



Particle and Astroparticle Physics School in Tirana 2020

Machine learning techniques: an introduction

Nicola Amoroso, PhD
nicola.amoroso@uniba.it



UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO

Outline

- Part 1: Basic definitions, bias, variance, cross-validation, performance
- Part 2: Exploratory data analysis and Unsupervised learning
- Part 3: Supervised learning, state-of-the-art classifiers
- Part 4: Deep learning
- Part 5: Case study



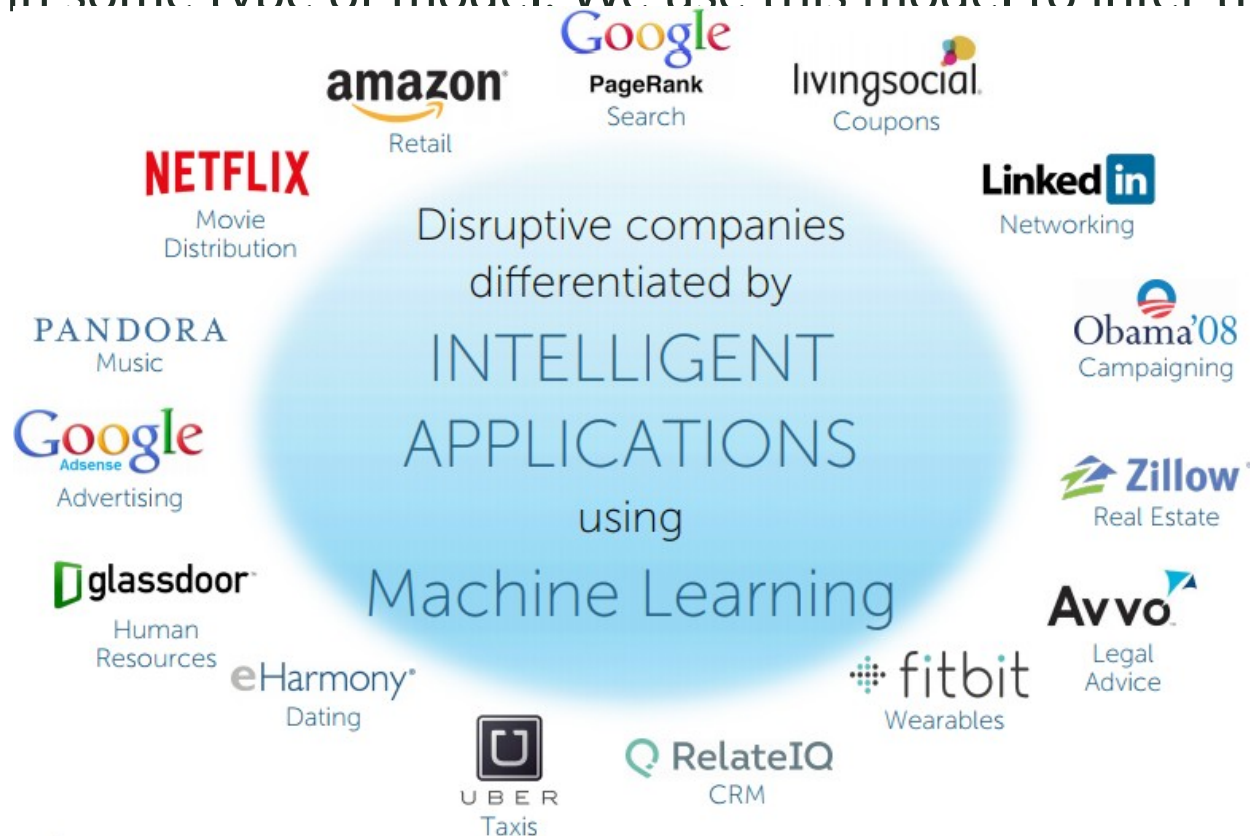
Part 1:

Basic definitions, bias, variance, cross-validation,
performance

Machine Learning

Interest in **machine learning** has exploded over the past decade. You see machine learning in computer science programs, industry conferences, and the Wall Street Journal almost daily.

Fundamentally, machine learning is using algorithms to extract information from raw data and represent it in some type of model. We use this model to infer things about other data we have not yet modeled.

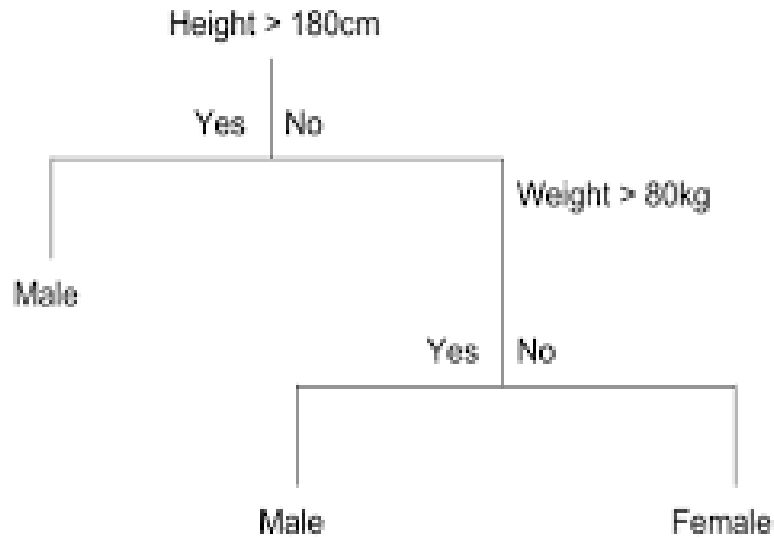


Top applications

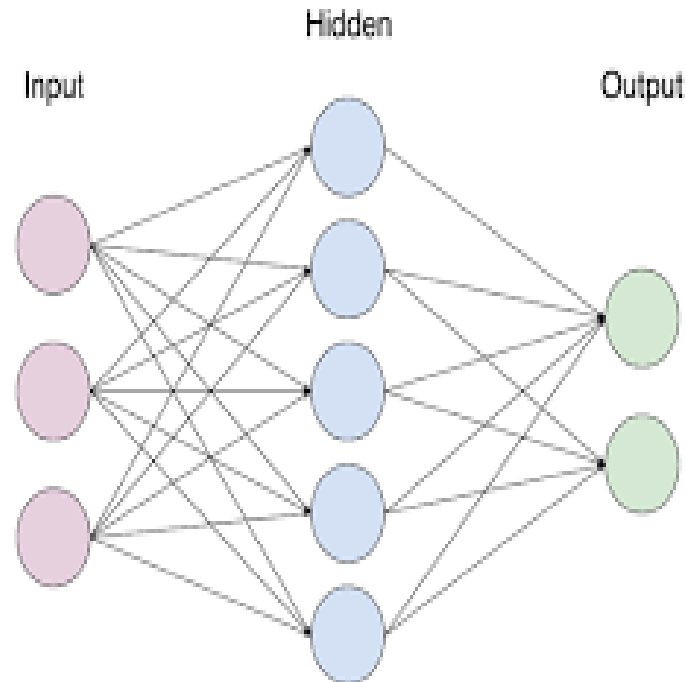
1. Medical diagnosis
2. Finance and fraud detection
3. Retail
4. Travel
5. Social networks

How can machines learn?

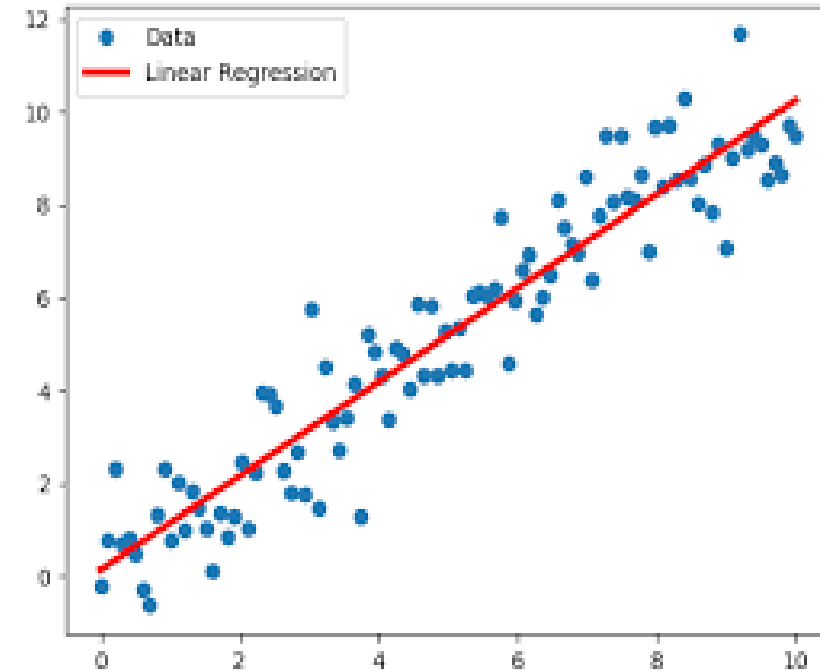
To define how machines can learn, we need to define what we mean by **learning**. We can think of machine learning as using algorithms for acquiring structural descriptions from data examples. Structural descriptions are models containing the information extracted from the raw data; we can use those structures or models to predict unknown data.



Decision tree



Neural network



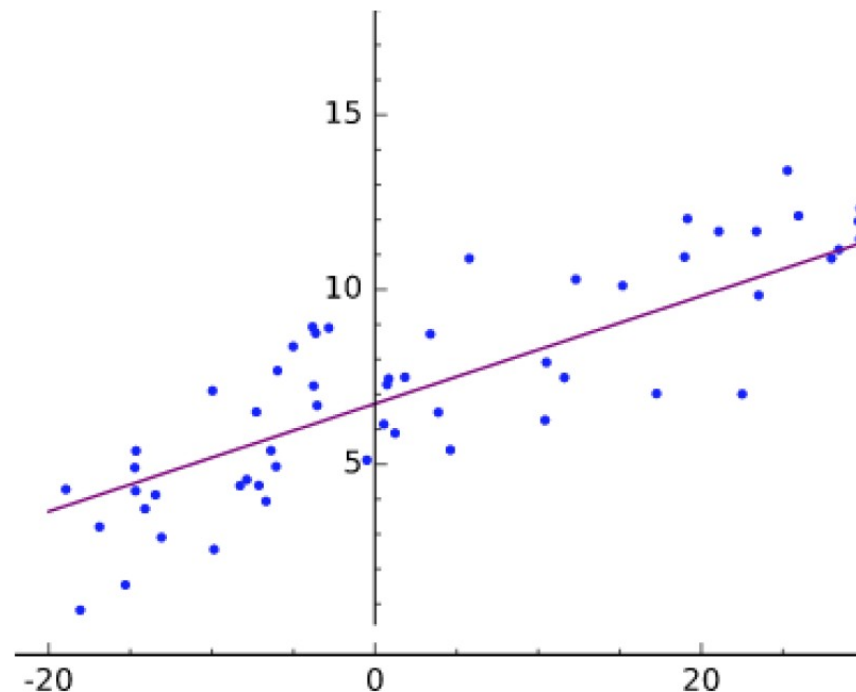
Linear regression

A simple regression

Let's assume we have N features characterizing a set of M observations; we want to build a linear model:

We want to solve the system of M equations or, at least, find the best approximation. In this case we want to learn **the best coefficients** (A,b) which minimize the error .

	Feature 1	Feature 2	...	Feature N
Obs 1				
Obs 2				
...			x_{ij}	
Obs M				



Building the model

The goal is to solve three distinct problems about the parameters (A,b) in order to get a model:

1. A **Hypothesis** about the data: exploring the model parameters;
2. A **Cost** function: evaluate the performance of the model;
3. An **Update** function: improve the performance by iteration.

Classification is modeling based on delineating classes of output based on some set of input features. If regression give us an outcome of "how much," classification gives us an outcome of "what kind." The dependent variable y is categorical rather than numerical.

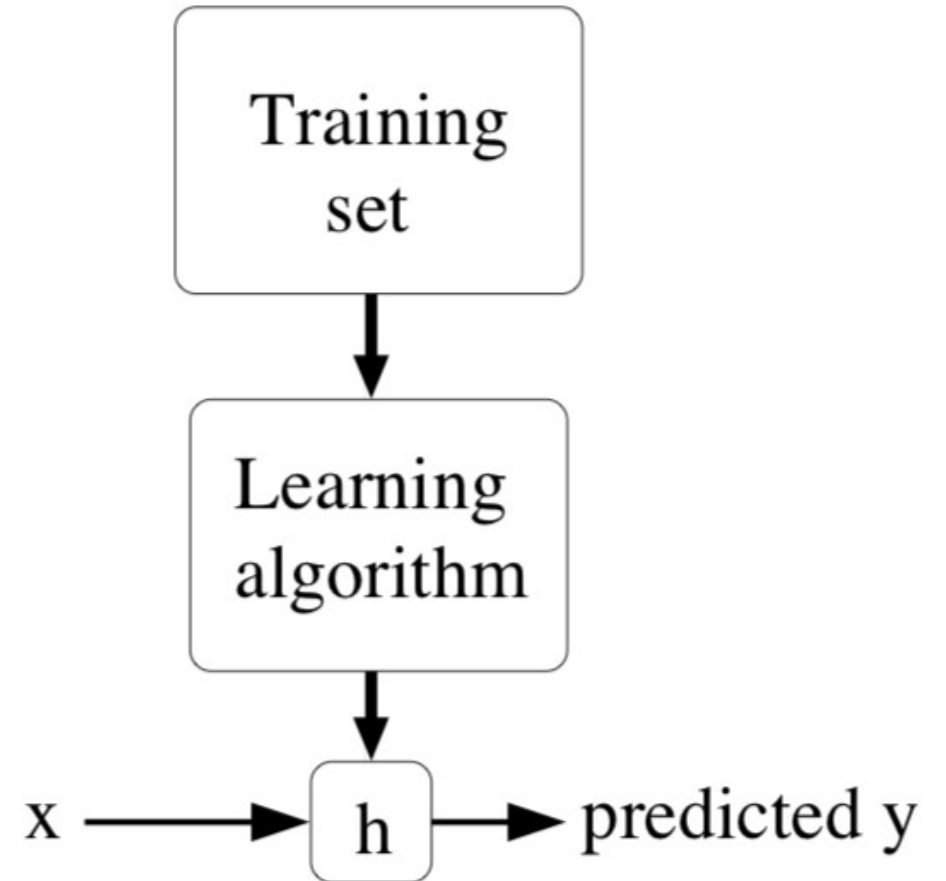
Supervised learning

To establish notation for future use, we'll use $x^{(i)}$ to denote the “input” variables (**features**) and $y^{(i)}$ to denote the “output” or **target** variable that we are trying to predict.

A pair $(x^{(i)}, y^{(i)})$ is called a training example, and the dataset that we'll be using to learn—a list of M training examples—is called a **training set**.

Our goal is, given a training set, to learn a function $h : X \rightarrow Y$ so that $h(x)$ is a “good” predictor for the corresponding value of y .

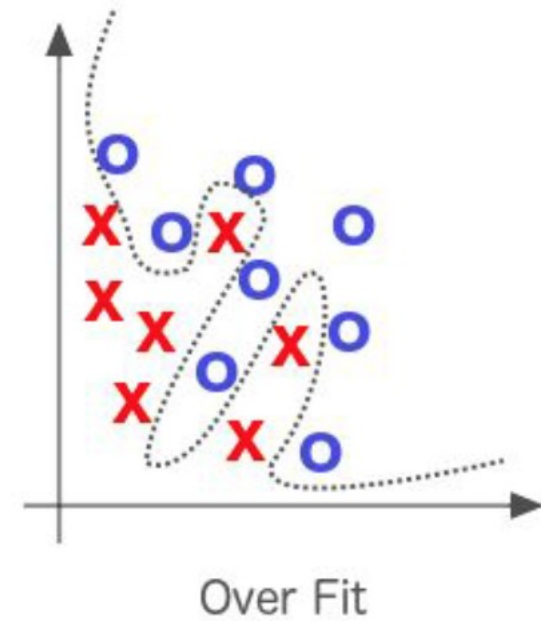
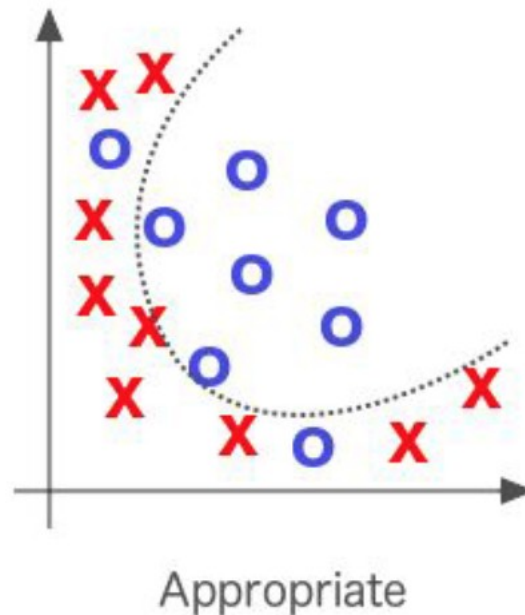
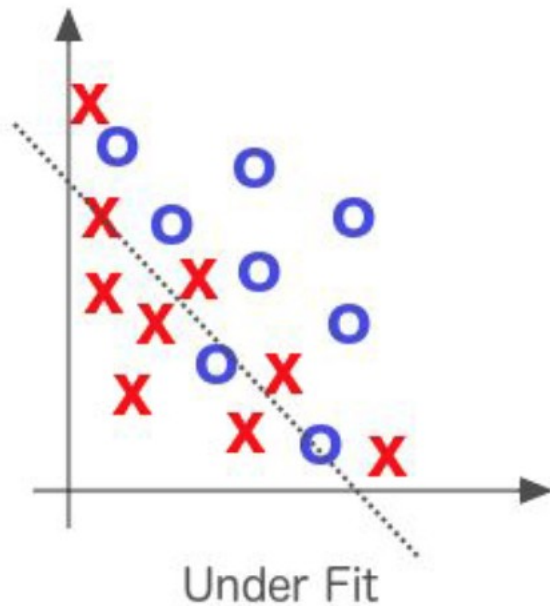
For historical reasons, this function h is called a hypothesis. Seen pictorially, the process is therefore like this:



Basic concepts: Bias and Variance

The **bias-variance** dilemma or problem is the conflict in trying to simultaneously minimize two sources of error that prevent learning algorithms from generalizing beyond their training set.

