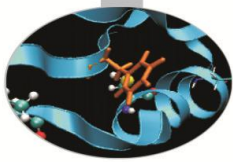
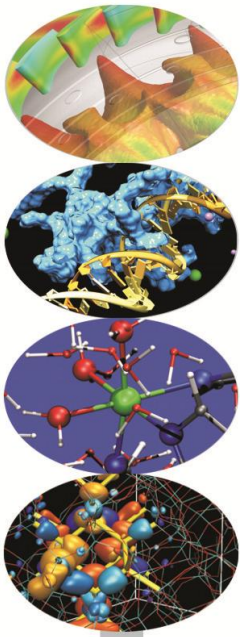
CINECA SCAI
SuperComputing Applications and Innovation



Machine Learning with Spark

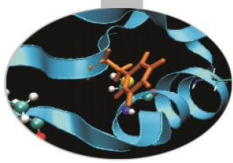


Giorgio Pedrazzi, *CINECA-SCAI*

Bologna, 14/04/2016

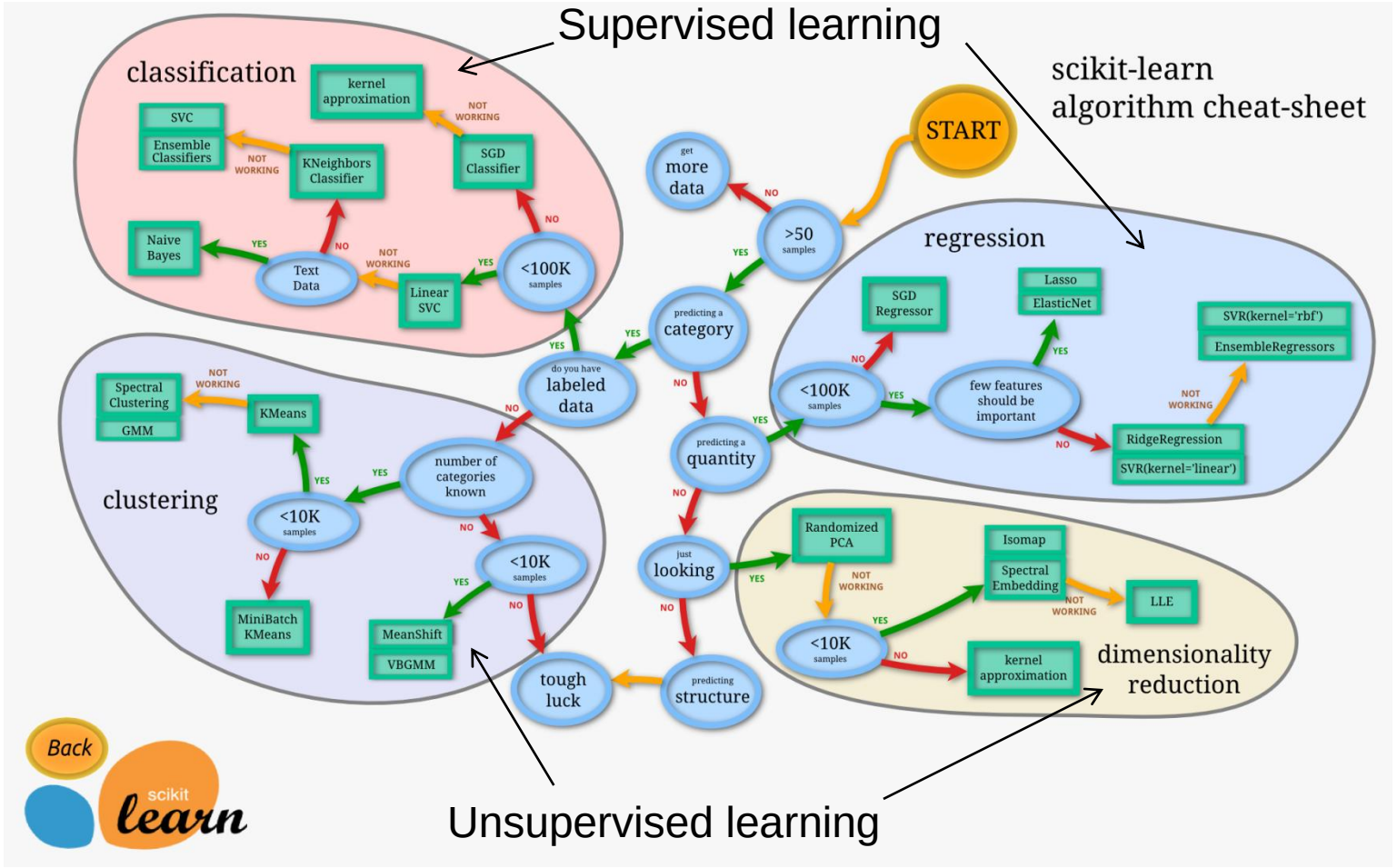
CINECA

Roadmap

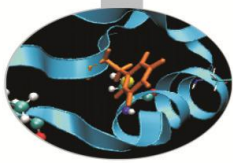


- Unsupervised learning: Clustering
  - Distance measures
  - K-means, Density based, Model based, Spectral
  - Clustering validation
- Supervised learning: Classification
  - Training and test
  - Evaluation metrics
  - Decision tree
  - Naïve Bayes
- Examples with Spark MLlib in Scala and Python

Algorithm cheat-sheet



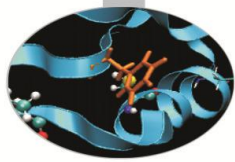
http://scikit-learn.org/stable/tutorial/machine_learning_map/



Unsupervised learning

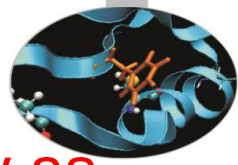
- Clustering (or cluster analysis)
  - no predefined classes for a training data set
  - find *similarities* between data according to characteristics underlying the data and grouping similar data objects into clusters
  - Two general tasks: identify the “natural” clustering number and properly grouping objects into “sensible” clusters
- Clustering Typical aims
  - as a stand-alone tool to gain an insight into data distribution
  - as a preprocessing step of other algorithms in intelligent systems

