Machine Learning

Methods of Knowledge Based Software Development

S. Nõmm

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

01.12.2017

Frameworks of the following four lectures

- Following four lectures are mainly devoted to the problems of supervised learning. Some problems and techniques related to unsupervised learning will be briefly mentioned. Those who are willing to study Machine learning problems in a more detailed way are welcome to join ML course during the spring term.
- Sometimes supportive code in Python programming language will be provided.
- Lecture slides are meant to support the lecture, not to replace it.

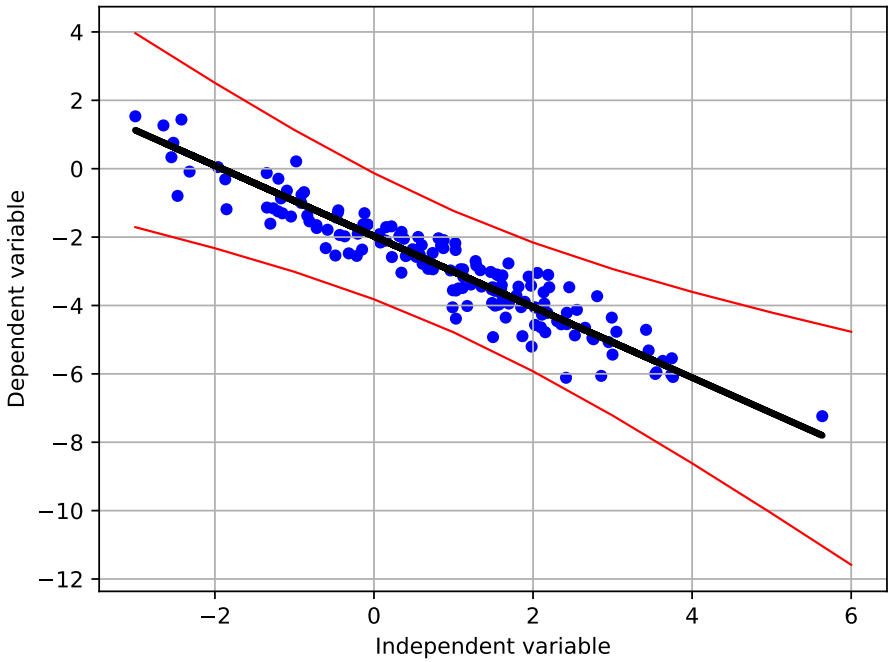
Three questions. What? Why? How? (Organisation)

Let us suppose that you have a particular problem to solve.

- **What** are you going to do? One is expected to choose method to solve the problem.
- **Why** are you going to use this method? One is expected to explain underlying theory to support choice of the method.
- **How** are you going to implement this? Explain technical details of your solution.
- **Validate**. This question was not stated in title but it is still expected to be answered. How good proposed solution is?

If one is assigned a machine learning exercise, these four questions are expected to be answered!

Linear regression: probably the oldest machine learning technique



- Find leaner correlation coefficient.
- Compute coefficients of the linear equation

$$\hat{y} = ax + b$$

- Evaluate the model

- In multivariate case it is required to identify coefficients of the model

$$\hat{y} = a_1 x_1 + a_2 x_2 + \ldots + a_n x_n + b.$$

This leads the necessity to choose variables (perform model building).

Data mining and Machine Learning

Data mining

- Clustering Grouping the elements of unlabeled set based on certain similarity criteria.
- Classification Learning existing grouping on the basis of the labeled (training) set.
- Association pattern mining Discovering rule based relations between the elements of the data set.
- Outlier analysis Finding and analyzing elements that differ much from the other elements in the data set.

Machine learning

- Supervised learning Inferring function (classifier) parameters on the basis of labeled data set (training data).
- Unsupervised learning Finding inferences on the basis of unlabeled data set.

Main steps

- Data acquisition.
- Data preparation.
- Feature selection.
- Model training.
- Tuning (sometimes).
- Model validation.

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 measure of similarity or distance between two elements.

Definition

A function $S : X \times X \to \mathbb{R}$ is called 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 function: Examples 1

- Euclidean distance

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

```
from scipy.spatial.distance import euclidean
e_dist_1_2 = euclidean(p_1,p_2)
```

- Manhattan distance also referred as city block distance or taxicab distance

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

```
from scipy.spatial.distance import cityblock
m_dist_1_2 = cityblock(p_1,p_2)
```