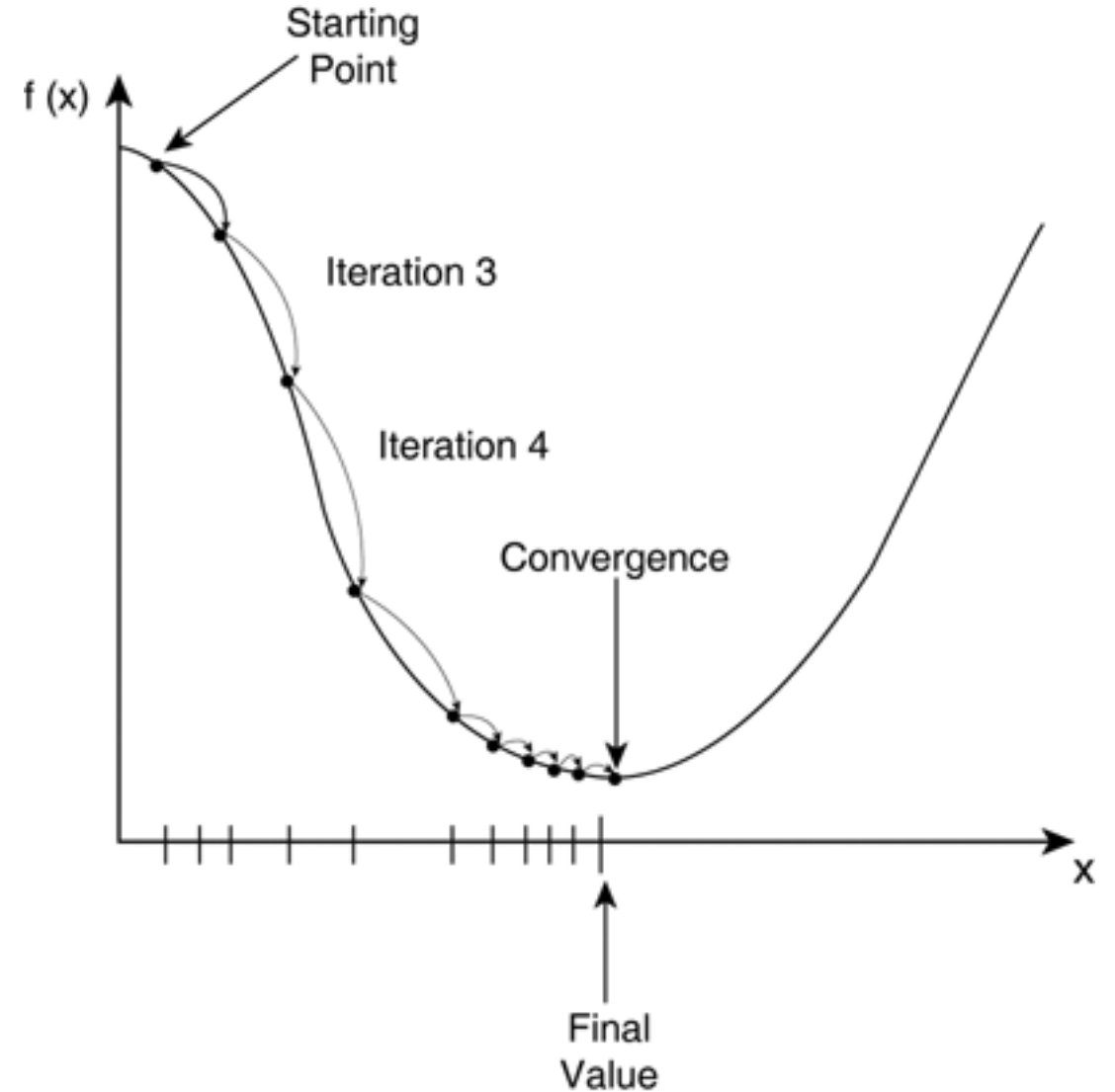
- The **bias** is an error from erroneous assumptions in the learning algorithm (underfitting).
- The **variance** is an error from sensitivity to small fluctuations in the training set. High variance can cause an algorithm to model the random noise in the training data, rather than the intended outputs (overfitting).



Cost function

Let us say we want to fit a simple linear model with two features and three parameters, and:

Using the notation θ . We want to determine the best parameters using our training sample. The optimal parameters are found minimizing a suitable **cost function**, for example:



Gradient descent

Start with an initial random guess for the parameter values:

is the **learning rate**. The update is simultaneously performed for all values of j . In order to implement this algorithm, we have to work out what is the partial derivative term on the right hand side:

Cross-validation

Let us suppose we are given a training set and we want to learn a model describing this data. Consider choosing the order of a polynomial. The higher the order of the polynomial, the better it will fit the training set. Hence, this method will always select a high-variance model, which we saw previously is often poor choice (overfitting).

Here's an algorithm that works better: **cross validation**.

```
1 Define sets of model parameter values to evaluate
2 for each parameter set do
3     for each resampling iteration do
4         Hold-out specific samples
5         [Optional] Pre-process the data
6         Fit the model on the remainder
7         Predict the hold-out samples
8     end
9     Calculate the average performance across hold-out predictions
10 end
11 Determine the optimal parameter set
12 Fit the final model to all the training data using the optimal parameter set
```

Evaluating models

Evaluating models is the process of understanding how well they give the correct classification/regression and then **measuring** the value of the prediction in a certain context.

For what concerns regression we want to measure how close predictions and actual values are:

- RMSE =
- MAE =

For classification, usual metrics are:

- Accuracy
- Area under the ROC curve
- Sensitivity and Specificity

Confusion matrix

Let's take a look at the basic tool for evaluating models: the **confusion matrix**, also called table of confusion, is a table of rows and columns that represents the predictions and the actual outcomes (labels).

	P' (Predicted)	N' (Predicted)
P (Actual)	True Positive	False Negative
N (Actual)	False Positive	True Negative

We measure the following quantities:

- *True positives.*
Positive prediction - Positive Label;
- *False positives.*
Positive prediction - Negative Label;
- *True negatives.*
Negative prediction - Negative Label;
- *False negatives.*
Negative prediction - Positive Label.

Performance metrics

Sensitivity = measures how often we correctly classify the positive class.

Specificity = measures how often we correctly classify the negative class.

Accuracy = measures how often we correctly classify both classes.

Precision = measures how often the positive predictions are correct.

Dice or F1 = an accuracy measure weighted over the misclassified examples.

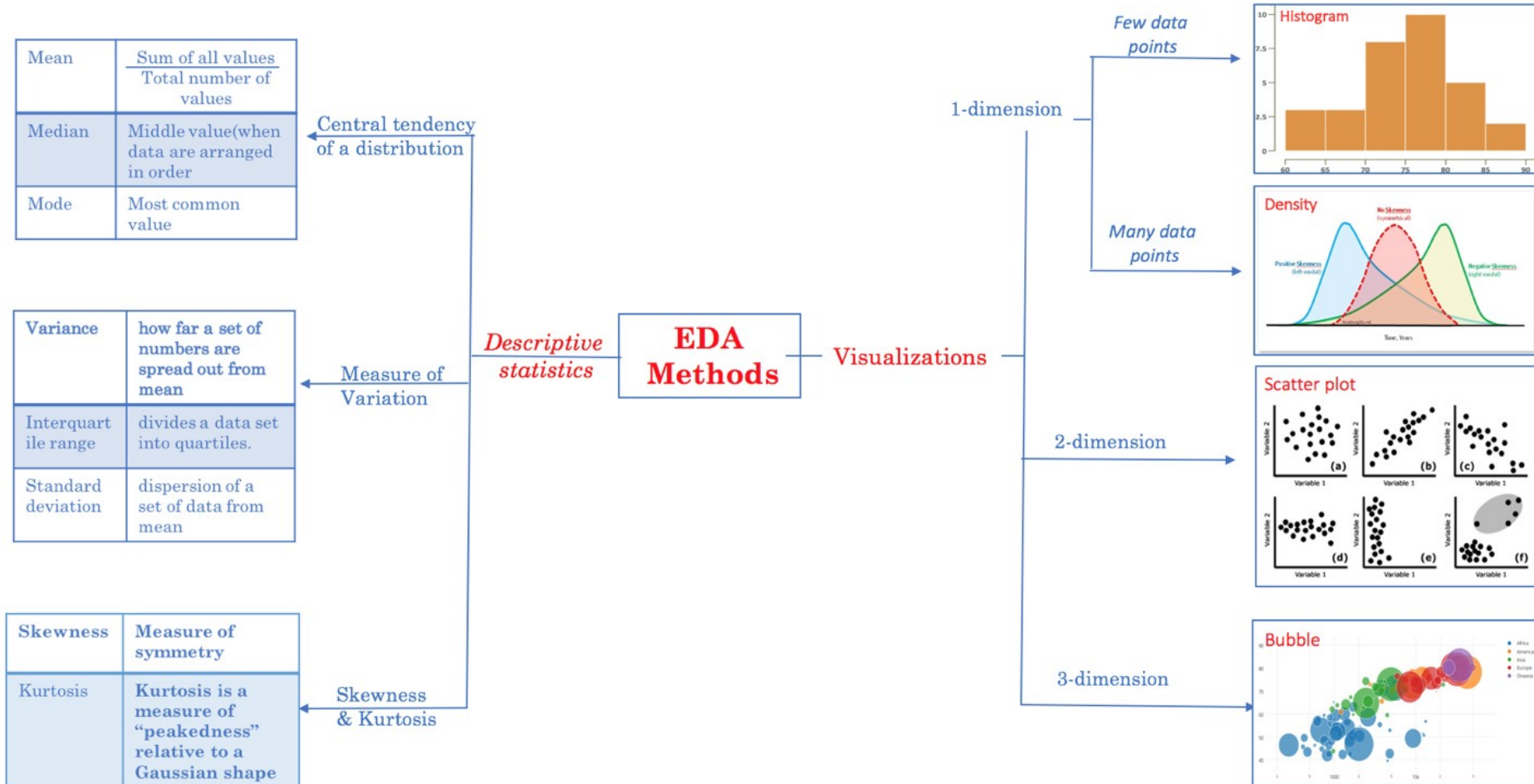


Part 2:

Exploratory data analysis and Unsupervised learning

Exploratory data analysis

The exploratory analysis of the data should not be underestimated. It's a very important phase. In short, the data begins to tell us a story, and to tell this story, we can make use of visualization techniques, summarization, transformation, and handling of data.



The Iris example

1. Summary statistics;
2. Basic visualization:
 - Histograms;
 - Barplots;
 - Boxplots.
3. Exploring relations in data: clustering and dimensionality reduction

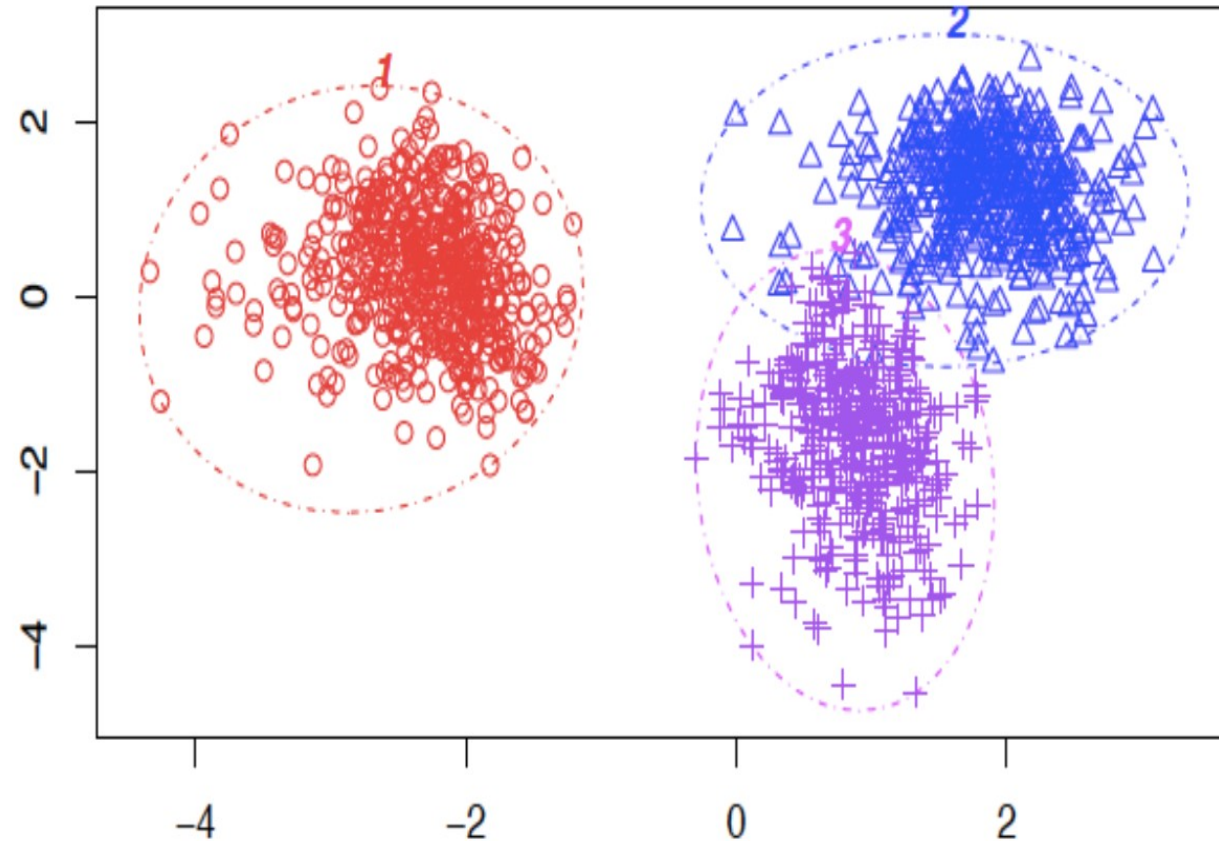
Let's go to Rstudio.

Clustering analysis

One of the most used techniques in unsupervised learning is the **Clustering Analysis**. Identifying groups can uncover and help to explain some patterns hiding in data.

Typically, the application of clustering techniques involves five phases:

1. developing a working dataset;
2. preprocessing and standardization of data;
3. finding clusters in the data;
4. interpreting those clusters;
5. finding conclusions.



Rescaling data

Whichever model we use will assume different things about the data. In the case of clustering models, it is very important that data, from the different numerical variables, expressed on a **similar scale**.

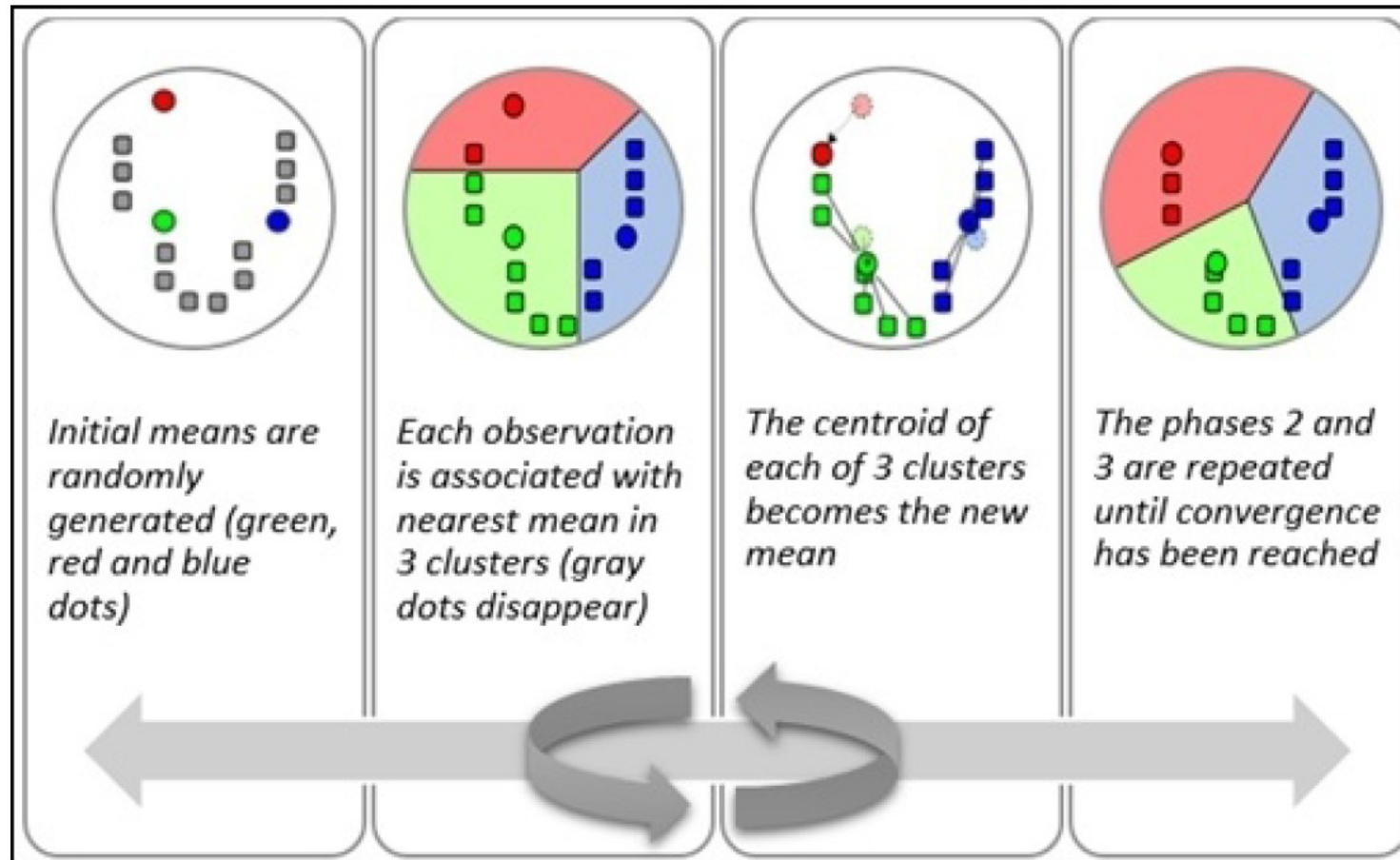
In order to mitigate the problem of the scale of the data, we can use methods of standardization. The standardization is the process of adjusting the data to a specific range. There are several techniques for normalization:

- Recenter ;
- Scale ;
- Robust recenter ;
- Natural log

Let's see some examples in **Ex. 2**.

