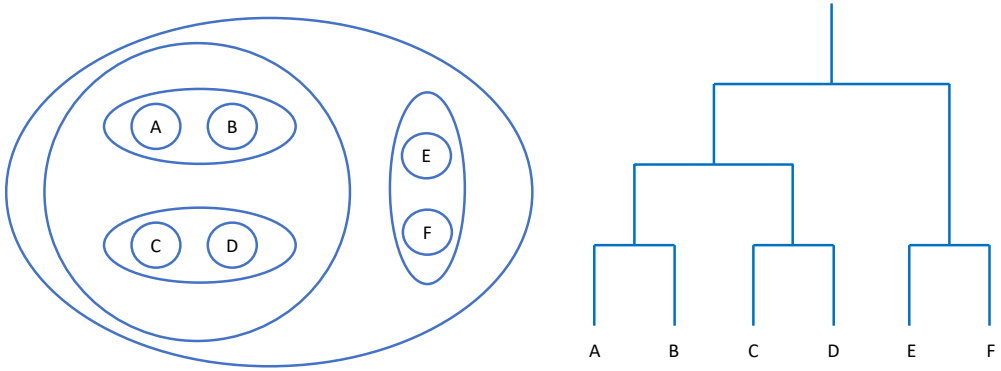


Hierarchical clustering: Agglomerative clustering

Sometimes referred ad bottom-up

Algorithm

- Initialize $n \times n$ distance matrix $\mathcal{M}$
- **Repeat**
  - Choose closest pair of clusters (i, j) based on $\mathcal{M}$.
  - Merge clusters i and j and update matrix $\mathcal{M}$.
- **Until** termination criterion.
- Return cluster labels for each point.



Group-based statistics

Also referred as linkage.

- Best (single) linkage. Distance is equal to the minimum distance between all pairs of elements (from two groups). Suitable to discover clusters of arbitrary shape. Drawback noise points may merge distant clusters.
- Worst (complete) linkage.(Complete linkage method) Distance is equal to the maximum distance between all pairs of elements (from two groups). Attempts to minimize maximual diameter of the cluster.
- Group average linkage. Distance between two groups is equal to the average of the distances between all pairs of elements (from two groups).
- Closest centroid. Clusters with closest centroid are merged.
- Variance based criterion. Minimizes the change in the objective function a result of merging.
- Ward's method, like previous but instead of variance observes changes in some od squared error.

Grid- and Density-based clustering

Explores the idea, that clusters are of a different density than space between them. May be see as the sub class of agglomerative methods.

Generic Grid:

Hyperparameters: Ranges and density threshold τ.

- Discretize each dimension into p ranges.
- Determine dense grid cells at level τ.
- Create graph in whichdense grids are connected if they are adjacent.
- Determine connected components of the graph.
- Return cluster indexes for each point.

DBSCAN

Let $\mathcal{D}$ denote the data set, τ - density threshold and ϵ - radius of the neighborhood.

Definition

Core point: A data point is defined as the core point, if its ϵ - neighbourhood contains at least τ data points.

Definition

Border point: A data point is defined as the border point, if its ϵ - neighbourhood contains at least one another data point of $\mathcal{D}$ and at least one core point.

Definition

Noise point: Is defined as data point of $\mathcal{D}$ which neither core point nor border point.

DBSCAN

Algorithm:

- Determine Core, border and noise points for given ϵ and τ.
- Create graph in which core points are connected (if they are within ϵ of one another).
- Assign each border point to a connected component.
- Return cluster indexes for each point.

Outliers and anomalies

1. An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism.
2. Outliers are also referred to as abnormalities, discordant, deviants, or anomalies in the data mining and statistics literature.
3. Alternatively, one may distinguish the notions of **outlier** and **anomaly** keeping (1) as definition of the outlier and treating anomalies as the sets of outliers which could not be considered as clusters or classes (for example do not satisfy conditions of being a cluster (number of elements, density, etc.))

Principles of outlier detection

- Majority of outlier detection methods create a model of normal patterns.
- Outliers are defined as data points that do not naturally fit within this normal model.
- The outlierness of a data point is quantified by a numeric value, known as the outlier score.
  - Real-valued outlier score quantifies the tendency for a data point to be considered an outlier.
  - Binary label is output, indicating whether or not a data point is an outlier.

Models of the normal patterns

- **Extreme values:** A data point is an extreme value, if it lies at one of the two ends of a probability distribution. [Hawkins].
- **Clustering models:** Clustering is considered a complementary problem to outlier analysis.
- **Distance-based models:** In these cases, the k-nearest neighbor distribution of a data point is analyzed to determine whether it is an outlier. Distance-based models can be considered a more fine-grained and instance-centered version of clustering models.
- **Density-based models:** The local density of a data point is used to define its outlier score.
- **Probabilistic models:** The steps are almost analogous to those of clustering algorithms, except that the EM algorithm is used for clustering, and the probabilistic fit values are used to quantify the outlier scores of data points (instead of distance values).
- **Information-theoretic models:** Constrain the maximum deviation allowed from the normal model and then examine the difference in space requirements for constructing a model with or without a specific data point. If the difference is large, then this point is reported as an outlier.

Supervised learning

Is a task of inferring function (training a model) on the basis of labeled training data. The goal is to construct a function (train a model) that would mimic (in a certain sense) behavior of the underling process.

