# Classification

Classification is a very important area of supervised machine learning. A large number of important machine learning problems fall within this area. Classification is an area of supervised machine learning that tries to predict which class or category some entity belongs to, based on its features.

For example, you might analyze the employees of some company and try to establish a dependence on the features or variables, such as the level of education, number of years in a current position, age, salary, odds for being promoted, and so on. The set of data related to a single employee is one observation.

There are two main types of classification problems:

1. Binary or binomial classification: exactly two classes to choose between (usually 0 and 1, true and false, or positive and negative)
2. Multiclass or multinomial classification: three or more classes of the outputs to choose from

Methods used for classification are

* Logistic Regression
* k-Nearest Neighbors
* Naive Bayes classifiers
* Support Vector Machines
* Decision Trees
* Random Forests
* Neural Networks

# Classification Performance

Binary classification has four possible types of results:

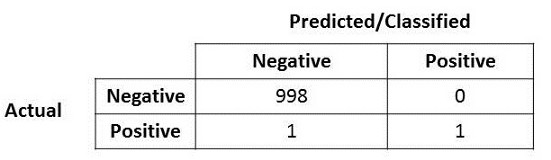
* **True negatives**: correctly predicted negatives (zeros)
* **True positives**: correctly predicted positives (ones)
* **False negatives**: incorrectly predicted negatives (zeros)
* **False positives**: incorrectly predicted positives (ones)

You usually evaluate the performance of your classifier by comparing the actual and predicted outputs and counting the correct and incorrect predictions.

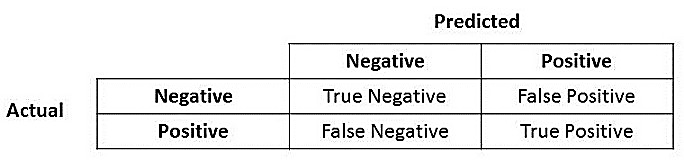
The most straightforward indicator of classification accuracy is the ratio of the number of correct predictions to the total number of predictions (or observations). Other indicators of binary classifiers include the following:

* The **positive predictive** value is the ratio of the number of true positives to the sum of the numbers of true and false positives.
* The **negative predictive** value is the ratio of the number of true negatives to the sum of the numbers of true and false negatives.
* The **sensitivity** (also known as recall or true positive rate) is the ratio of the number of true positives to the number of actual positives.
* The **specificity** (or true negative rate) is the ratio of the number of true negatives to the number of actual negatives.

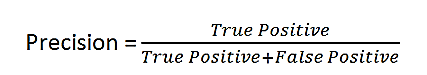
Firstly, let us look at the following confusion matrix. What is the accuracy for the model?



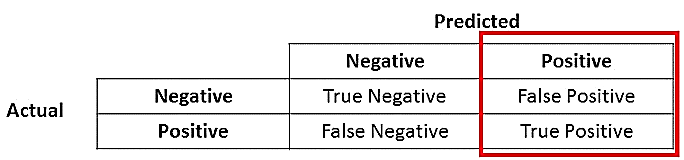
Very easily, you will notice that the accuracy for this model is very high, at 99.9%. what if I mentioned that the positive over here is actually represents terrorist that the model says it’s a non-terrorist? And vice versa.

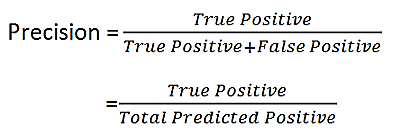


## ****Precision****



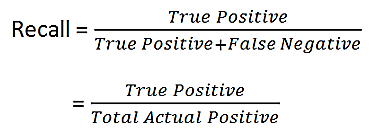
What do you notice for the denominator? The denominator is actually the Total Predicted Positive! So the formula becomes

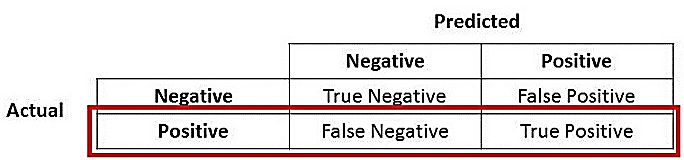




Precision is a good measure to determine, when the costs of False Positive is high. For instance, email spam detection. In email spam detection, a false positive means that an email that is non-spam (actual negative) has been identified as spam (predicted spam). The email user might lose important emails if the precision is not high for the spam detection model.