Typical applications



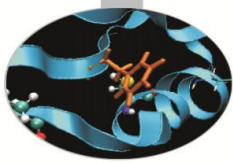
- Scientific applications
  - **Gene expression data**: Discover genes with similar functions in DNA microarray data.
  - …
- Business applications
  - **Customer segmentation:** Discover distinct groups in customer bases (insurance, bank, retailers) to develop targeted marketing programs.
  - …
- Internet applications
  - **Social network analysis:** in the study of social networks, clustering may be used to recognize communities within large groups of people.
  - **Search result grouping:** in the process of intelligent grouping of the files and websites, clustering may be used to create a more relevant set of search results.
  - …

Data Matrix



The problem must be formulated in a mathematical way as a matrix of data containing information on N objects (cases or observations ; rows of the matrix) specified by the values assigned to V variables (columns of the matrix)

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

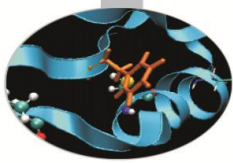


Clustering steps

- Pre processing
- Select a clustering algorithm
- Select a distance or a similarity measure (*)
- Determine the number of clusters (*)
- Validate the analysis

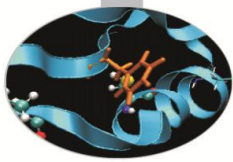
(*) if needed by the method used

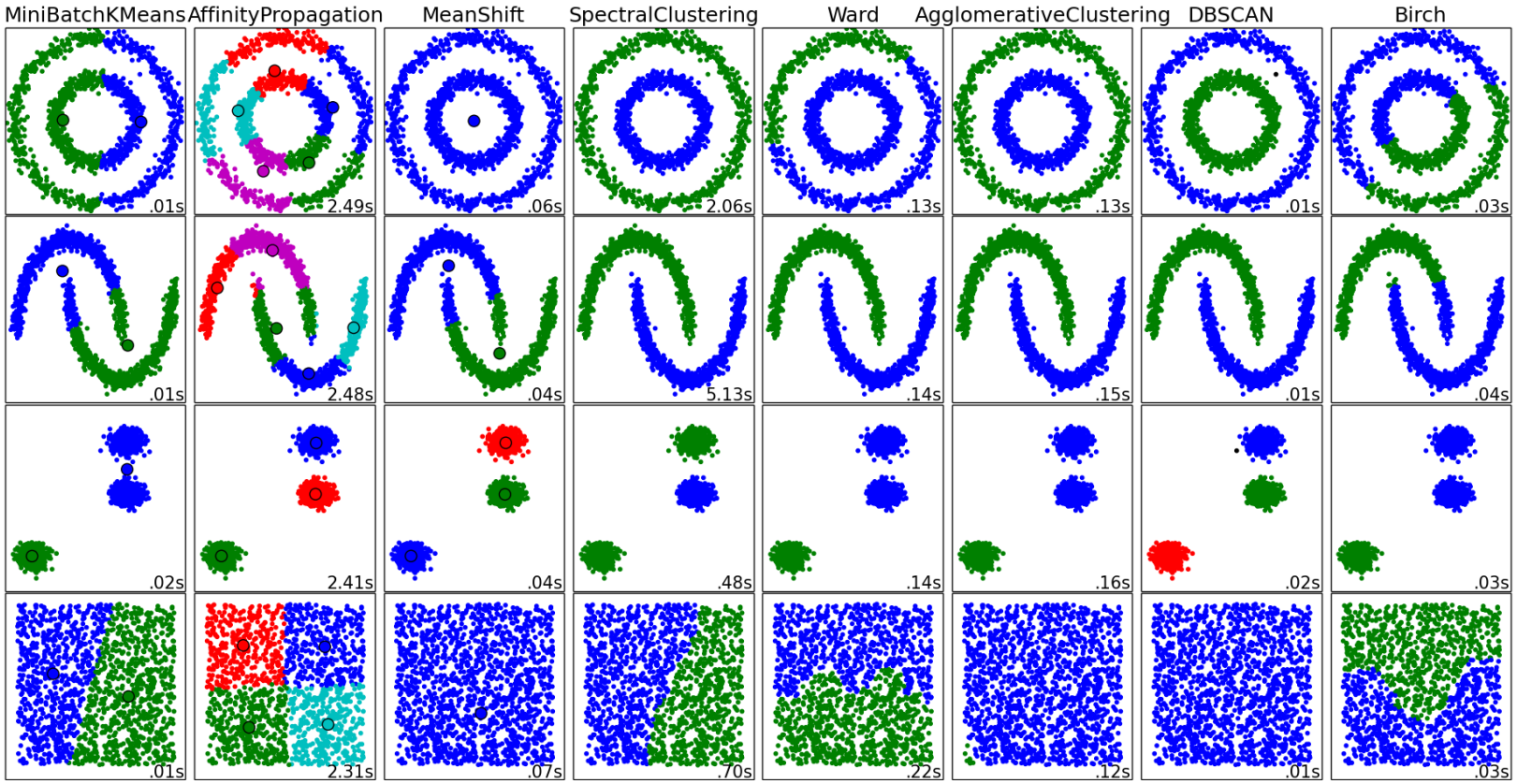
Classification of methods



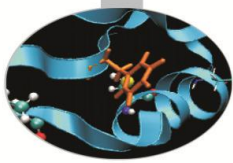
- Distance-based
  - partitioning algorithms
  - hierarchical algorithms
- Density based (DBSCAN)
- Model based
- Spectral clustering
- Combination of methods

Comparison of algorithms





Distance measure



Minkowski distance (L_p Norm)