Finding clusters: K-means

In very general terms, the **K-Means** algorithm aims to partition a set of observations into clusters so that each observation belongs to the cluster that possesses the nearest mean, see **Ex. 3**.



Hierarchical clustering

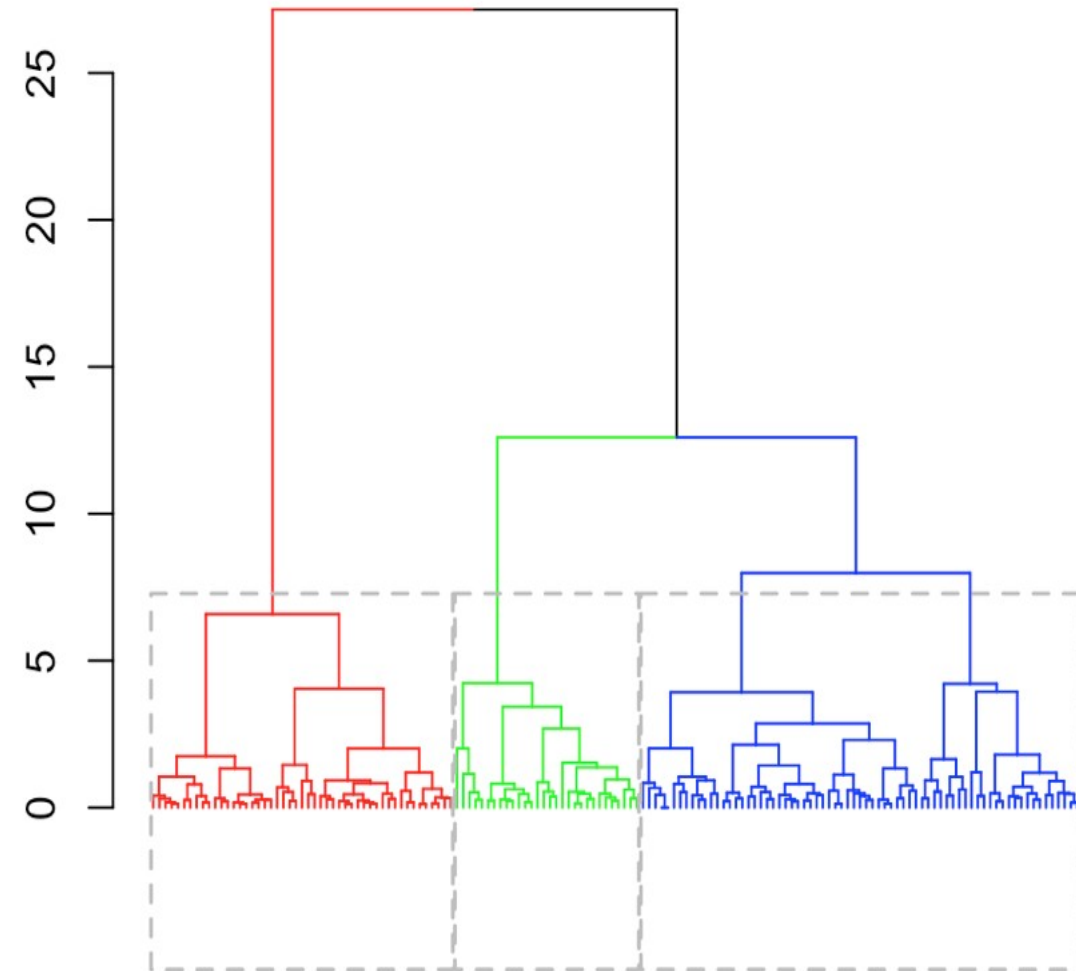
Another fundamental method is the **Hierarchical Clustering**. This method, as its name suggests, aims to build a hierarchy of clusters, see **Ex. 4**.

Agglomerative methods:

This method uses a bottom-up approach in which each observation begins in its own clusters and pairs of clusters are recursively merged.

Divisive methods: This method uses a top-down approach in which each observation begins in one cluster and split recursively.

In a hierarchical clustering, there are two very important parameters: the *distance metric* and the *linkage method*.



Distance metric

Defining closeness is a fundamental aspect. If you don't use a metric for distance that makes sense with your dataset, it is quite possible that you won't get any useful information from your cluster analyses.

Distance Metric	Definition
Euclidean Distance	Usual square distance between the two vectors (2 norm)
Maximum Distance	Maximum distance between two components of x and y (supremum norm)
Manhattan Distance	Absolute distance between the two vectors (1 norm)
Canberra Distance	It is a weighted version of L1 (Manhattan) distance. $\sum(x_i - y_i / x_i + y_i)$. Terms with zero numerator and denominator are omitted from the sum and treated as if the values were missing
Binary Distance	The vectors are regarded as binary bits, so non-zero elements are <i>on</i> and zero elements are <i>off</i> . The distance is the proportion of bits in which only one is <i>on</i> among those in which at least one is <i>on</i>
Pearson Distance	Also named <i>not centered Pearson</i> $\sum(x_i y_i) / \sqrt{[\sum(x_i^2) \sum(y_i^2)]}$
Correlation	Also named <i>Centered Pearson</i> $1 - \text{corr}(x,y)$
Spearman Distance	Compute a distance based on rank. $\sum (d_i^2)$ where d_i is the difference in rank between x_i and y_i

Distance metric

The linkage methods determine how the distance between two clusters is defined.

A linkage rule is necessary for calculating the inter-cluster distances. It is important to try several linkage methods to compare their results. Depending on the dataset, some methods may work better.

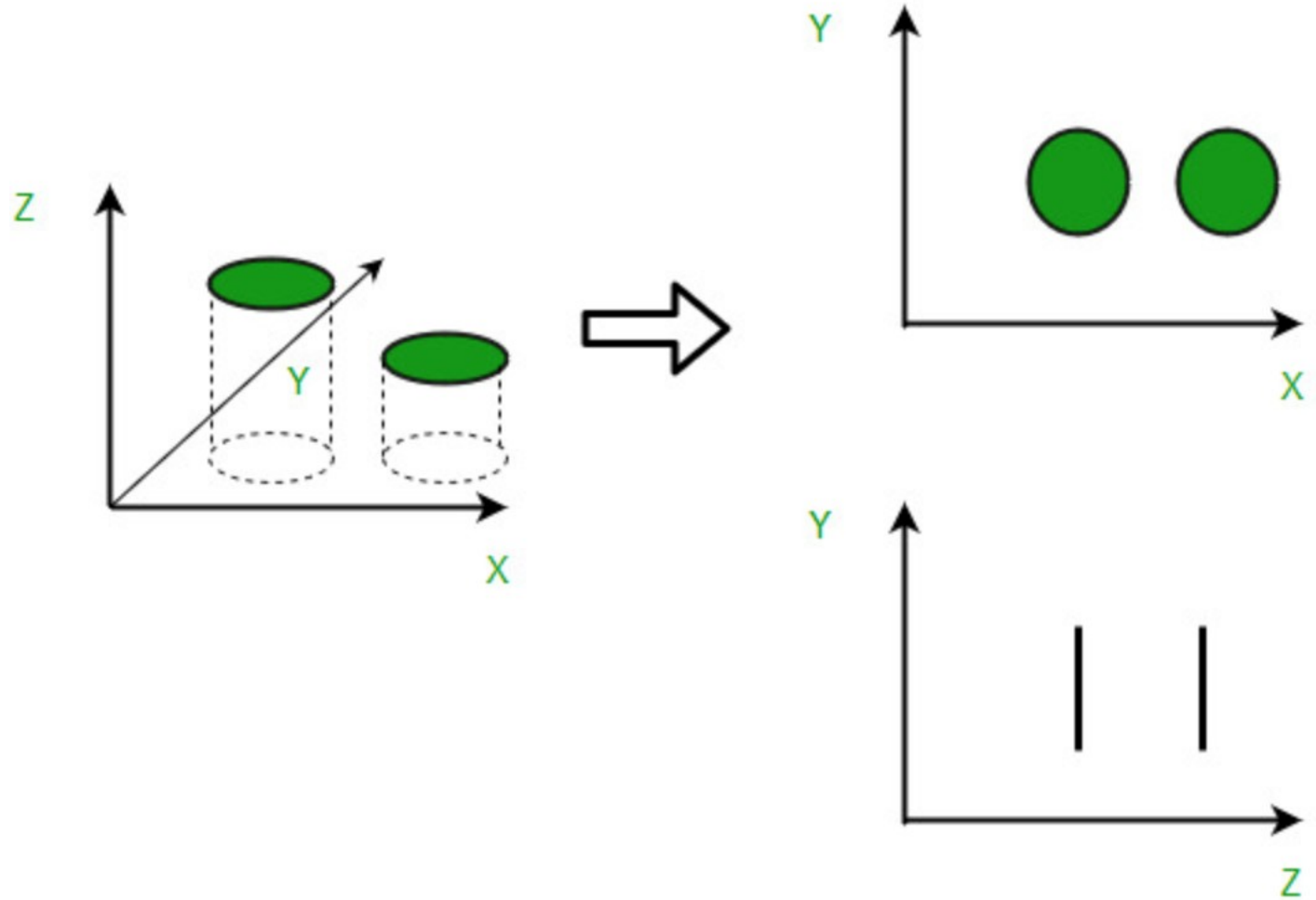
Linkage Methods	Definition
Single Linkage	The distance between two clusters is the minimum distance between an observation in one cluster and an observation in the other cluster. A good choice when clusters are obviously separated.
Complete Linkage	The distance between two clusters is the maximum distance between an observation in one cluster and an observation in the other cluster. It can be sensitive to outliers.
Average Linkage	The distance between two clusters is the mean distance between an observation in one cluster and an observation in the other cluster.
Centroid Linkage	The distance between two clusters is the distance between the cluster centroids or means.
Median Linkage	The distance between two clusters is the median distance between an observation in one cluster and an observation in the other cluster. It reduces the effect of outliers.
Ward Linkage	The distance between two clusters is the sum of the squared deviations from points to centroids. Try to minimize the within-cluster sum of squares. It can be sensitive to outliers.
McQuitty Linkage	When two clusters A and B are be joined, the distance to new cluster C is the average of distances of A and B to C. So, the distance depends on a combination of clusters instead of individual observations in the clusters.

Dimensionality reduction

Dimensionality reduction techniques refer to the processes of reducing the number of variables:

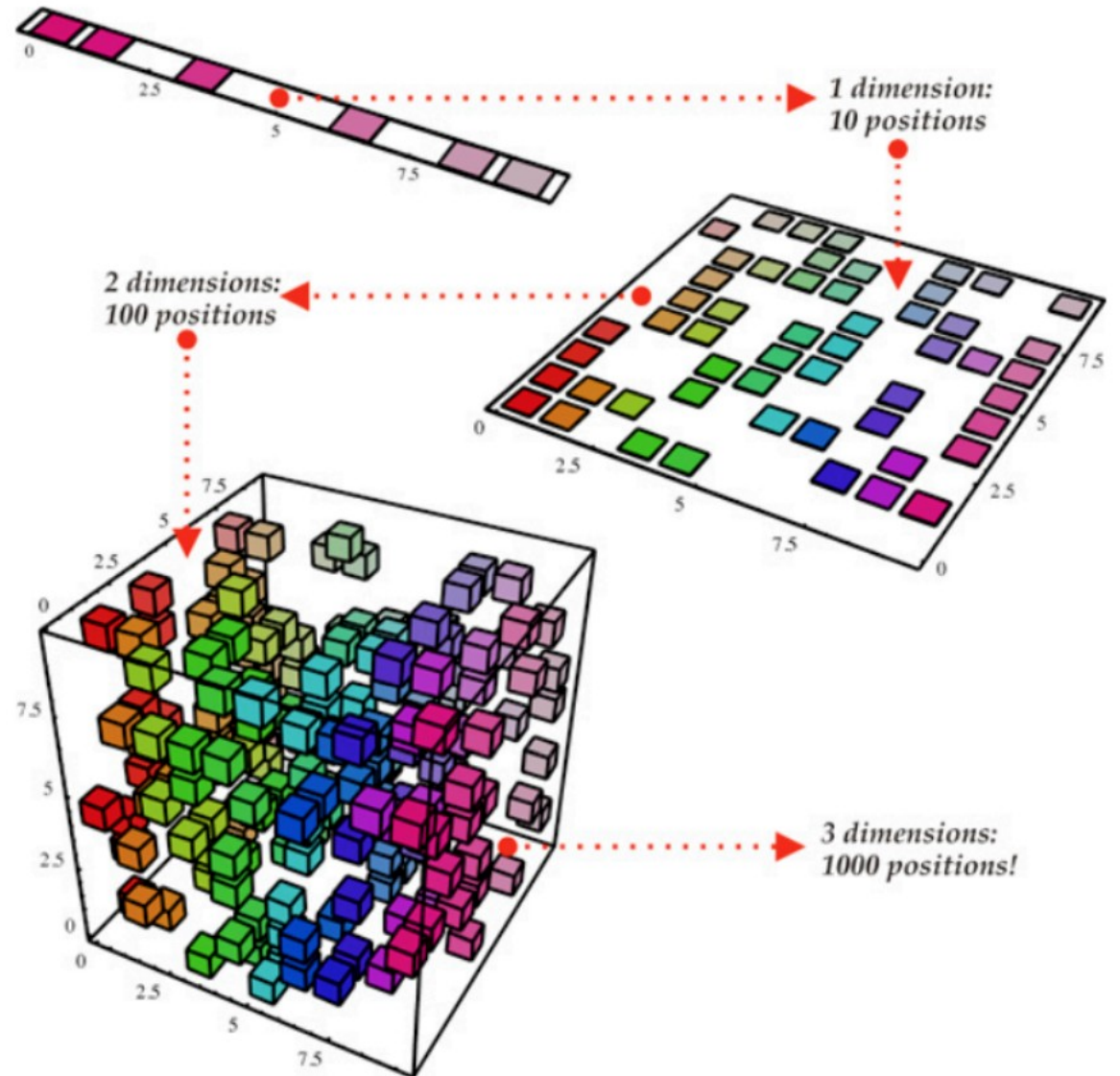
- feature selection;
- feature extraction.

The key is to reduce the number of dimensions, while preserving most parts of the information.



Why is it useful?

1. It reduces the computational requirements;
2. It can unveil data patterns hidden by high dimensionality;
3. Therefore, it help learning processes

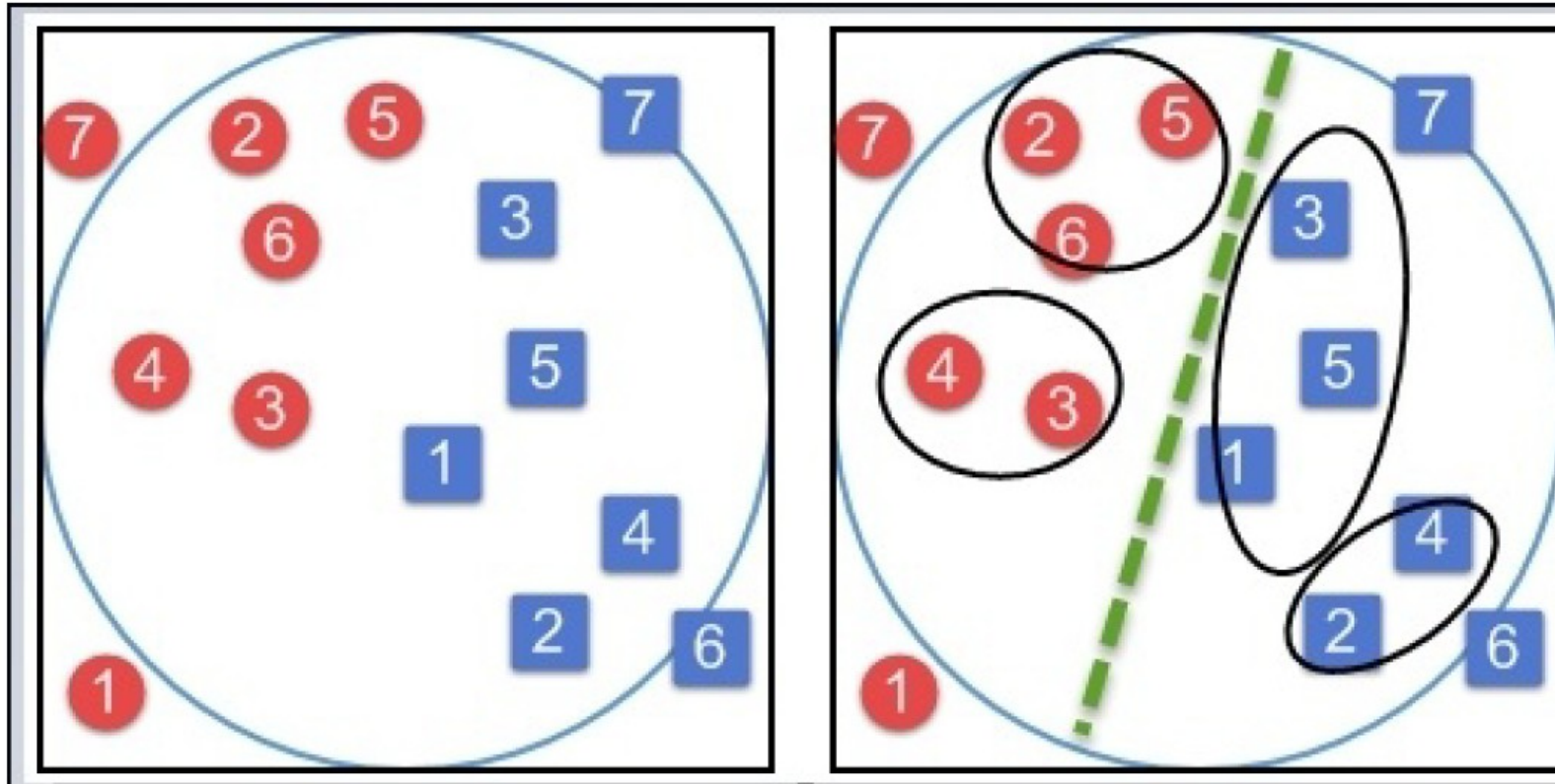


Dimensionality reduction: tips

1. The problem of high dimensionality;
2. Alternatives to mitigate the effects of the curse of dimensionality;
3. Principal Component Analysis (PCA): Concept construction and graphical analysis;
4. Hierarchical clustering on PCA;

The curse of dimensionality

The **curse of dimensionality** is a term used to refer to the phenomena occurring in mathematics and statistics, when analyzing data in multidimensional spaces. When the dimensionality increases, the volume of space increases exponentially, causing the available data to become scattered; this affects its statistical significance. Suppose we have a clustering problem. The observations that do not fall within this circle are like outliers and hard to cluster.