- Regression: The dependent variable (continuous) plays the role of labels.
  - Linear
  - Nonlinear
  - Application of trees and SVM for regression.
  - Advanced methods like Neural Networks, etc.
- Classification labels are discrete (categorical values).
  - k-nearest neighbors.
  - Decision trees.
  - Support Vector Machines.
  - Neural networks.
  - Ensemble (committee).
  - Boosted techniques.

Classification

- Learning existing grouping based on the labeled set (training).
- The goal is to generate (choose the structure and train) a model which would mimic the existing grouping.
- Based on the features of the element model should estimate which class element belongs to or estimate value of dependent variable.
- Unlike the case of unsupervised learning, miss classification may be precisely measured.

Feature selection for classification

- Case of numeric data: Fisher's score

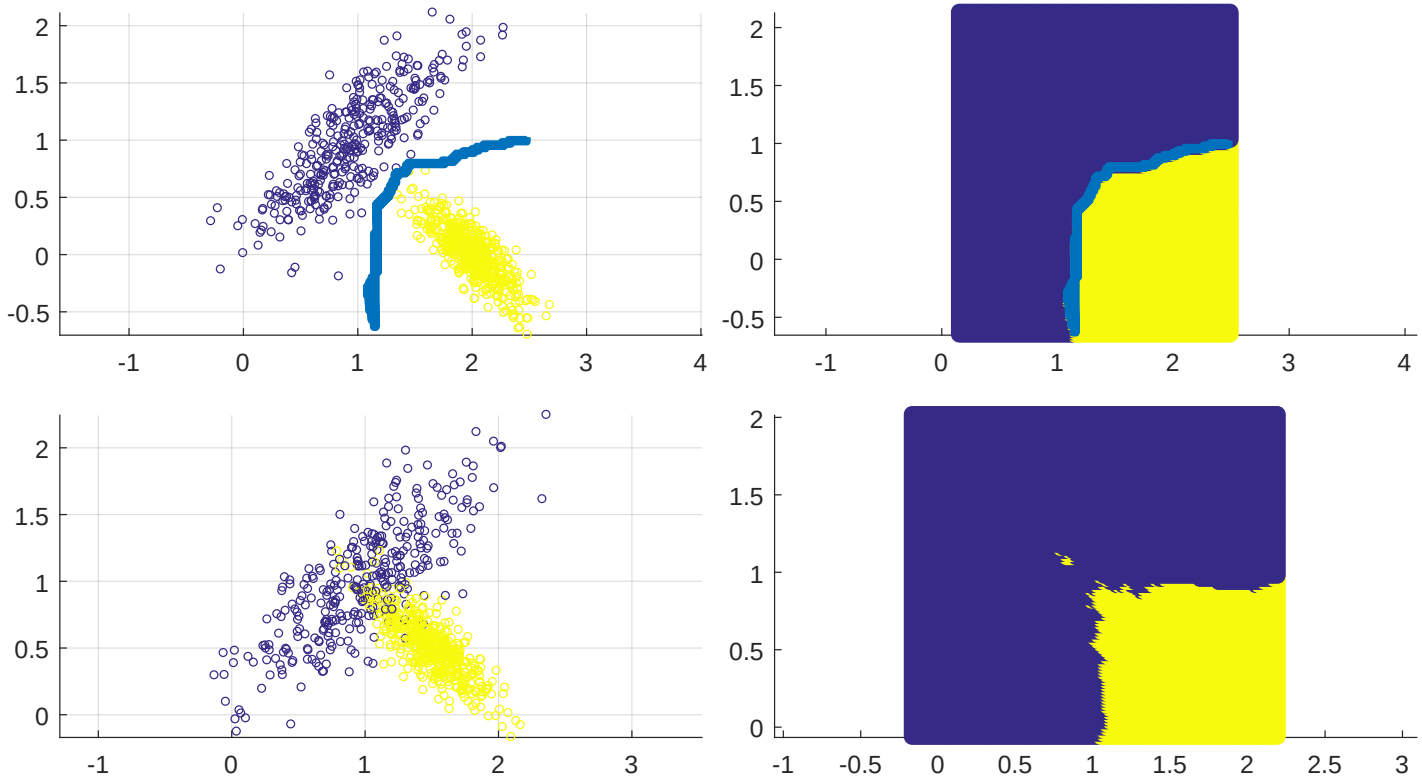
$$F = \frac{\sum_{j=1}^{k} p_j(\mu_j - \mu)^2}{\sum_{j=1}^{k} p_j \sigma_j^2}$$

  Greater values imply greater discriminating power of the variable.
- Wrapper methods.

k - nearest neighbours (k-NN)

- Let D denote the training (labeled) data set.
- For each unlabeled point (point to be classified).
  - Find k - the nearest neighbors.
  - Assign the mode (majority) label of k - nearest neighbours.