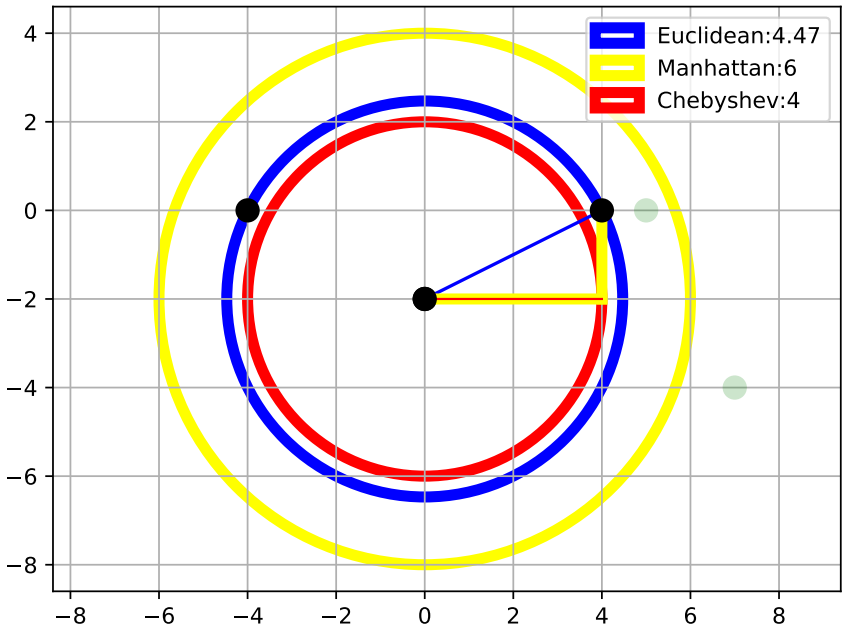
- Chebyshev distance

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

```
from scipy.spatial.distance import chebyshev
c_dist_1_2 = chebyshev(p_1,p_2)
```

where $p_1 = (0,-2)$ and $p_2 = (4,0)$

Distance function: Examples 2



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

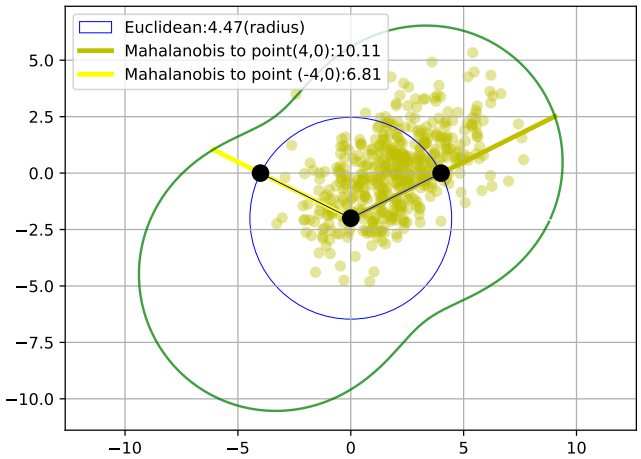
$$S(x,y) = \max_i\big(\mid x_i - y_i \mid\big)$$

