**Machine learning:**

* **Machine learning** is defined as the use of computer power to analyze and draw inferences from patterns in the data.
* **Trainer:** the term trainer is used to refer to the data that is used to test and train the system under development. Data is compiled and collected from the known set of previous occurrences of similar events that can be used to train the system under test.
* **Types:**  
  ML is of following types:
  + **Un-supervised learning**: where trainer is not present.
  + **Supervised learning**: where trainer is present.

It is of 2 types:

* **Classification:** where the ML system is required to classify the object into one of the classes.  
  **Example:** check whether or not the borrower will default on the loan.
* **Regression:** where the ML system is required to predict a numerical value.  
  **Example:** find out the price at which an article will be sold.

Stats review:

* **Mean / Average:**

Sum of all observations divided by number of observations.

* **Median:**  
  It is the measure of the central quantity.  
  The central or middle value with all the values arranged in ascending or descending order.

When total no of observations are even in no, no clear central value is present. In such a case, the median is the mean of both the central values.

* **Mode:**The most frequently occurring value is called mode.
* **Percentiles:**

The percentiles are the measure of how distributed the data is.

The 50th percentile indicates the central measure of data. It is also known as median.

Similarly, the 25th percentile is the measure of one quarter of through the data.

The 75th percentile is the measure of three quarters through the data.

* **Standard deviation and variance:**

SD and variance are the measure of how spread or distributed the data is.

It is the measure of how far each datapoint is from the median.

Variance = /n

SD =

* **Stats with Python library- numpy:**
  + **Numpy** is the python library that is used to perform the statistical manipulation on data.
* **Load data to python project using Pandas.**

**Target:** The thing that we are trying to predict is called a target.

So, Classification problem is where the target is a categorical value while Regression problems are those where the target value is numerical.

**Classification Terminology:**

* **Target:** The thing that we are trying to predict is target. In classification problem, the target will be a categorical value.
* **Feature:** Feature is the data that we use to predict the target.
* **Predictors:** Features are also called predictors.

**Idea of Solving Classification problem using ML:**

In order to solve the classification problem using ML, the data is needed to be mathematically sampled. We would need some set of historic data to model with features and known target. This will be used to test and train the model under development.

The idea to solve a classification problem in ML is that, this trainer data can be represented via scattering plot. In order to clearly distinguish between the classes, we could use a color coding for each data point that would represent the class to which each datapoint belongs.

Based on the plot, we would aim to find a line (ax + by + c = 0), that separates the area under the plot into two regions such that most data points belonging to a class lie on either side of the line.

For predicting the target of a datapoint, we would evaluate the equation of line at features of the datapoint and would classify as follows:

**Case 1:** If the equation evaluates to +ve value, then the target will lie on the right side if the line, and would be classified under the class that is dominating in that region.

**Case 2:** If the equation evaluates to -ve value, then the target will lie on the left side of the line and would be classified under the class that is dominating in that region.

**Case 3:** if the equation evaluates to 0, that means the target datapoint lie on the line and we are unable to predict its value.

Hence, solving a classification problem using ML, drills down to finding out the best possible line to divide the area under the plot into 2 regions.

**Likelihood:**

It tells us how good is our line.

To calculate how good our line is, we need to score whether or not our predictions are correct. If the prediction is correct, the model is rewarded and if the prediction is in-correct, the model is penalized.

**Building the Logistic regression model in python:**

Different models are supported by the library:

sklearn.

Install “scikit-learn” package using pip to use the library.

The LR model internally does what we discussed in **Idea of Solving Classification problem using ML.**

More details are present here: <https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html>

**Terms related to the ML Model predictions:**

**(Evaluation Metrics)**

* **Accuracy:**

It is the measure how many datapoints were predicted correctly by the model designed**.** Measured in %

* **Confusion Matrix:**

In a ML Model, we are not only concerned of correct predictions, but we are also concerned about the incorrect predictions. Hence, Confusion Matrix represents the correct and incorrect predictions made by the model.

The Confusion Matrix is a table showing four values:

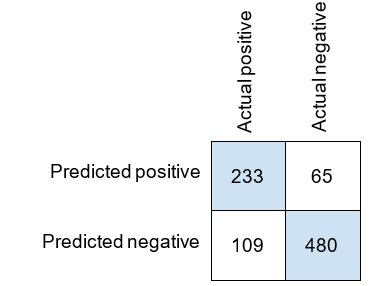
• Datapoints we predicted positive that are actually positive

• Datapoints we predicted positive that are actually negative

• Datapoints we predicted negative that are actually positive

• Datapoints we predicted negative that are actually negative

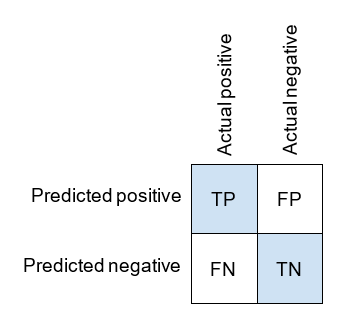
It is a 2-d matrix, that looks like:

****

The shaded area represents the correct predictions while the white area represents the incorrect predictions**.**

* **A true positive (TP)** is a datapoint we predicted positively that we were correct about.
* **A true negative (TN)** is a datapoint we predicted negatively that we were correct about.
* **A false positive (FP)** is a datapoint we predicted positively that we were incorrect about.
* **A false negative (FN)** is a datapoint we predicted negatively that we were incorrect about.

Based on above terms: the confusion matrix can be generalized as below:



* **Precision:**

Precision is the measure of how precise the model is with respect to the positive predictions.

* **Recall:**

Recall is a measure of how many of the positive cases the model can recall.

Note:  
There will always be a trade off between precision and recall.

Of the two, what we would want to maximize depends on the use case and how we want to handle the categorical predicted data.

For example, let’s say we’re building a model to predict if a credit card charge is fraudulent. The positive cases for our model are fraudulent charges and the negative cases are legitimate charges.

Let’s consider two scenarios:

1. If we predict the charge is fraudulent, we’ll reject the charge.

2. If we predict the charge is fraudulent, we’ll call the customer to confirm the charge.

In case 1, it’s a huge inconvenience for the customer when the model predicts fraud incorrectly (a false positive). In case 2, a false positive is a minor inconvenience for the customer.

Since, Precision and recall are two different values, it may become difficult to decide which model to select if one has higher Precision and other has higher recall.

In such a case, the F1 score comes handy.  
  
**F1 Score:** is the average of the recall and precision.