Machine Learning Algorithms II

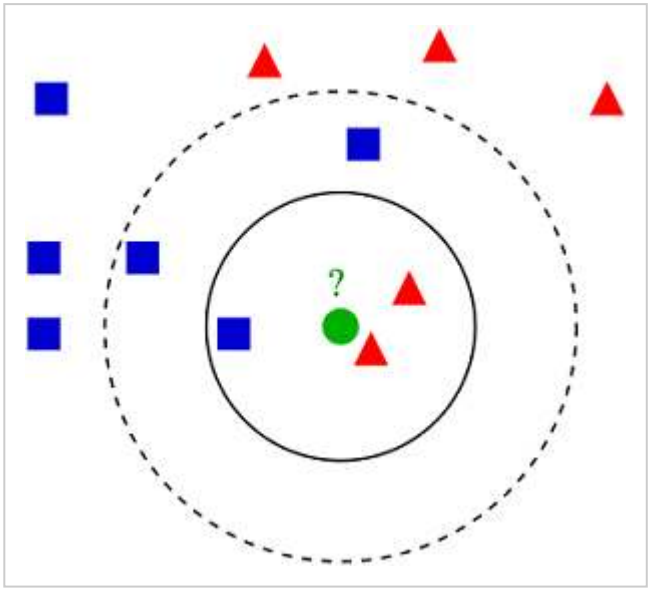
WE'LL COVER THE FOLLOWING

- 6. K-Nearest Neighbors (KNN)
- 7. K-Means
- 8. Random Forest
- 9. Dimensionality Reduction
- 10. Artificial Neural Networks (ANN)
- Final Thoughts

6. K-Nearest Neighbors (KNN)

KNN algorithm is a very simple and popular technique. It is based on the following idea from real life: ***You are the average of the five people you most associate with!***

KNN classifies an object by searching through the entire training set for the k most similar instances, the k neighbors, and assigning a common output variable to all those k instances. The figure below represents a classification example. The test sample (green dot) should be classified either to blue squares or to red triangles. If $k = 3$ (solid line circle) it is assigned to the red triangles because there are 2 triangles and only 1 square inside the inner circle. If $k = 5$ (dashed line circle) it is assigned to the blue squares (3 squares vs. 2 triangles inside the outer circle):



Example of k-NN classification

The selection of *k* is critical here; a small value can result in a lot of noise and inaccurate results, while a large value is not feasible and defeats the purpose of the algorithm.

Although mostly used for classification, this technique can also be used for regression problems. For example, when dealing with a regression task, the output variable can be the mean of the *k* instances, while for classification problems this is often the mode class value.

The distance functions for assessing similarity between instances can be Euclidean, Manhattan, or Minkowski distance. ***Euclidean distance***, the most commonly used one, is simply an ordinary straight-line distance between two points. To be specific, it is the square root of the sum of the squares of the differences between the coordinates of the points.

7. K-Means

K-means is a type of unsupervised algorithm for data clustering. It follows a simple procedure to classify a given data set. It tries to find *K* number of clusters or groups in the dataset. Since we are dealing with unsupervised learning, all we have is our training data *X* and the number of clusters, *K*, that we want to identify, but no labelled training instances (i.e., no data with known final output category that we could use to train our model). For example, K-Means could be used to segment users into *K* groups based on their purchase history.

their purchase history.

The algorithm iteratively assigns each data point to one of the *K* groups based on their features. Initially, it picks *k* points for each of the K-clusters, known as the *centroid*. A new data point is put into the cluster having the closest centroid based on feature similarity. As new elements are added to the cluster, the cluster centroid is re-computed and keeps changing. The new centroid becomes the average location of all the data points currently in the cluster. This process is continued iteratively until the centroids stop changing. At the end, each centroid is a collection of feature values that define the resulting group.