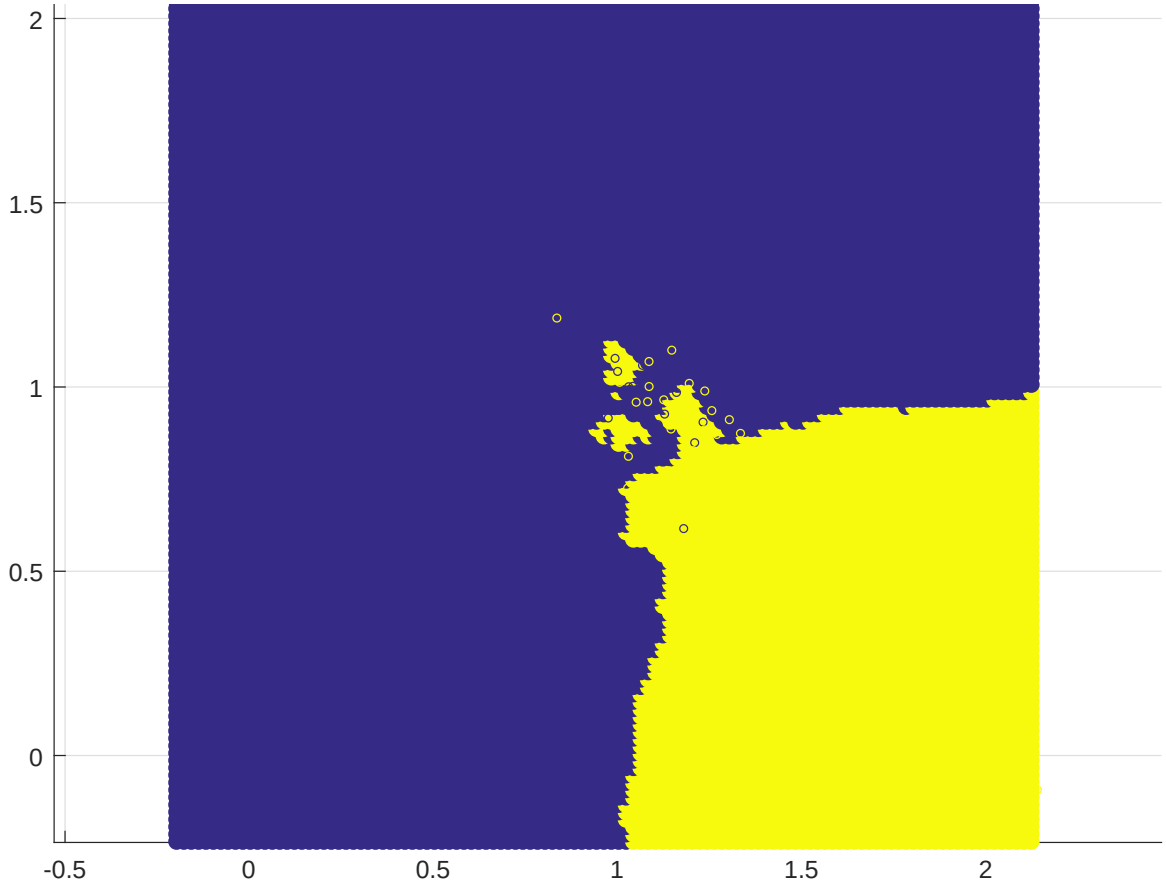
k - nearest neighbors, geometric interpretation, 2D



- The decision boundary (decision surface) (statistical classification with two classes) is a hypersurface that partitions the data set into two subsets, one for each class.
- The classifier tries to learn (construct) a decision boundary that will lead to minimal empirical error.

Accuracy

During the training (learning) process classifier tries to learn (construct) decision boundary that will lead minimal empirical error.



How good is trained classifier?

Validation

- Overall accuracy and confusion matrix (table),computed for the validation subset, are the goodness parameters of the trained classifier.

	Predicted Class 1	Predicted class 2
Actual class 1	58	2
Actual class 2	6	134

- How reliable are these parameters?

Cross validation: k- fold validation

- Divide the training data (after removing the test data) randomly into k - folds.
- Perform the following k experiments:
  - Compose the training data by concatenating k-1 folds leaving one fold out.
  - Train the model on those k-1 folds
  - Test it on the left-out fold
  - Record the result
- Report the average of the k experiments.

Learning: Underfitting and overfitting

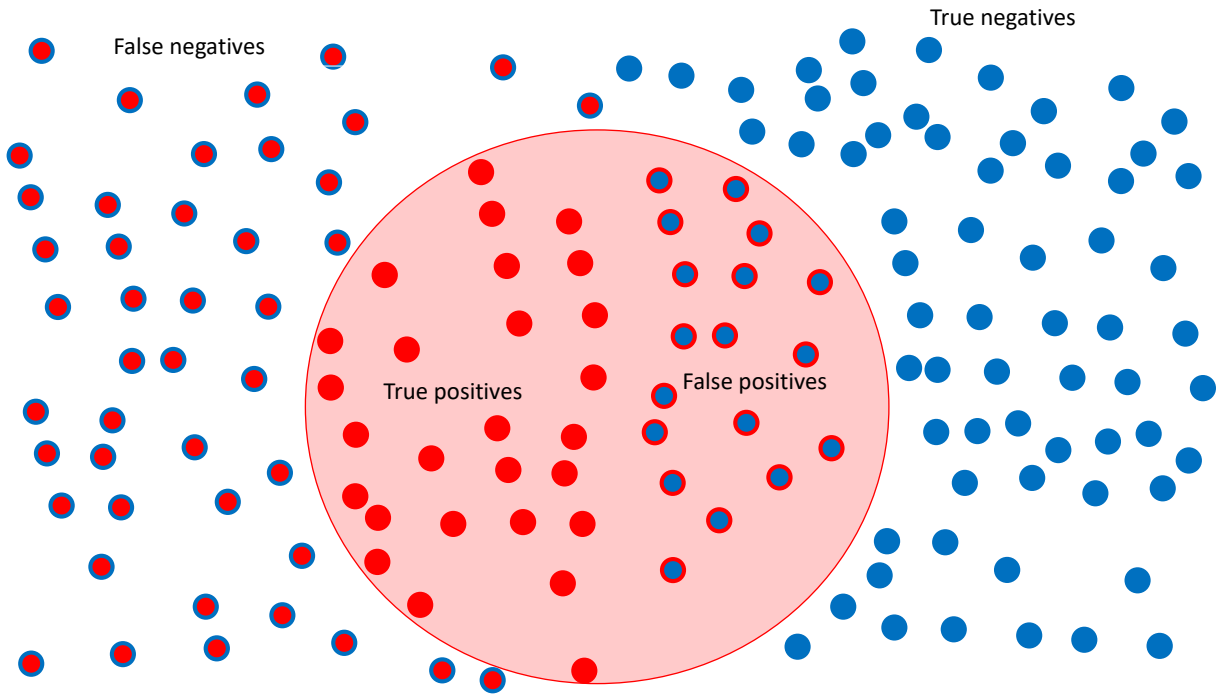
- Underfitting the learned function is too simple in the context of human learning: Underfitting similar to the case where one learns too little.
- Overfitting the learned function is too complex In the context of human learning: Overfitting is more similar to memorizing than learning.

