Introduction to Machine Learning

With Scikit-Learn and OpenCV

Machine Learning + Big Data in Real Time + Cloud Technologies

=> The Future of Intelligent Systems

Where to Find The Code and Materials?

https://github.com/iproduct/course-ml

Agenda for This Lesson - I

- Machine learning
- Supervised, Semi-supervised and unsupervised ML
- Classification, clustering and reinforcement learning
- Mean, median, standard deviation, percentiles, histograms.
- Normal distribution
- Data visualization using Matplotlib
- Training-testing-evaluation
- Receiver operating characteristic (ROC)

Agenda for This Lesson - II

Practical ML using Scikit-learn and Python:

- Prediction: linear regression, polynomial regression, multiple regression
- Classification: logistic regression, K Nearest Neighbors (KNN)
- Gradient descent
- Clustering : K-Means Clustering
- Curse of dimensionality: Principal Component Analysis & Feature Selection
- Linear discriminant analysis – LDA
- Support Vector Machines (SVM). Kernels
- Decision Trees. Random Forests

Introduction to Machine Learning

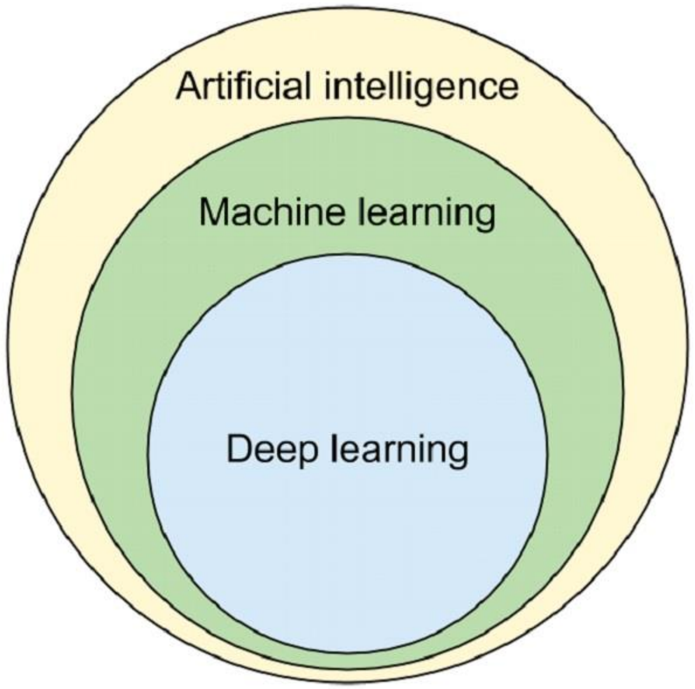
What Is Machine Learning?

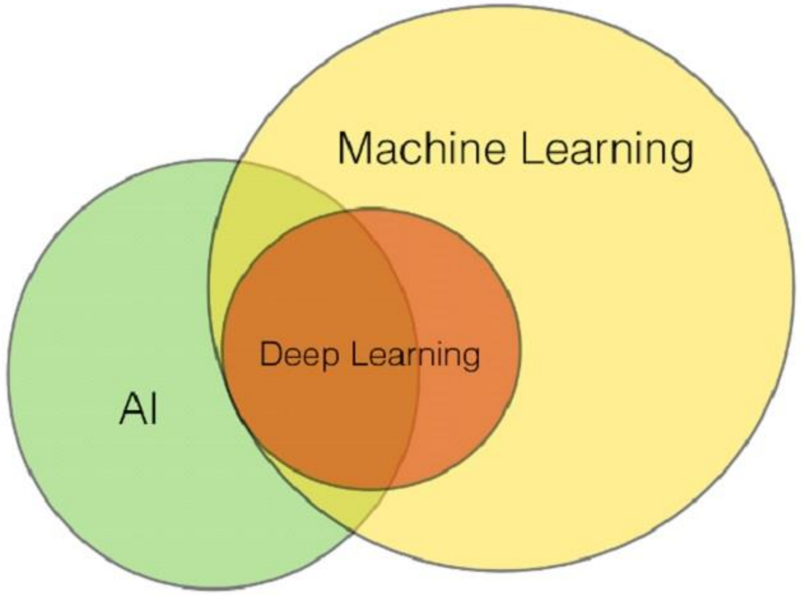
- **Machine Learning (ML)** –the study of computer algorithms that improve automatically through experience. A subset of Artificial Intelligence (AI).
- ML algorithms build a model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so.
- ML algorithms are used in a wide variety of applications, where it is difficult or unfeasible to develop conventional algorithms for the task – e.g. computer vision, email filtering, digital assistants, natural language processing (NLP), recommendations, personalized online advertising, chatbots, fraud detection, cybersecurity, medical image analysis, self-driving cars, self-driving databases, and much, much more ...
- ML is closely connected with computational statistics, mathematical optimization, data mining, predictive analytics.

Machine Learning and Artificial Intelligence

- ML learns and predicts based on passive observations, whereas AI implies an agent interacting with the environment to learn and take actions that maximize its chance of successfully achieving its goals.

Judea Pearl, The Book of Why, 2018





Pictures by: Yakoove - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=95589683
By Yakoove - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=95589710

Types of Learning Algorithms: Supervised Learning

- **Supervised learning** – build a mathematical model of a set of data that contains both the inputs and the desired outputs.
- Training data – a set (matrix) of training examples (feature vectors), having one or more inputs and the desired output. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs.
- Examples: active learning, classification and regression.
  - Active learning – learning algorithm can interactively query a user (teacher or oracle) to label new data points with the desired outputs (optimal experimental design).
  - Classification algorithms – when the outputs are restricted to a limited set of values
  - Regression algorithms – when the outputs may have any numerical value within a range.
  - Similarity learning – using a similarity function measuring how similar two objects are – ranking, recommender systems, visual identity tracking, face and speaker verification.

Types of Learning Algorithms: Unsupervised Learning

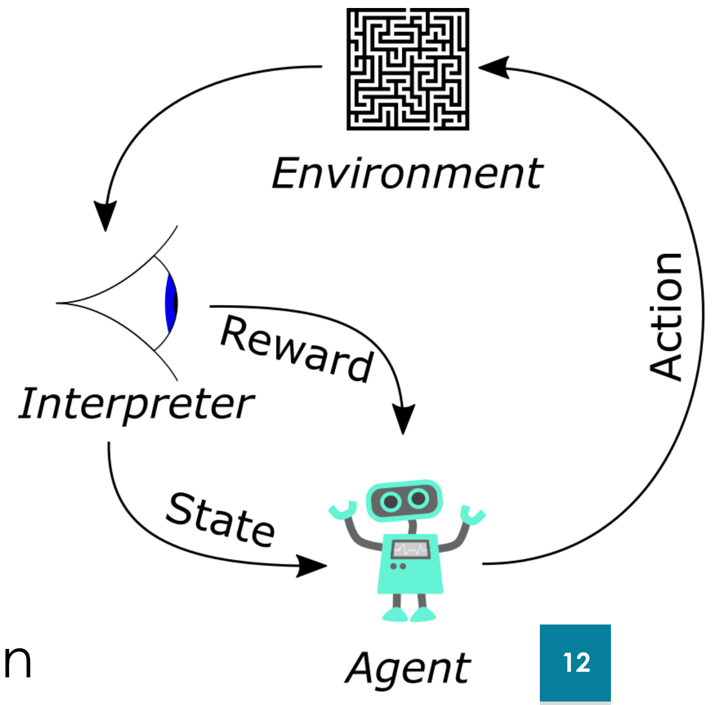
- **Unsupervised learning** – takes a data set that contains only inputs, and find structure in the data, like grouping or clustering of data points by identifying data commonalities
- Algorithms learn from test data that has not been labeled, classified or categorized.
- Applications: density function estimation, summarizing and explaining data features.
- Cluster analysis is the assignment of a set of observations into subsets (called clusters) so that observations within the same cluster are similar according to one or more predesignated criteria, while observations drawn from different clusters are dissimilar.
- Different clustering techniques make different assumptions on the structure of the data, often defined by some similarity metric and evaluated, for example, by internal compactness, or the similarity between members of the same cluster, and separation, the difference between clusters. Other methods are based on estimated density and graph connectivity.

Types of Learning Algorithms: Semi-supervised Learning

- **Semi-supervised learning** – falls between unsupervised learning (without any labeled training data) and supervised learning (with completely labeled training data). During training, it uses a smaller labeled data set to guide classification and feature extraction from a larger, unlabeled data set.
- In **weakly supervised learning**, the training labels are noisy, limited, or imprecise; however, these labels are often cheaper to obtain, resulting in larger effective training sets.

Types of Learning Algorithms: Reinforcement Learning

- **Reinforcement learning** – concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward
- Due to its generality, the field is studied in many other disciplines, such as game theory, control theory, operations research, information theory, simulation-based optimization, multi-agent systems, swarm intelligence, statistics and genetic algorithms.
- In ML, the environment is typically represented as a Markov decision process (MDP). Many reinforcement learning algorithms use dynamic programming techniques.
- Reinforcement learning algorithms do not assume knowledge of an exact mathematical model of the MDP, and are used when exact models are infeasible.
- Examples: reinforcement learning algorithms are used in autonomous vehicles or in learning to play a game against human



Types of Learning Algorithms: Self Learning

- **Self learning** – as a machine learning paradigm was introduced in 1982 along with a neural network capable of self-learning named crossbar adaptive array (CAA). It is a learning with no external rewards and no external teacher advice.
- The CAA self-learning algorithm computes, in a crossbar fashion, both decisions about actions and emotions (feelings) about consequence situations. The system is driven by the interaction between cognition and emotion.
- In situation s perform an action a; Receive consequence situation s'; Compute emotion of being in consequence situation $v(s')$; Update crossbar memory $w'(a,s) = w(a,s) + v(s')$.
- The backpropagated value (secondary reinforcement) is the emotion toward the consequence situation. The CAA exists in two environments, one is the behavioral environment where it behaves, and the other is the genetic environment, wherefrom it initially and only once receives initial emotions about situations to be encountered in the behavioral environment.

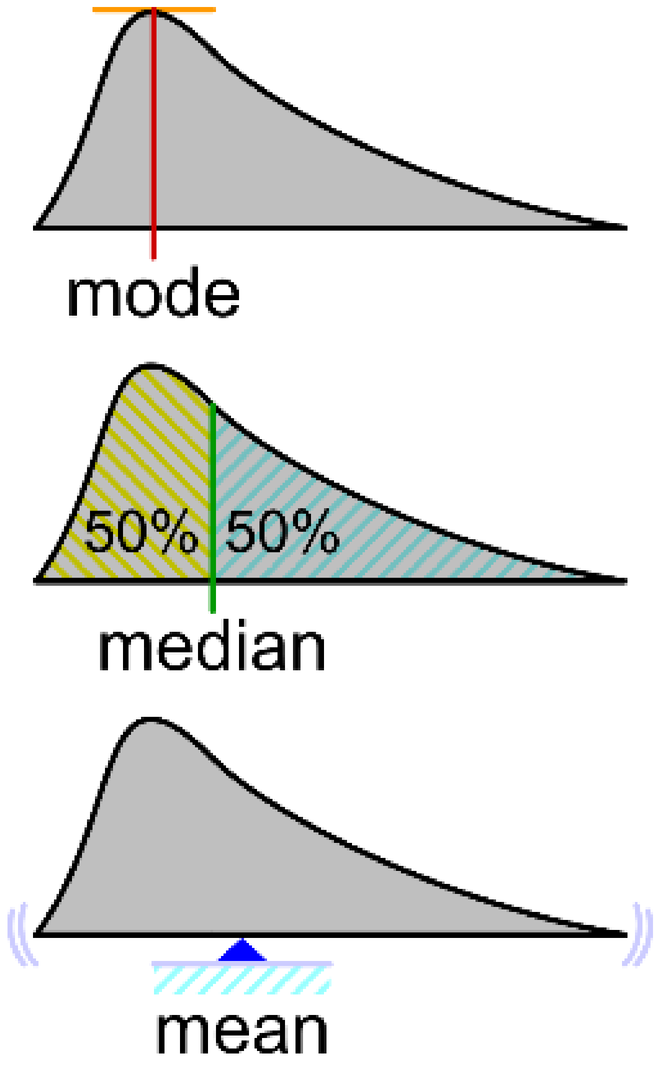
Types of Learning Algorithms: Feature learning – I