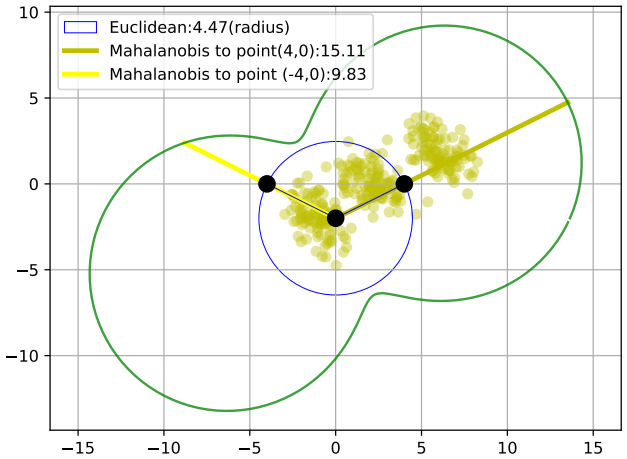
Distance function: Examples 3

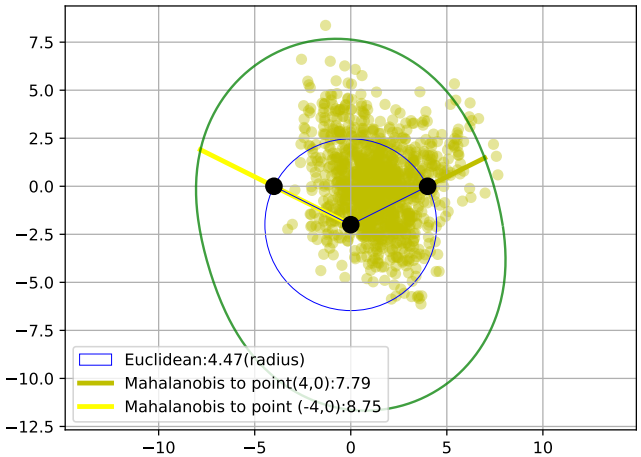
Mahalanobis distance

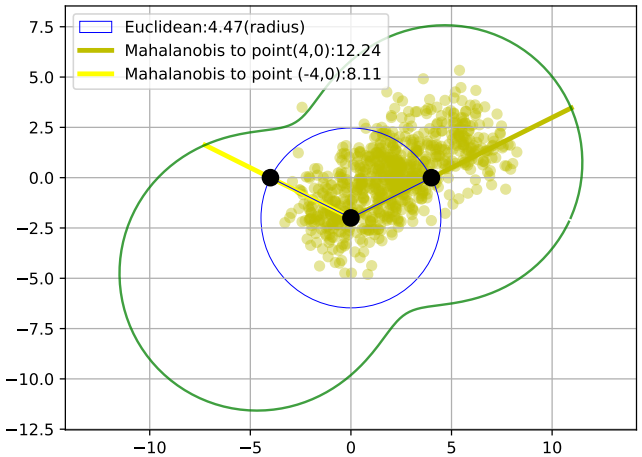
$$S(x,y) = \sqrt{(x-y)^T C^{-1}(x-y)}$$

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

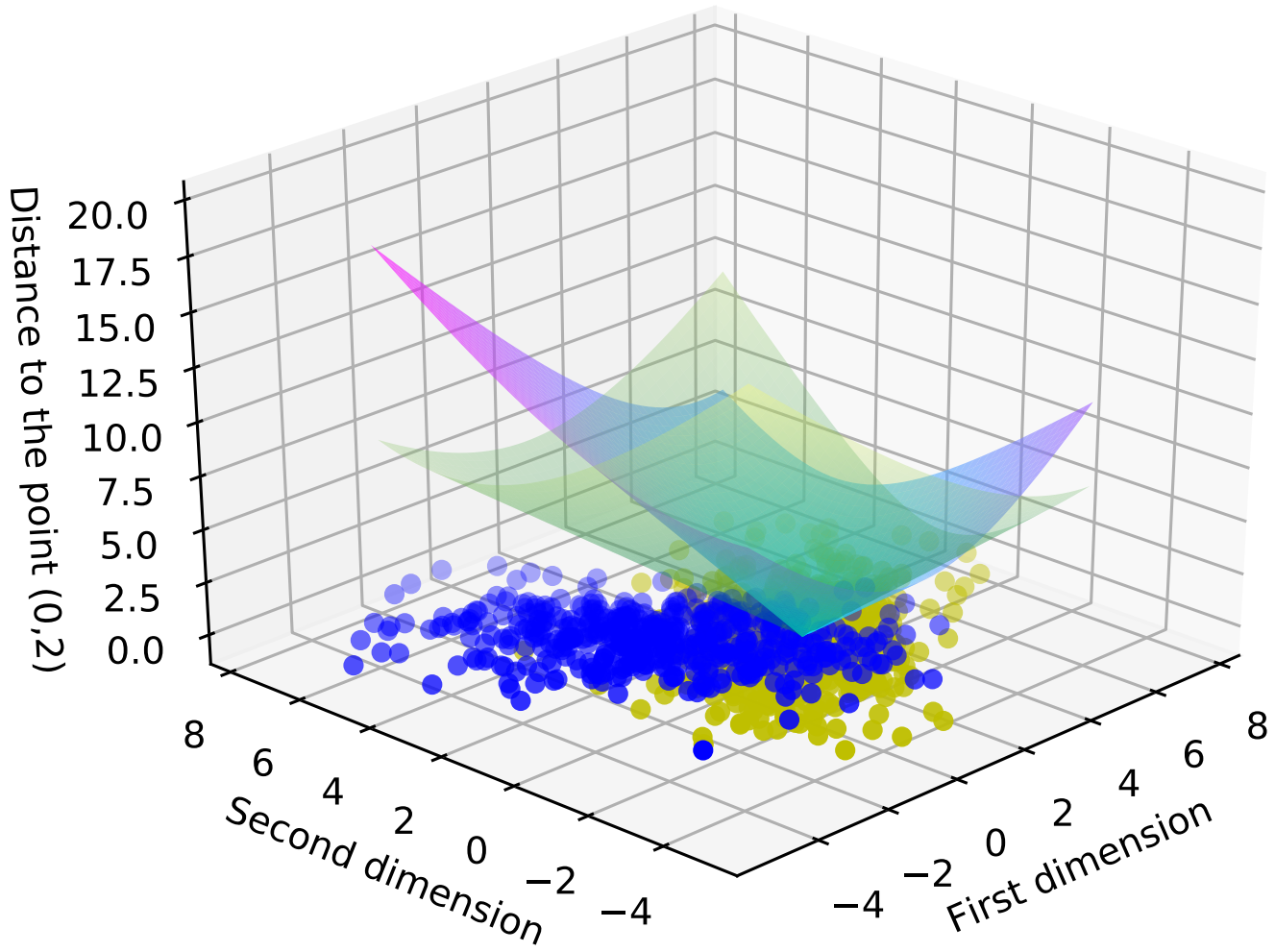








Distance function: Examples 4



- Impact of the rotation of 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. Cosine distance is defined as follows