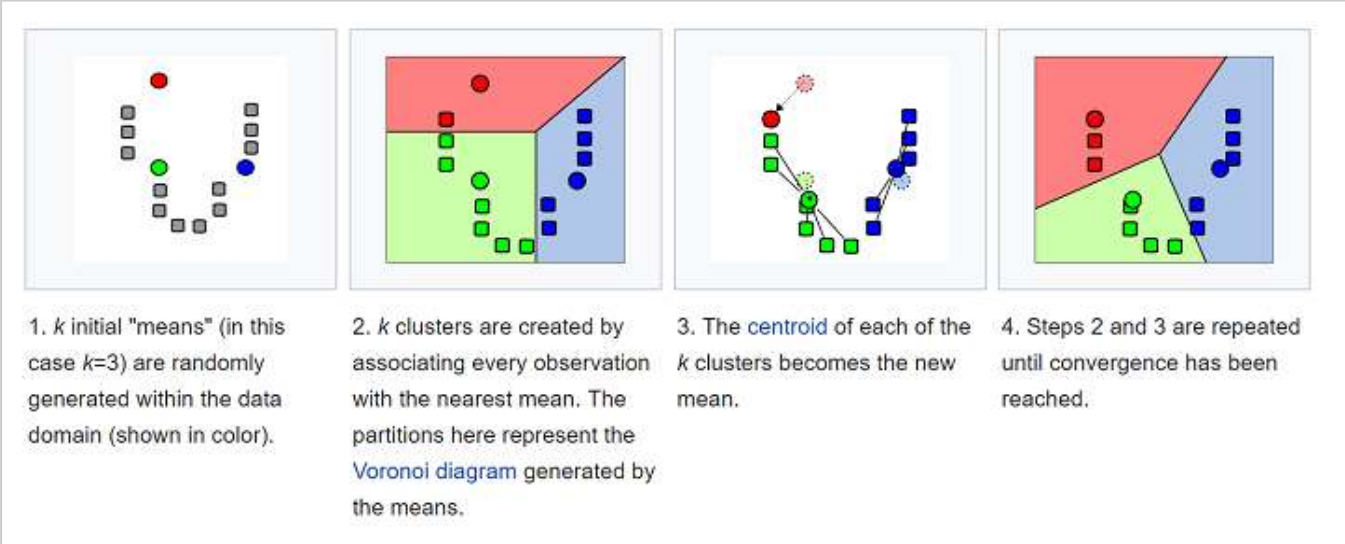


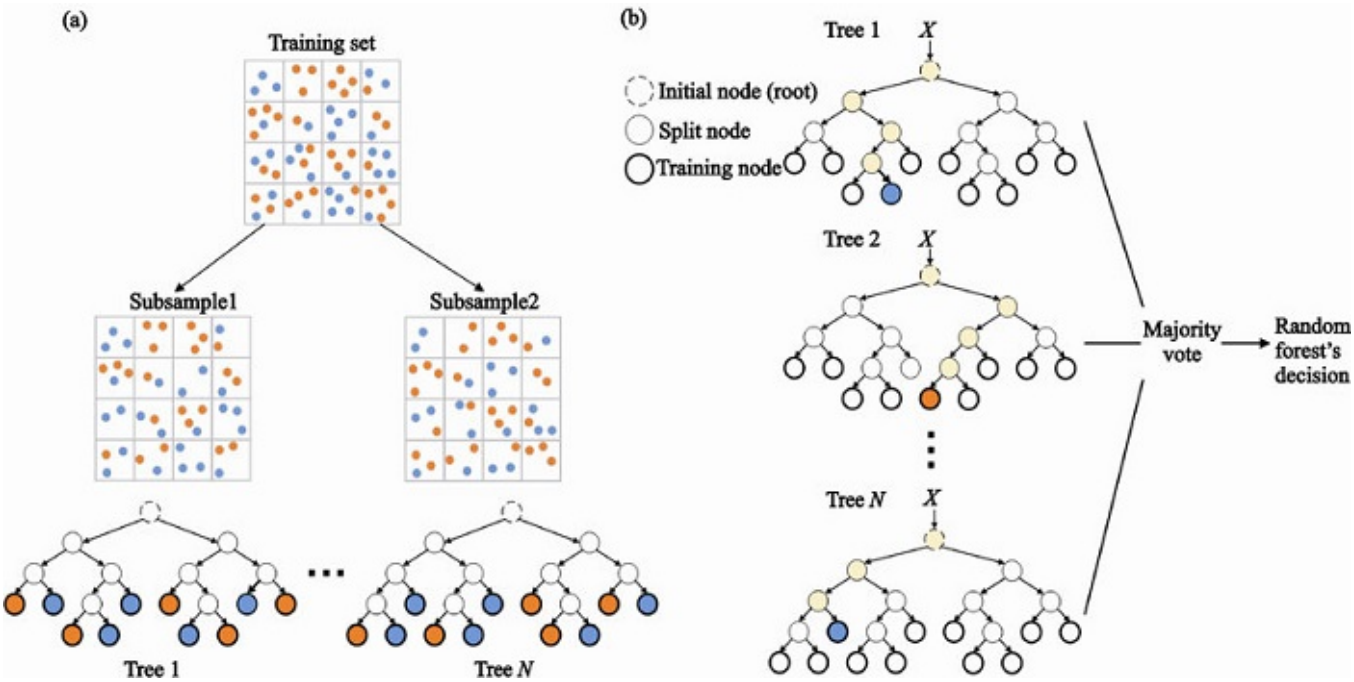
Image Credits: Wikipedia

Continuing with the purchase history example, the red cluster might represent users that like to buy tech gadgets and the blue one might be users interested in buying sports equipment. Now the algorithm will keep moving the centroid for each user-segment until it is able to create *K* groups. And it will do so by trying to maximize the separation between groups and users outside of the group.