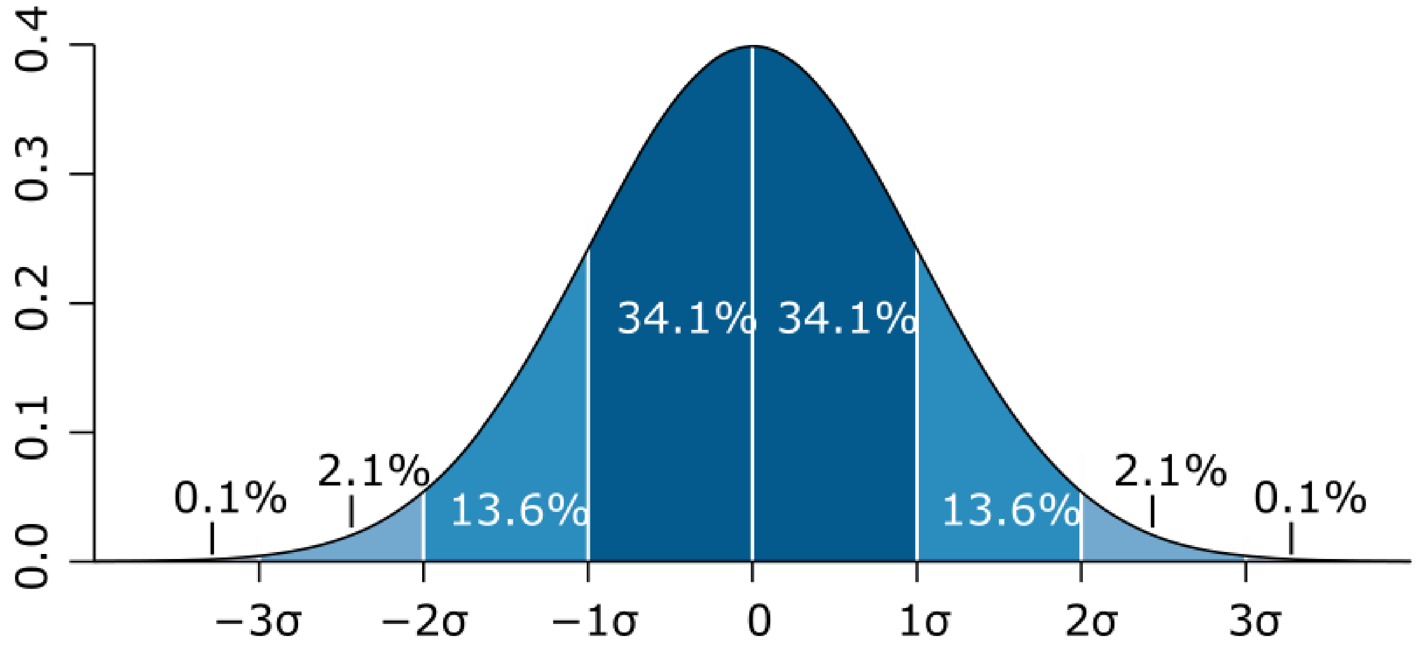
- **Feature learning (representation learning algorithms)** – discover better representations of the inputs provided during training. Examples: principal components analysis and cluster analysis.
- Attempt to preserve the information in their input but also transform it in a way that makes it useful, often as a pre-processing step before performing classification or predictions.
- Allows a machine to both learn the features and use them to perform a specific task.
- Feature learning can be either supervised or unsupervised. In supervised feature learning, features are learned using labeled input data. Examples: artificial neural networks, multilayer perceptrons, and supervised dictionary learning. In unsupervised feature learning, features are learned with unlabeled input data. Examples: dictionary learning, independent component analysis, autoencoders, matrix factorization, and clustering.

Types of Learning Algorithms: Feature learning – II

- Manifold learning algorithms attempt to do so under the constraint that the learned representation is low-dimensional.
- Sparse coding algorithms attempt to do so under the constraint that the learned representation is sparse, meaning that the mathematical model has many zeros.
- Multilinear subspace learning algorithms aim to learn low-dimensional representations directly from tensor representations for multidimensional data, without reshaping them into higher-dimensional vectors.
- Deep learning algorithms discover multiple levels of representation, or a hierarchy of features, with higher-level, more abstract features defined in terms of (or generating) lower-level features. It has been argued that an intelligent machine is one that learns a representation that disentangles the underlying factors of variation that explain the observed data.

Mean, Median, Mode, Standard Deviation

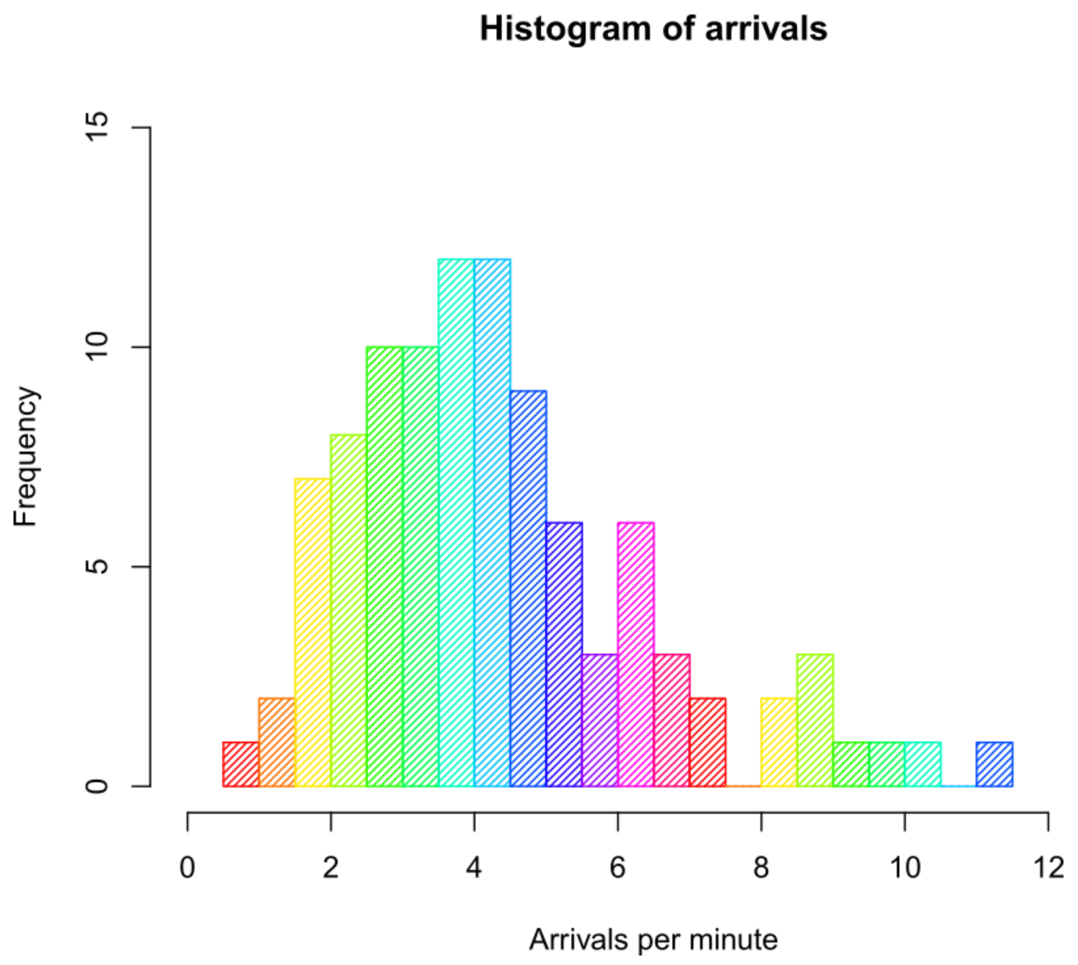




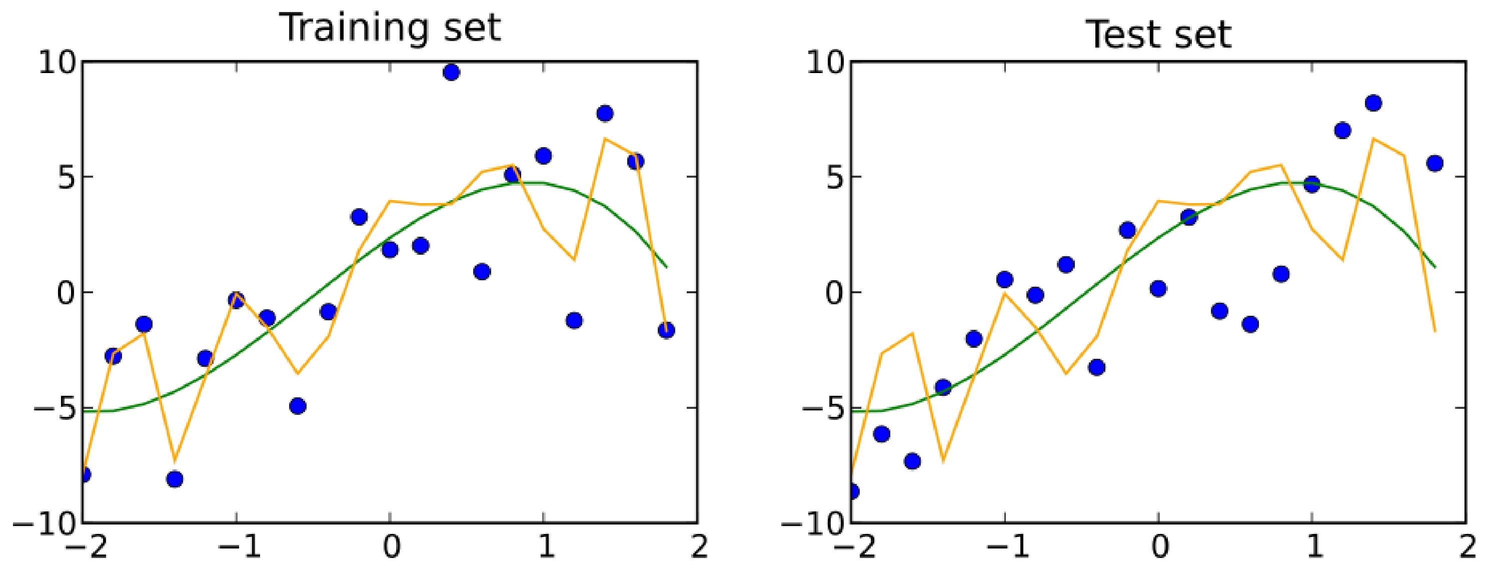
$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}$$

Percentiles. Histograms

- Percentile (or a centile) is a type of quantile which divides the given probability distribution, or sample, into 100 equal-sized intervals; this allows the data to be analyzed in terms of percentages. For example, the 20th percentile is the value (or score) below which 20% of the observations are found, and above which 80% are found.
- Histogram – approximate representation of the distribution of numerical data. To construct a histogram, the first step is to "bin" (or "bucket") the range of values—that is, divide the entire range of values into a series of intervals—and then count how many values fall into each interval.



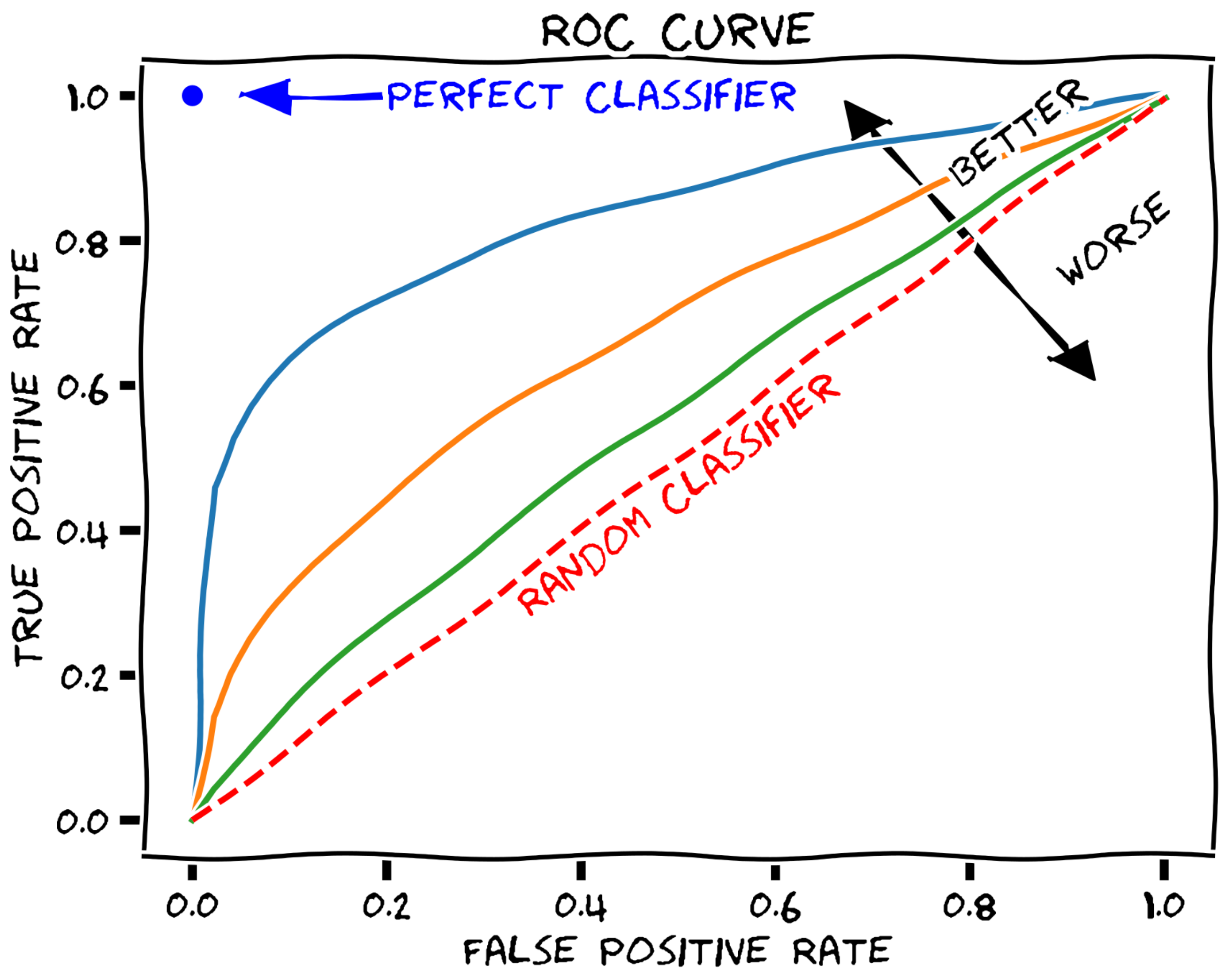
Training and Testing Data Sets



$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

A training set (left) and a test set (right) from the same statistical population are shown as blue points. Two predictive models are fit to the training data. Both fitted models are plotted with both the training and test sets. In the training set, the MSE of the fit shown in orange is 4 whereas the MSE for the fit shown in green is 9. In the test set, the MSE for the fit shown in orange is 15 and the MSE for the fit shown in green is 13. The orange curve severely overfits the training data, since its MSE increases by almost a factor of four when comparing the test set to the training set. The green curve overfits the training data much less, as its MSE increases by less than a factor of 2.