Classification model goodness!

- How good is the model?
- What is the goal of modeling?

Classification outcome

- Consider binary classifier.
- In the data set there are two classes: Positive (P) and negative (N)
- Outcomes of the classification: True positive, true negative, false positive (type I error), false negative(type II error).



Context of information retrieval

NB! Observe notions!

- Relevant elements of the data set. One is interested in finding (retrieving elements of a certain class).
- Precision is defined as:

$$\text{precision} = \frac{|\text{relevant} \cap \text{retrieved}|}{|\text{retrieved}|}$$

- Recall (sensitivity, hit rate, True Positive Rate) is defined as:

$$\text{recall} = \frac{|\text{relevant} \cap \text{retrieved}|}{|\text{relevant}|}$$

Context of classification I

Denote: tp true positive, tn true negative, fp false positive, and fn false negative.

- Precision (positive predictive value):

$$\text{Precision} = \frac{tp}{tp + fp}$$

- Recall (sensitivity, hit rate, TPR):

$$\text{Recall} = \frac{tp}{tp + fn}$$

- True negative rate (Specificity, selectivity):

$$\text{TNR} = \frac{tn}{tn + fp}$$

- Accuracy:

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

- Predicted positive condition rate

$$\text{Predicted positive condition rate} = \frac{tp + fp}{tp + tn + fp + fn}$$

F_1-score

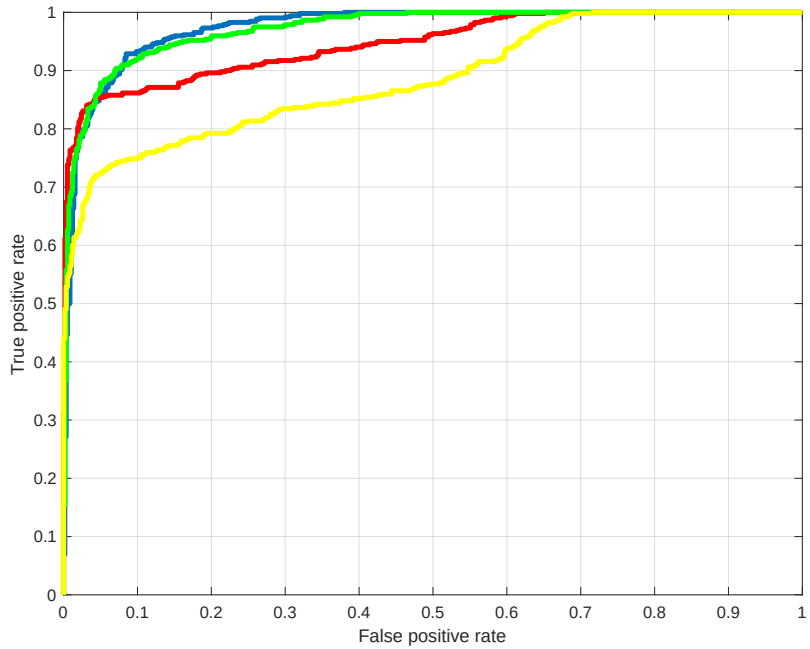
F_1-score is harmonic average of precision and recall.

- $$F = 2 * \frac{\text{precision} \quad \text{recall}}{\text{precision} + \text{recall}}$$
- More general definition:

$$F_\beta = (1 + \beta^2)\frac{\text{precision} \quad \text{recall}}{\beta^2\text{precision} + \text{recall}}$$

Receiver Operating Characteristic or ROC curve

- Let $\mathcal{D} = \{x_i, y_i\}$ is the labeled data set.
- Assume also that $\delta(x) = \mathbb{I}(f(x) > \tau)$ - decision rule. $f(x)$ is the confidence function and τ threshold parameter
- Each particular value of τ corresponds to a certain decision rule.
- For each decision rule one may compute recall and false positive rate.
- Associate recall values with the axis Y and false positive rate values with axis X.



Decision trees

- Non-parametric supervised learning technique.
- Tree-like graph is used to represent the model of decision making and possible consequences of such decisions.
- Internal nodes are conditions (questions). terminal nodes represent labels of classes.
- Questions or conditions play a role of features. Answers to the questions are referred as feature values.
- Training a tree model is referred as tree growing.

Information gain

Definition

Information gain G_I of an action is the decrease of the ambiguity achieved as the result of the action.