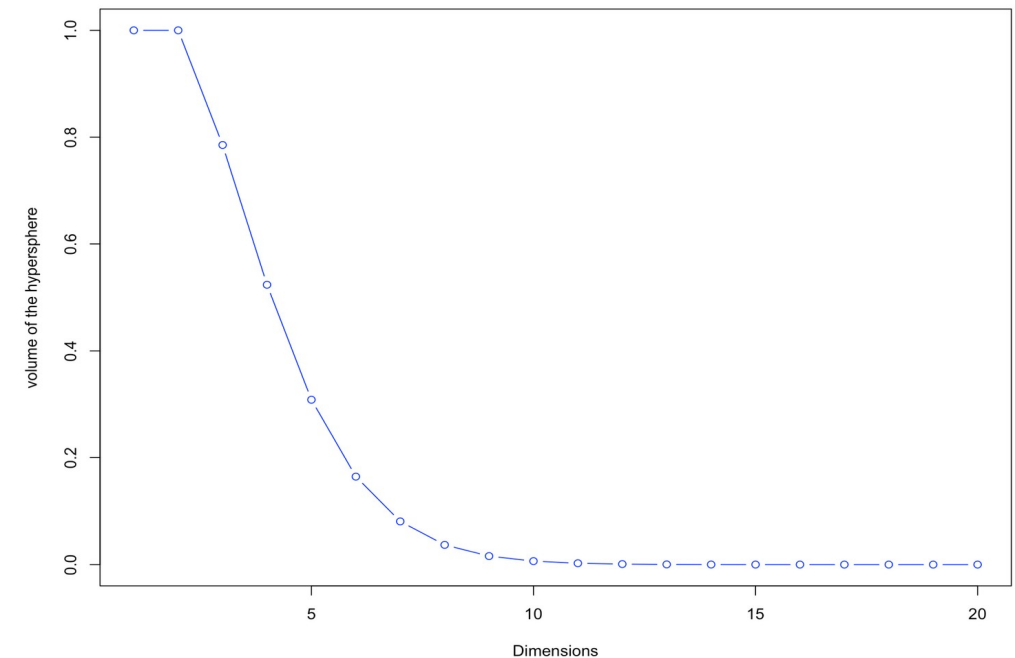
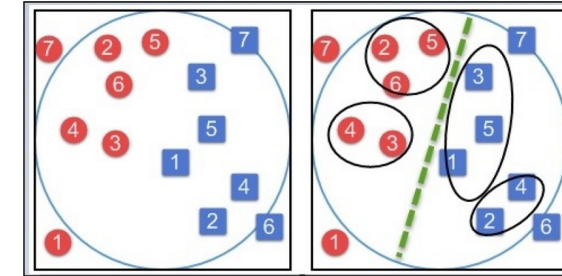


The curse of dimensionality

By increasing the dimension of the square, it becomes an n-dimensional hypercube. Furthermore, the circle becomes an n-dimensional hypersphere.

The hypersphere has a very interesting behavior that helps us better understand the curse of dimensionality. The volume of the inscribed hypersphere of dimension d and with radius 0.5 can be calculated with the following formula:

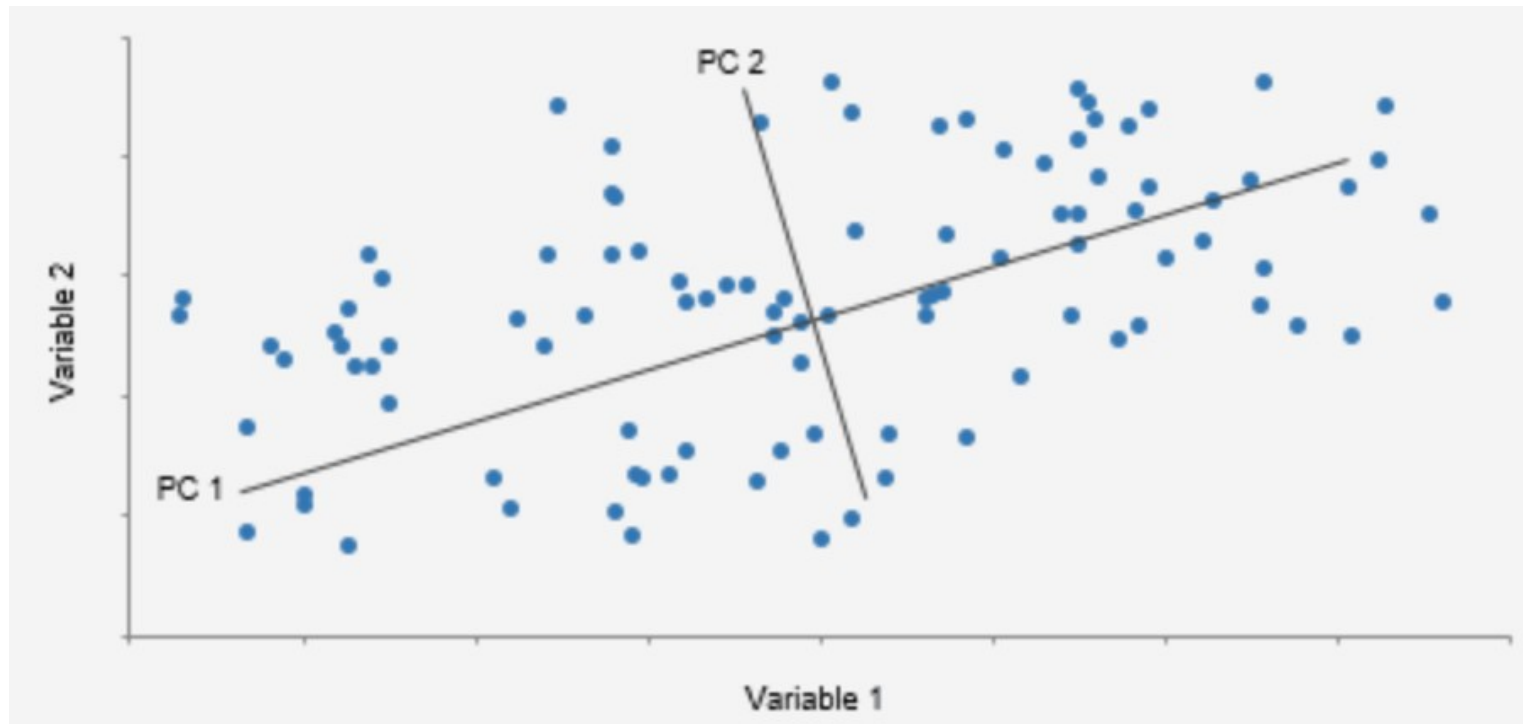
As the dimensionality tends to reach infinity, the hypersphere shrinks to insignificance.



Feature Extraction

Dimensionality reduction includes a set of techniques to deal with the curse of dimensionality. These techniques are aimed at reducing the number of variables to be considered.

In the context of machine learning, the term **feature extraction** is associated with techniques that seek to build a dataset derived and transformed from the original data. One of the best known and most used techniques to reduce the dimensionality is **Principal Components Analysis** or PCA.

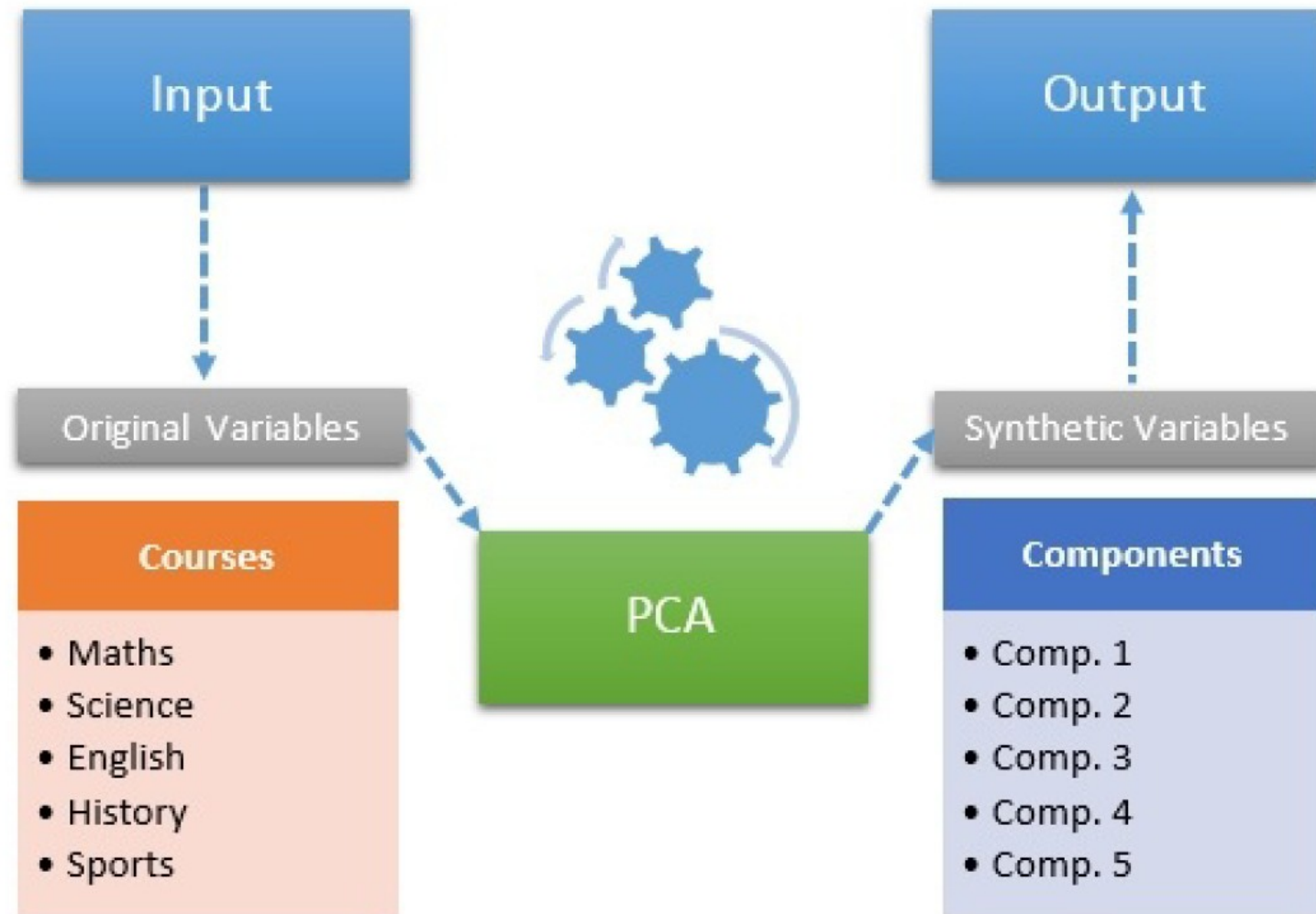


PCA

5 Dimensions

10 Observations

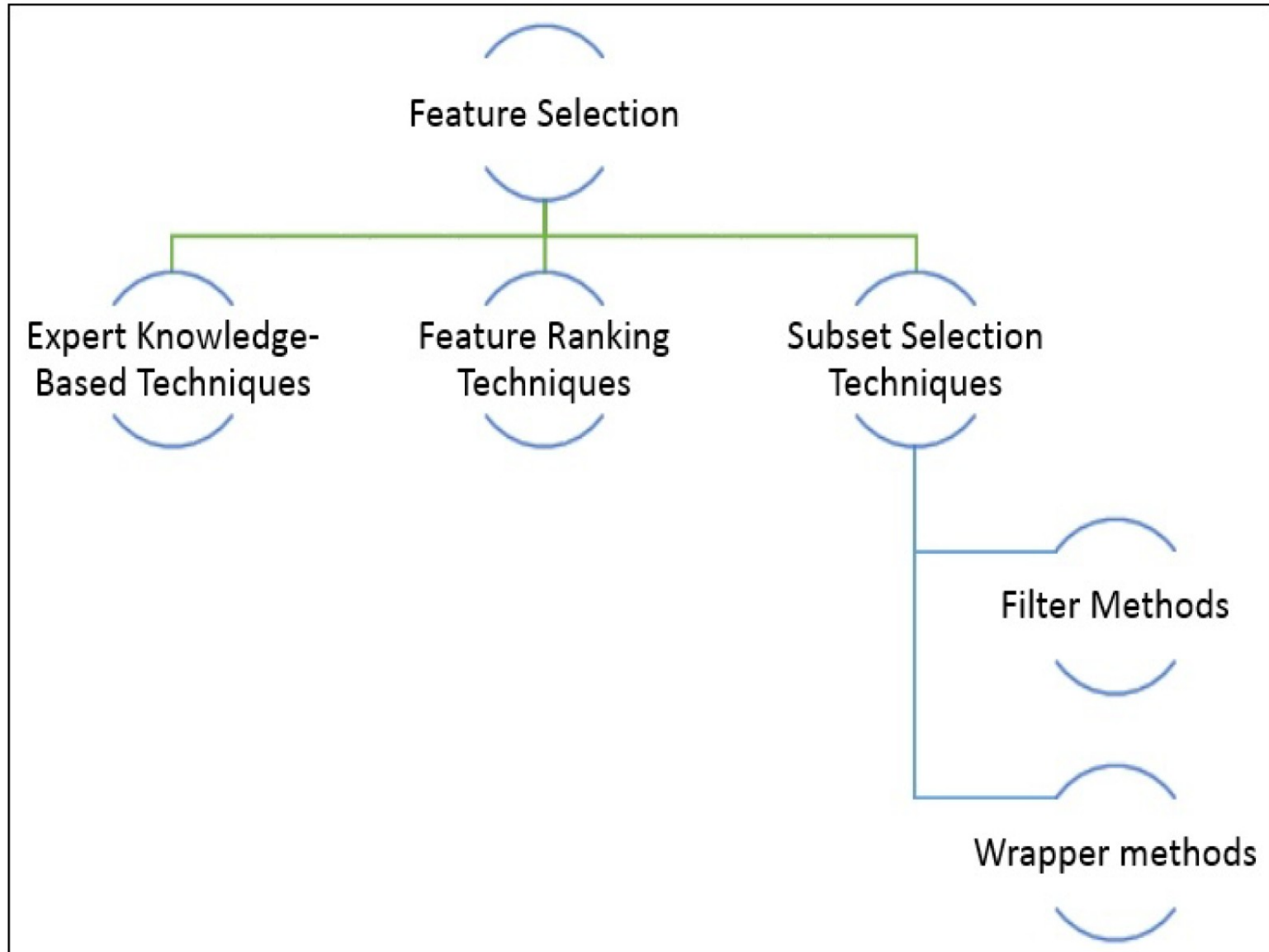
Student	Maths	Science	English	History	Sports
Rosa	7	6,5	9,2	8,6	8
Denis	7,5	9,4	7,3	7	7
Edgar	7,6	9,2	8	8	7,5
Yeison	5	6,5	6,5	7	9
Silvia	6	6	7,8	8,9	7,3
Arturo	7,8	9,6	7,7	8	6,5
Elizabeth	6,3	6,4	8,2	9	7,2
Erik	7,9	9,7	7,5	8	6
Dante	6	6	6,5	5,5	8,7
Sasha	6,8	7,2	8,7	9	7



Feature selection

Feature selection is the process of selecting a subset of relevant features for use in model construction. The most commonly used techniques are:

- Expert Knowledge-based techniques;
- Feature ranking;
- Subset selection;
- Embedded methods;
- Wrapper methods;
- Filter methods.



Feature ranking

Feature ranking is a technique aimed at supporting expert knowledge, and consists of defining a scoring criteria, calculating the performance of each variable individually.

The first step is to proceed to define a qualification criterion for each variable and the number of variables to use.

Subsequently, a validation can be performed to compare the performance of the model with all data versus a model that uses only the «important» variables.