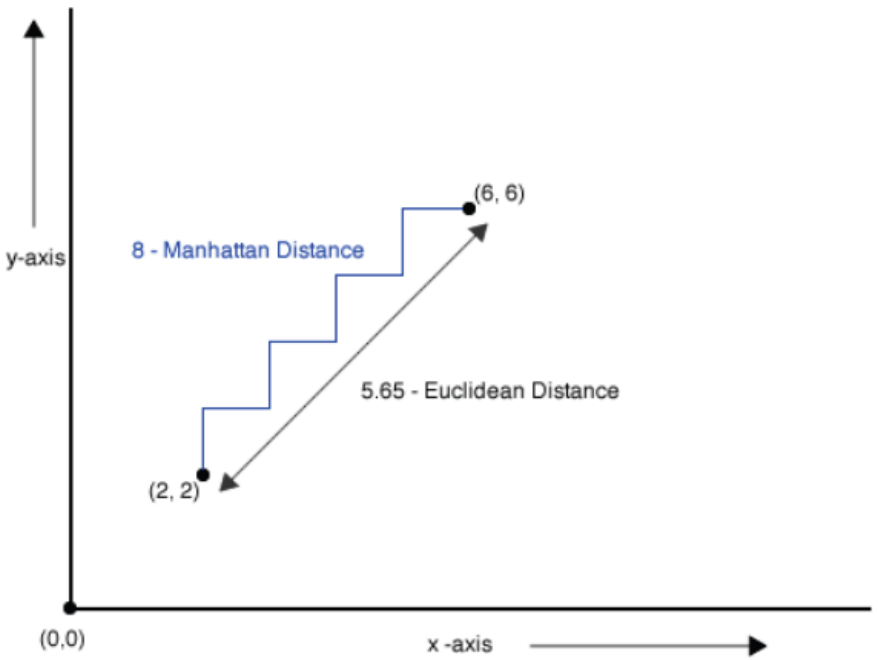
$$d(i,k) = \left[\sum_{j=1}^{d} | x_{ij} - x_{kj} |^{p} \right]^{1/p}$$

Euclidean distance (L_2 Norm)

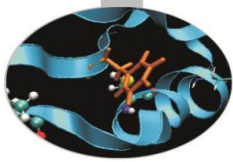
$$d(i,k) = \left[\sum_{j=1}^{d} (x_{ij} - x_{kj})^{2} \right]^{1/2}$$

Manhattan distance (city block distance)

$$d(i,k) = \sum_{j=1}^{d} | x_{ij} - x_{kj} |$$



Distance Measures



- Cosine Measure (Similarity vs. Distance)

  For $\mathbf{x} = (x_1\, x_2 \cdots x_n)$ and $\mathbf{y} = (y_1\, y_2 \cdots y_n)$

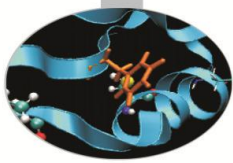
$$\cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \bullet \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|} = \frac{x_1 y_1 + \cdots + x_n y_n}{\sqrt{x_1^2 + \cdots + x_n^2}\sqrt{y_1^2 + \cdots + y_n^2}}$$

$$d(\mathbf{x}, \mathbf{y}) = 1 - \cos(\mathbf{x}, \mathbf{y}) = 1 - \frac{\mathbf{x} \bullet \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|}$$

$$0 \le d(\mathbf{x}, \mathbf{y}) \le 2$$

  - Nonmetric vector objects: keywords in documents, gene features in micro-arrays, …
  - Applications: information retrieval, biologic taxonomy, ...

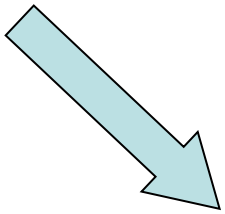
Similarity measures

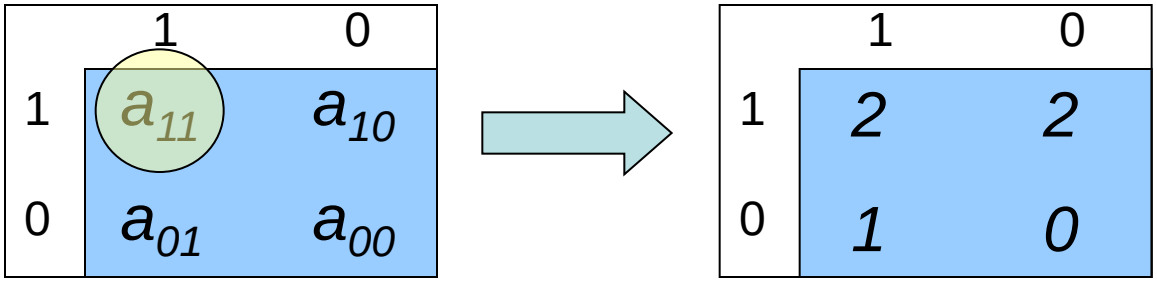


Correspondent 1's

x_k: 0 1 1 0 1

x_j: 1 1 0 1 1





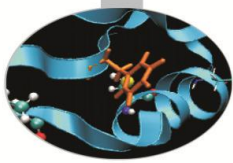
Jaccard: $d(i,k) = (a_{11}) / (a_{11} + a_{10} + a_{01})$

Condorcet: $d(i,k) = a_{11} / [a_{11} + 0.5(a_{10} + a_{01})]$

Dice bis: $d(i,k) = a_{11} / [a_{11} + 0.25(a_{10} + a_{01})]$

Go to Supervised learning

Intra-Cluster Distance



- Minimize intra-cluster, equivalent to maximize inter-cluster distance
- Intra-cluster distance

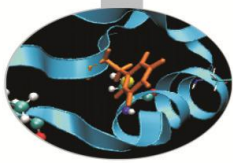
$$W(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\sum_{C(j)=k} d(x_i, x_j)$$

- Total distance, which is fixed

$$T = \frac{1}{2}\sum_{i}\sum_{j} d(x_i, x_j)$$

- Inter-cluster distance: $B(C) = T - W(C)$

k-Means



- Distance measure: Squared Euclidean Distance

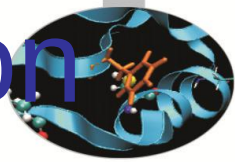
$$d(x,y) = \|x - y\|^2$$

- Minimize the sum of squared error distance

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

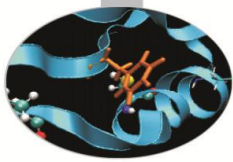
where $$\bar{x}_k = \frac{1}{n_k} \sum_{C(i)=k} x_i$$

