# LOGISTICAL REGRESSION:

When you’re implementing the logistic regression of some dependent variable 𝑦 on the set of independent variables 𝐱 = (𝑥₁, …, 𝑥ᵣ), where 𝑟 is the number of predictors ( or inputs), you start with the known values of the predictors 𝐱ᵢ and the corresponding actual response (or output) 𝑦ᵢ for each observation 𝑖 = 1, …, 𝑛.

Your goal is to find the **logistic regression function** 𝑝(𝐱) such that the **predicted responses** 𝑝(𝐱ᵢ) are as close as possible to the **actual response** 𝑦ᵢ for each observation 𝑖 = 1, …, 𝑛. Remember that the actual response can be only 0 or 1 in binary classification problems! This means that each 𝑝(𝐱ᵢ) should be close to either 0 or 1. That’s why it’s convenient to use the sigmoid function.

Once you have the logistic regression function 𝑝(𝐱), you can use it to predict the outputs for new and unseen inputs.

𝑓(𝐱): 𝑝(𝐱) = 1 / (1 + exp(−𝑓(𝐱))

**PACKAGES:**

**NumPy**: scientific and numerical computing in Python; enables high-performance operations on single- and multi-dimensional arrays

**scikit-learn:** popular [data science](https://realpython.com/tutorials/data-science/) and [machine learning](https://realpython.com/tutorials/machine-learning/) libraries; **Preprocess** data, **Reduce** the dimensionality of problems, **Validate** models, **Select** the most appropriate model, **Solve** regression and classification problems, **Implement** cluster analysis

**Matplotlib:** visualize the results of your classification and high-quality plotting

# DECISION TREE:

A Decision tree is a flowchart-like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.

**PACKAGES:**

**Sklearn: machine learning package which include a lot of ML algorithms**

NumPy: scientific and numerical computing in Python; enables high-performance operations on single- and multi-dimensional arrays

Pandas: Used to read and write different files; Data manipulation can be done easily with dataframes

# RANDOM FOREST CLASSIFIER:

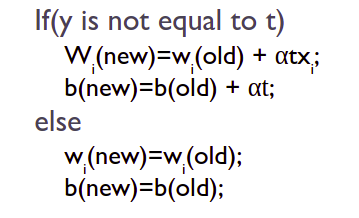
The Random forest or Random Decision Forest is a supervised Machine learning algorithm used for classification, regression, and other tasks using decision trees. The Random forest classifier creates a set of decision trees from a randomly selected subset of the training set. It is basically a set of decision trees (DT) from a randomly selected subset of the training set and then it collects the votes from different decision trees to decide the final prediction.

Random forests are a set of multiple decision trees.

# MULTILAYER PERCEPTRON NN:

Multi-Layer Perceptron(MLP) is the simplest type of artificial neural network. It is a combination of multiple perceptron models. Perceptrons are inspired by the human brain and try to simulate its functionality to solve problems. In MLP, these perceptrons are highly interconnected and parallel in nature. This parallelization helpful in faster computation.

Perceptron network consists of three units: Sensory Unit (Input Unit), Associator Unit (Hidden Unit), and Response Unit (Output Unit). The Perceptron consists of an input layer and an output layer which are fully connected. MLPs have the same input and output layers but may have multiple hidden layers in between.



Multi-Layer Perceptron trains model in an iterative manner. In each iteration, partial derivatives of the loss function used to update the parameters. We can also use regularization of the loss function to prevent overfitting in the model.

# ADABOOST CLASSIFIER:

Ada-boost or Adaptive Boosting is one of ensemble boosting classifier which combines multiple classifiers to increase the accuracy of classifiers. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier. The basic concept behind Adaboost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observations. Any machine learning algorithm can be used as base classifier if it accepts weights on the training set. Adaboost should meet two conditions:

1. The classifier should be trained interactively on various weighed training examples.
2. In each iteration, it tries to provide an excellent fit for these examples by minimizing training error.

# GRADIENT BOOSTING CLASSIFIER:

[Gradient boosting classifiers](https://en.wikipedia.org/wiki/Gradient_boosting) are a group of machine learning algorithms that combine many weak learning models together to create a strong predictive model. Decision trees are usually used when doing gradient boosting. Gradient boosting models are becoming popular because of their effectiveness at classifying complex datasets.

**PACKAGES:**

**scikit-learn:** popular [data science](https://realpython.com/tutorials/data-science/) and [machine learning](https://realpython.com/tutorials/machine-learning/) libraries; **Preprocess** data, **Reduce** the dimensionality of problems, **Validate** models, **Select** the most appropriate model, **Solve** regression and classification problems, **Implement** cluster analysis