8. Random Forest

Random Forest is one of the most popular and powerful machine learning algorithms. It is a type of ensemble algorithm. The underlying idea for **ensemble learning** the is ***wisdom of crowds***, the idea that the **collective opinion of many is more likely to be accurate than that of one**. The outcome of each of the models is combined and a prediction is made.

In Random Forest, we have an ensemble of decision trees, seen earlier in, algorithm 3. When we want to classify a new object, we take the vote of each decision tree and combine the outcome to make a final decision; majority vote wins.



(a) In the training process, each decision tree is built based on a bootstrap sample of the training set, which contains two kinds of examples (green labels and red labels). (b) In the classification process, decision for the input instance is based on the majority voting results among all individual trees. Image Source: Scientific Figure on ResearchGate, https://www.researchgate.net/figure/Illustration-of-random-forest-a-In-the-training-process-each-decision-tree-is-built_fig3_317274960

9. Dimensionality Reduction

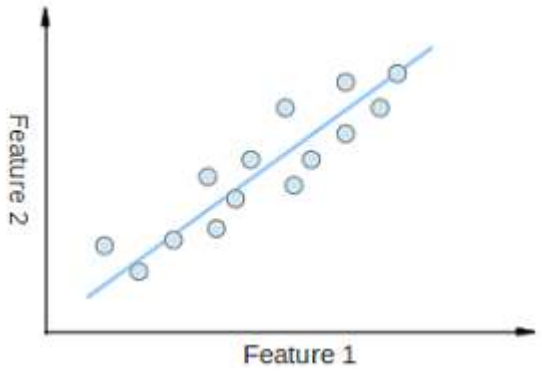
In the last years, there has been an exponential increase in the amount of data captured. This means that many machine learning problems involve thousands or even millions of features for each training instance! This not only makes training extremely slow but makes finding a good solution much harder. This problem is often referred to as the **curse of dimensionality**. In real-world problems, it is often possible to reduce the number of features considerably, making problems tractable.

For example, in an image classification problem, if the pixels on the image borders are almost always white, these pixels can completely be dropped from the training set without losing much information.

In simple terms, dimensionality reduction is about assembling specific features into more high-level ones without losing the most important information. Principal Component Analysis (PCA) is the most popular

information. Principal Component Analysis (PCA) is the most popular dimensionality reduction technique. Geometrically speaking, PCA reduces the

dimension of a dataset by squashing it onto a lower-dimensional line, or more generally a hyperplane/subspace, which retains as much of the original data's salient characteristics as possible.



Say we have a set of 2D points as shown in the figure above. Each dimension corresponds to a feature we are interested in. Although the points seem to be scattered quite randomly, if we pay close attention, we can see that we have a linear pattern (blue line). As we said, the key point in PCA is Dimensionality Reduction, the process of reducing the number of the dimensions of the given dataset; it does this by finding the direction along which our data varies the most.

In the example above, it is possible to achieve dimensionality reduction by approximating all the data points to a single line. The projection onto a line reduces the dimensionality of our dataset from 2D to 1D.

Last but definitely not the least, let's look into Artificial Neural Networks, which are at the very core of Deep Learning.

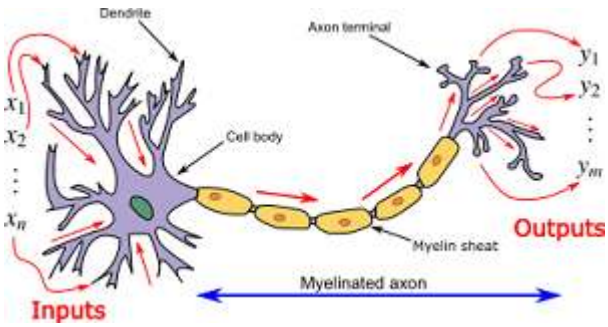
10. Artificial Neural Networks (ANN)

ANN are ideal for tackling large and highly complex machine learning tasks, such as recommending the best videos to watch to hundreds of millions of users every day (e.g., YouTube), powering speech recognition services (e.g., Siri, Cortana) or learning to beat the world champion at the game of Go (DeepMind's AlphaGo).

ANN require a **huge amount of training data, high computational power and long training time** but, in the end, they are able make **very accurate predictions**.