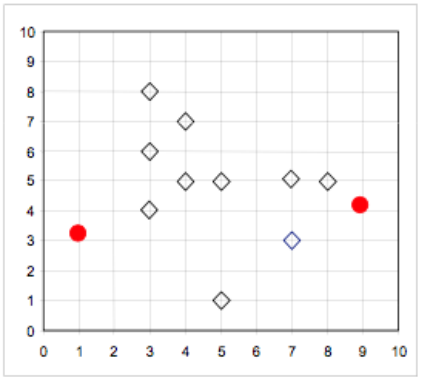
k-Means Iterative Optimization



- Initialize: Partition the data into k initial clusters
- Step 1: Compute the mean of each cluster
- Step 2: Assign each data point to the closest partition
- Step 3: If any data point changed its cluster membership, then repeat from Step 1

Example: k-Means

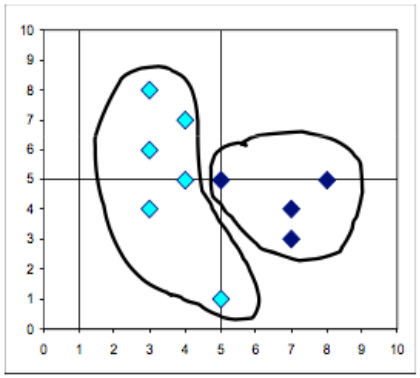




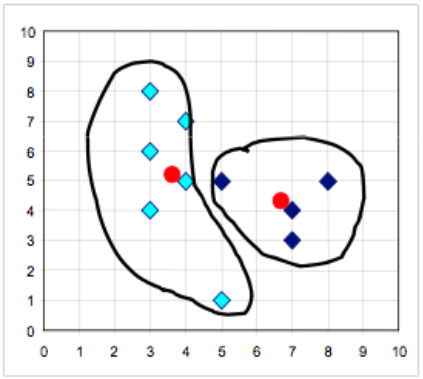
K=2

Arbitrarily choose K object as initial cluster center

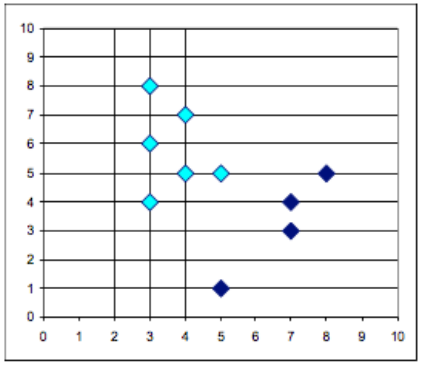
Assign each objects to most similar center



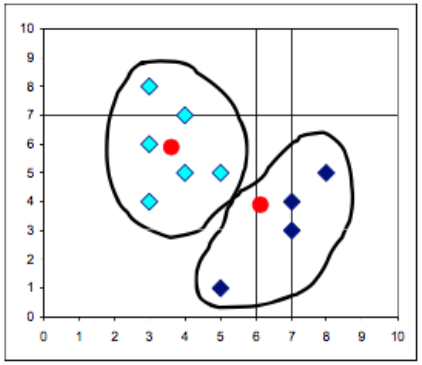
Update the cluster means



reassign

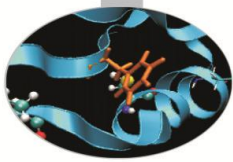


Update the cluster means



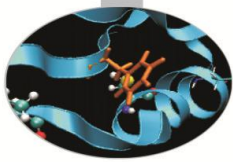
reassign

Partitioning Approach

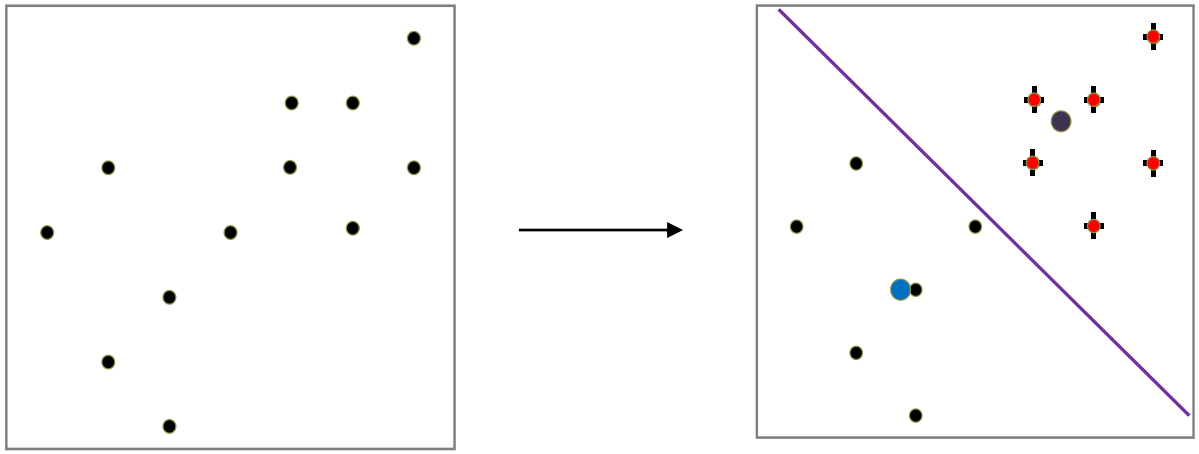


- Partitioning method: Discovering the groupings in the data by optimizing a specific objective function and iteratively improving the quality of partitions
- K-partitioning method: Partitioning a dataset D of n objects into a set of K clusters so that an objective function is optimized (e.g., the sum of squared distances is minimized, where ck is the centroid or medoid of cluster Ck)
  - A typical objective function: Sum of Squared Errors (SSE)
- Problem definition: Given K, find a partition of K clusters that optimizes the chosen partitioning criterion
  - Global optimal: Needs to exhaustively enumerate all partitions
  - Heuristic methods (i.e., greedy algorithms): K-Means, K-Medians, K-Medoids, etc.