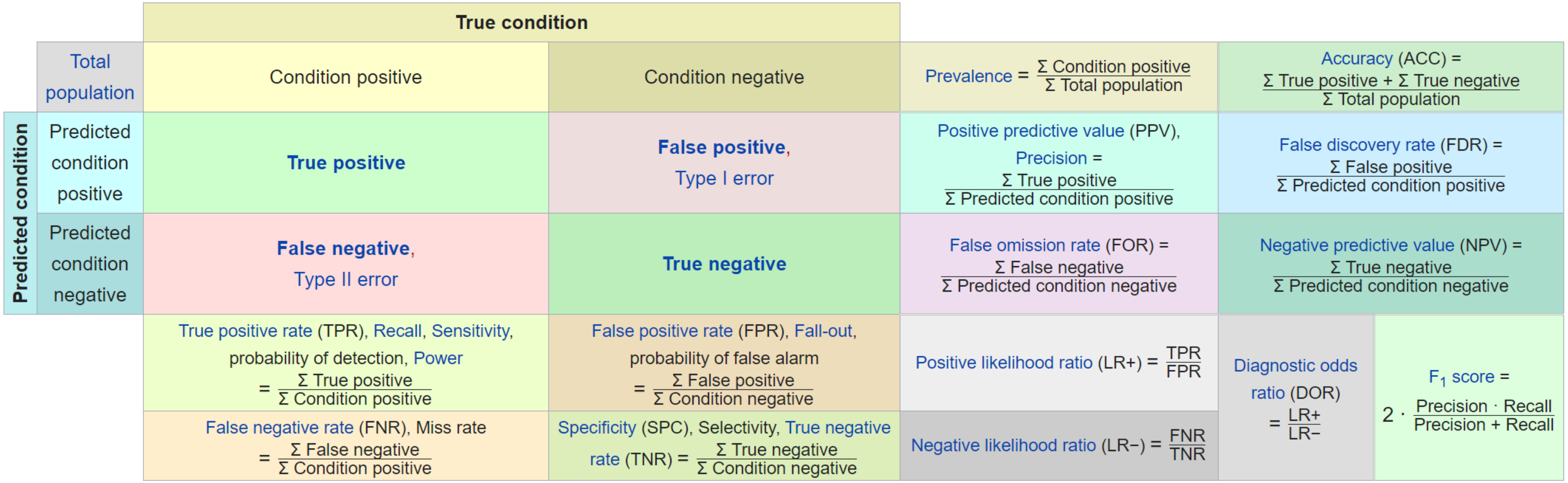
Receiver Operating Characteristic (ROC)



Receiver operating characteristic (ROC) curve - graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.

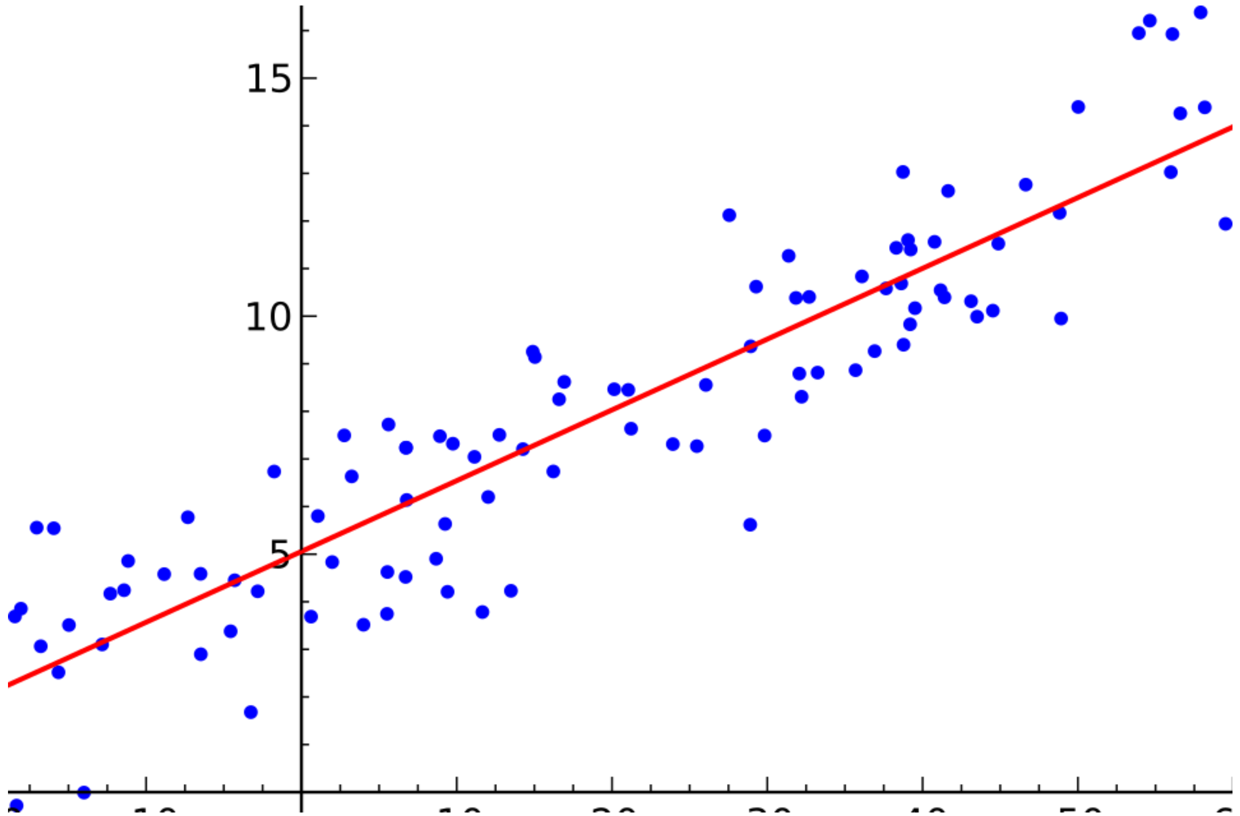
Receiver Operating Characteristic (ROC)

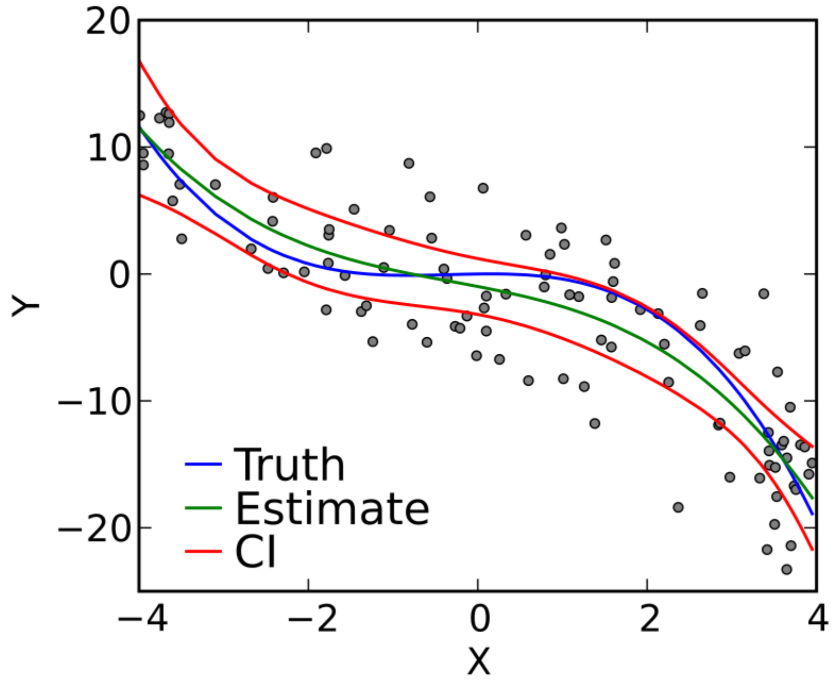
- condition positive (P) / negative (N) - the number of real positive / negative cases in the data
- true positive (TP) - eqv. with hit, true negative (TN) - eqv. with correct rejection
- false positive (FP) - eqv. with false alarm, Type I error, false negative (FN) - eqv. with miss, Type II error



		True condition				
	Total population	Condition positive	Condition negative	Prevalence = $\frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error	Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$	F_1 score = $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
		False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) = $\frac{\text{FNR}}{\text{TNR}}$		

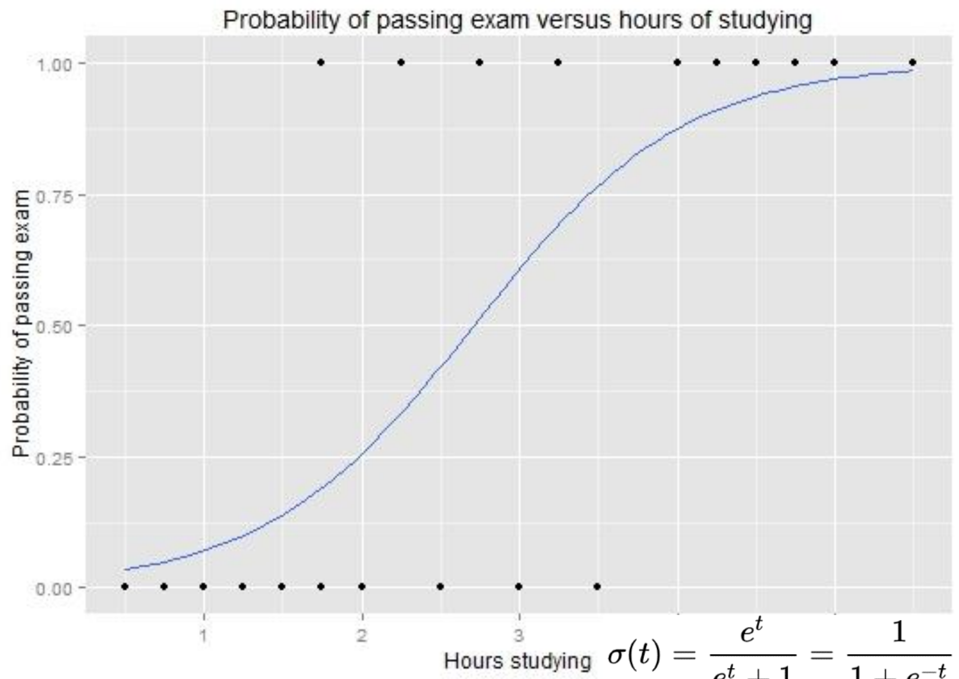
Prediction: Linear Regression, Polynomial Rregression



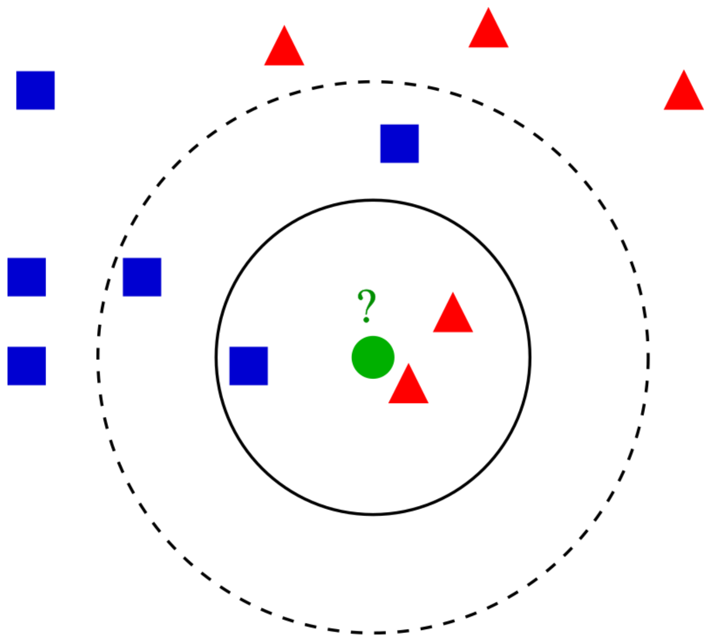


$$Y_i = f(X_i, \beta) + e_i$$

Classification: logistic regression, K Nearest Neighbors



$$\sigma(t) = \frac{e^t}{e^t + 1} = \frac{1}{1 + e^{-t}}$$



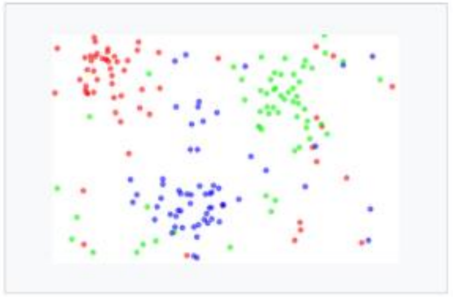


Fig. 1. The dataset.

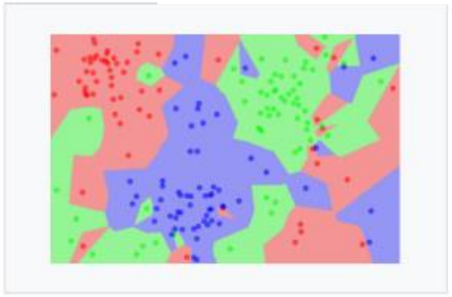


Fig. 2. The 1NN classification map.