Ranking	Variable Name	Score
01	V07	0,94
02	V14	0,81
03	V04	0,78
04	V05	0,76
05	V03	0,75
06	V11	0,73
07	V01	0,63
08	V02	0,57
09	V10	0,33
10	V13	0,24
11	V08	0,22
12	V06	0,10
13	V12	0,09
14	V09	0,01

Threshold=0.75



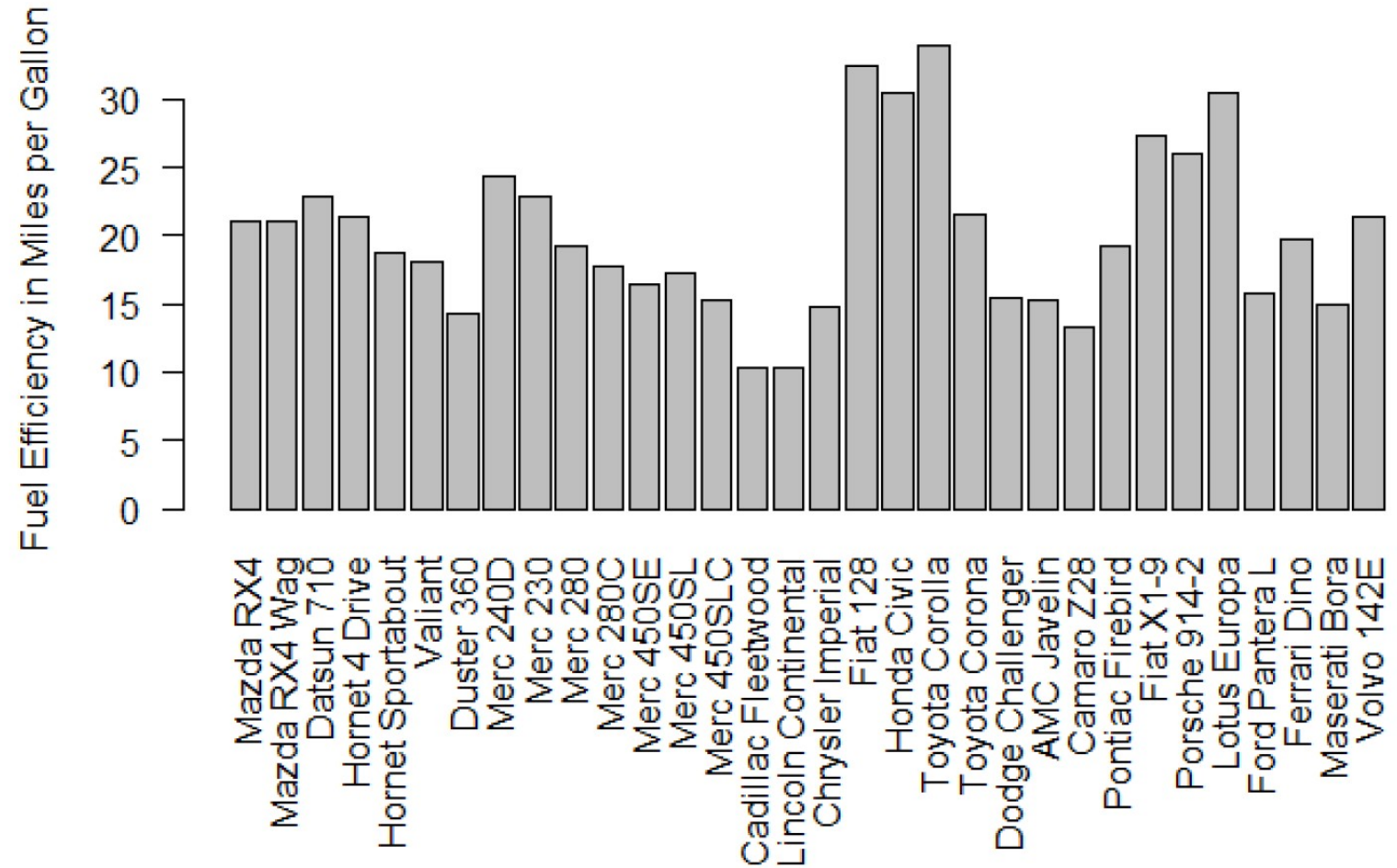
Part 3:

Supervised learning, state-of-the-art classifiers

Reports and predictions

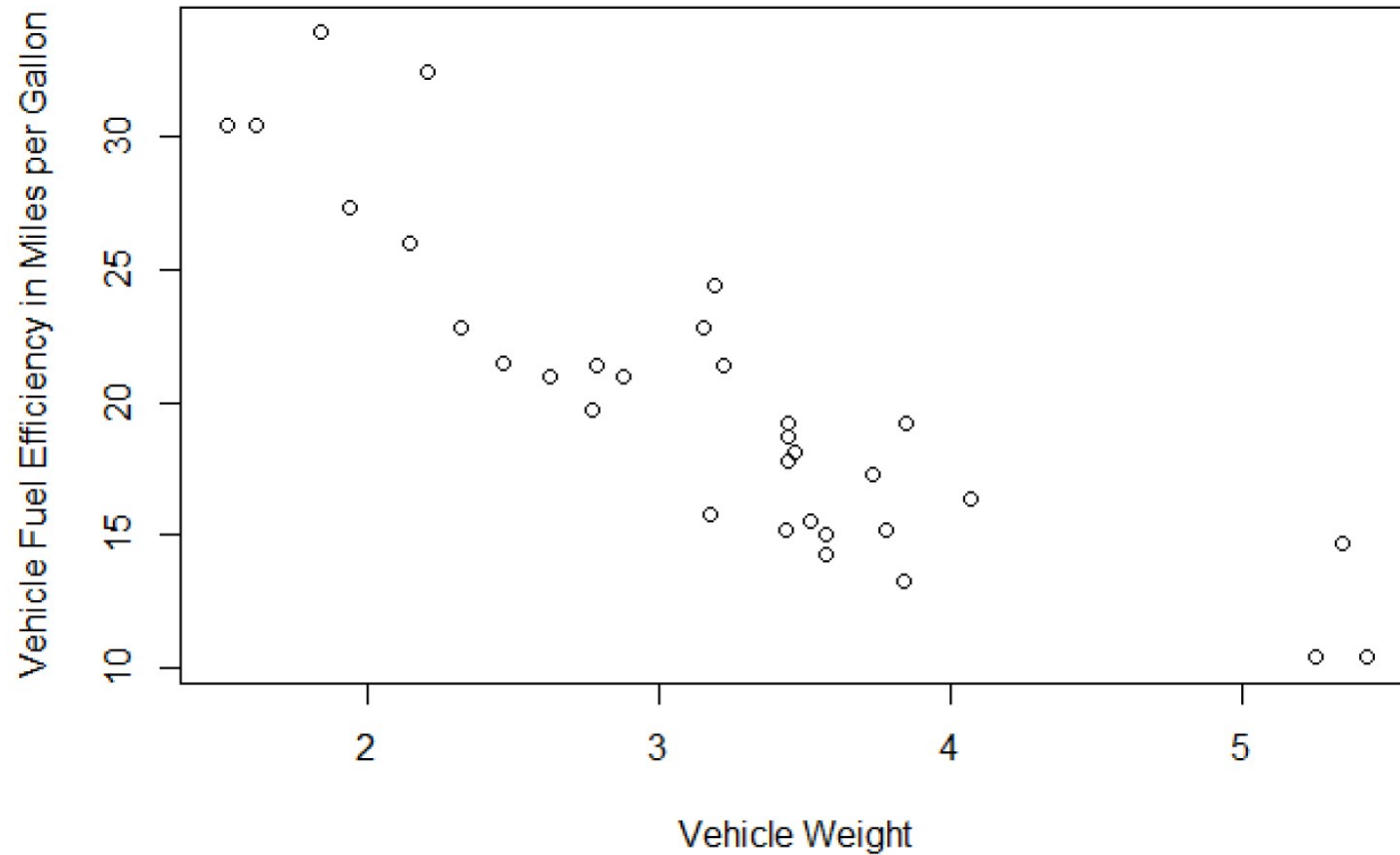
The figure depicts a plot of the mtcars dataset that comes prebuilt with R. It shows a number of cars plotted by their fuel efficiency in miles per gallon.

This **report** isn't very interesting. It doesn't give us any **predictive power**. Seeing how the efficiency of the cars is distributed is nice, but how can we make predictions from it?



Models

This is a model:



Supervised models

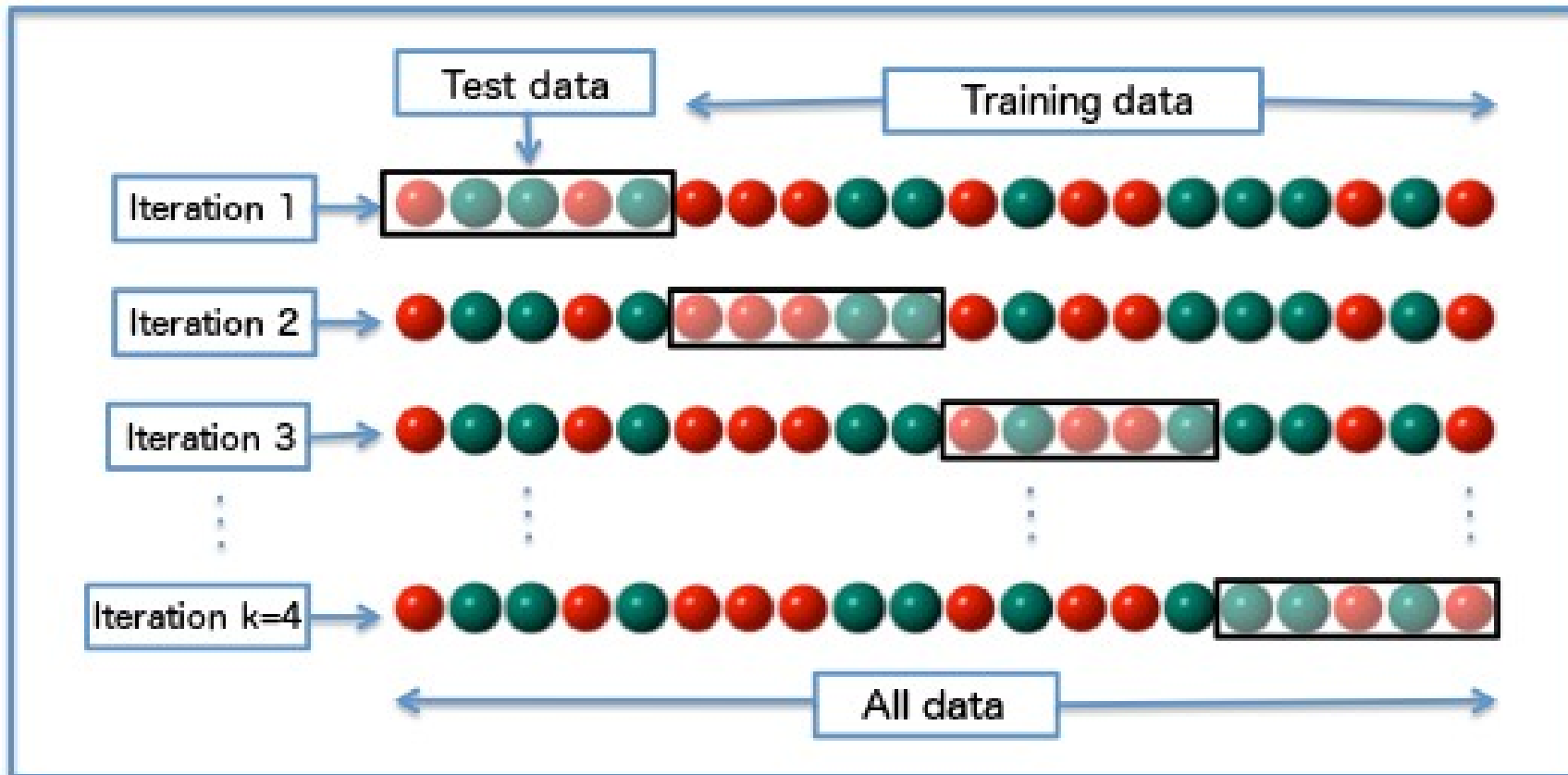
These are three distinct types of supervised models:

- Regression.
These models are very common. They are primarily used for looking at how data evolves with respect to another variable (e.g., time).
- Classification.
These models are used to organize your data into schemes that make categorical sense.
- Mixed.
These models can often rely on parts of regression to inform how to do classification, or sometimes the opposite.

Training and Testing of Data

In the world of statistics, you measure the accuracy of the model by taking a dataset you have and splitting it into training data and test (validation) data.

Cross-validation consists in repeating training-test split and measuring the model accuracy.



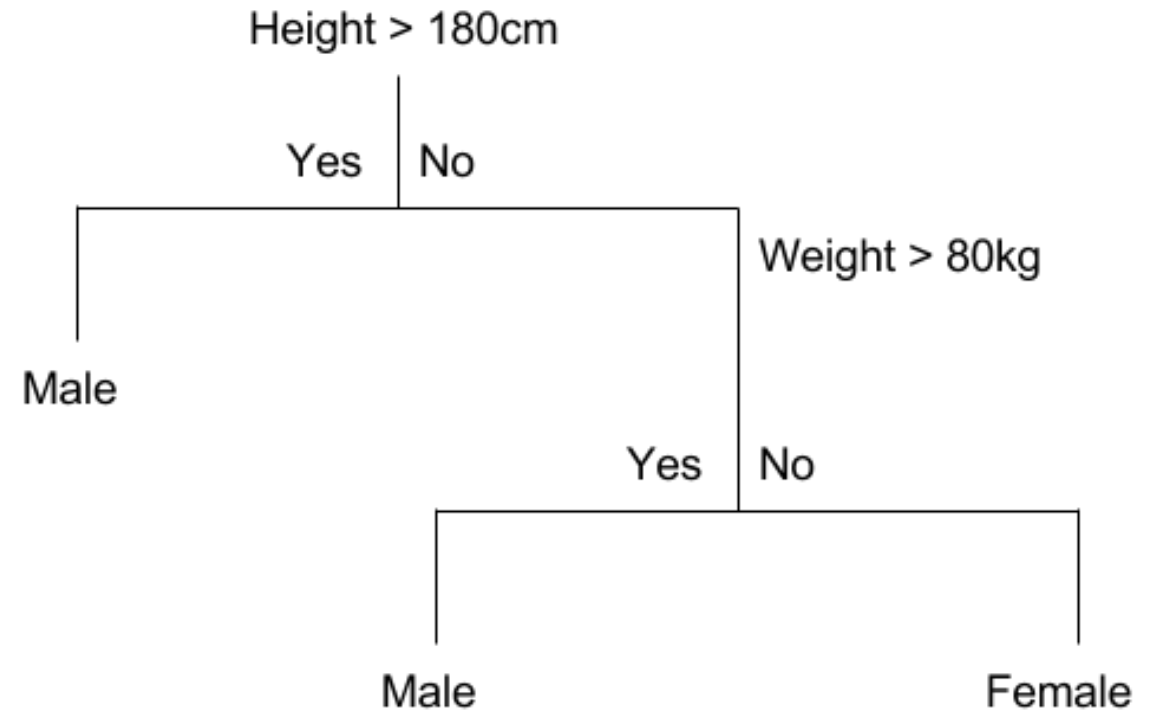
Classification trees

Part of the universe of machine learning models includes **tree-based** methods.

At each node data is splitted according to a particular feature.

The goal is finding the optimal cuts to separate the classes. Trees can also be used for regression.

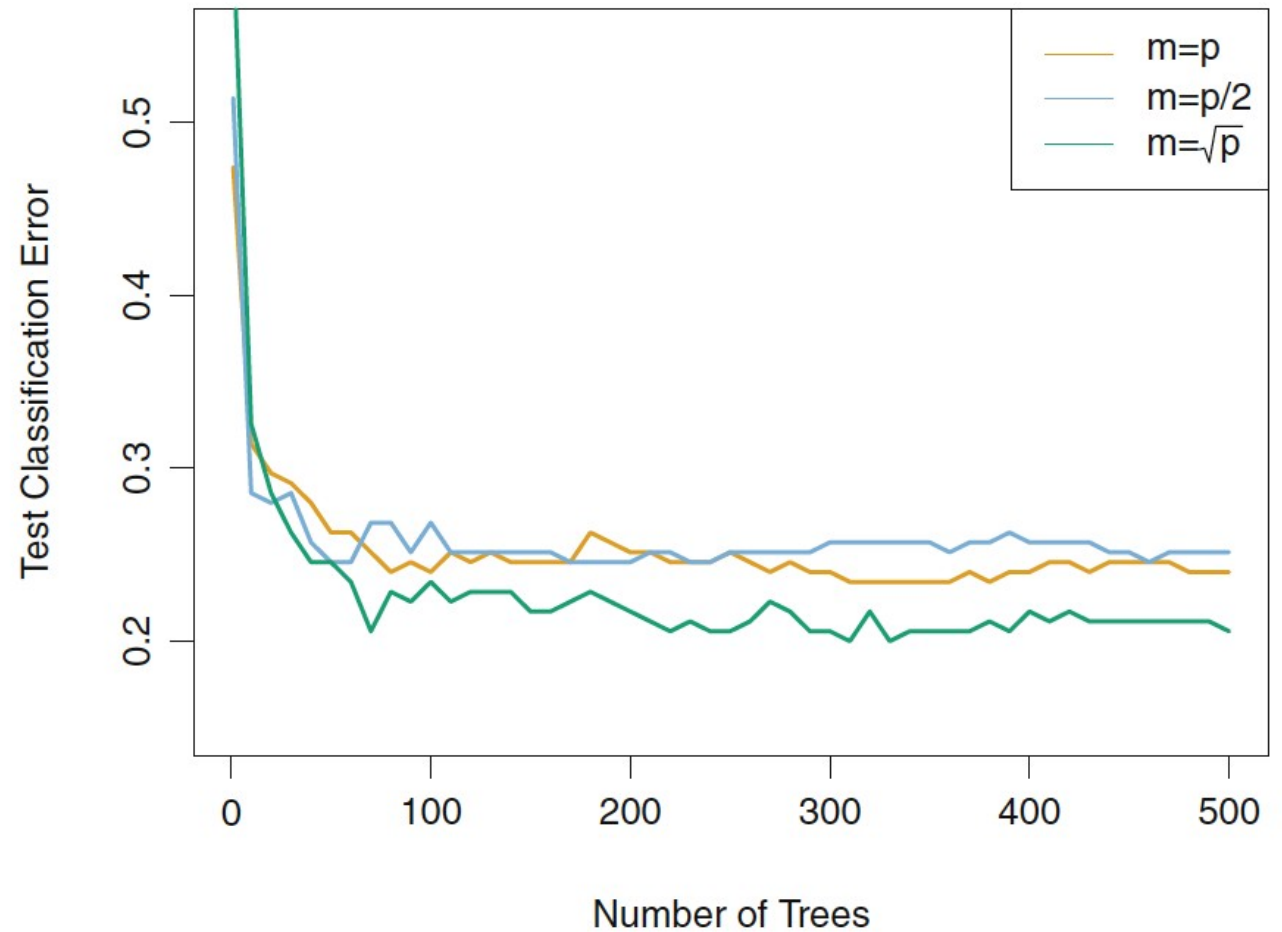
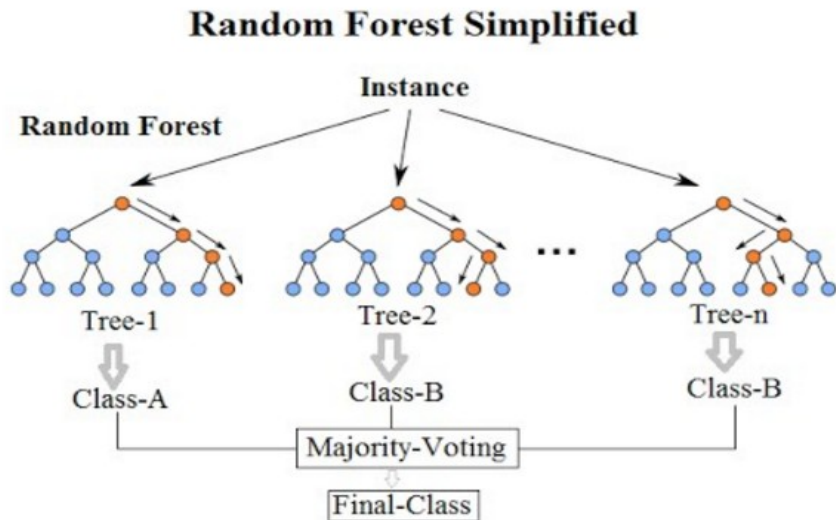
A particular algorithm based on trees is Random Forests.