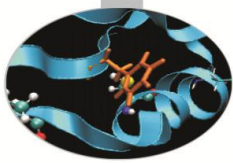
Partitioning Approach



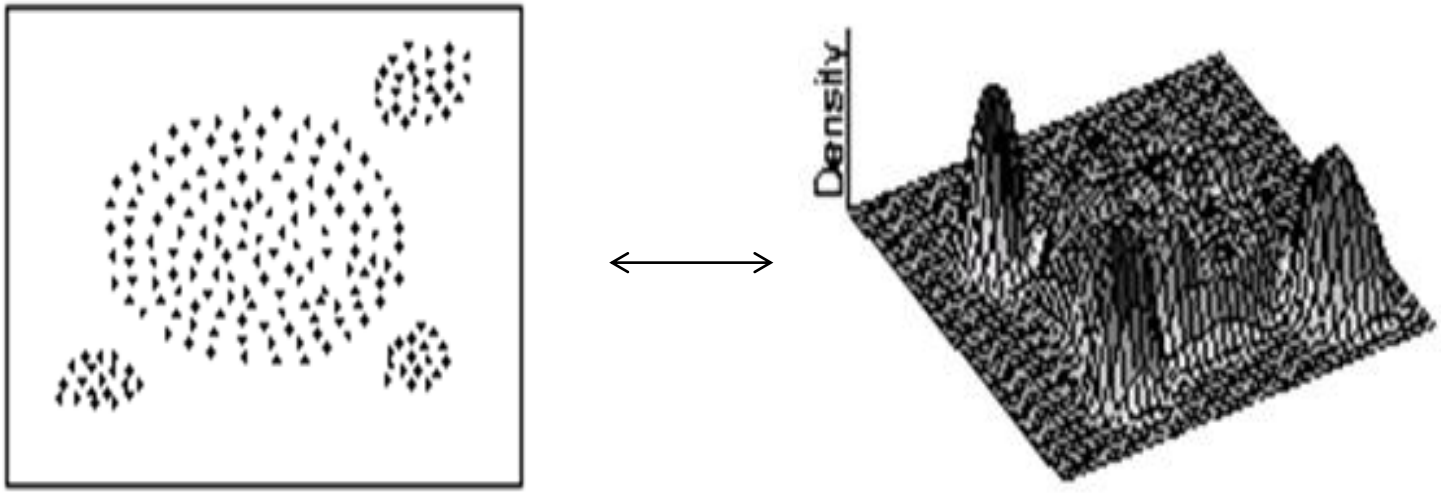
- Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square distance cost
- Typical methods: K-Means, K-Medoids, K-Medians, ……



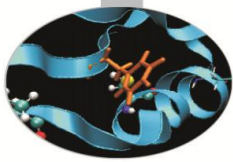
Density based Approach



- Based on connectivity and density functions
- Typical methods: DBSCAN, OPTICS, DenClue, ……

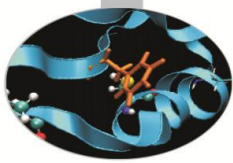


Density based approach

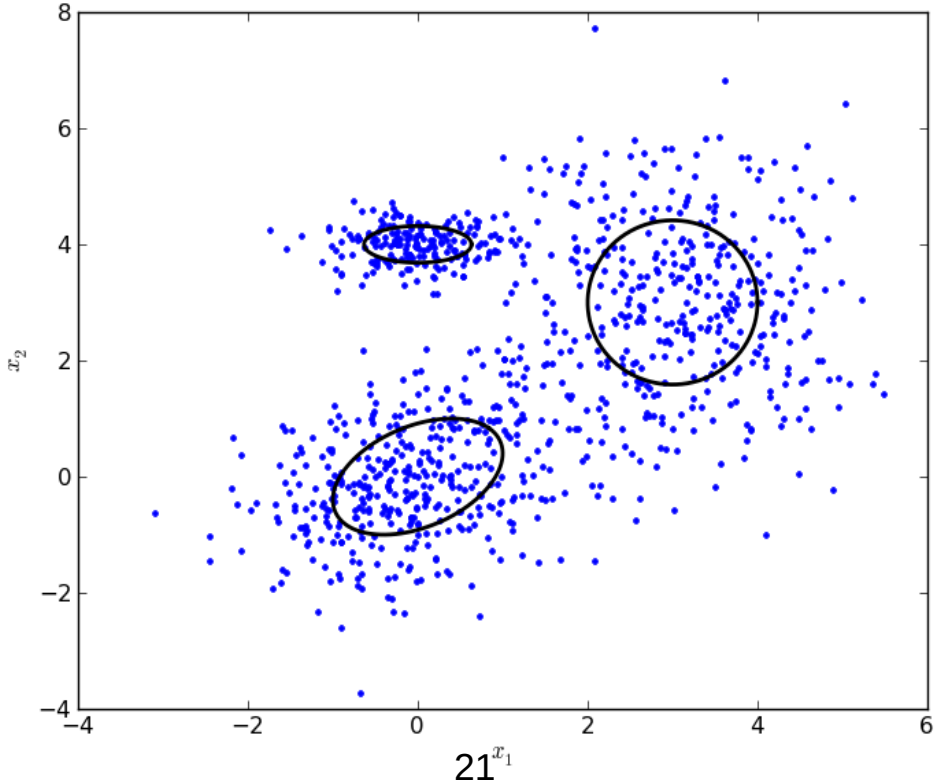


- Density-based methods assume that the points that belong to each cluster are drawn from a specific probability distribution.
- The overall distribution of the data is assumed to be a mixture of several distributions. The aim of these methods is to identify the clusters and their distribution parameters.
- These methods are designed for discovering clusters of arbitrary shape which are not necessarily convex, namely: xi , xj ∈ Ck This does not necessarily imply that: α · xi + (1 − α) · xj ∈ Ck The idea is to continue growing the given cluster as long as the density (number of objects or data points) in the neighborhood exceeds some threshold. Namely, the neighborhood of a given radius has to contain at least a minimum number of objects. When each cluster is characterized by local mode or maxima of the density function, these methods are called mode-seeking

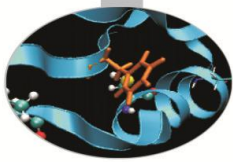
Model-based Approach



- For each cluster, a theoretical model is hypothesized in order to find the best fit.
- Typical methods: Gaussian Mixture Model (GMM), COBWEB, ……