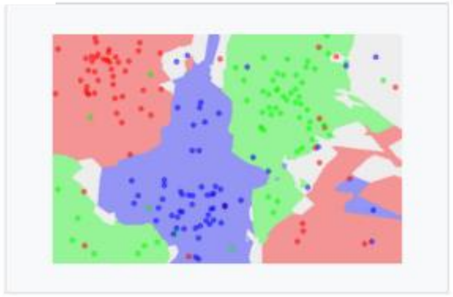


Fig. 3. The 5NN classification map.

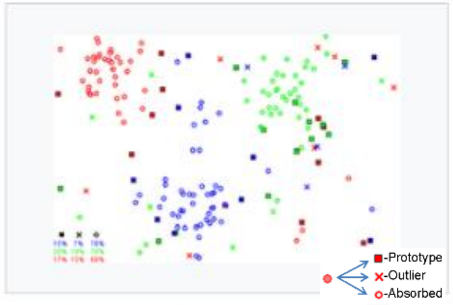


Fig. 4. The CNN reduced dataset.

CNN = Condensed nearest neighbors

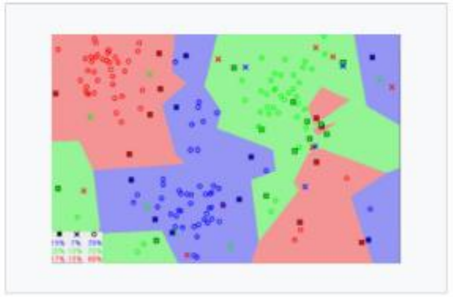
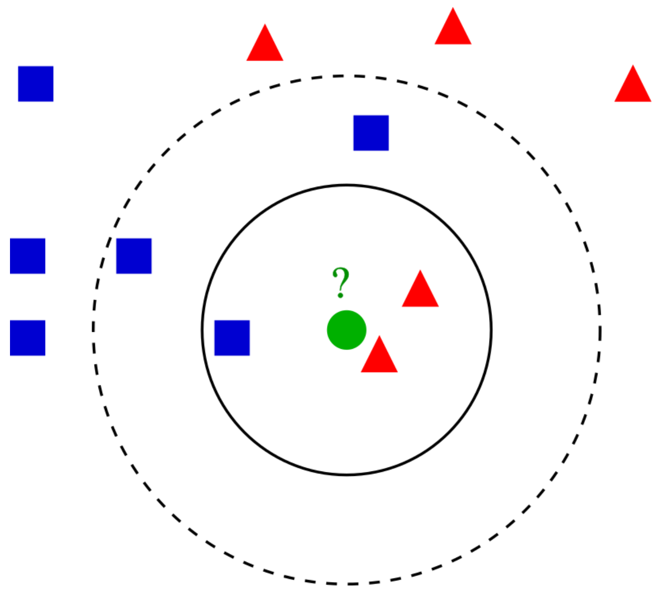


Fig. 5. The 1NN classification map based on the CNN extracted prototypes.

K Nearest Neighbors Classification

- k-nearest neighbors (k-NN) – a non-parametric algorithm proposed by Thomas Cover for classification and regression.
- Input consists of the k closest training examples in the feature space.
- In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.



Distance measures (continuous):

Euclidean $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

Manhattan $\sum_{i=1}^{k}|x_i - y_i|$

Minkowski $\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$

Hamming distance

$$D_H = \sum_{i=1}^{k}|x_i - y_i|$$

$$x = y \Rightarrow D = 0$$

$$x \neq y \Rightarrow D = 1$$

K Nearest Neighbors Algorithm – Simple Form

1. Load the data
2. Choose the number of neighbors to be considered – K
3. For each example in the data
    i. Calculate the distance between the query example and the current example from the data.
    ii. Add the distance and the index of the example to an ordered list
4. Sort the ordered list of distances and indices in ascending order (from smallest to largest distances)
5. Pick the first K entries from the sorted list and get their labels
6. If regression, return the mean of the K labels
7. If classification, return the mode of the K labels

K Nearest Neighbors Algorithm Details

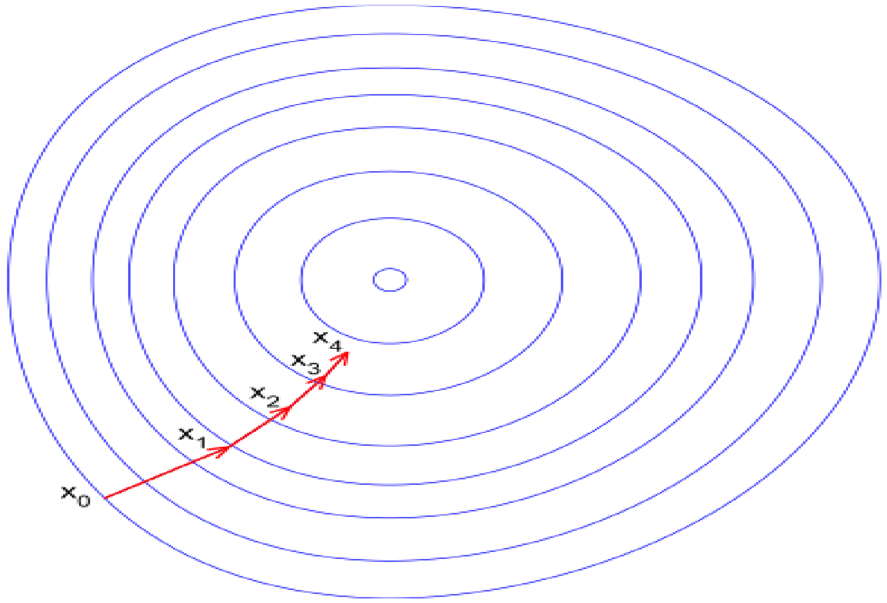
- A drawback of the basic "majority voting" classification occurs when the class distribution is skewed – examples of a more frequent class tend to dominate the prediction of the new example, because they tend to be common among the **k** nearest neighbors due to their large number => weighting the classification, taking into account the distance from the test point to each of its k nearest neighbors – e.g. by a weight proportional to the inverse of the distance from that point to the test point.
- Choosing **k** – larger values of **k** reduces effect of the noise on the classification, but make boundaries between classes less distinct.
- In binary classification problems, it is helpful to choose k to be an odd number as this avoids tied votes. One popular way of choosing the empirically optimal k in this setting is via bootstrap method.
- Often, the classification accuracy of **k-NN** can be improved significantly if the distance metric is learned with specialized algorithms such as Large Margin Nearest Neighbor or Neighbourhood components analysis

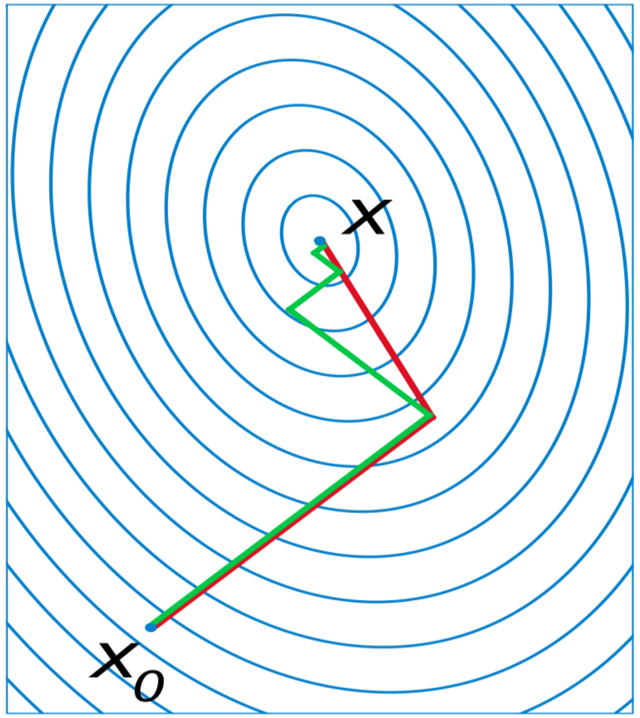
K Nearest Neighbors Algorithm – Feature Extraction

- Feature extraction – when input data is too large to be processed/redundant, then the input data will be transformed into a reduced representation set of features (features vector). For example (Open CV):
  1. Haar face detection
  2. Mean-shift tracking analysis
  3. PCA or Fisher LDA projection into feature space, followed by k-NN classification.
- For high-dimensional data (e.g., >= 10) dimension reduction is usually performed prior to applying the k-NN algorithm in order to avoid the effects of the curse of dimensionality.
- Feature extraction and dimension reduction can be combined in one step using principal component analysis (PCA), linear discriminant analysis (LDA), or canonical correlation analysis (CCA) techniques as a pre-processing step, followed by clustering by k-NN on feature vectors in reduced-dimension space (low-dimensional embedding)
- For very-high-dimensional datasets (e.g. similarity search on live video streams, DNA data or high-dimensional time series) running a fast approximate k-NN search using locality sensitive hashing, "random projections", "sketches" or other high-dimensional similarity search techniques from the VLDB toolbox might be the only feasible option.

Gradient Descent

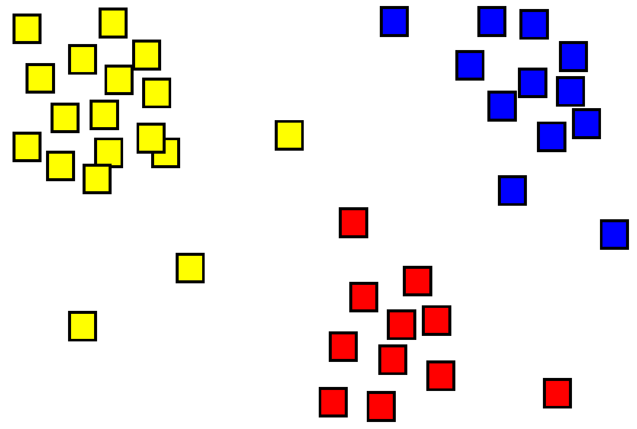
- Gradient descent – a first-order iterative optimization algorithm for finding a local minimum of a differentiable function. The idea is to take repeated steps in the opposite direction of the gradient (or approximate gradient) of the function at the current point, because this is the direction of steepest descent. Conversely, stepping in the direction of the gradient will lead to a local maximum of that function; the procedure is then known as gradient ascent.
- Conjugate gradient method - numerical solution algorithm for systems of linear equations whose matrix is symmetric and positive-definite. The method is often implemented as an iterative algorithm, applicable to sparse systems that are too large for Cholesky decomposition to be applied.

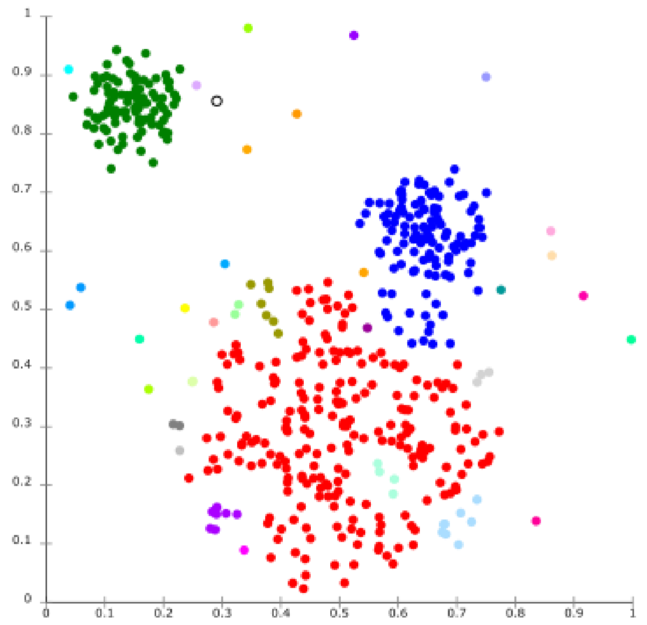




Clustering - I

- Cluster analysis (clustering) – grouping a set of objects in a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters).
- Usages: exploratory data mining, statistical data analysis, pattern recognition, image analysis, information retrieval, bioinformatics, data compression, computer graphics and machine learning.
- Cluster analysis can be achieved by various algorithms that differ in notion of what constitutes a cluster and how to efficiently find them – e.g. groups with small distances between members, dense areas of the data space, intervals or particular statistical distributions.