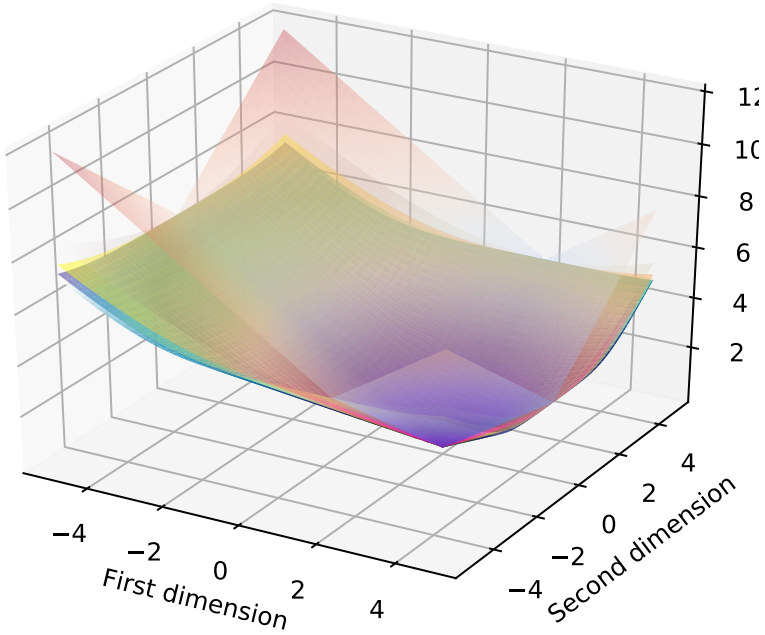
$$S_C(x,y) = 1 - S_c(x,y)$$

- Minkowski distance

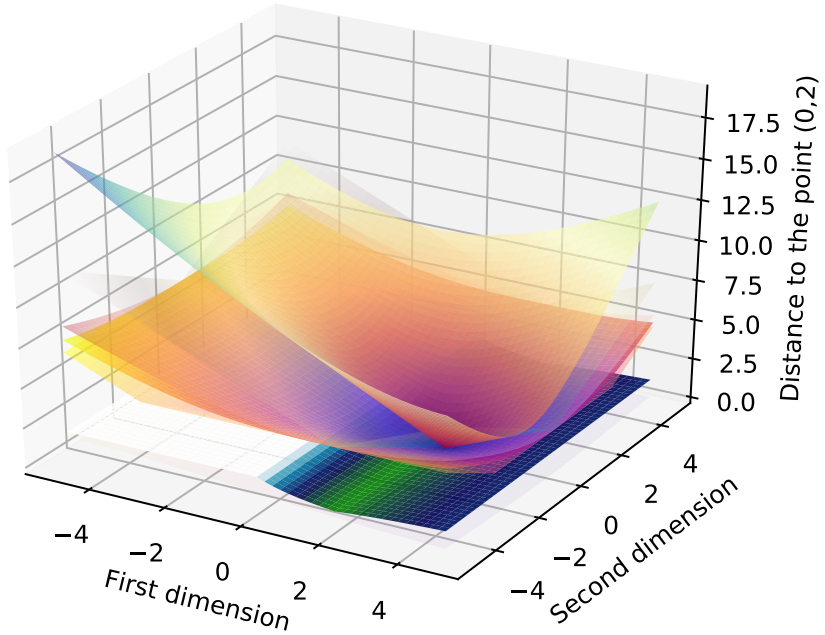
$$S(x,y) = \left(\sum_{i=1}^{d} \mid x_i - y_i \mid^p\right)^{1/p}$$

Mahalnobis and distance functions discussed on this page are called from in Python in a similar way to those presented in slide 8

Distance function: Examples 6



- 3D representation of the Minkovski distances for different values of parameter p



- 3D representation of all distance functions discussed above.

Distance function: Examples 7

- 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".
- Hamming distance Similar to Levenshtein but with substitution operation only. Frequently used with categorical and binary data.
- Specialized similarity measures Distance and similarity functions applicable to the graphs, temporal data etc. These topics are left outside of the framework of the present course.

Impact of High Dimensionality (Curse of Dimensionality)

Curse of dimensionality - term introduced by Richard Bellman. Referred to the phenomenon of efficiency loss by distance based data-mining methods. Let us consider the following example.