- In the context of decision tree growing the action is splitting the node.
- If entropy is chosen as the cost function then information gain is defined as follows:

$$G_I = E - (E_l \cdot p_l + E_r \cdot p_r)$$

where E is the entropy before splitting E_l is the entropy of left child and E_r is the entropy of the right child. Indexes r and l have the same meaning for the proportions p.

Growing a tree 1

Greedy heuristic is the most popular technique. Let F be the possible set of features and S is the subset of data. The idea is to find most useful feature (among remaining) at each node.

$$j(S) = \arg\min_{j \in F} \text{cost}(\{x_i, y_i : x_i \in S, x_{i,j} = c_k\}) + \text{cost}(\{x_i, y_i \; x_i \in S, x_{i,j} \neq c_k\})$$

Classification cost:

$$\hat{\pi}_c = \frac{1}{|S|} \sum_{x_i \in S} \mathbb{1}\{y_i = c\}$$

Cost functions

- Entropy:

$$E(\hat{\pi}) = -\sum_{c=1}^{C} \hat{\pi}_c \log_2 \hat{\pi}_c$$

  Minimizing entropy is equivalent to maximizing information gain which is $\mathbb{H}(Y) - \mathbb{H}(Y|X_j)$.

- Gini index:

$$G = \sum_{c=1}^{C} \hat{\pi}_c (1 - \hat{\pi}_c)$$

Growing a tree 2

- Repeat:
  - For each feature divide data into corresponding subsets. Evaluate accuracy of such split with respect to response variable.
  - "Most accurate" feature wins. It will become condition at a given node.
  - Exclude chosen feature from the feature set.
- Until no more features left.

Growing the tree: case of continues features

Denote X the matrix where columns correspond to different features and rows correspond to the different observation points.

- If all the data points are of the same class return the leaf node that predicts this class.
- Among all splitting points for each column find the one giving largest information gain.
- Then chose the column with the maximum gain.
- Perform splitting.
- If stopping criteria is satisfied return the tree.
- If stopping criteria is not satisfied apply tree growing procedure to each child.

Pruning

- In order prevent overfitting stop growing the tree when the decrease is not sufficient to justify adding extra subtree.
- Grow a full tree and then prune the branches giving less decrease in error.

Support Vector Machines: Separability

Linear separability

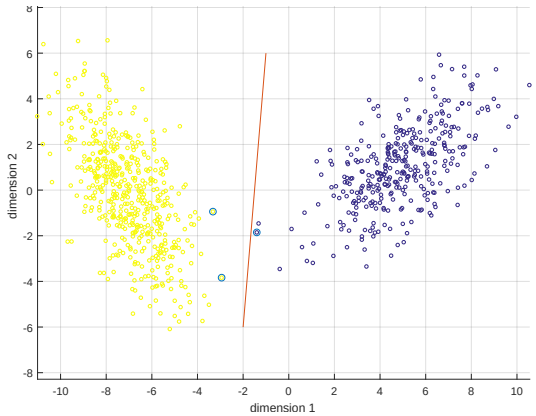
Two classes are said to be linearly separable in $\mathbb{R}^n$ -if there exists a hyperplane dividing the space into two subspaces such that all the elements of the first class belong to one subspace and the elements of the second class belong to the other subspace.

Or

-if there exist n - dimensional vector a and scalar b such that for the elements of one class $x^T a > b$ and for the elements of the second class $x^T a < b$

Separability

- If two classes are linearly separable it is possible to construct two hyperplanes, parallel to the "separating" hyperplane, such that first hyperplane would contain at least one point of the first class and second hyperplane will contain at least one point of the second class.
- The training data points belonging to these hyperplanes are referred as support vectors and the distance between the hyperplanes is referred as margin.
- In order to determine maximum margin hyperplane nonlinear programming optimization is required. First margin is expressed as the function of the coefficients of separating hyperplane. Second optimization problem is solved.