Clustering - II

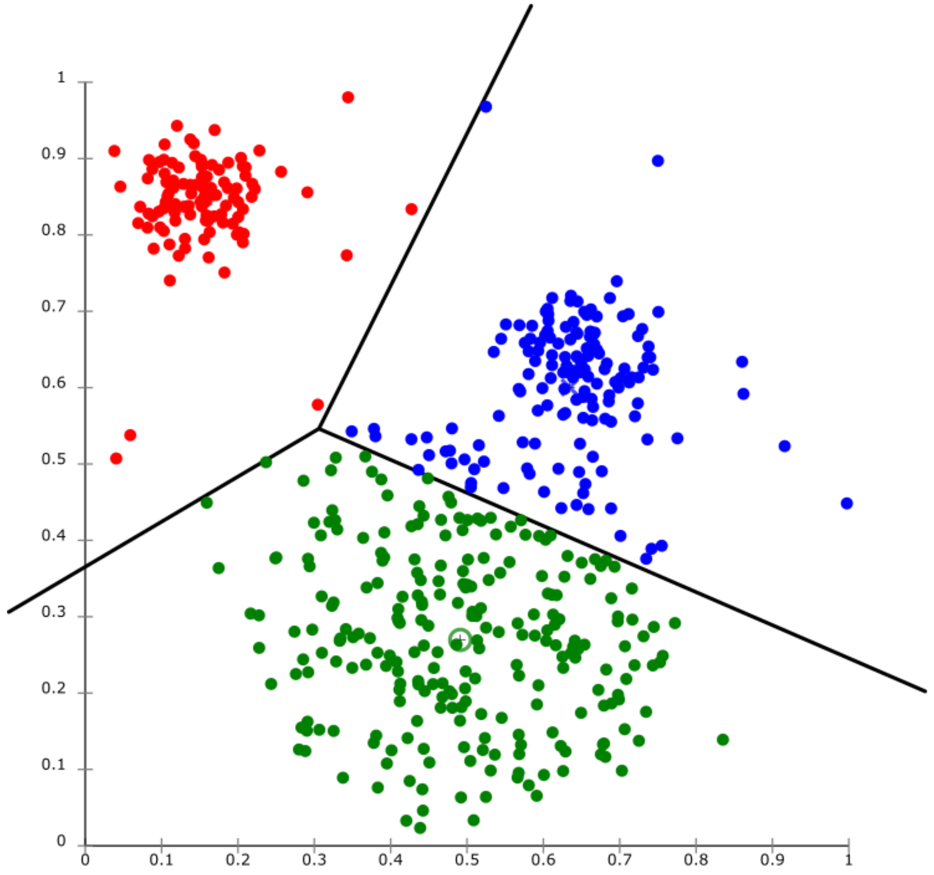
- Clustering = multi-objective optimization problem, parameter settings (distance function, density threshold / number of clusters) depend on data set and usage.
- Cluster analysis – iterative process of knowledge discovery or interactive multi-objective optimization that involves trial and failure. It is often necessary to modify data preprocessing and model parameters until the result achieves the desired properties.
- Algorithms: connectivity models (e.g. distance), centroid models (e.g. k-means), distribution models (statistical distributions, expectation-maximization), density models (DBSCAN, OPTICS), subspace models (two-mode-clustering – members + attributes), group models (just provide the grouping), graph-based models (cliques, quasi-cliques, HCS), signed graph models, neural models (self-organizing map).
- Clusterings can be roughly distinguished as: hard, soft (fuzzy); strict, overlapping hierarchical clustering, subspace clustering (strict in a defined subspace).

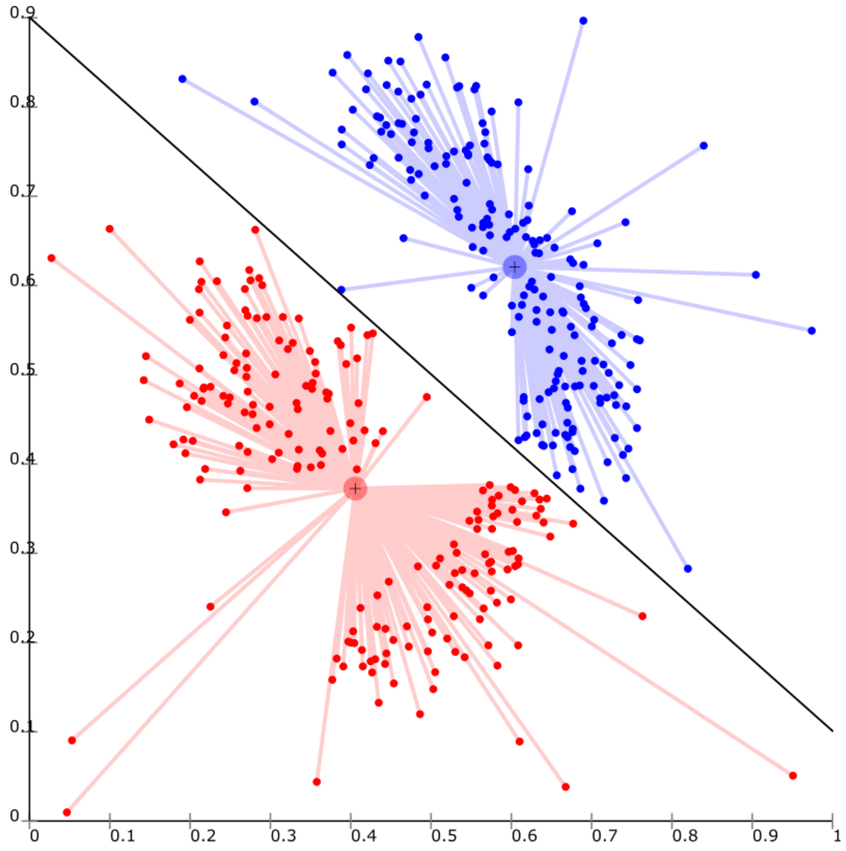
K-Means Clustering (Centroid-Based Clustering)

- Centroid-based clustering – clusters are represented by a central vector, which may not necessarily be a member of the data set. When the number of clusters is fixed to K, K-means clustering gives a formal definition as an optimization problem: find the K cluster centers and assign the objects to the nearest cluster center, such that the squared distances from the cluster are minimized.
- NP-hard problem -> approximate solutions – e.g. Lloyd's algorithm (k-means algorithm, local optimum, run multiple times with different random initializations),
- Variations: restricting the centroids to members of the data set (k-medoids), choosing medians (k-medians clustering), choosing the initial centers less randomly (k-means++) or allowing a fuzzy cluster assignment (fuzzy c-means).
- Conceptually close to nearest neighbor classification.

K-Means Clustering (Centroid-Based Clustering)

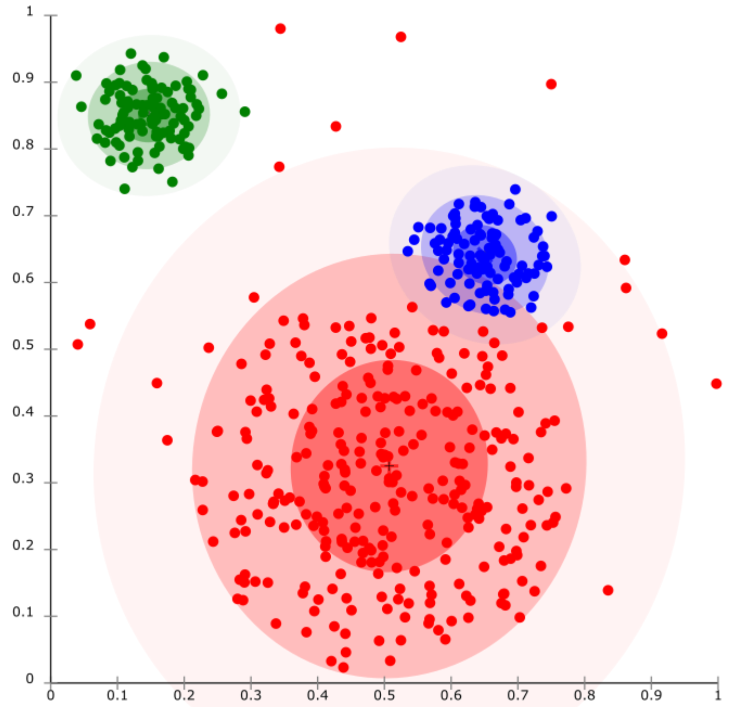
- Partitions the data space into a structure known as a Voronoi diagram

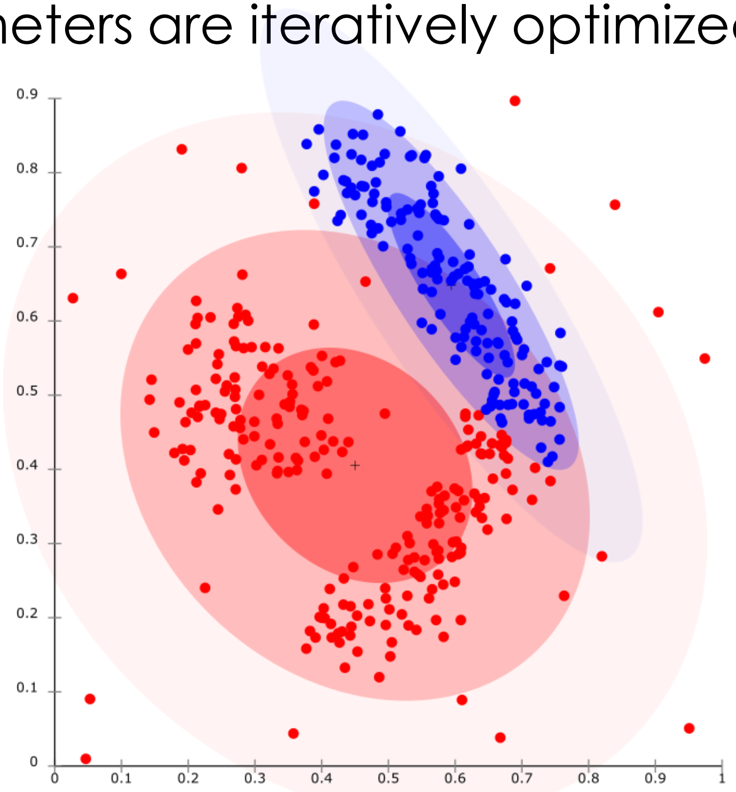




Distribution-Based Clustering

- Clusters are defined as objects most likely belonging to the same distribution.
- Gaussian mixture models (using the expectation-maximization algorithm) – the data set is modeled with a fixed (to avoid overfitting) number of Gaussian distributions that are initialized randomly and whose parameters are iteratively optimized to better fit the data.

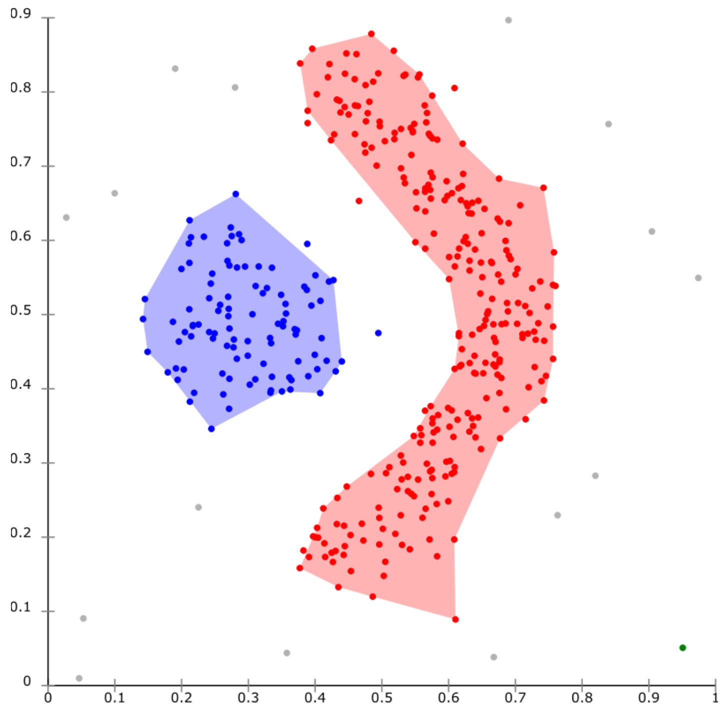


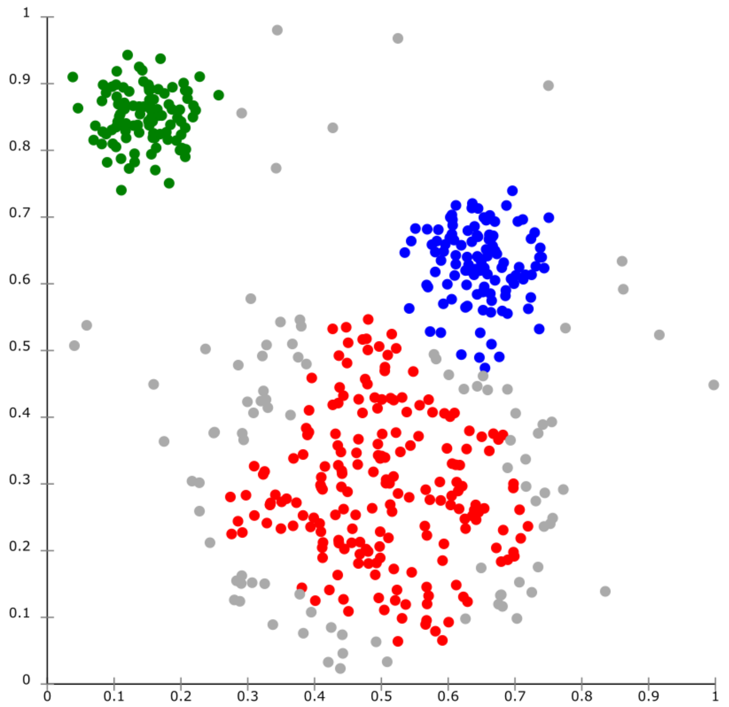


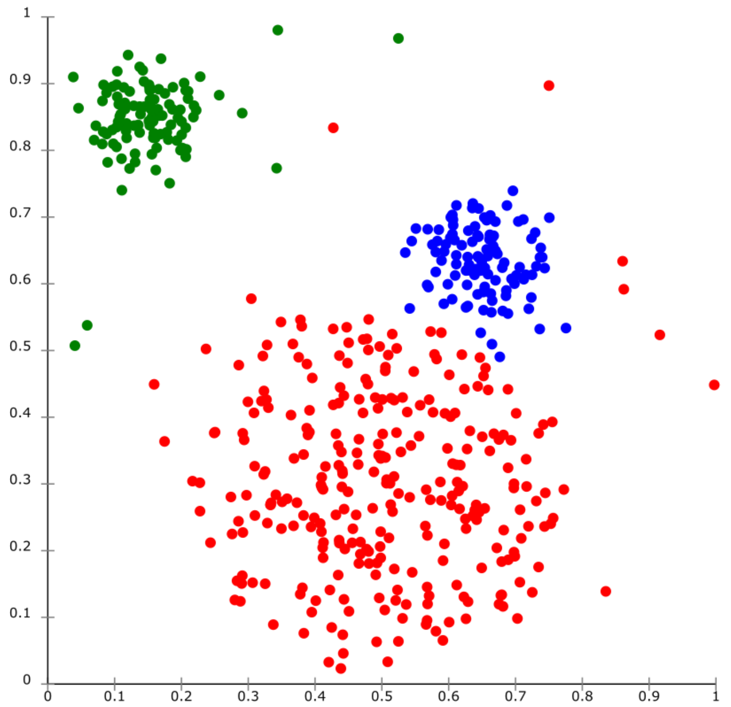
Pictures by: By Chire - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=17085713, https://commons.wikimedia.org/w/index.php?curid=17085331