- Consider the unit cube in d - dimensional space, with one corner at the origin.
- What is the Manhattan distance from the arbitrary chosen point inside the cube to the origin?

$$S(\bar{0}, \bar{Y}) = \sum_{i=1}^{d} (Y_i - 0)$$

  Note that Y_i is random variable in $[0, 1]$
- The result is random variable with a mean $\mu = d/2$ and standard deviation $\sigma = \sqrt{d/12}$
- The ratio of the variation in the distances to the mean value is referred as *contrast*

$$G(d) = \frac{S_{max} - S_{min}}{\mu} = \sqrt{\frac{12}{d}}$$

Clustering: overview

Let $\mathcal{D}$ be n -dimensional data set. Divide $\mathcal{D}$ into:

- k subsets. Usually solved using representative based algorithms like k-*means*, k -medoids, etc. Sometimes probabilistic EM algorithm is used.
- subsets of certain density. Usually solved by grid- and density- based algorithms, like DBSCAN, DENCLUE etc.
- subsets, such that minimal/maximal subset power is m. Usually solved by hierarchical clustering algorithms, like bottom-up agglomerative method or top-down divisive methods
- subgraphs. Special case related to graph and social networks analysis.
- ... .

For the case of unsupervised learning any parameter which value is not defined by the algorithm is referred as *hyper parameter*.

Clustering: Feature selection

The role of feature selection is to remove noisy attributes. Feature selection methods for clustering are frequently divided into two groups:

- Filter models: Term strength, Entropy, Hopkins statistic, etc.
  - Entropy. Let ϕ be the number of grid regions. For k-dimensional there will be $m = \phi^k$ regions. denote p_i fraction of the data points belonging to grid region i, then the probability based entropy is defined as

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

- Wrapper models To be used in conjunction with internal validation criteria.

Clustering: Validation

- Sum of square distances to centroid.
- Intracluster to intercluster ratio.

$$II = \frac{\sum_{(\bar{X}_i, \bar{X}_j) \in P} s(\bar{X}_i, \bar{X}_j)/|P|}{\sum_{(\bar{X}_i, \bar{X}_j) \in Q} s(\bar{X}_i, \bar{X}_j)/|Q|}$$

  where P is the set of all pairs of elements belonging to the same cluster and Q is the set of all remaining pairs. Smaller values indicate better clustering behaviour.
- Silhouette coefficient.

$$S_i = \frac{D^{\text{out}}_{\text{min},i} - D^{\text{in}}_{\text{avg},i}}{\max\{D^{\text{out}}_{\text{min},i}, D^{\text{in}}_{\text{avg},i}\}}$$

  Large positive values indicate highly separated clusters.
- Probabilistic measures

Classification: k - nearest neighbours

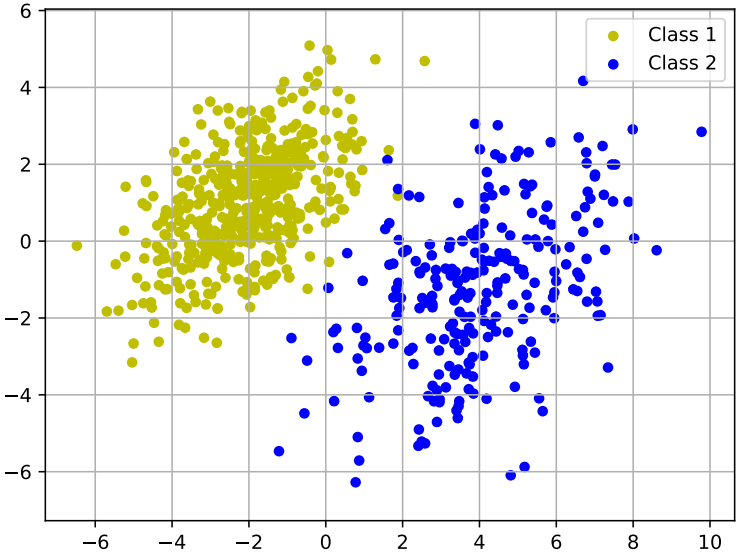
Learning existing grouping on the basis of the labeled (training) set.

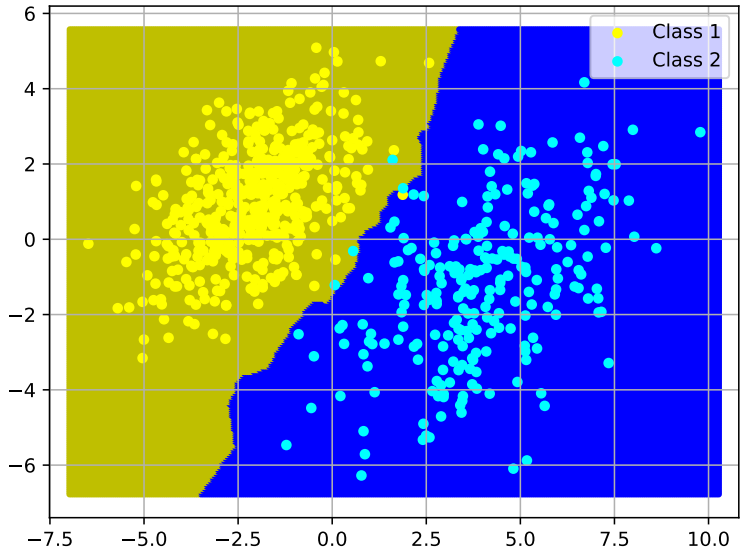
- Let D denote training (labeled) data set.
- For each unlabeled point (point to be classified)
  - Find k - nearest neighbours. (On this step distance function is necessary!!!)
  - Assign mode (majority) label for k - nearest neighbours.