Random Forests

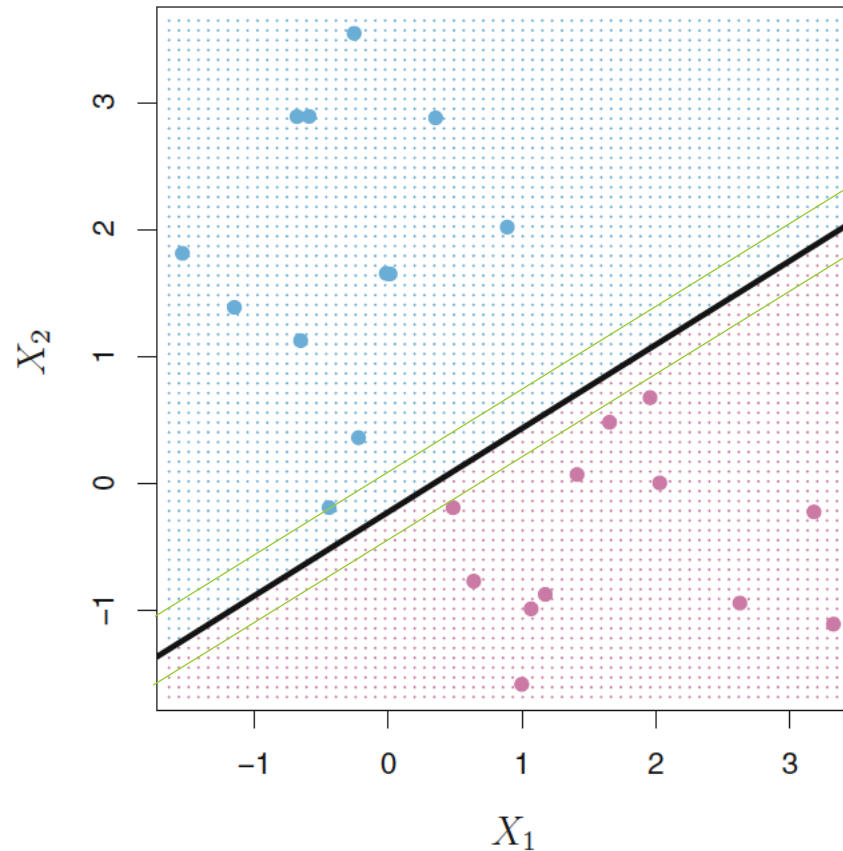
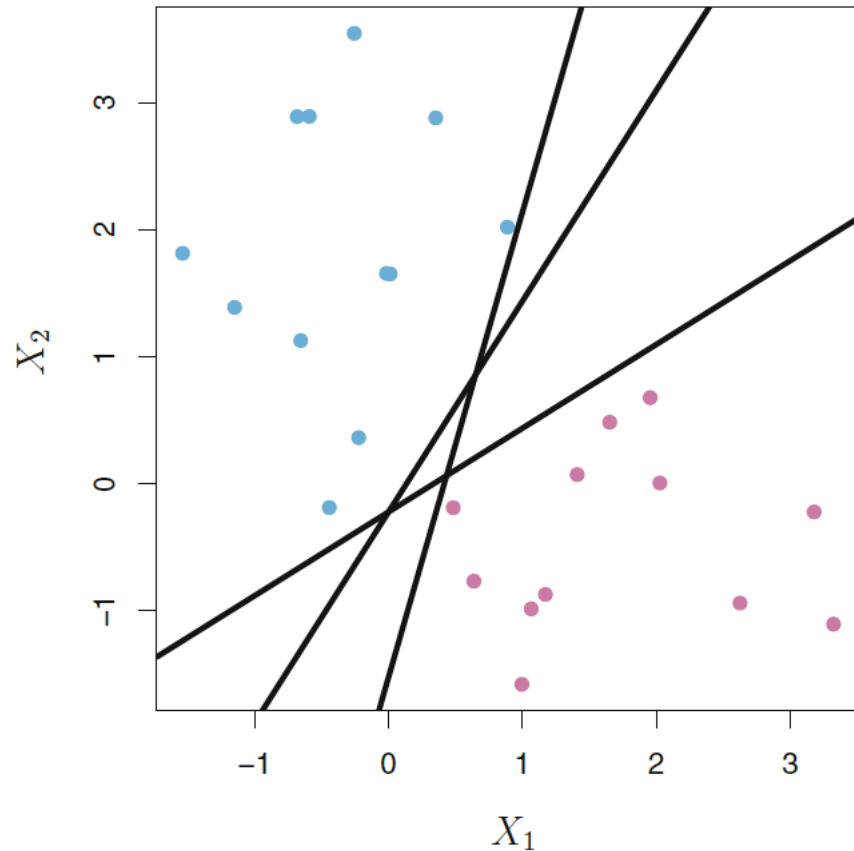
We build a forest of decision trees on bootstrapped training samples.

But when building these decision trees, each time a split in a tree is considered, a random sample of m features is chosen, typically where p is the number of predictors.



Support Vector Machines

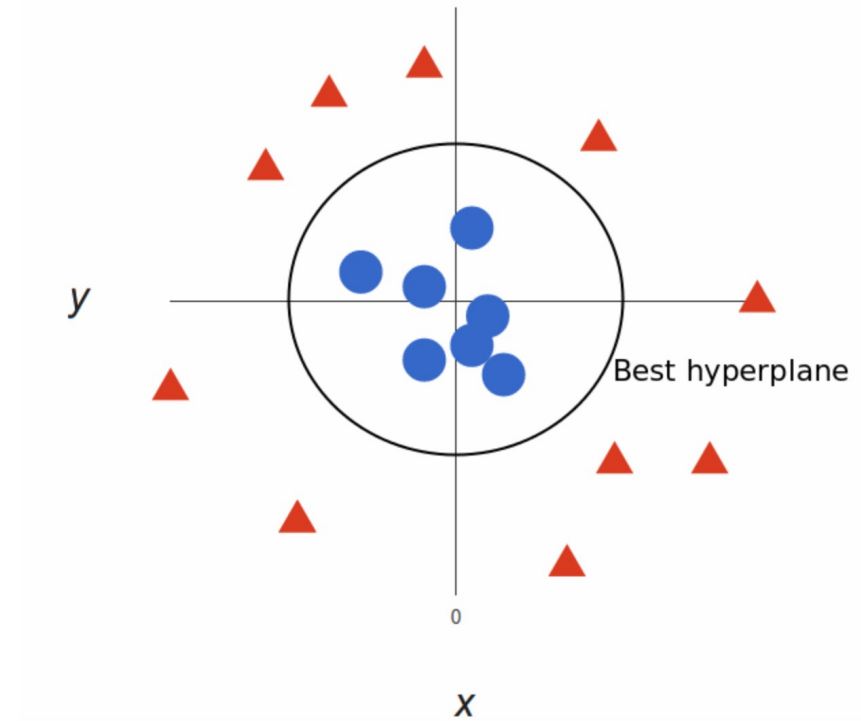
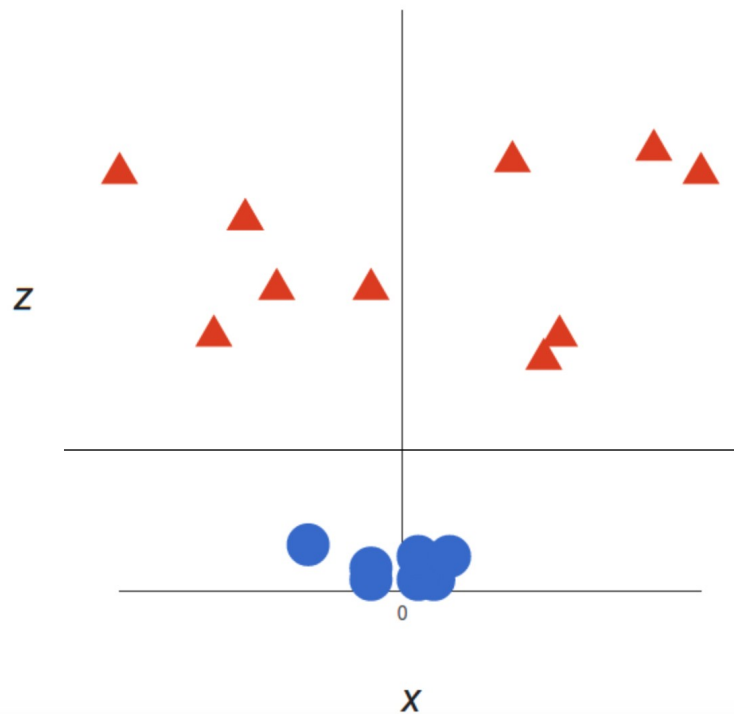
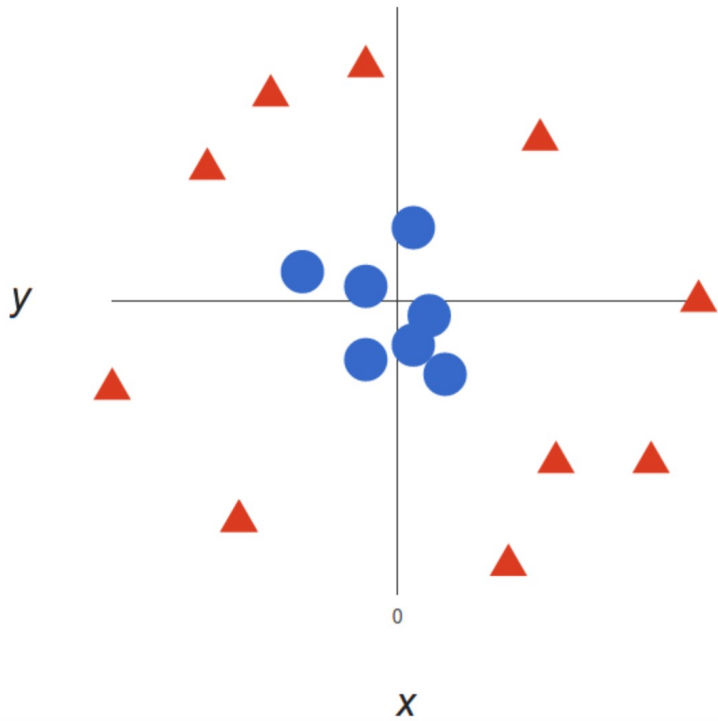
There are two classes of observations, shown in blue and in purple, each of which has measurements on two variables. The maximal margin hyperplane is shown as a solid line. The margin is the distance from the solid line to either of the green lines. Two observations are equidistant from the maximal margin hyperplane and lie along the green lines. These two observations are known as **support vectors**.



Linear vs non-Linear

Sometimes a linear decision boundary does not exist. However, the classes can be clearly segregated. In this case, we can add a third dimension: . The **kernel** trick:

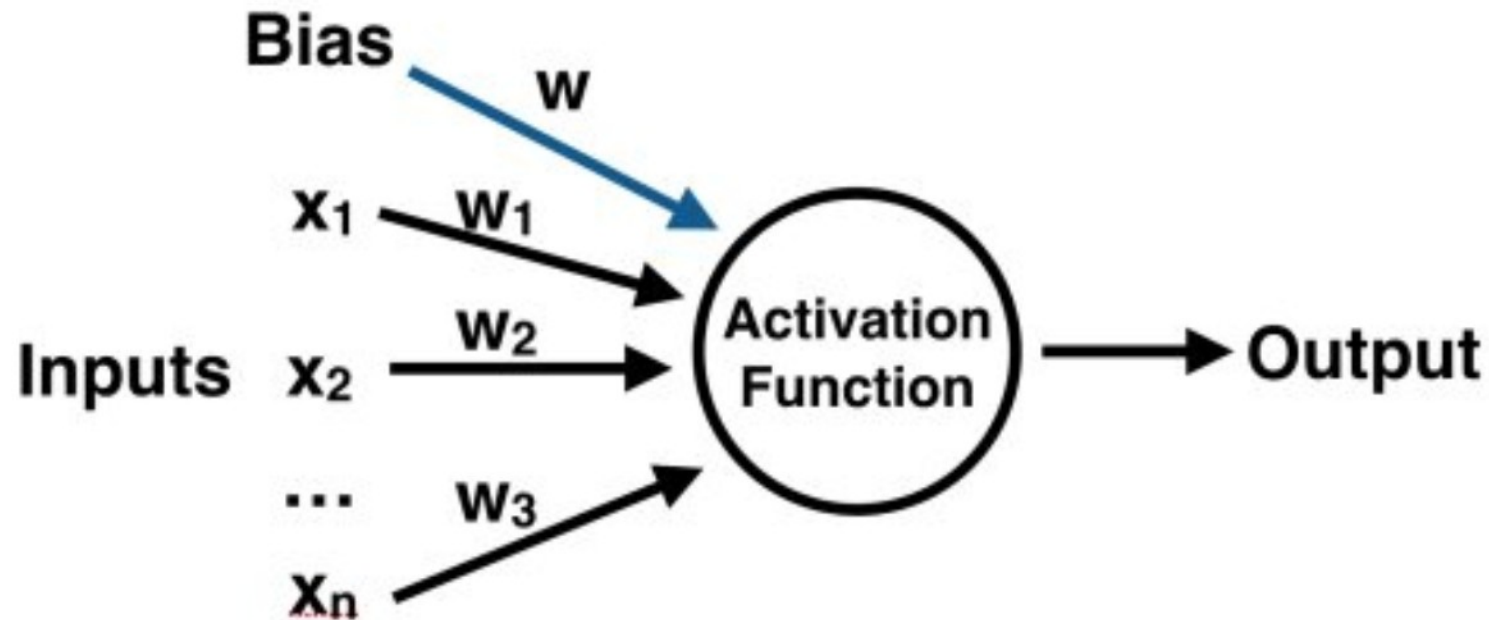
1. Add the new dimension
2. Define the dot product:
3. Tell SVM to use this dot product in the original space.



Neural Networks: the Perceptron

Neural Networks are a machine learning framework that attempts to mimic the learning pattern of natural biological neural networks.

Let's start our discussion by talking about the **Perceptron**. A perceptron has one or more inputs, a bias, an activation function, and a single output. The perceptron receives inputs, multiplies them by some weight, and then passes them into an activation function to produce an output.