## ****Recall****





Recall actually calculates how many of the Actual Positives our model capture through labeling it as Positive (True Positive). Applying the same understanding, we know that Recall shall be the model metric we use to select our best model when there is a high cost associated with False Negative.

For instance, in fraud detection or sick patient detection. If a fraudulent transaction (Actual Positive) is predicted as non-fraudulent (Predicted Negative), the consequence can be very bad for the bank.

Similarly, in sick patient detection. If a sick patient (Actual Positive) goes through the test and predicted as not sick (Predicted Negative). The cost associated with False Negative will be extremely high if the sickness is contagious.

## F1 Score

F1 which is a function of Precision and Recall.

C:\Users\Vinsmon TP\Pictures\1_T6kVUKxG_Z4V5Fm1UXhEIw.png

F1 Score is needed when you want to seek a balance between Precision and Recall. Right…so what is the difference between F1 Score and Accuracy then? We have previously seen that accuracy can be largely contributed by a large number of True Negatives which in most business circumstances, we do not focus on much whereas False Negative and False Positive usually has business costs (tangible & intangible) thus F1 Score might be a better measure to use if we need to seek a balance between Precision and Recall AND there is an uneven class distribution (large number of Actual Negatives).

# Logistic Regression

Logistic regression is a fundamental classification technique. It belongs to the group of linear classifiers and is somewhat similar to polynomial and linear regression. Logistic regression is fast and relatively uncomplicated, and it’s convenient for you to interpret the results. Although it’s essentially a method for binary classification, it can also be applied to multiclass problems.

Logistic Regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). In other words, the logistic regression model predicts P(Y=1) as a function of X.

## Logistic Regression Assumptions

* Binary logistic regression requires the dependent variable to be binary.
* For a binary regression, the factor level 1 of the dependent variable should represent the desired outcome.
* Only the meaningful variables should be included.
* The independent variables should be independent of each other. That is, the model should have little or no multi collinearity.
* The independent variables are linearly related to the log odds.
* Logistic regression requires quite large sample sizes.

## Working of Logistic Regression

It is a special case of linear regression where the target variable is categorical in nature. It uses a log of odds as the dependent variable. Logistic Regression predicts the probability of occurrence of a binary event utilizing a logit function.

Linear Regression Equation:

C:\Users\Vinsmon TP\Pictures\image1_ga8gze.png

Where, y is dependent variable and x1, x2 ... and Xn are explanatory variables.

Sigmoid Function:

C:\Users\Vinsmon TP\Pictures\image2_kwxquj.png

Apply Sigmoid function on linear regression:

C:\Users\Vinsmon TP\Pictures\image3_qldafx.png

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, assuming that the underlying mathematical dependence is unchanged.

To get the best weights, you usually maximize the log**-**likelihoodfunction(LLF) for all observations 𝑖 = 1, …, 𝑛. This method is called the maximumlikelihoodestimation and is represented by the equation LLF = Σᵢ(𝑦ᵢ log(𝑝(𝐱ᵢ)) + (1 − 𝑦ᵢ) log(1 − 𝑝(𝐱ᵢ))).