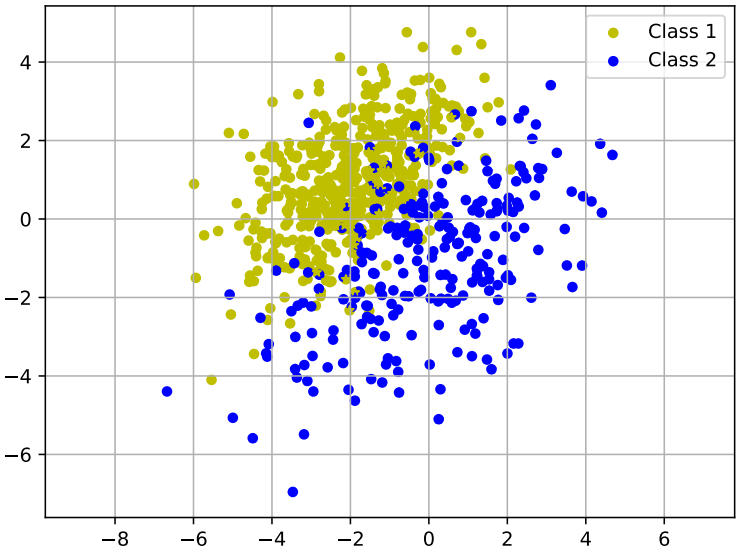
```
from sklearn.neighbors import KNeighborsClassifier
knn_1 = KNeighborsClassifier(n_neighbors=neighbors)
knn_1.fit(D_train, labels_train)
```

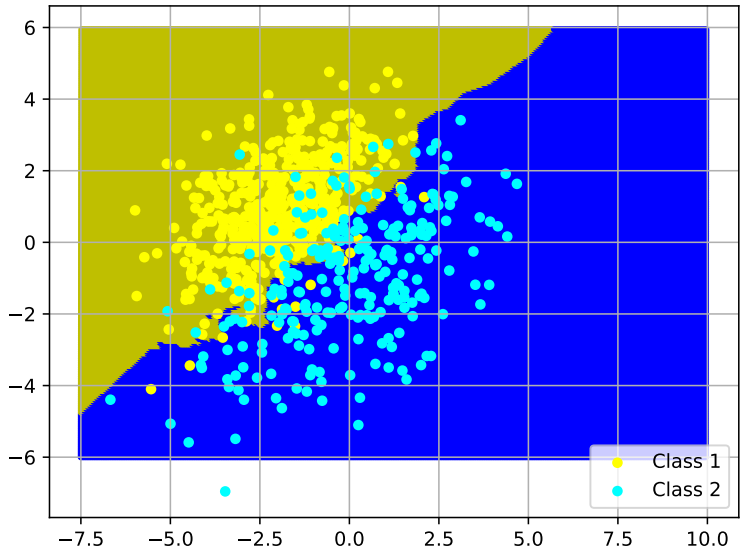
where D_train is the training set and $labels\_train$ are the labels.