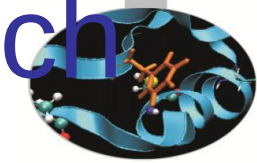


Model-based Approach

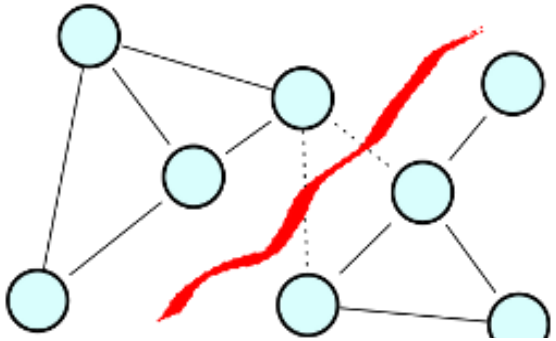


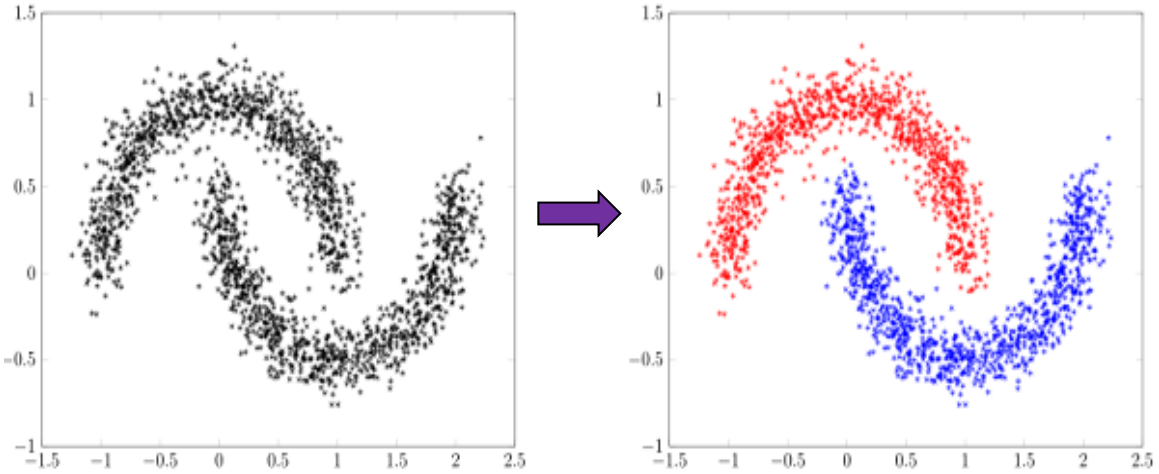
- Probabilistic model-based clustering
  - In model-based clustering, it is assumed that the data are generated by a mixture of underlying probability distributions in which each component represents a different group or cluster.
  - Cluster: Data points (or objects) that most likely belong to the same distribution
  - Clusters are created so that they will have a maximum likelihood fit to the model by a mixture of K component distributions (i.e., K clusters)

Spectral Clustering Approach

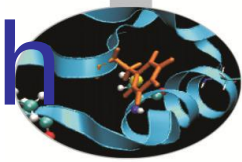


- Convert data set into weighted graph (vertex, edge), then cut the graph into sub-graphs corresponding to clusters via spectral analysis
- Typical methods: Normalised-Cuts ……





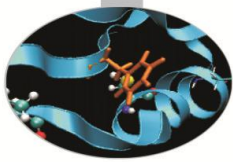
Spectral Clustering Approach



- In multivariate statistics, spectral clustering techniques make use of eigenvalue decomposition (spectrum) of the similarity matrix of the data to perform dimensionality reduction before clustering in fewer dimensions. The similarity matrix is provided as an input and consists of a quantitative assessment of the relative similarity of each pair of points in the dataset.

- In application to image segmentation, spectral clustering is known as segmentation-based object categorization.

Combination of methods

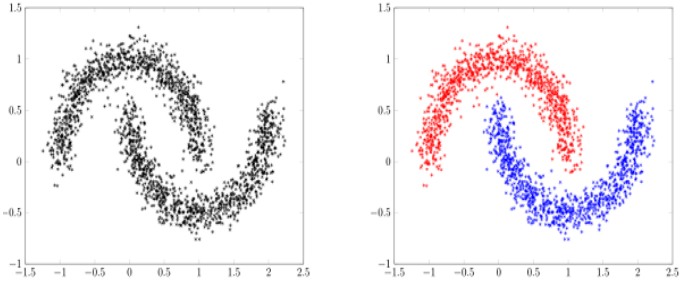


Using different methods can be useful for overcome the drawbacks of a single methods.

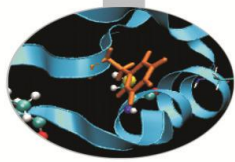
For example it is possible to generate a large number of clusers with K-means and then cluster them together using a hierarchical method.

It is important using the “single-link” method, in which the distance between two clusters is defined by the distance between the two closest data points we can find, one from each cluster.

This method has been applied to find cluster in non-convex set.



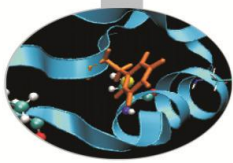
Clustering validation



Clustering validation is a technique to find a set of clusters that best fits natural partitions (number of clusters K). Generally speaking, there are two types of validation techniques, which are based on internal criteria and external criteria.

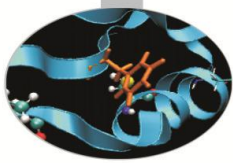
- Internal validation: based on the information intrinsic to the data alone
- External validation: based on previous knowledge about data

Supervised learning: classification



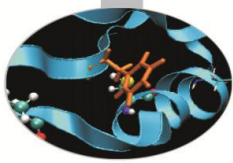
- Human learning from past experiences.
- A computer does not have “experiences”.
- A computer system learns from data, which represent some “past experiences” of an application domain.
- Learn a target function that can be used to predict the values of a discrete class attribute,
- The task is commonly called: Supervised learning, classification, or inductive learning.