Density-Based Clustering

- Clusters are defined as areas of higher density than the remainder of the data set. Objects in sparse areas - that are required to separate clusters - are usually considered to be noise and border points.
- The most popular density based clustering method is DBSCAN / OPTICS (different densities)



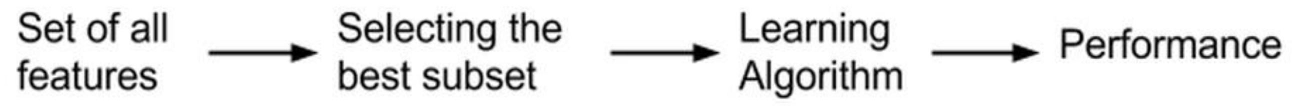


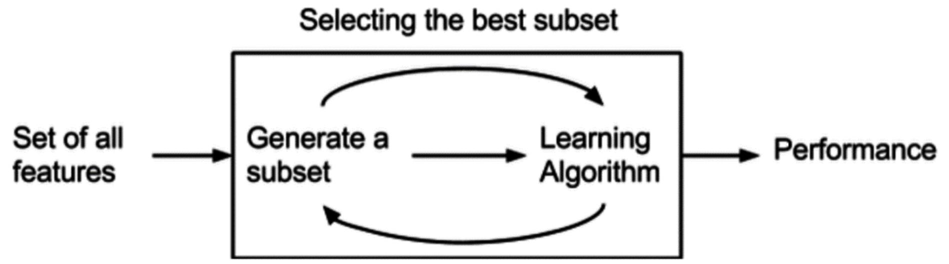


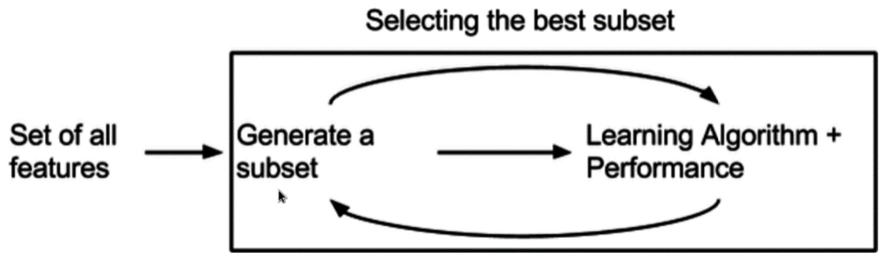
Pictures by: Chire - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=17085332, https://commons.wikimedia.org/w/index.php?curid=17085715, https://commons.wikimedia.org/w/index.php?curid=17085716

Feature Selection

- Feature selection (variable selection, attribute selection) - a process of selecting a subset of relevant features (variables, predictors) for use in model construction by removing features that are redundant or irrelevant. Reasons to use:
  - simplification of models to make them easier to interpret by researchers/users;
  - shorter training times;
  - to avoid the curse of dimensionality;
  - enhanced generalization by reducing (formally, reduction of variance)
- Metaheuristics – filter (correlation with the variable to predict), wrapper and embedded:



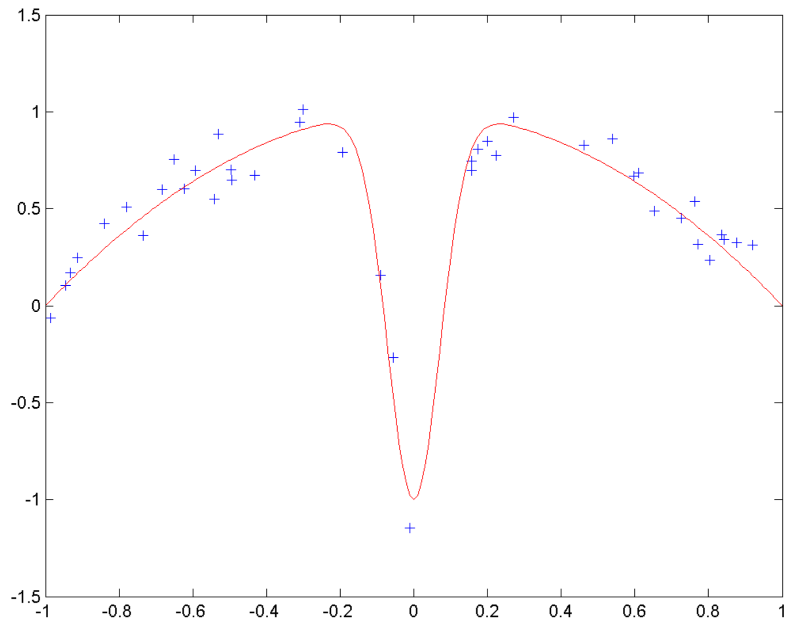




Pictures by: By Lucien Mousin - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=37776286,
By Lastdreamer7591 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=37208688, https://commons.wikimedia.org/w/index.php?curid=37208689

Bias–Variance Tradeoff - I

- Bias error is an error from erroneous assumptions in the learning algorithm. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting).
- Variance error – sensitivity to small fluctuations in the training set. High variance can cause an algorithm to model the random noise in the training data, rather than the intended outputs (overfitting).
- Dimensionality reduction and feature selection can decrease variance by simplifying models. Similarly, a larger training set tends to decrease variance. Adding features (predictors) tends to decrease bias, at the expense of introducing additional variance.

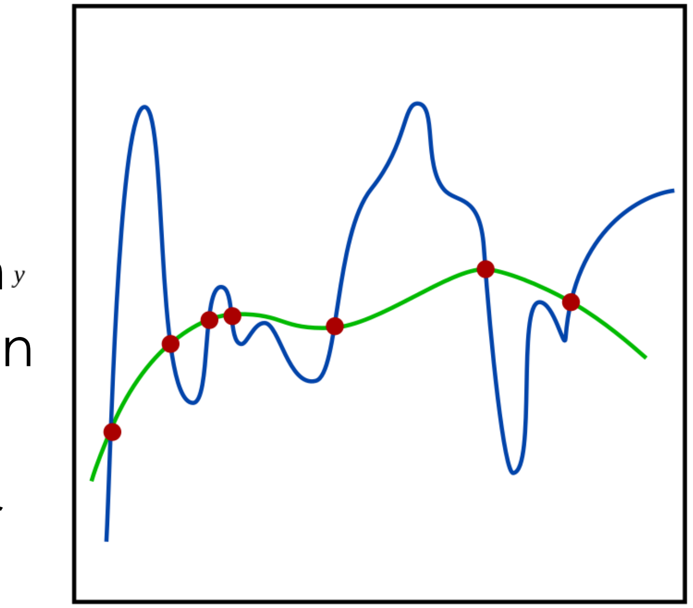


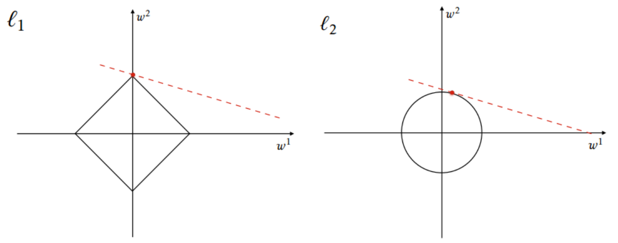
Bias–Variance Tradeoff – Controlling Bias & Variance

- Linear and Generalized Linear Models (GLM) can be regularized to decrease their variance at the cost of increasing their bias.
- Artificial neural networks, the variance increases and the bias decreases as the number of hidden units increase. Like in GLMs, regularization is typically applied.
- k-nearest neighbor models, a high value of k leads to high bias and low variance.
- Instance-based learning, regularization is achieved varying prototypes/exemplars mixture.
- Decision trees – depth of the tree determines the variance. Decision trees are commonly pruned to control variance.
- One way of resolving the trade-off is to use mixture models and ensemble learning. For example, boosting combines many "weak" (high bias) models in an ensemble that has lower bias than the individual models, while bagging combines "strong" learners in a way that reduces their variance.
- Model validation methods such as cross-validation (statistics) can be used to tune models so as to optimize the trade-off.

Regularization - I

- In mathematics, statistics, machine learning, regularization is the process of adding information in order to solve an ill-posed problem or to prevent overfitting.
- Empirical learning of classifiers (from a finite data set) is always an underdetermined problem, because it attempts to infer a function of any **f(x)** given only examples.
- Tikhonov regularization (ridge regression) – when learning a linear function **f(x) = w·x**, characterized by unknown vector **w**, we can use **L_2-norm** of the vector **w** to the loss expression in order to prefer solutions with smaller norms.
- Early stopping - can be viewed as regularization in time.
- Regularizers for sparsity – The **L_1** norm can be used to approximate the optimal **L_0** norm via convex relaxation, inducing sparsity. In the case of least squares -> **LASSO** (least abs. shrinkage and selection).



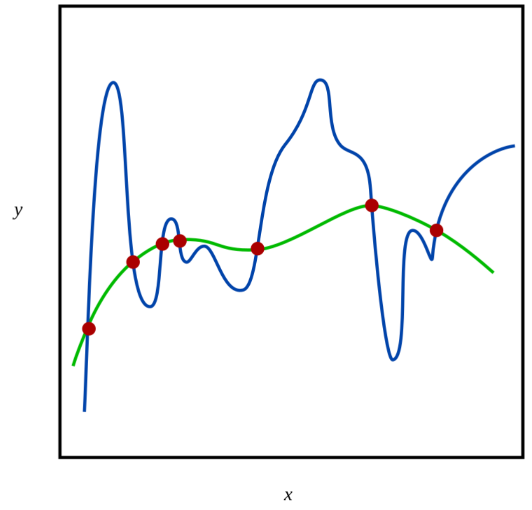


$$\min_{w \in \mathbb{R}^p} \frac{1}{n} \|\hat{X}w - \hat{Y}\|^2 + \lambda \|w\|_1$$

Regularization - II

- A regularization term (or regularizer) is added to a loss function:

$$\min_f \sum_{i=1}^{n} V(f(x_i), y_i) + \lambda R(f)$$



where **V** is an underlying loss function that describes the cost of predicting **f(x)** when the label is **y** and **λ** is a parameter which controls the importance of the regularization term, typically chosen to impose a penalty on the complexity of **f**. Concrete notions of complexity used include restrictions for smoothness and bounds on the vector space norm.

- A theoretical justification for regularization is that it attempts to impose Occam's razor on the solution (as depicted in the figure above, where the green function, the simpler one, may be preferred). From a Bayesian point of view, many regularization techniques correspond to imposing certain prior distributions on model parameters.

The Curse of Dimensionality

- The curse of dimensionality – refers to various phenomena that arise when analyzing and organizing data in high-dimensional spaces that do not occur in low-dimensional settings such as the three-dimensional physical space of everyday experience.
- When the dimensionality increases, the volume of the space increases so fast that the available data become sparse. This sparsity is problematic for any method that requires statistical significance. To obtain a statistically sound and reliable result, the amount of data needed to support the result often grows exponentially with the dimensionality.
- Also, organizing and searching data often relies on detecting areas where objects form groups with similar properties; in high dimensional data, however, all objects appear to be sparse and dissimilar in many ways, which prevents common data organization strategies from being efficient (e.g. k-NN clustering).

The Curse of Dimensionality – Problems [Zimek et al. 2012]

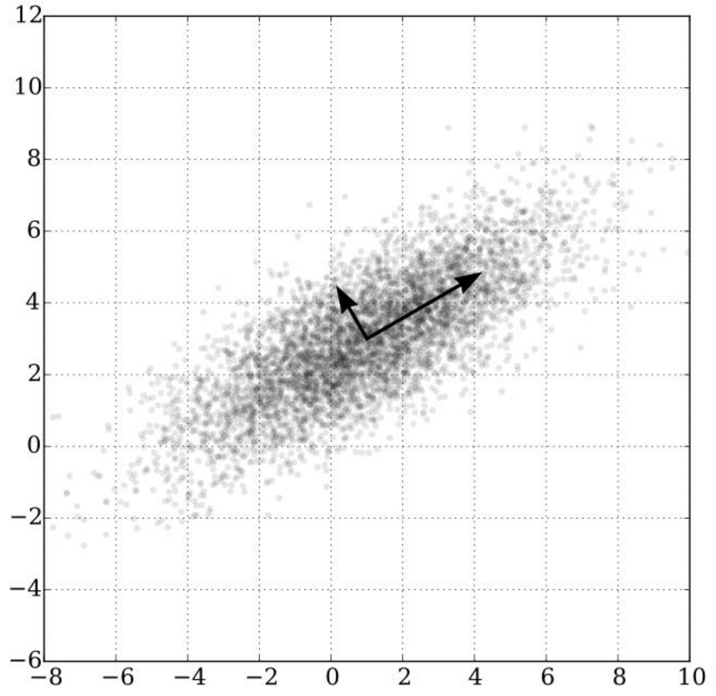
- Concentration of scores and distances – derived values such as distances become numerically similar;
- Irrelevant attributes – significant number of attributes may be irrelevant;
- Definition of reference sets – for local methods, reference sets are often k-NN based;
- Incomparable scores for different dimensionalities – different subspaces produce incomparable scores;
- Interpretability of scores – the scores often no longer convey a semantic meaning;
- Exponential search space – the search space can no longer be systematically scanned;
- Data snooping bias: given the large search space, for every desired significance a hypothesis can be found;
- Hubness – certain objects occur more frequently in neighbor lists than others (e.g. in k-NN).