k - nearest neighbors, 2D

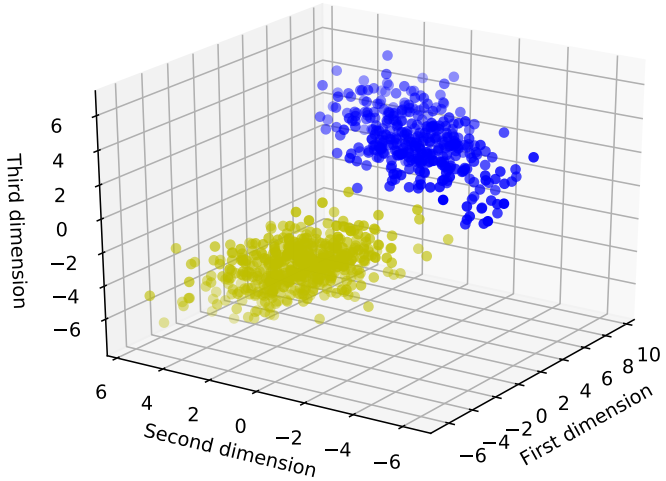


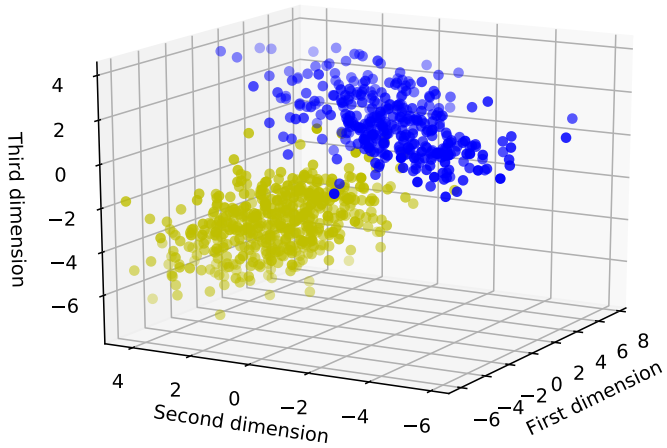


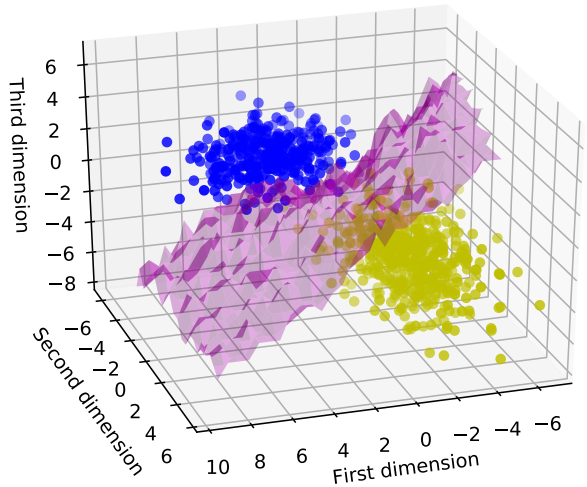


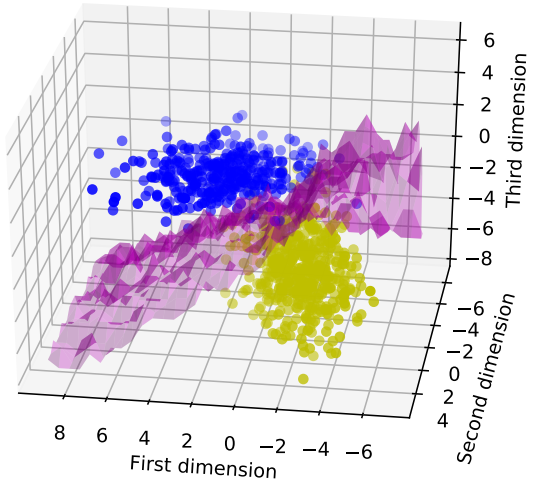


k - nearest neighbors, 3D

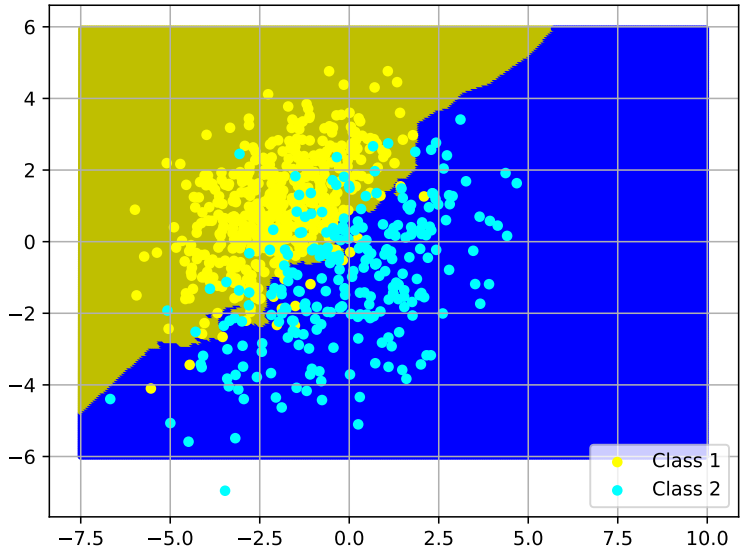


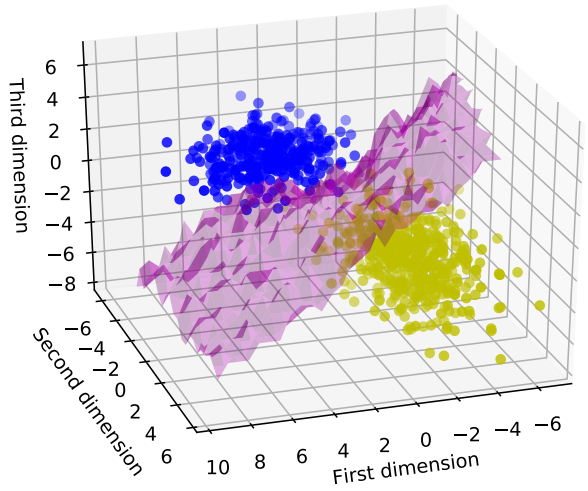






Decision boundary





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

Validation

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

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

```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
labels_predicted = knn_1.predict(D_valid)
cm_1 = confusion_matrix(labels_list_valid,labels_predicted)
as_1 = accuracy_score(labels_list_valid,labels_predicted)
```

- How reliable these parameters are ?