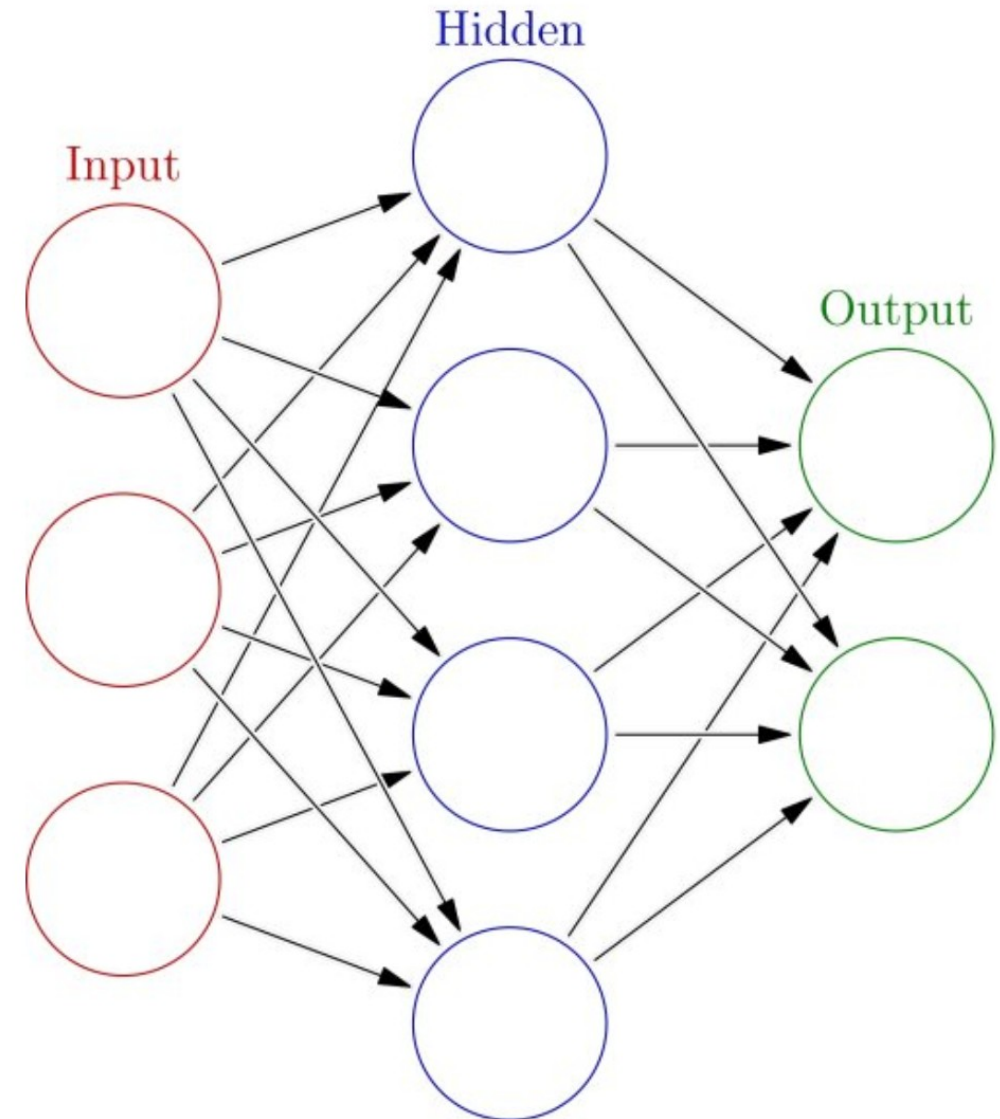


The multilayer Perceptron

To create a neural network, we simply begin to add layers of perceptrons together, creating a **multi-layer perceptron** model of a neural network.

There are:

- an *input layer* which directly takes in your feature inputs;
- an *output layer* which will create the resulting outputs;
- *Hidden layers* which are in between. They are said hidden because they don't directly "see" the feature inputs or outputs.



Neural Networks

Neural networks are a computational model that shares some properties with the animal brain in which many simple units are working in parallel with no centralized control unit.

The **weights** between the units are the primary means of long-term information storage in neural networks. Updating the weights is the primary way the neural network learns new information.

A network's architecture can be defined (in part) by the following:

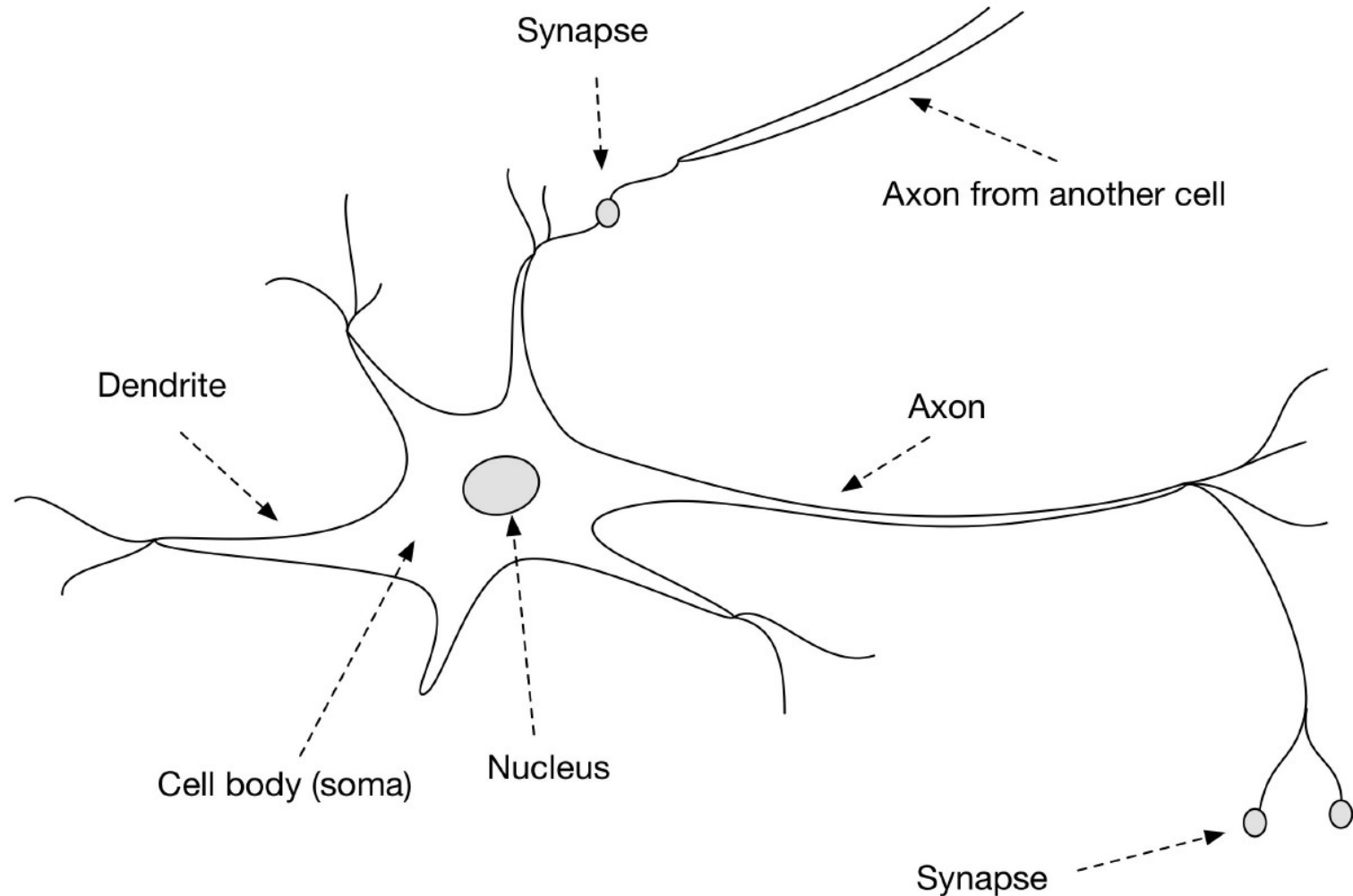
- Number of neurons;
- Number of layers;
- Types of connections between layers.

The Biological Neuron

The **biological neuron** is a nerve cell that provides the fundamental functional unit for the nervous systems of all animals.

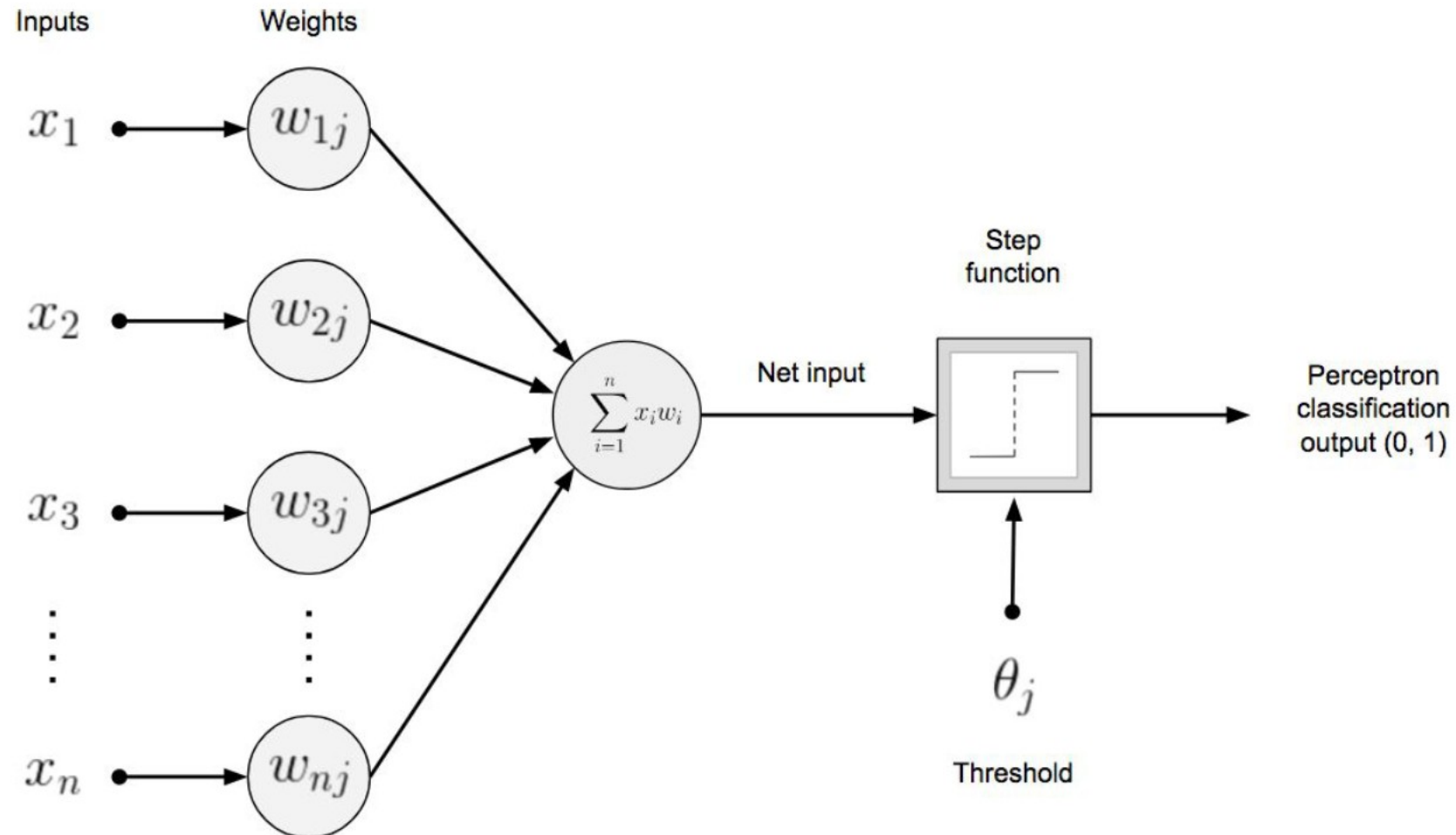
The major parts of the nerve cell are:

- Soma;
- Dendrites;
- Axons;
- Synapses.



The Perceptron

The **perceptron** is a linear model used for binary classification. In the field of neural networks the perceptron is considered an artificial neuron using the Heaviside step function for the activation function.



The perceptron learning algorithm

The **perceptron learning** algorithm changes the weights in the perceptron model until all input records are all correctly classified.

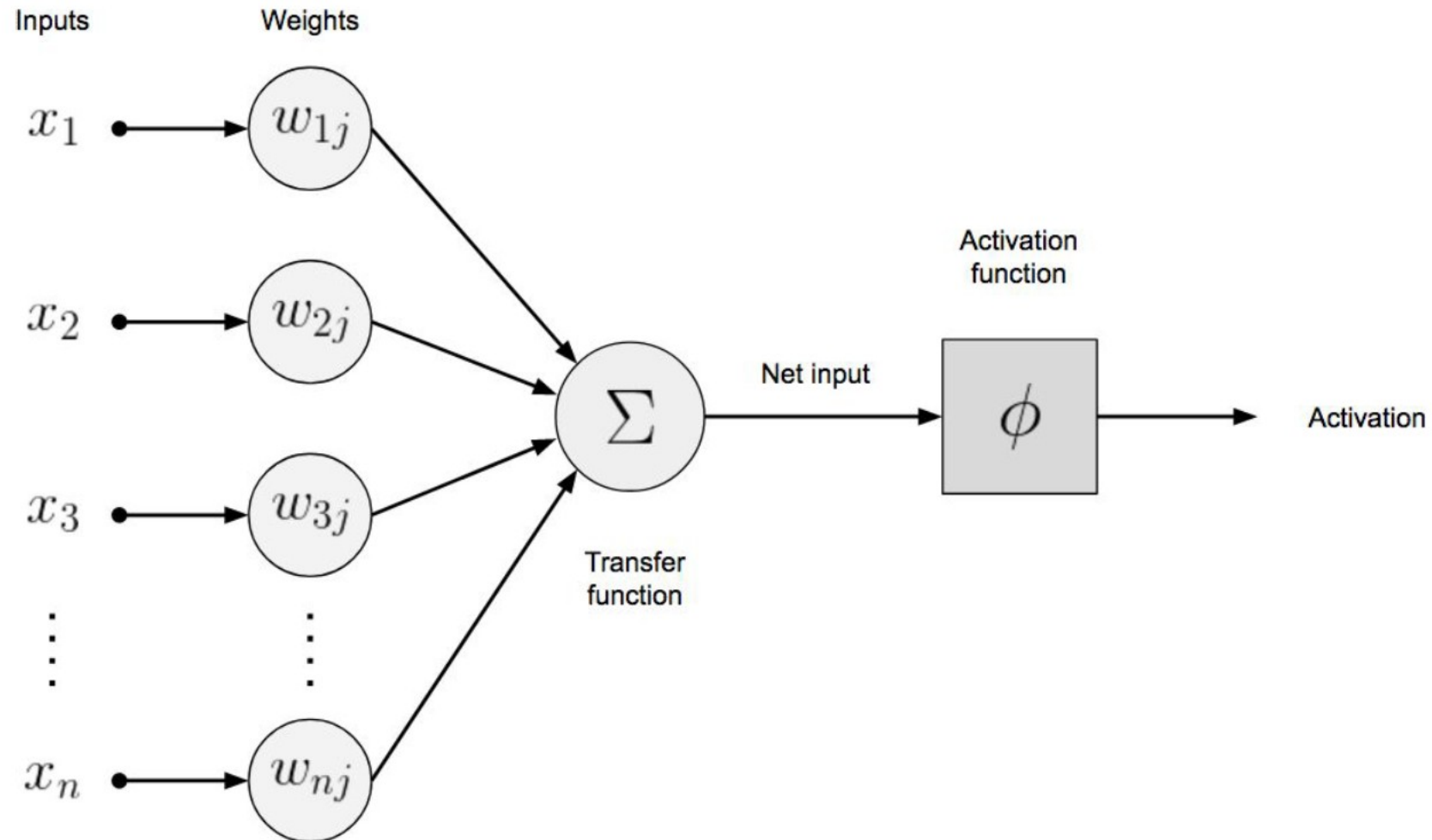
The perceptron learning algorithm initializes the weight vector with small random values. The perceptron learning algorithm takes each input record and computes the output classification. If the classification is correct, no weight changes are made. If the classification is incorrect, the weights are adjusted accordingly.

This loop continues until all of the input examples are correctly classified. If the dataset is not linearly separable, the training algorithm will not terminate. An example is the XOR logic function.

x_0	x_1	y
0	0	0
0	1	1
1	0	1
1	1	0

Multilayer Feed-Forward Networks

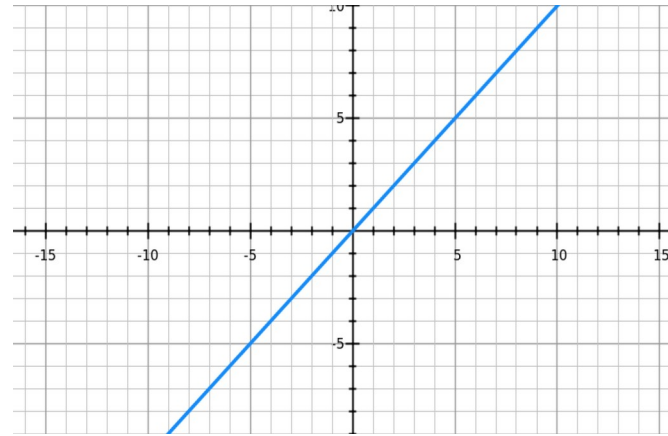
The **multilayer feed-forward network** is a neural network with an input layer, one or hidden layers, and an output layer.



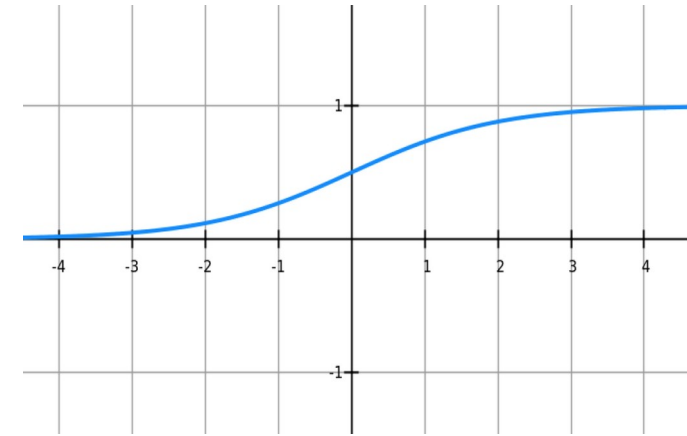
Activation Functions

We use **activation functions** to propagate the output of one layer's nodes forward to the next layer. Activation functions are a scalar functions, yielding neuron's activation. The basic types are:

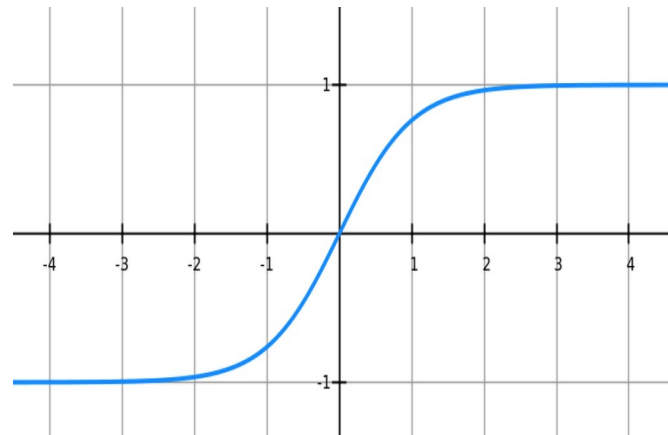
- Linear (a);
- Sigmoid (b);
- Tanh (c);
- Rectified Linear (d).