When 𝑦ᵢ = 0, the LLF for the corresponding observation is equal to log(1 − 𝑝(𝐱ᵢ)). If 𝑝(𝐱ᵢ) is close to 𝑦ᵢ = 0, then log(1 − 𝑝(𝐱ᵢ)) is close to 0. This is the result you want. If 𝑝(𝐱ᵢ) is far from 0, then log(1 − 𝑝(𝐱ᵢ)) drops significantly. You don’t want that result because your goal is to obtain the maximum LLF. Similarly, when 𝑦ᵢ = 1, the LLF for that observation is 𝑦ᵢ log(𝑝(𝐱ᵢ)). If 𝑝(𝐱ᵢ) is close to 𝑦ᵢ = 1, then log(𝑝(𝐱ᵢ)) is close to 0. If 𝑝(𝐱ᵢ) is far from 1, then log(𝑝(𝐱ᵢ)) is a large negative number.

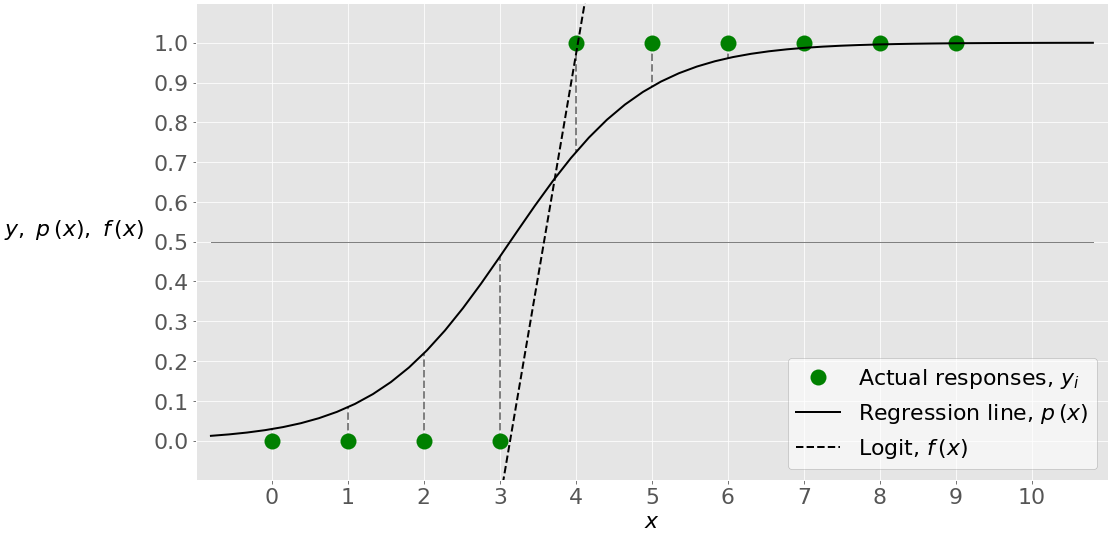
There’s one more important relationship between 𝑝(𝐱) and 𝑓(𝐱), which is that log(𝑝(𝐱) / (1 − 𝑝(𝐱))) = 𝑓(𝐱). This equality explains why 𝑓(𝐱) is the logit. It implies that 𝑝(𝐱) = 0.5 when 𝑓(𝐱) = 0 and that the predicted output is 1 if 𝑓(𝐱) > 0 and 0 otherwise.



Linear regression is estimated using Ordinary Least Squares (OLS) while logistic regression is estimated using Maximum Likelihood Estimation (MLE) approach. The MLE is a "likelihood" maximization method, while OLS is a distance-minimizing approximation method. Maximizing the likelihood function determines the parameters that are most likely to produce the observed data. From a statistical point of view, MLE sets the mean and variance as parameters in determining the specific parametric values for a given model. This set of parameters can be used for predicting the data needed in a normal distribution.