Cross validation

- Non-exhaustive do not use all possible ways of splitting into training and validation sets
  - k - fold.
  - Holdout.
  - Repeated random sub-sampling.
- Exhaustive: use all possible ways to divide the data set into training and validation sets
  - Leave p-out cross validation.
  - Leave one out cross validation.

Cross validation: k- fold validation

- Divide the training data (after removing test data) randomly into k - folds.
- Perform 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 when 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.

Feature selection for classification

- Case of categorical data: Gini Index or Entropy

$$G(v_i) = 1 - \sum_{j=1}^{k} p_j^2; \qquad E(v_i) = -\sum_{j=1}^{k} p_j \log_2(p_j)$$

  where p_j is the fraction of data points containing attribute value v_i. Lower values of Gini index or Entropy imply greater discriminative power.

- Case of numeric data: Fisher 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 discriminative power of the variable.

- Wrapper methods.

Decision Trees

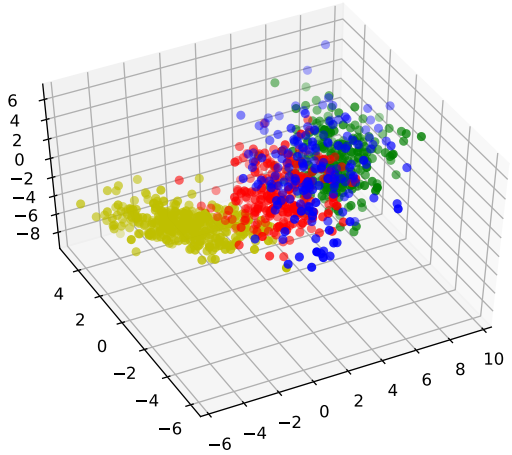
Decision Trees is a non-parametric supervised learning technique for classification and regression. Tree-like graph model is used to describe classification decisions.

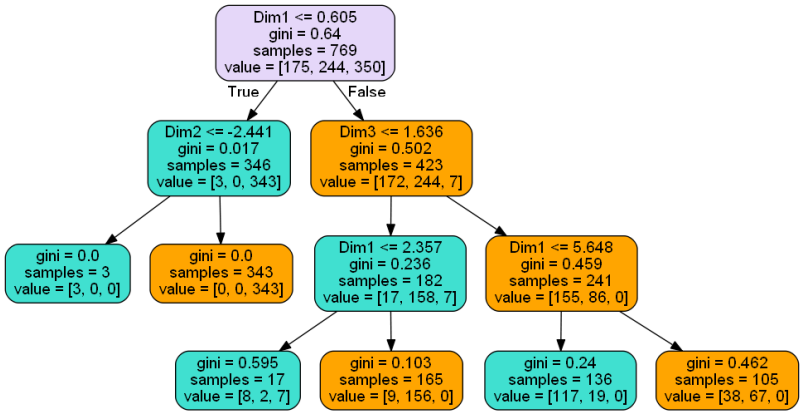
- Structure
  - Internal tree nodes are questions.
  - Leaves represent the answers - *labels* or *classes*.
  - Data is usually given in tabulated form.
  - Each row represents one data- or observation- point.
- Training
  - What questions to ask?
  - In what order to ask?
  - Which label to predict?

Decision trees: training

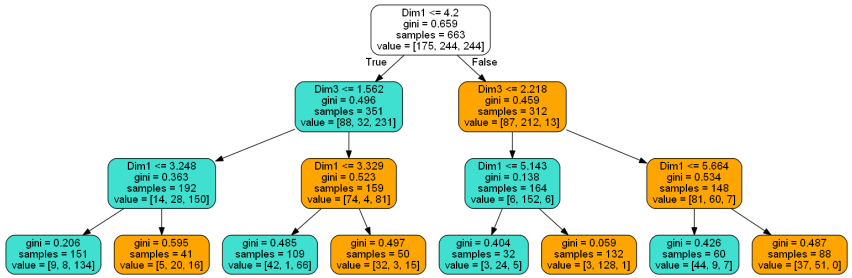
- Greedy heuristic: find the most useful feature for guessing the answer. In other words "If one could ask only one question then what would it be?"
- Continue by using divide and conquer strategy for each set obtained find the most useful feature.
- Continue until: No more features left, some feature at some point in the tree classifies the data 100% accurately, other stopping criteria are possible.

Decision trees: example





Classifier for the "yellow","green" and "blue" sets.



Classifier for the "yellow","green" and "red" sets.

Comparison

- Regression: Easy to represent, stable model structure.
- kNN: Easy to explain, suited for a certain cases, consumes a lot of time, easy to explain.
- Decision tree: Easy to trace, very sensitive, able to handle high dimensionality.

Other techniques

- Logistic regression provides simplicity of the model for the case of categorical labels.
- kNN regression provides relative simplicity of implementation.
- Regression trees.

To be continued ...