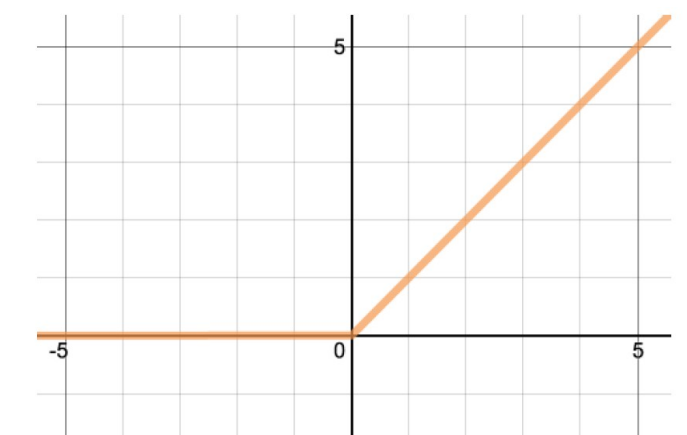
a




b



c



d



Part 4: Deep learning

Defining Deep Learning

Deep neural networks are multi-layer networks with:

- More neurons than previous networks;
- More complex connections;
- Huge computational requirements;

In one of four fundamental architectures:

- Unsupervised pretrained networks;
- Convolutional neural networks;
- Recurrent neural networks;
- Recursive neural networks.

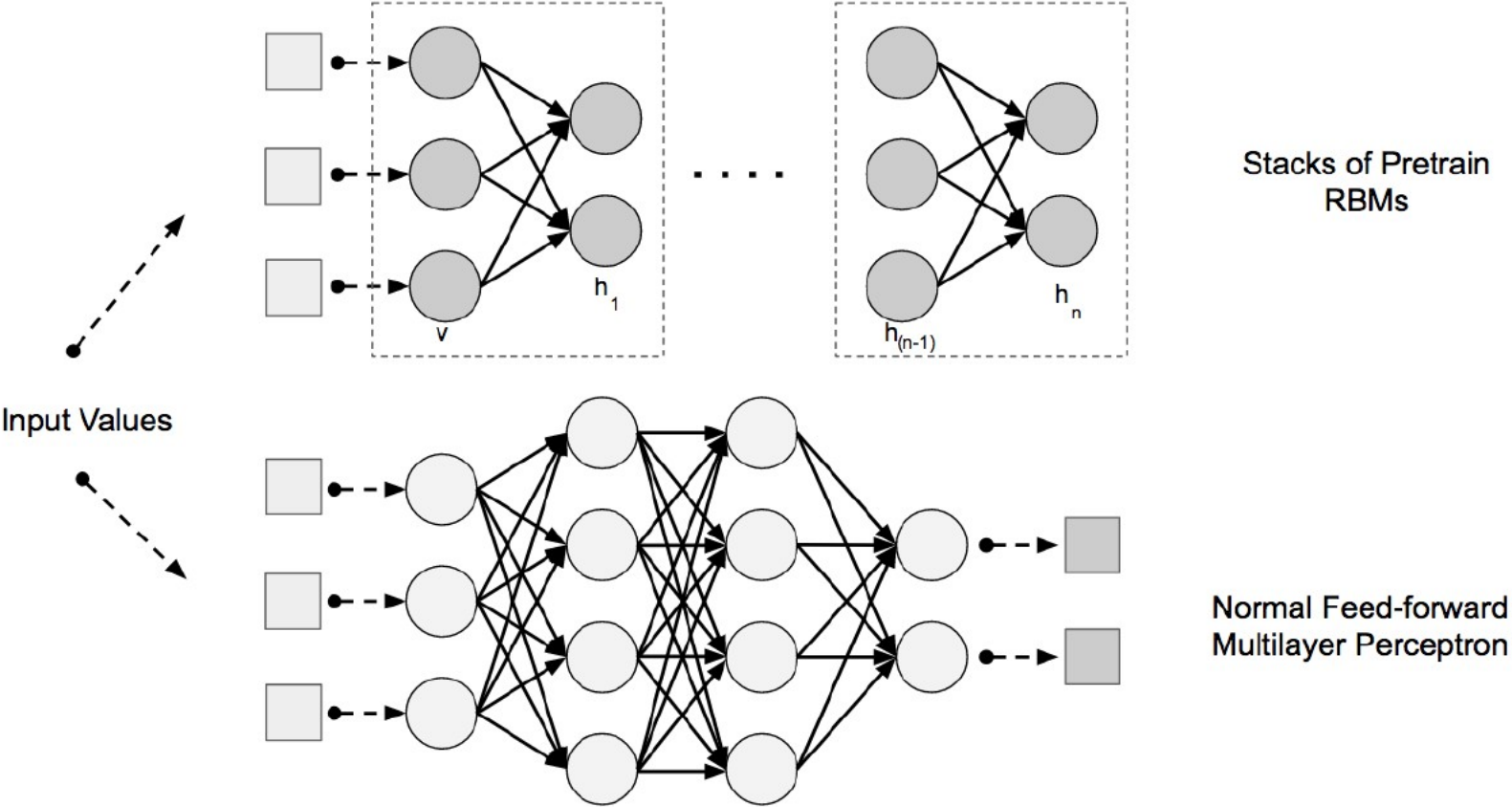
Deep Learning records

Here are just a few of the benchmark records deep learning has achieved in the last few years:

- Text-to-speech synthesis (Fan et al., Microsoft, Interspeech 2014);
- Language identification (Gonzalez-Dominguez et al., Google, Interspeech 2014);
- Large vocabulary speech recognition (Sak et al., Google, Interspeech 2014);
- English-to-French translation (Sutskever et al., Google, NIPS 2014);
- Social signal classification (Brueckner & Schuler, ICASSP 2014);
- Optical character recognition (Breuel et al., ICDAR 2013);
- Image caption generation (Vinyals et al., Google, 2014);
- Video-to-textual description (Donahue et al., 2014);
- Syntactic parsing for natural language processing (Vinyals et al., Google, 2014);
- Photo-real talking heads (Soong and Wang, Microsoft, 2014)

Major Architectures of Deep Networks

Deep Belief Networks are composed of layers of Restricted Boltzmann Machines (RBMs) for the pretrain phase and then a feed-forward network for the fine-tune phase.

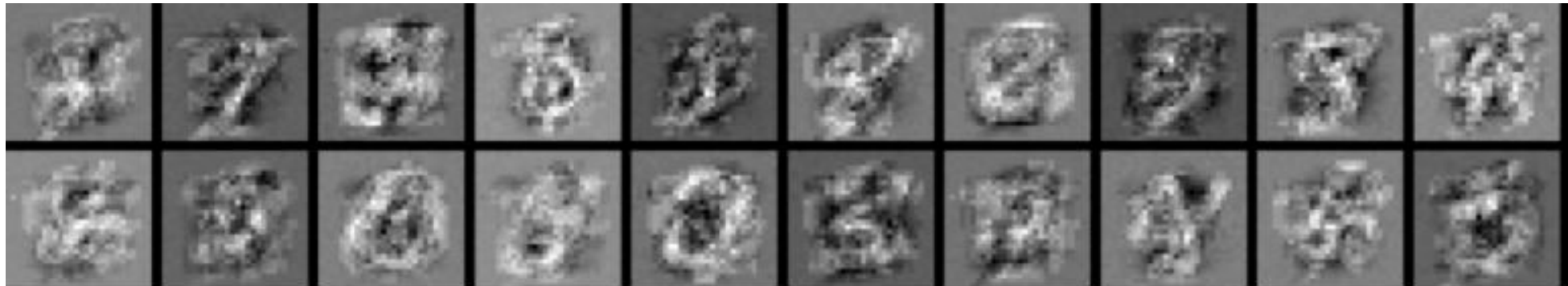
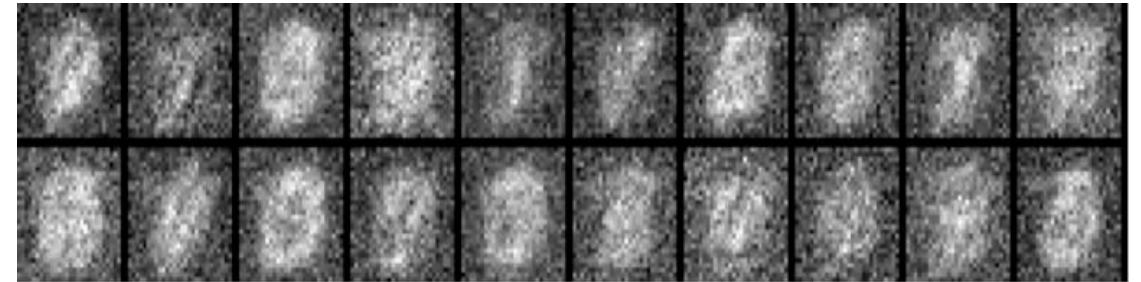


Learning and training

Activation render at the beginning of training



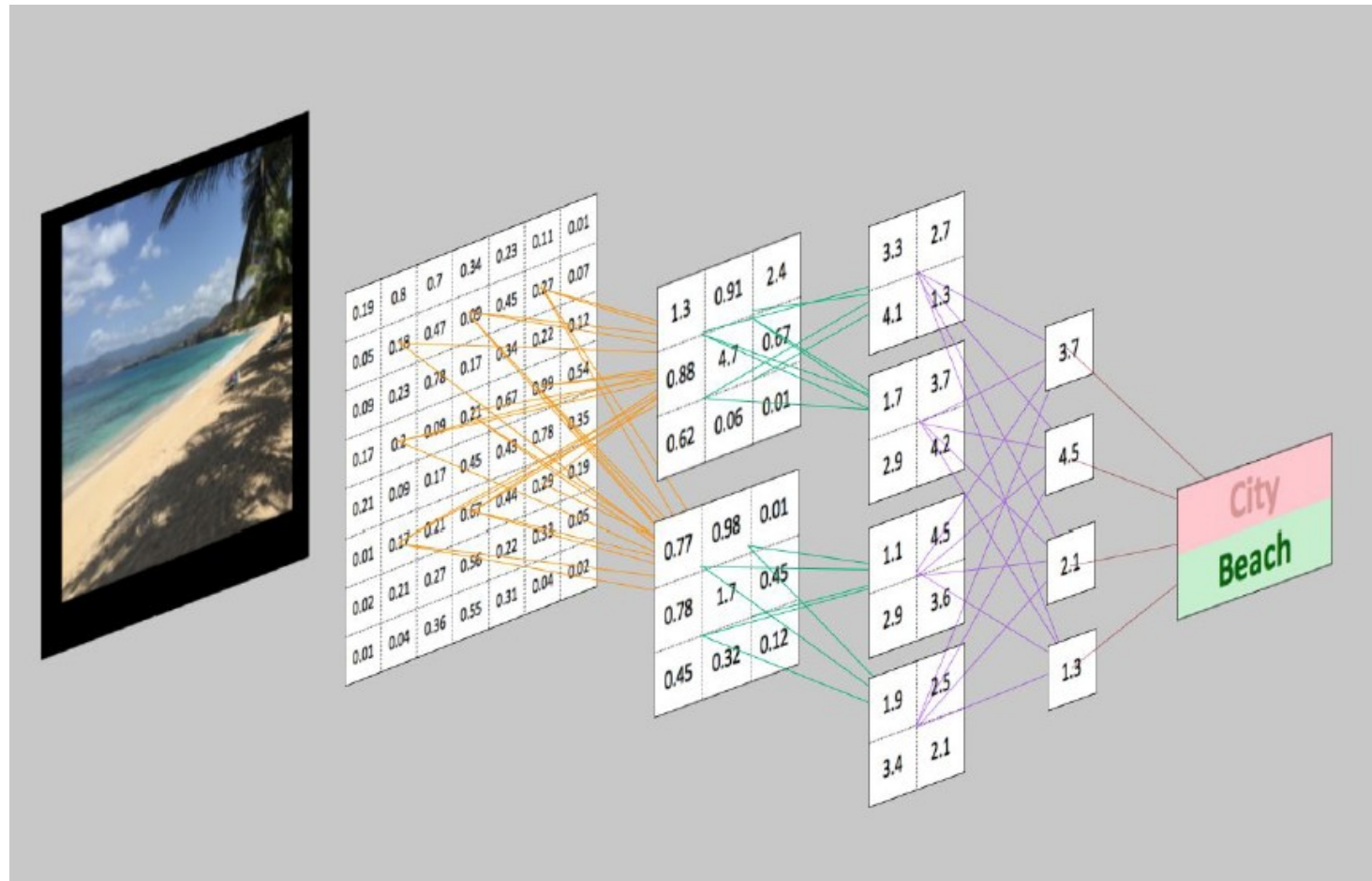
Features emerge in later activation render



Portions of MNIST digits emerge towards end of training

Convolutional Neural Networks

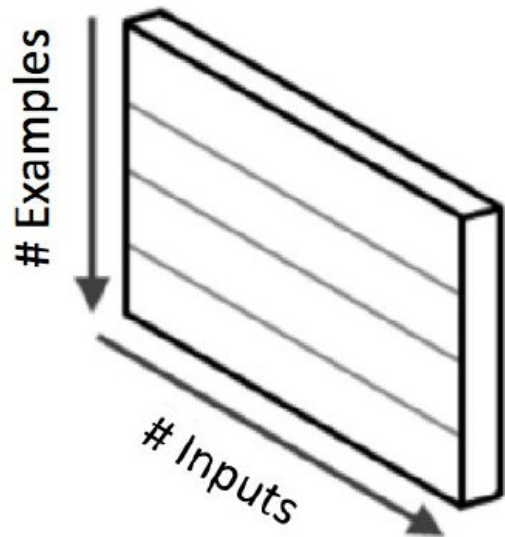
The goal of a **CNN** is to learn higher-order features in the data via convolutions. They are well suited to object recognition with images and consistently top image classification competitions.



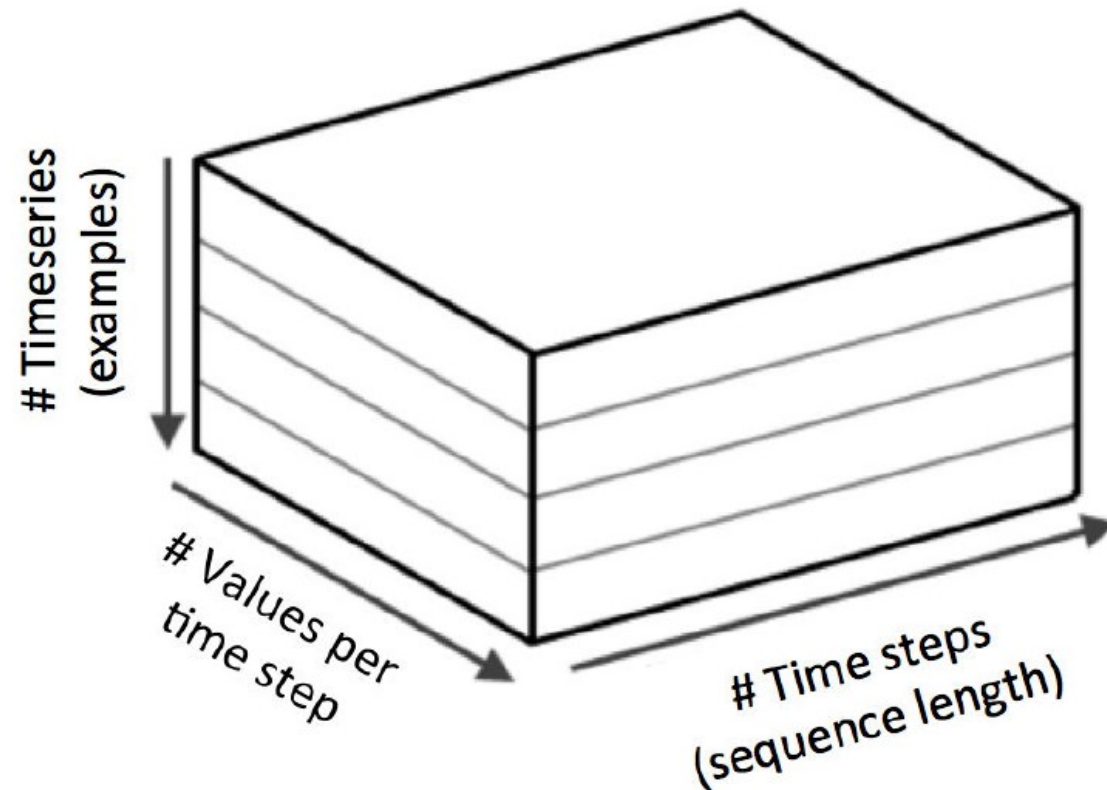
Recurrent Neural Networks

Recurrent Neural Networks are in the family of feed-forward neural networks. They are different from other feed-forward networks in their ability to send information over time-steps.

Feed-Forward Network Data



Recurrent Network Data



Recursive Neural Networks

Recursive Neural Networks, like Recurrent Neural Networks, can deal with variable length input.

The primary difference is that Recurrent Neural Networks have the ability to model the hierarchial structures in the training dataset. Images commonly have a scene composed of many objects. Deconstructing scenes is often a problem domain of interest yet is nontrivial.

An RNN architecture is composed of a **shared-weight matrix** and a **binary tree** that allows the RNN to learn varying sequences of patterns of an image.