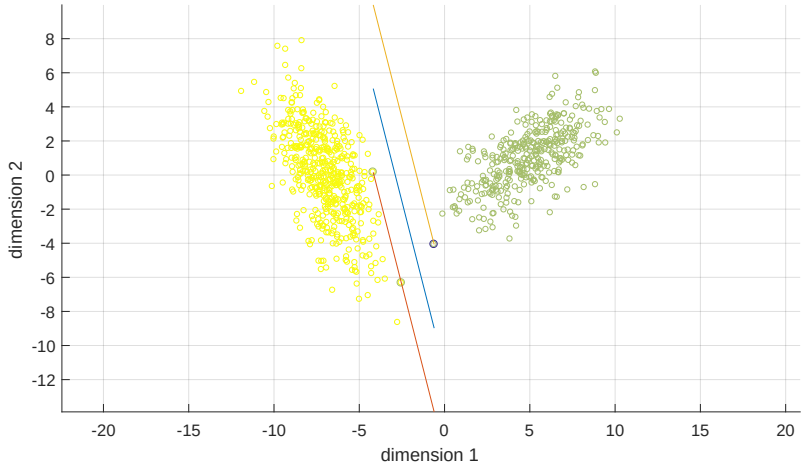


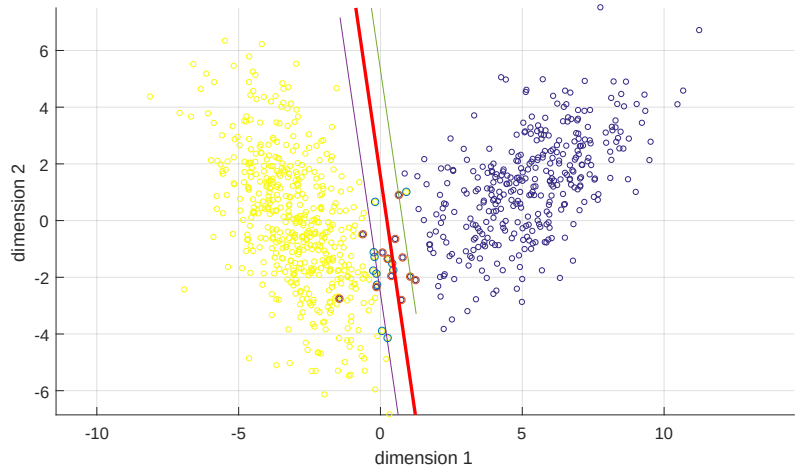
Maximum margin hyperplane

- For the hyperplane $x^T a + b$ vector $a = (a_1, \ldots, a_n)$ is n - dimensional vector representing the normal direction to the hyperplane.
- Then the distance (margin) from the separating hyperplane to the hyperplanes containing points of each class (see previous slide) would be $M = ||a||^{-1}$.
- The optimization problem may be stated in terms of finding vector a that would maximize margin

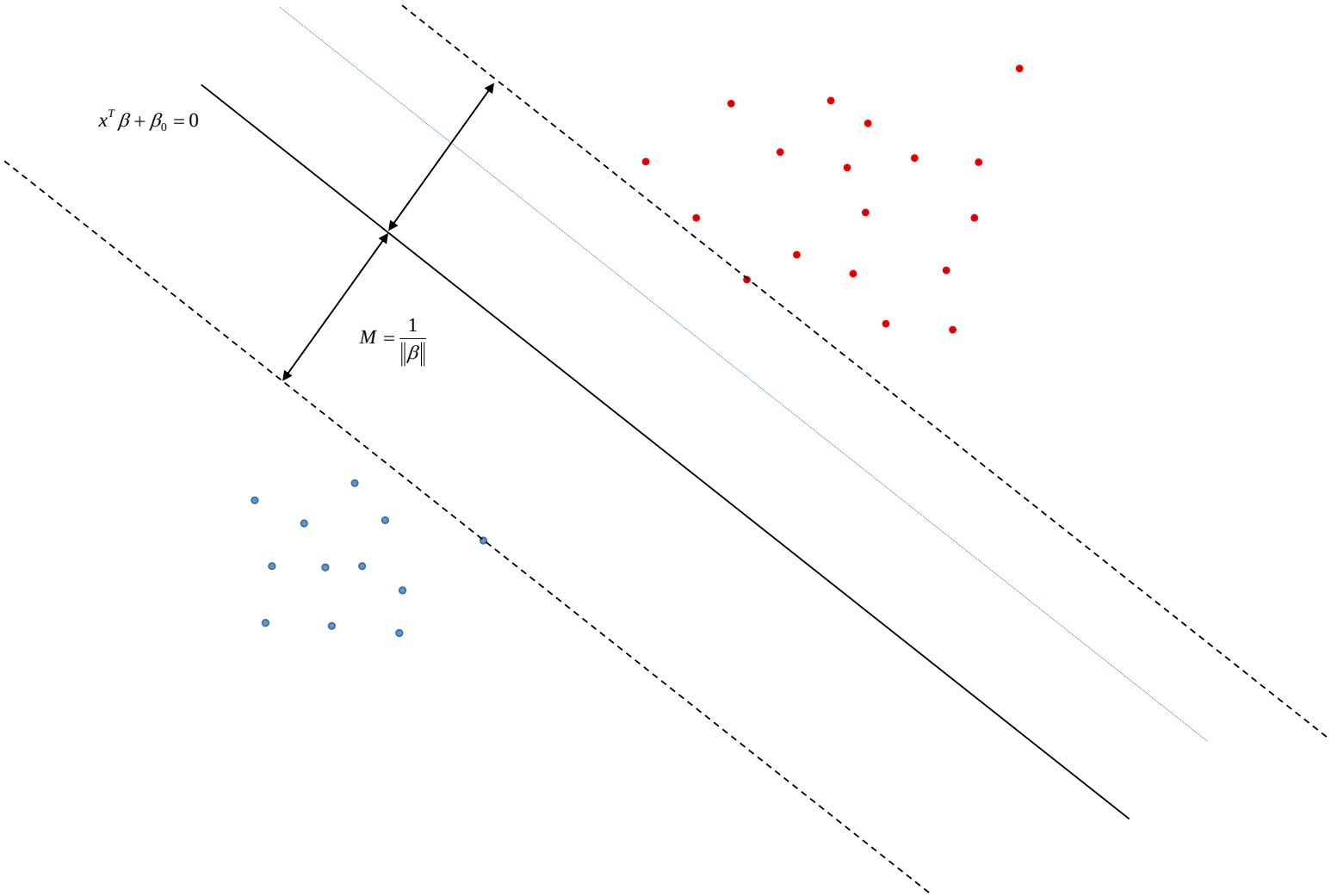
Hard and Soft Margin cases

Hard margin case is depicted by the left figure and soft margin by the figure on the right side.