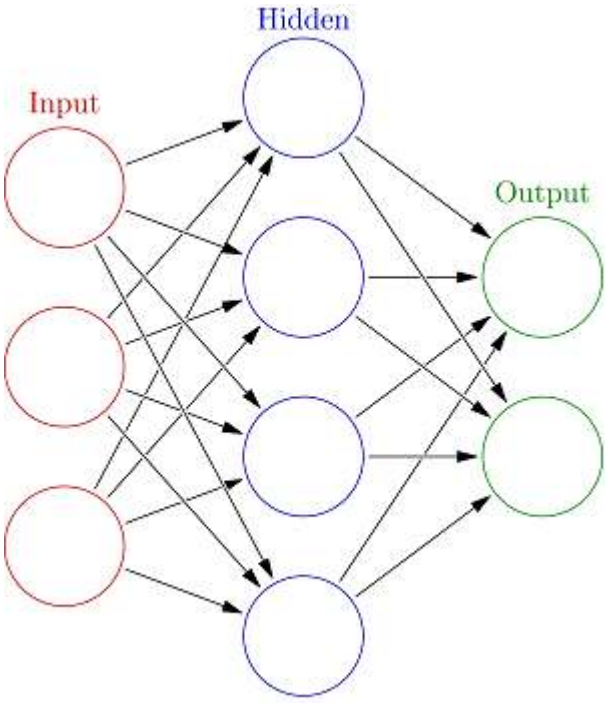
predictions.

The key idea behind ANN is to use the brain's architecture for inspiration on how to build intelligent machines.



Neuron with signal flow from inputs at dendrites to outputs at axon terminals

To train a neural network, a set of neurons are mapped out and assigned a random weight which determines how the neurons process new data, images, text, sounds, etc. The correct relationship between inputs and outputs is learned from training the neural network on input data. Since during the training phase the system gets to see the correct answers, if the network doesn't accurately identify the input – doesn't see a face in an image, for example — then the system adjusts the weights. Eventually, after sufficient training, the neural network will consistently recognize the correct patterns in speech, text or images.



An artificial neural network is an interconnected group of nodes, inspired by a simplification of neurons in a brain. Here, each circular node represents an artificial neuron and an arrow represents a connection from the output of one artificial neuron to the input of another.

As a neural network is a essentially a set of interconnected layers with weighted edges and nodes called neurons. Between the input and output layers we can insert multiple hidden layers. ANN make use of only two hidden layers. However, if we increase the depth of these layers then we are dealing with the famous Deep Learning.

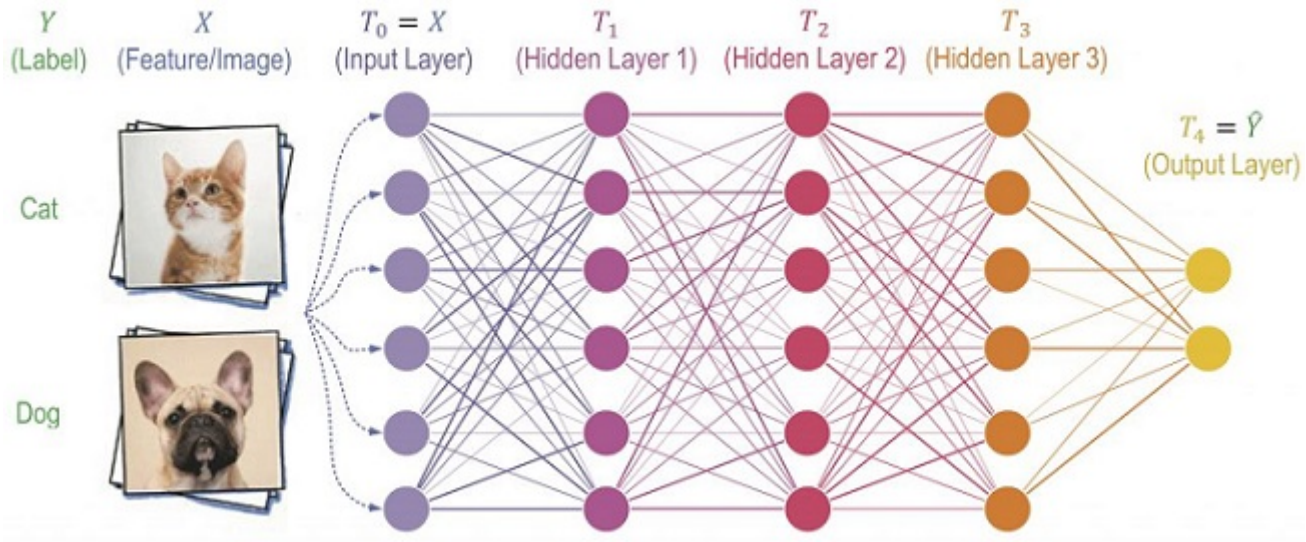


Image Credits: https://www.ibm.com/blogs/research/2019/06/deep-neural-networks

> **Note**: For a deep-dive into Neural Networks, I highly encourage you to read **this article**.

Final Thoughts

Now we have a very good overview of the most commonly used machine learning algorithms. Hope you enjoyed this walk-through!

With the basics covered, we are now in a good place to look at the implementation of these algorithms in Python. However, before we move onto the hands-on part, we have a few more theoretical topics to cover. So stay tuned!