The Blessing of Dimensionality [Donoho, 2012]

- Common-sense heuristics based on the most straightforward methods "can yield results which are almost surely optimal" for high-dimensional problems.
- Example: linear separability of a random point from a large finite random set with high probability even if this set is exponentially large: the number of elements in this random set can grow exponentially with dimension. This linear functional can be selected in the form of the simplest linear Fisher discriminant.
- Two sides of the same coin – e.g. the squared distance of random points to a selected point is, with high probability, close to the average (or median) squared distance. This property significantly simplifies the expected geometry of data and indexing of high-dimensional data (blessing), but, at the same time, it makes the similarity search in high dimensions difficult and even useless (curse).
- Zimek et al.: data becomes easier with each attribute that adds signal, and harder with attributes that only add noise (irrelevant error) to the data. In particular for unsupervised data analysis this effect is known as swamping.

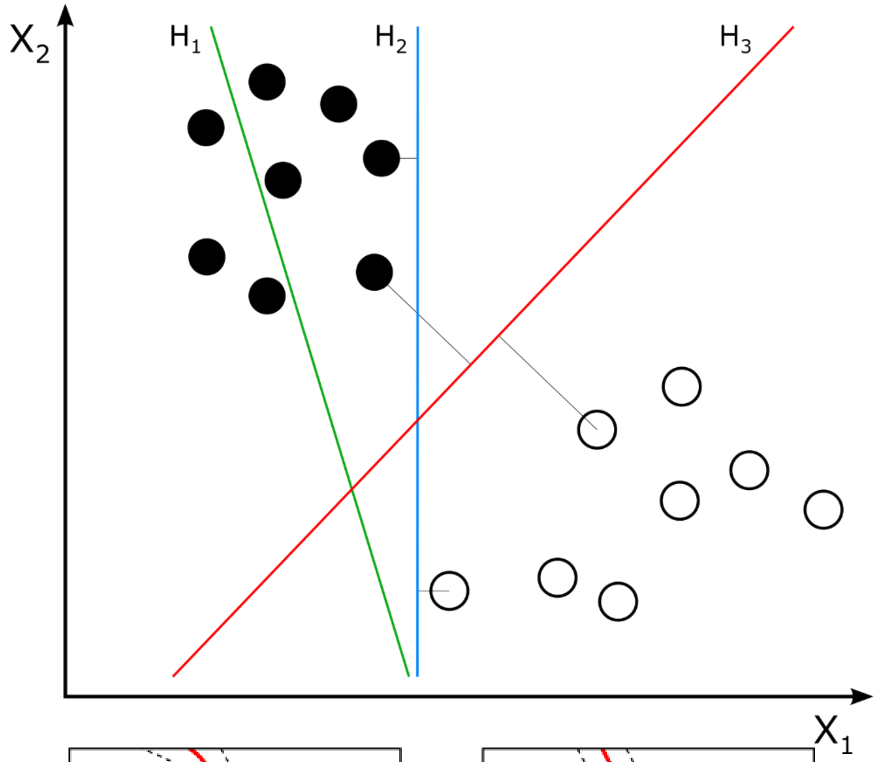
Principal Component Analysis (PCA)

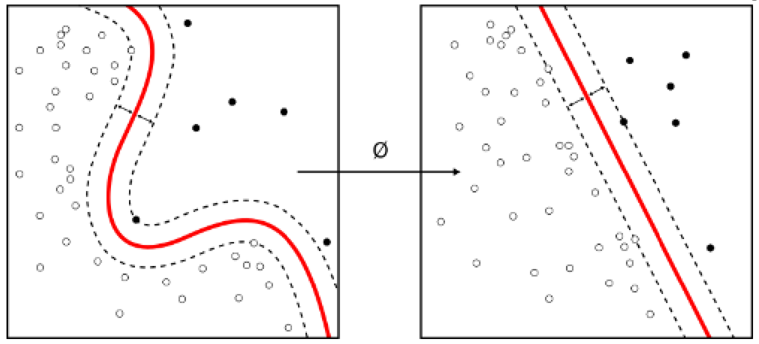
- Principal components of a collection of points in a real p-space are a sequence of direction vectors, where the vector is the direction of a line that best fits the data while being orthogonal to the first vectors. Here, a best-fitting line is defined as one that minimizes the average squared distance from the points to the line. These directions constitute an orthonormal basis in which different individual dimensions of the data are linearly uncorrelated.
- Principal component analysis (PCA) is the process of computing the principal components and using them to perform a change of basis on the data, sometimes using only the first few principal components and ignoring the rest.

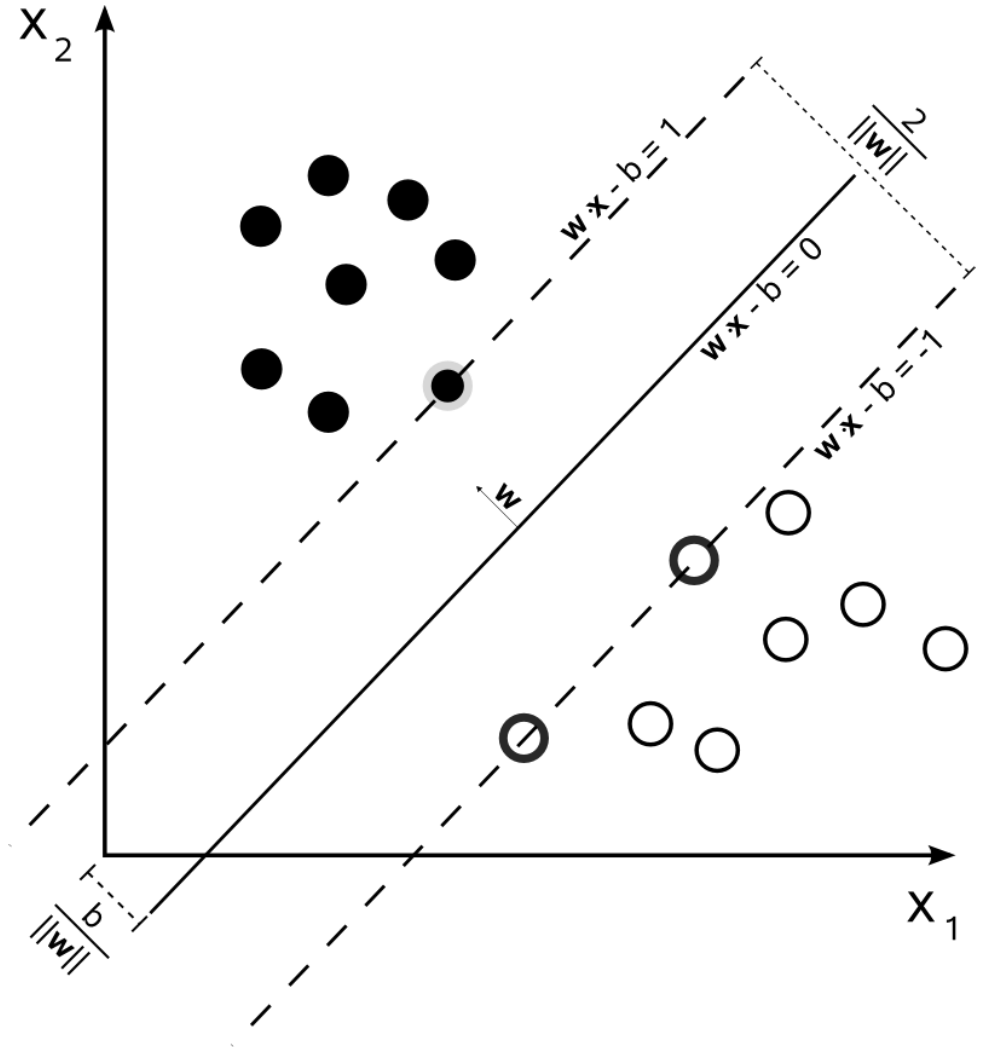


Tutorial: https://builtin.com/data-science/step-step-explanation-principal-component-analysis

Support Vector Machines (SVM) – I





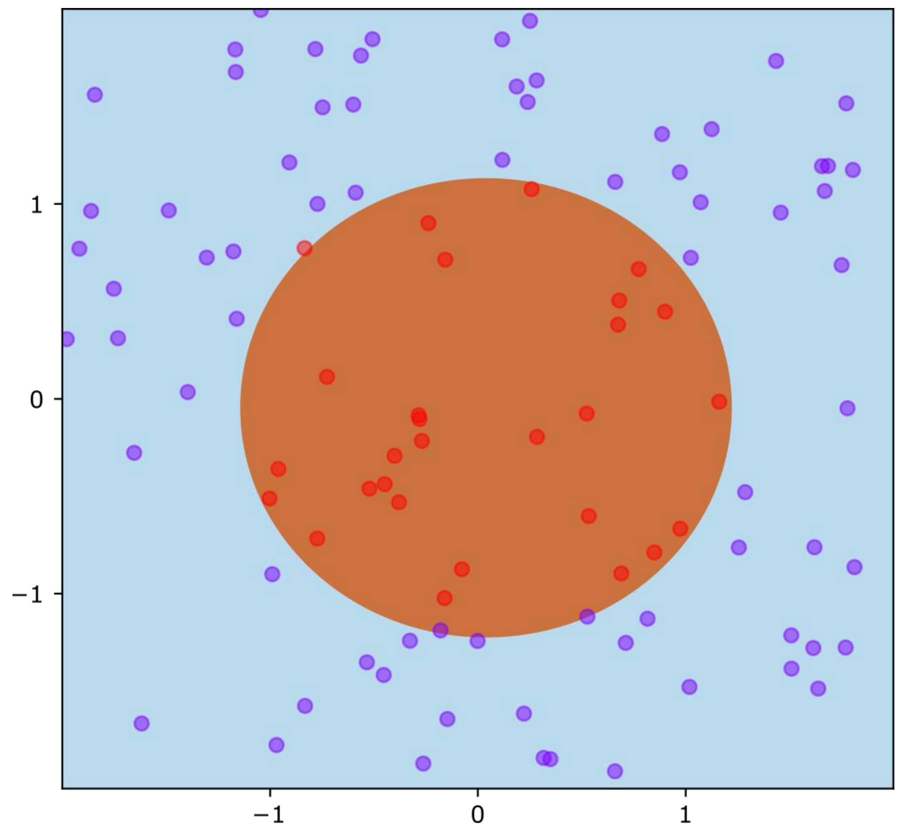


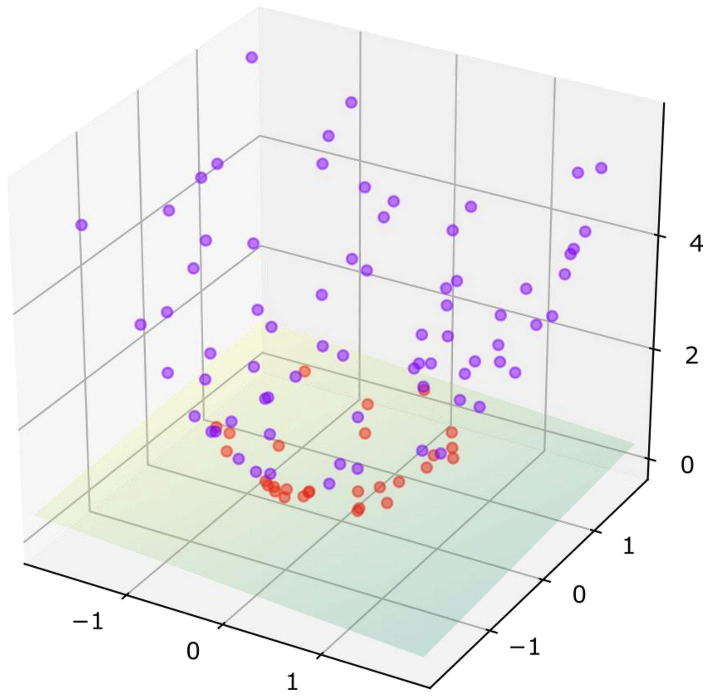
X_2
X_1
$\mathbf{w}\cdot\mathbf{x} - b = 1$
$\mathbf{w}\cdot\mathbf{x} - b = 0$
$\mathbf{w}\cdot\mathbf{x} - b = -1$
$\frac{2}{\|\mathbf{w}\|}$
$\frac{b}{\|\mathbf{w}\|}$
$\mathbf{w}$

Support Vector Machines (SVM) - II

- SVMs (also support-vector networks) – supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis, one of the most robust prediction methods, based on the statistical learning framework.
- Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.
- An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall.
- In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