## Single-Variate Logistic Regression

Single-variate logistic regression is the most straightforward case of logistic regression. There is only one independent variable (or feature), which is 𝐱 = 𝑥. This figure illustrates single-variate logistic regression:

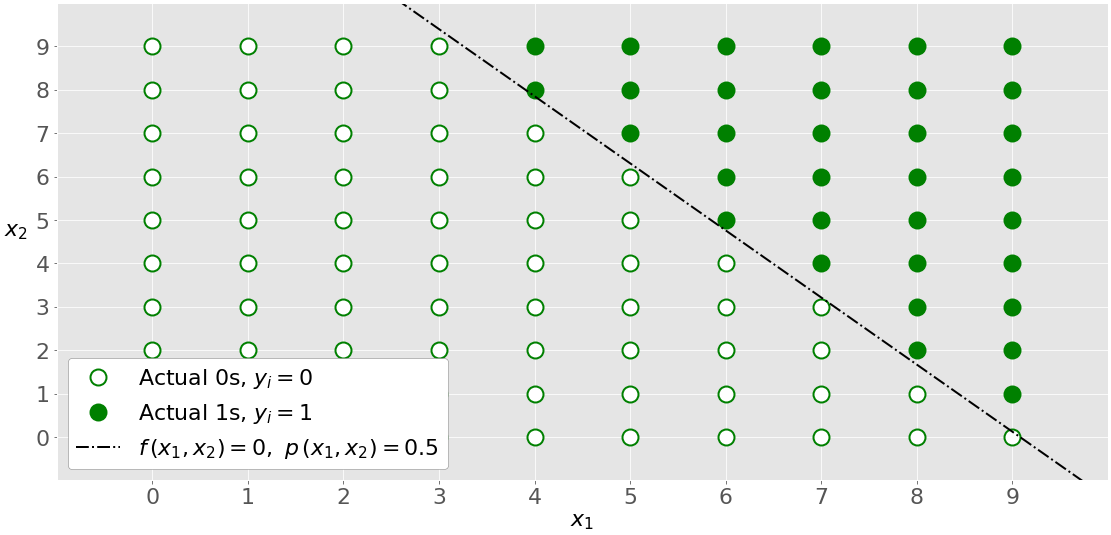
[](https://files.realpython.com/media/log-reg-2.e88a21607ba3.png)

Here, you have a given set of input-output (or 𝑥-𝑦) pairs, represented by green circles. These are your observations. Remember that 𝑦 can only be 0 or 1. For example, the leftmost green circle has the input 𝑥 = 0 and the actual output 𝑦 = 0. The rightmost observation has 𝑥 = 9 and 𝑦 = 1.

Logistic regression finds the weights 𝑏₀ and 𝑏₁ that correspond to the maximum LLF. These weights define the logit 𝑓(𝑥) = 𝑏₀ + 𝑏₁𝑥, which is the dashed black line. They also define the predicted probability 𝑝(𝑥) = 1 / (1 + exp(−𝑓(𝑥))), shown here as the full black line. In this case, the threshold 𝑝(𝑥) = 0.5 and 𝑓(𝑥) = 0 corresponds to the value of 𝑥 slightly higher than 3. This value is the limit between the inputs with the predicted outputs of 0 and 1.

## Multi-Variate Logistic Regression

Multi-variate logistic regression has more than one input variable. This figure shows the classification with two independent variables, 𝑥₁ and 𝑥₂:

[](https://files.realpython.com/media/log-reg-3.b1634d335c4f.png)

The graph is different from the single-variate graph because both axes represent the inputs. The outputs also differ in color. The white circles show the observations classified as zeros, while the green circles are those classified as ones.

Logistic regression determines the weights 𝑏₀, 𝑏₁, and 𝑏₂ that maximize the LLF. Once you have 𝑏₀, 𝑏₁, and 𝑏₂, you can get:

* The logit 𝑓(𝑥₁, 𝑥₂) = 𝑏₀ + 𝑏₁𝑥₁ + 𝑏₂𝑥₂
* The probabilities 𝑝(𝑥₁, 𝑥₂) = 1 / (1 + exp(−𝑓(𝑥₁, 𝑥₂)))

The dash-dotted black line linearly separates the two classes. This line corresponds to 𝑝(𝑥₁, 𝑥₂) = 0.5 and 𝑓(𝑥₁, 𝑥₂) = 0.