Linear Case: Hard-Margin case



Linear Case: Hard-Margin case

- Let N pairs (x_i, y_i) where $i = 1, \dots N$ constitute the training data; $x_i \in \mathbb{R}$ and $y_i \in \{-1, 1\}$.
- Define the hyperplane as $\{x : f(x) = x^T b_1 + b_0 = 0\}$, where $||b|| = 1$.
- Classification rule induced by $f(x)$ is

$$G(x) = \text{sign}[x^T b_1 + b_0].$$

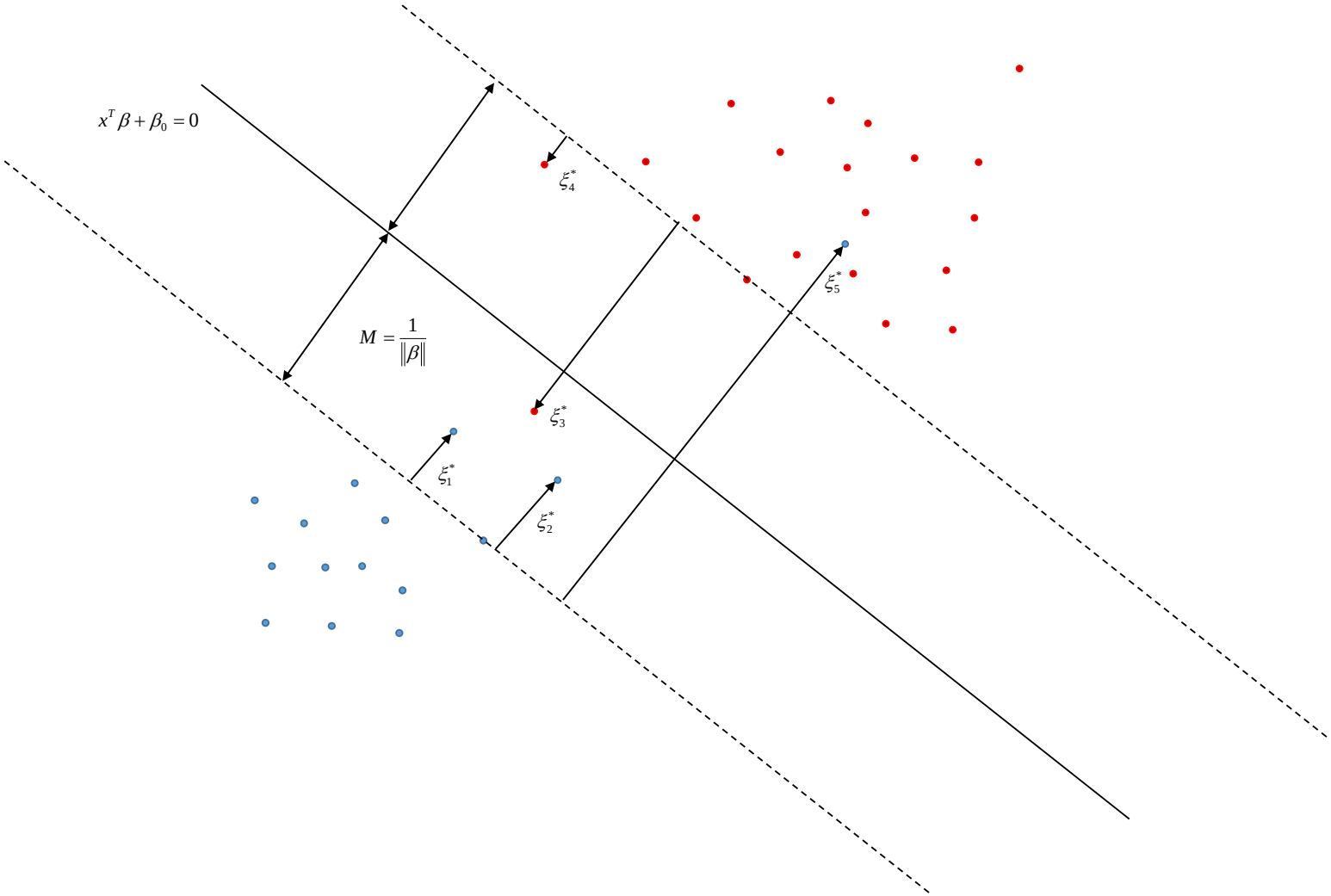
- Classes are separable $\Rightarrow \exists f(x) = x^T b_1 + b_0 : y_i f(x_i) > 0, \forall i$.
- One is able to find the hyperplane that creates the biggest margin between the training points leads following optimization problem:

$$\max_{b_1, b_0, ||b||=1} M$$
$$\text{subject} \quad \text{to} \quad y_i(x^T b_1 + b_0) \geq M, i = 1, \dots, N,$$

or in a more convenient form:

$$\min_{b_1, b_0} ||b||$$
$$\text{subject} \quad \text{to} \quad y_i(x^T b_1 + b_0) \geq 1, i = 1, \dots, N; \quad M = 1/||b||$$

Linear Case: Soft-Margin case



Linear Case: Soft-Margin case

If the classes in training data overlap then we are talking about soft margin case.

- One of the possible way is to maximize M, whereas it is allowed to some points to be on the "wrong" side of the plane.
- Define the slack variables $\xi = (\xi_1, \ldots, \xi_N)$
- The first way to modify optimization problem is $y_i(x^T b_1 + b_0) \geq M - \xi_i$. (results in a non-convex optimization problem).
- The second way is $y_i(x^T b_1 + b_0) \geq M(1 - \xi_i)$. (results in a convex optimization problem).
- $\forall i, \xi_i \geq 0, \sum_{i=1}^{N} \xi_i$ is limited by some constant
- The second way leads to the standard vector classifier.