Two-Step Process (1)

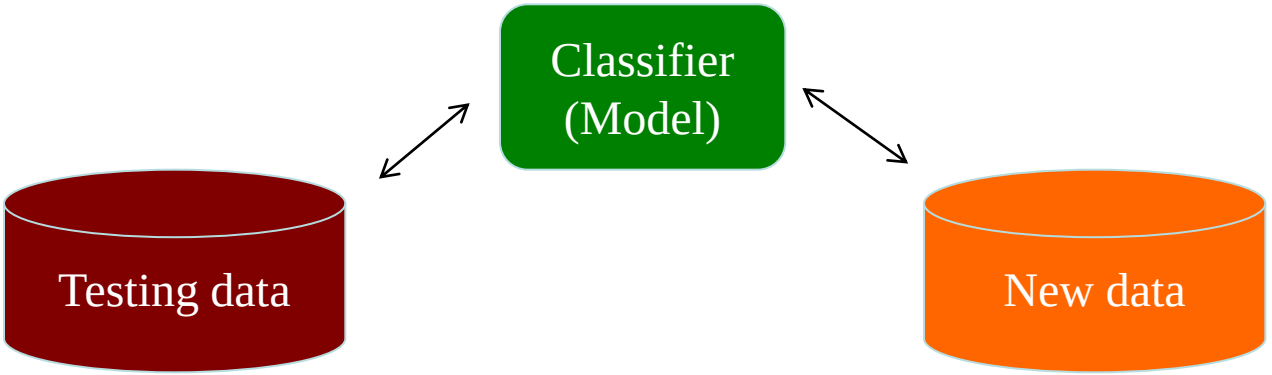


- Model construction
  - The set of samples used in this step is training data
  - Each sample belongs to a predefined class, suggested by its class label
  - The model is represented as classification rules, decision trees, or other functions

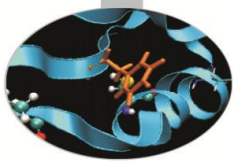
Two-Step Process (2)



- Model usage: classifying new objects
  - Estimate model accuracy
    - The set of samples for testing accuracy is testing data
    - Accuracy rate is the percentage of correctly classified samples
    - Testing data is independent of training data
  - If the accuracy is acceptable, apply it to new data

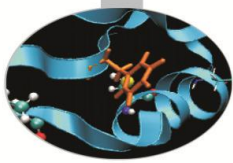


Typical applications



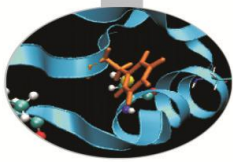
- Scientific applications
  - **Medical Diagnosis**: Given the symptoms exhibited in a patient and a database of anonymized patient records, predict whether the patient is likely to have an illness.
  - …
- Business applications
  - **Credit Card Fraud Detection**: Given credit card transactions for a customer in a month, identify those transactions that were made by the customer and those that were not.
  - **Stock Trading**: Given the current and past price movements for a stock, determine whether the stock should be bought, held or sold. A model of this decision problem could provide decision support to financial analysts.
  - …
- Internet applications
  - **Spam Detection**: Given email in an inbox, identify those email messages that are spam and those that are not.
  - …

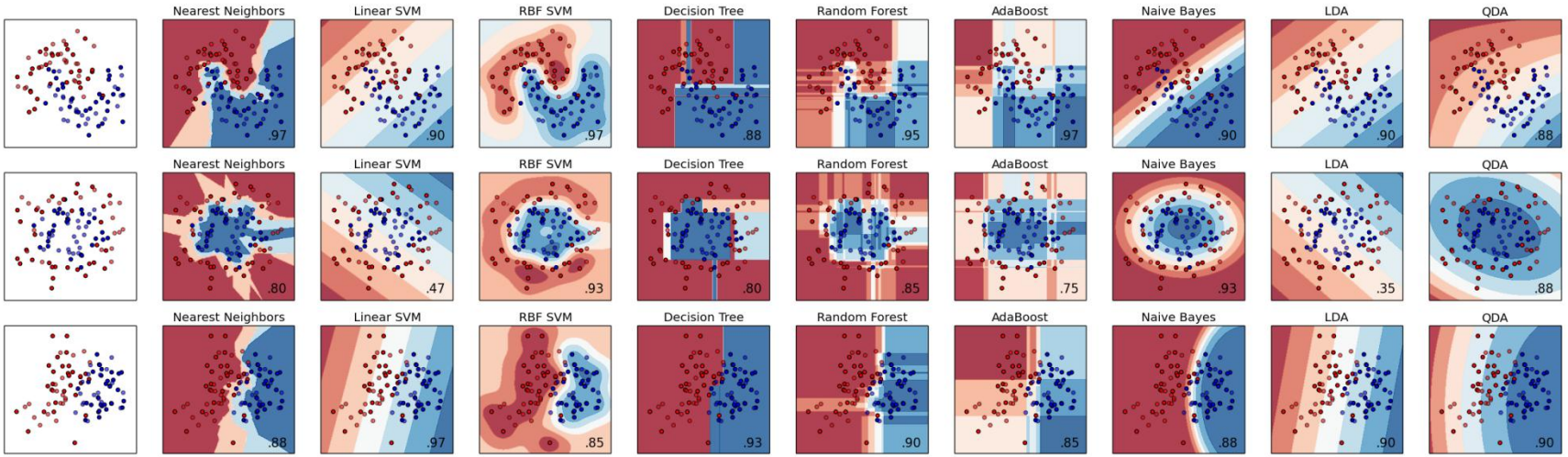
Classification Techniques



- Decision Tree based Methods
- Ensemble methods
- Naïve Bayes and Bayesian Belief Networks
- Rule-based Methods
- Memory based reasoning
- Neural Networks
- Support Vector Machines

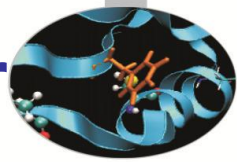
Comparison of algorithms





CINECA

Training and test a classifier

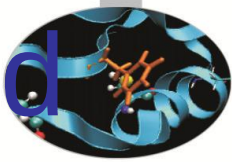


Is the model able to generalize? Can it deal with unseen data, or does it overfit the data? Test on hold-out data:

- **split** data to be modeled in training and test set
- **train** the model on training set
- evaluate the model on the training set
- **evaluate** the model on the test set
- difference between the fit on training data and test

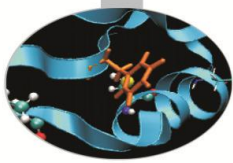
data measures the model's ability to *generalize*

Methods to create training and test data



- Fixed
  - Leave out random N% of the data
- K-fold Cross-Validation
  - Select K folds without replace
- Leave-One-Out Cross Validation
  - Special case of CV
- Bootstrap
  - Generate new training sets by sampling with replacement