Kernel Example

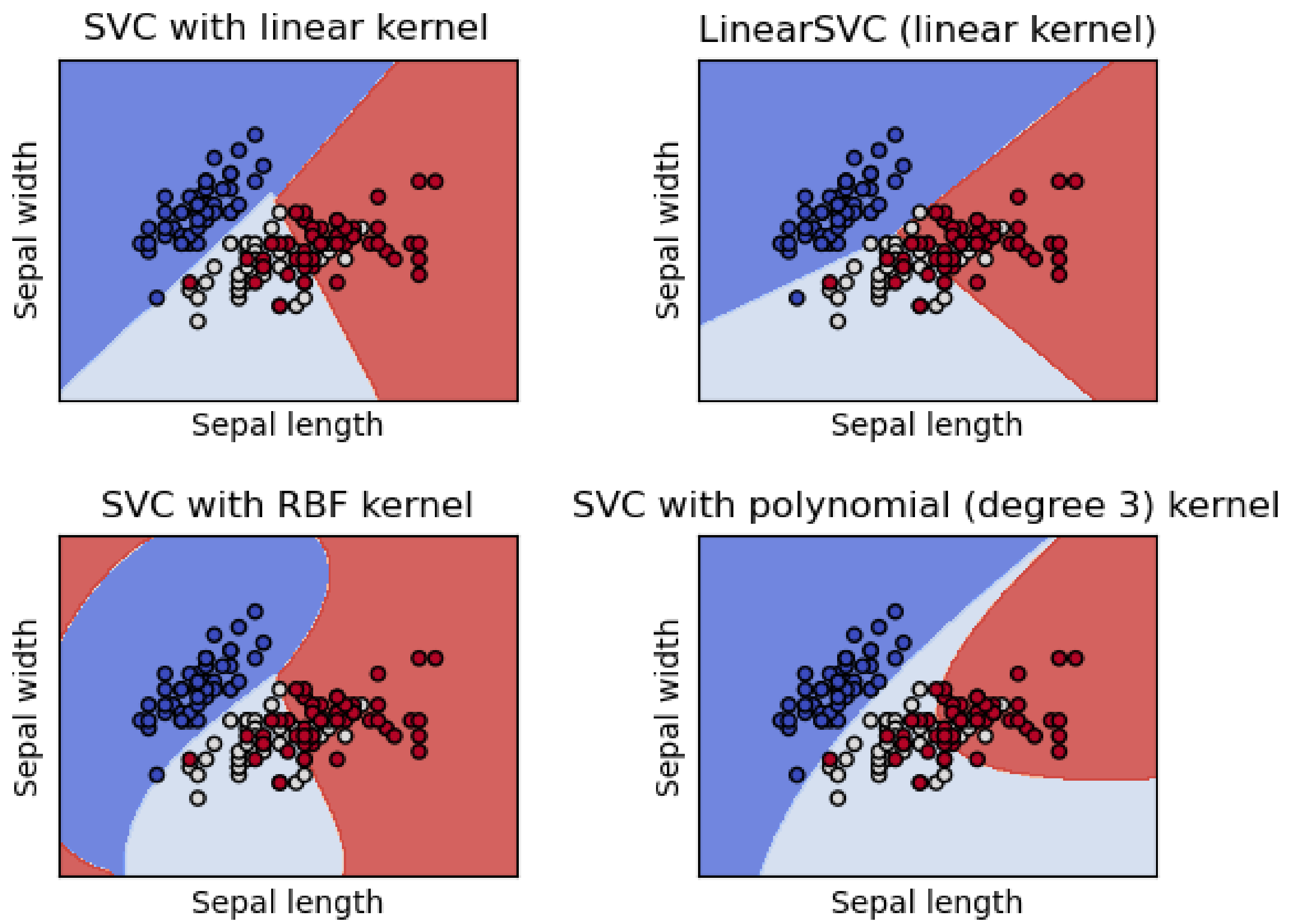




SVM with kernel given by φ((*a*, *b*)) = (*a*, *b*, *a*^{2} + *b*^{2}) and thus *K*(**x** , **y**) = $\mathbf{x} \cdot \mathbf{y} + \|\mathbf{x}\|^2 \|\mathbf{y}\|^2$. The training points are mapped to a 3-dimensional space where a separating hyperplane can be easily found.

By Shiyu Ji - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=60458994

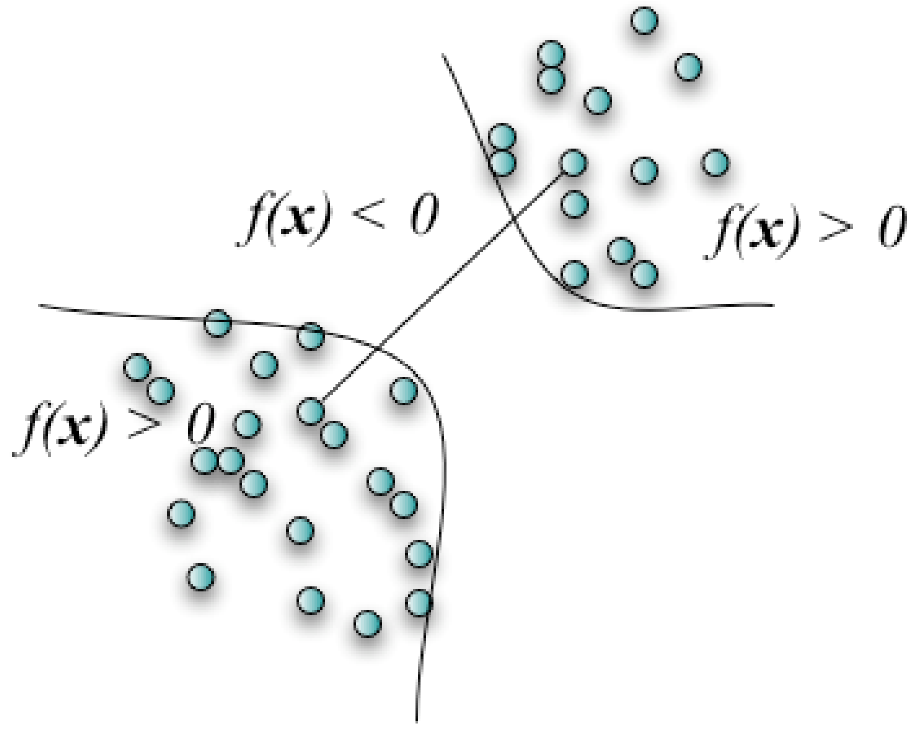
Support Vector Machines (SVM) – Different Kernels



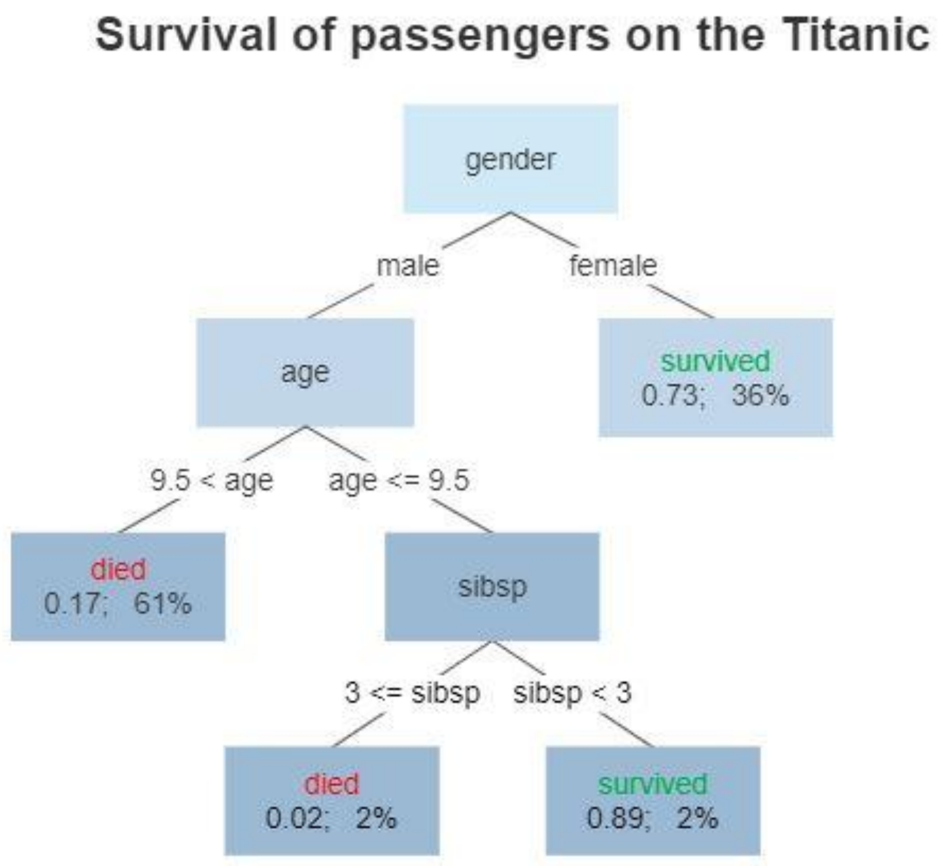
Picture by: Scikit-learn – https://scikit-learn.org/stable/auto_examples/svm/plot_iris_svc.html

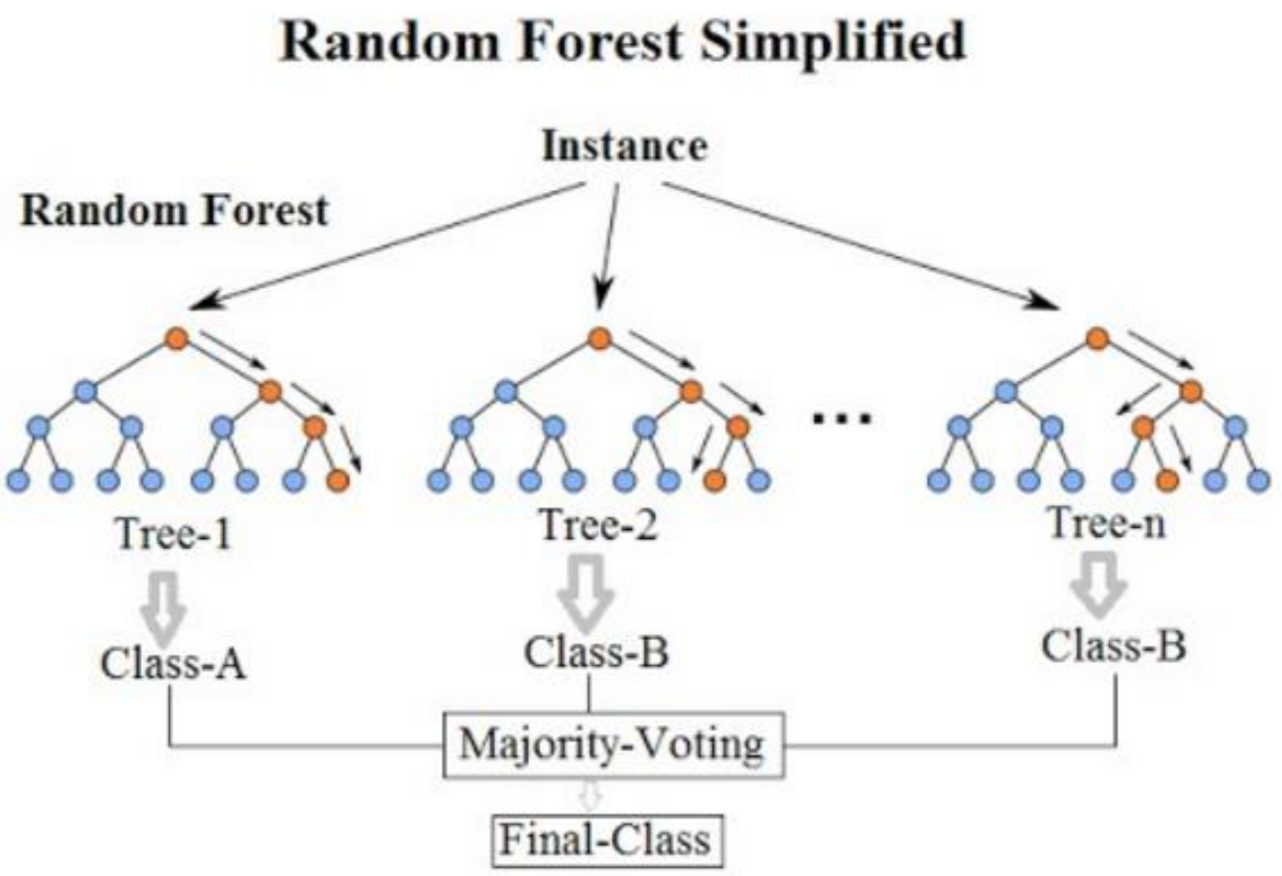
Support Vector Clustering

- When data are unlabelled, supervised learning is not possible, and an unsupervised learning approach is required, which attempts to find natural clustering of the data to groups, and then map new data to these formed groups.
- Support-vector clustering algorithm, created by Hava Siegelmann and Vladimir Vapnik, applies the statistics of support vectors, developed in the support vector machines algorithm, to categorize unlabeled data, and is one of the most widely used clustering algorithms in industrial applications.



Decision Trees. Random Forests





Thank’s for Your Attention!

Trayan Iliev

IPT – Intellectual Products & Technologies

http://iproduct.org/

https://github.com/iproduct

https://twitter.com/trayaniliev

https://www.facebook.com/IPT.EACAD