Soft-Margin case

- For the case when classes are allowed to overlap some points are allowed to be on the 'wrong' side of the hyperplane. The distances of these points from their margin are denoted as ξ_i.
- The first way to describe the constraint is

$$y_i(x_i^T b_1 + b_0) \geq M - \xi_i$$

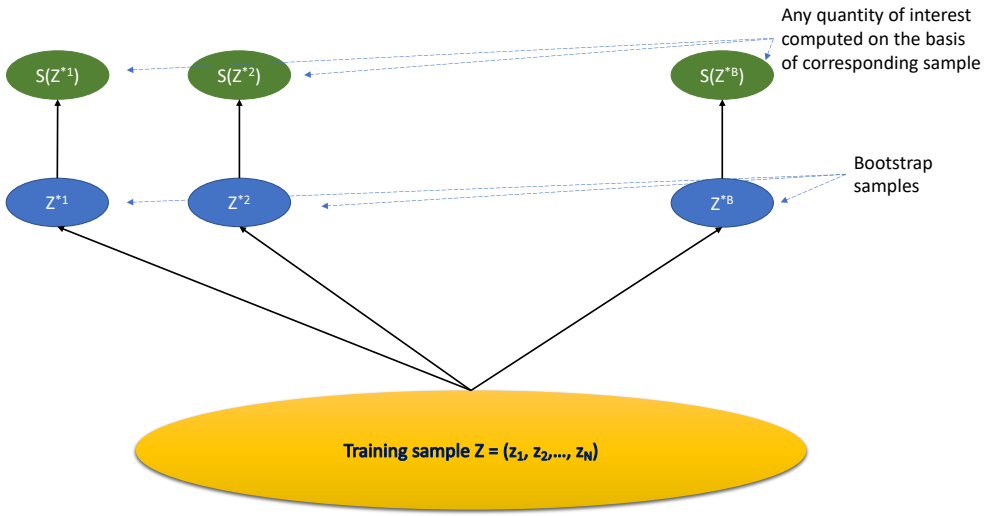
  In this form constraint described overlap in actual distance but lead non convex optimization problem.
- The second way is

$$y_i(x_i^T b_1 + b_0) \geq M(1 - \xi_i)$$

  In this form overlap is described in relative distance but lead convex optimization problem. ξ is the proportional amount by which the prediction $f(x_i) = x_i^T b_1 + b_0$ is on the wrong side of its margin.
- Bounding sum of ξ_i allows to bound the total proportion amount by which the predictions fall into 'wrong' side of their margin. Misclassifications occur when $\xi_i > 1$.

Model quality: Bootstrap I

- Let $Z = (z_1, \ldots, z_n)$ is the training set.
- Draw randomly data sets with replacement (the samples are independent) from Z. This will result in B bootstrap data sets.
- Fit the model for each of B data sets. Examine behaviour over B replacements.
- This approach allows to estimate any aspect of distribution $S(Z)$.



Model quality: Bootstrap II

- Let $f^{*b}(x_i)$ be the predicted value at x_i from the model fitted to the b^{th} bootstrap dataset.
- Error estimate is given by:

$$\mathcal{E}_{boot} = \frac{1}{B}\frac{1}{N}\sum_{b=1}^{B}\sum_{i=1}^{N} L\Big(y_i, \hat{f}^{*b}(x_i)\Big).$$

- Better bootstrap estimate may be derived by mimicking cross-validation. For each observation we will keep track of predictions from bootstrap samples not containing this observation. This is referred as leave-one-out bootstrap estimate of prediction error and is defined by the following equation.

$$\mathcal{E}^{(1)}_{boot} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{C^{-i}}\sum_{b\in C^{-i}} L\Big(y_i, f^{*b}(x_i)\Big).$$

- Notation here may cause a problem. You are welcome to fix it :) .

Bagging

- Induced from the bootstrap technique (which is used to assess accuracy of estimate).
- Draw B samples with replacements and train the model on each sample.
- The bagging estimate then is defined by:

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^{*b}(x).$$

Random Forests

The idea is to build large collection of de-correlated trees, and then average them.

- For $b = 1$ to B:
  - Draw a bootstrap sample Z^* of size N from the available training data.
  - Grow tree T_b. Repeat recursively for each terminal node until minimum node size is reached.
    - ★ Select m variables from p.
    - ★ Pick the best variable among m.
    - ★ Split the node.
- Output the ensemble of trees $\{T_b\}_1^B$.
- Prediction:
  - Regression: $\hat{f}_{\text{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^B T_b(x)$.
  - Classification: $\hat{C}_{\text{rf}}^B(x) = \text{mode}\{\hat{C}_b(x)\}_1^B$.

Committee learning

- Some times referred as ensemble learning.
- The idea is to combine a number of weak (accuracy is slightly larger than of random guessing) classifiers into a powerful committee.
- Motivation is to improve estimate by reducing variance and sometimes bias.

Boosting

- The final prediction is given by:

$$G(x) = \text{sign}\Big(\sum_{m=1}^{M} \alpha_m G_m(x)\Big).$$

  which is weighted majority vote of classifiers $G_m(x)$. Here α_m are weights describing contribution of each classifier.
- While on the first view result is very similar to the bagging, there are some major differences.
- Two class problem where output variable coded as $Y \in \{-1, 1\}$.
- For the classifier $G(X)$ error rate is given by:

$$\overline{\text{err}} = \frac{1}{N} \sum_{i=1}^{N} I(y_i \neq G(x_i)),$$

  where N is the power of training data set.

Thank you!

Dank je wel!!!