Evaluation metrics



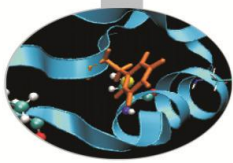
Confusion matrix

The known class of test samples is matched against the class predicted by the model		Predicted labels (model)		
		False	**True**	
Real labels (target)	**False**	TN	FP	Specificity TN / (FP+TN)
	True	FN	TP	Sensitivity TP / (TP+FN)
		Negative Predictive Value TN / (TN + FN)	Positive Predictive Value TP / (TP + FP)	Accuracy (TP+TN) / (TP+FP+TN+FN)

⇨ Recall

⇩ Precision

F-score = 2*Precision*Recall / (Precision + Recall)
Error rate = 1 – Precision
FP rate = 1 – Specificity

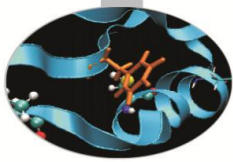


Evaluation metrics

Accuracy baselines

- Base Rate
  - Accuracy of trivially predicting the most-frequent class
- Random Rate
  - Accuracy of making a random class assignment
- Naive Rate
  - Accuracy of some simple default or pre-existing model

Building a Decision Tree

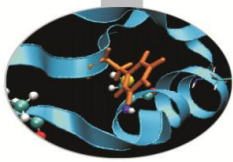


- Choose the attribute with the highest Information Gain
- Create branches for each value of attribute
- Partition examples on the basis of selected attributes
- Repeat with remaining attributes
- Stopping conditions
  - All examples assigned the same label
  - No examples left

Problems

- Expensive to train
- Prone to overfitting
  - perform well on training data, bad on test data
  - pruning can help: remove or aggregate subtrees that provide little discriminatory power

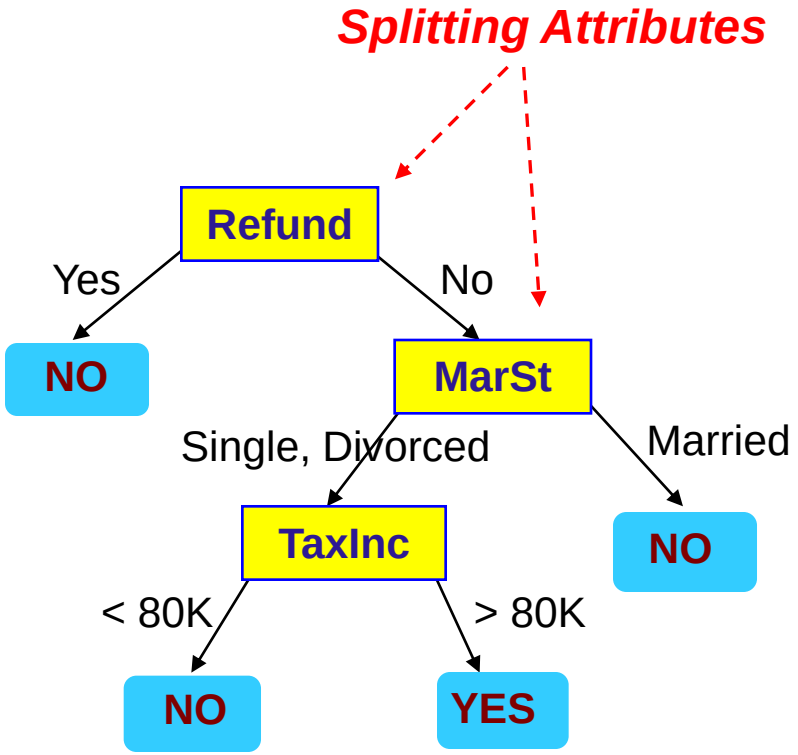
Example of a Decision Tree



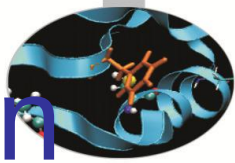
categorical | categorical | continuous | class

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data



Model: Decision Tree



Naïve Bayesian Classfication

Bayes theorem:

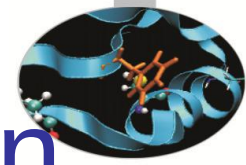
$$P(C|X) = P(X|C) \cdot P(C) / P(X)$$

P(X) is constant for all classes

P(C) = relative freq of class C samples

C such that **P(C|X)** is maximum =
C such that **P(X|C)·P(C)** is maximum

Problem: computing P(X|C) is unfeasible!



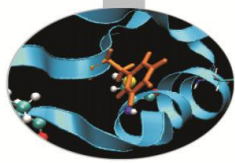
Naïve Bayesian Classification

- Here's where the "Naive" comes in. We're going to assume that the different features of the data are *independent* of each other, conditional on *C*=*c*.
- $P(x_1,\ldots,x_k|C) = P(x_1|C)\cdot\ldots\cdot P(x_k|C)$
- By making the decision to completely ignore the correlations between features, this method is blissfully unaware of the primary difficulty of high-dimensional (high-*p*) datasets, and training Naive Bayes classifiers becomes extremely easy.

- If i-th attribute is categorical:
  $P(x_i|C)$ is estimated as the relative freq of samples having value x_i as i-th attribute in class C
- If i-th attribute is continuous:

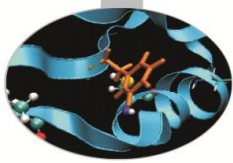
  $P(x_i|C)$ is estimated thruogh a Gaussian density function

Spark MLlib



- MLlib is a Spark subproject providing machine learning primitives:
- MLlib's goal is to make practical machine learning (ML) scalable and easy. Besides new algorithms and performance improvements that we have seen in each release, a great deal of time and effort has been spent on making MLlib *easy*.

Spark MLlib



- MLlib algorithms
  - classification: logistic regression, naive Bayes, decision tree, ensemble of trees (random forests)
  - regression: generalized linear regression (GLM)
  - collaborative filtering: alternating least squares (ALS)
  - clustering: k-means, gaussian mixture, power iteration clustering, latent Dirichelet allocation
  - decomposition: singular value decomposition (SVD), principal component analysis, singular value decompostion
- Spark packages availables for machine learning at http://spark-packages.org