Logistic Regression

- Logistic regression is a predictive modelling algorithm that is used when the Y variable is binary categorical.
- It can take only two values like 1 or 0.
- The goal is to determine a mathematical equation that can be used to predict the probability of event 1.
- It can be used to predict the Y when only the X's are known.
- Logistic regression can be used to model and solve such problems, also called as binary classification problems.
- Logistic regression is a classic predictive modelling technique and still remains a popular choice for modelling binary categorical variables.

Binary Classification Problems

- **Spam Detection** : Predicting if an email is Spam or not
- **Credit Card Fraud** : Predicting if a given credit card transaction is fraud or not
- **Health** : Predicting if a given mass of tissue is benign or malignant
- **Marketing** : Predicting if a given user will buy an insurance product or not
- **Banking** : Predicting if a customer will default on a loan.

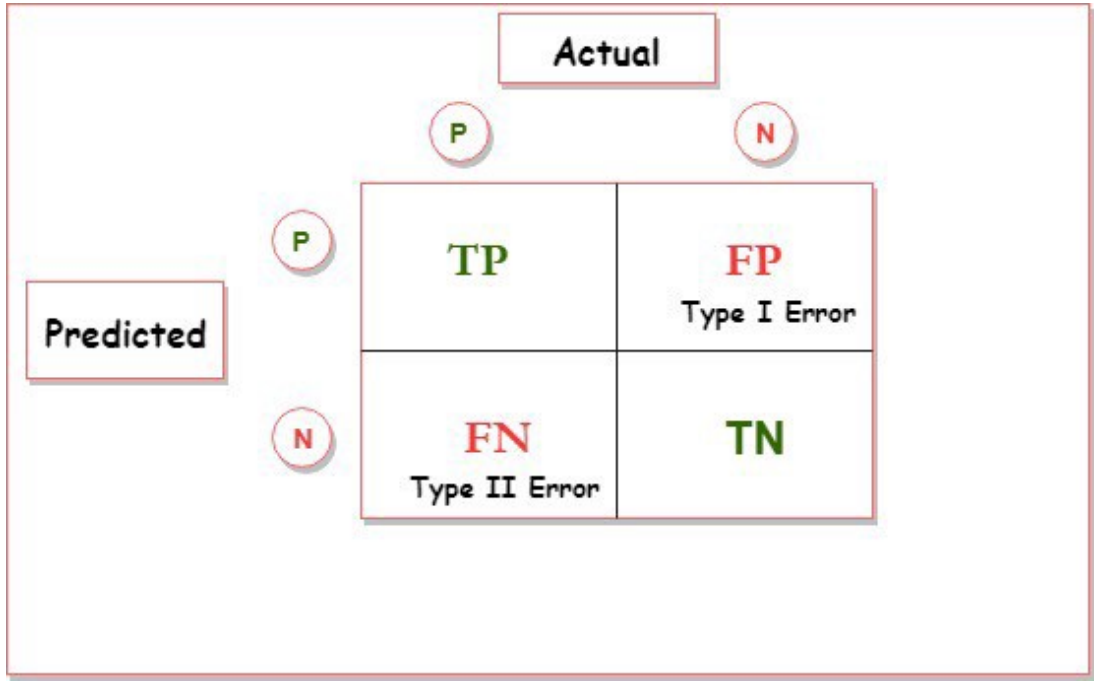
- When the response variable has only 2 possible values, it is desirable to have a model that predicts the value either as 0 or 1 or as a probability score that ranges between 0 and 1.
- In logistic regression, you get a probability score that reflects the probability of the occurence of the event

Maths behind Logistic Function

- **Step 1: Classifying inputs to be in class zero or one.**
- **Step 2: Defining a boundary values for the odds**

Threshold value

- Define a threshold boundary in-order to clearly classify each given input values into one of the classes.
- So if your probability values come out to be >0.5 - classify such observation into class 1 type, the rest into class 0.
- The choice of threshold value is generally based on error types, which are of two types , false positives, and false negatives.



Threshold value impacts

- ***Higher threshold value***

Suppose if P(y=1) > 0.7. The model is being more restrictive when classifying as 1´s, and therefore, more False Negative errors will be made.

- ***Lower Threshold value***

Suppose, if P(y=1) > 0.3.

The model is now being less strict and we are classifying more examples as class 1, therefore, we are making more false positives errors.

Confusion Matrix: A Way To Choose An Effective Threshold Value:

- A confusion matrix also known as error matrix is a predictor of model performance on a classification problem.
- The number of correct and incorrect predictions is summarized with count values and broken down by each class.
- The confusion matrix shows the ways in which your classification model is confused when it makes predictions on observations, it helps us to measure the type of error our model is making while classifying the observation into different classes.

Key points of Confusion matrix

- **True Positive** (TP): This refers to the cases in which we predicted "**YES**" and our prediction was actually **TRUE**
  (Eg. Patient is actually having diabetes and you predicted it to be True)
- **True Negative** (TN): This refers to the cases in which we predicted "**NO**" and our prediction was actually **TRUE**
  (Eg. Patient is not having diabetes and you predicted the same.)
- **False Positive** (FP): This refers to the cases in which we predicted "**YES**", but our prediction turned out **FALSE**
  (Patient was not having diabetes, but our model predicted that s/he is diabetic)
- **False Negative** (FN): This refers to the cases in which we predicted "**NO**" but our prediction turned out **FALSE**

Accuracy

- Overall, how often is the classifier correct*?*
- **Accuracy = (TP+TN)/total No of Classified Item = (TP+TN)/ (TP+TN+FP+FN)**

Precision

- ***When it predicts yes, how often is it correct?***
- Precision is usually used when the goal is to *limit the number of false positives*(FP). For example, with the spam, filtering algorithm, where our aim is to minimize the number of reals emails that are classified as spam.
- ***Precision = TP/ (TP+FP)***

Recall

- *When it is actually the positive result, how often does it predict correctly?*
- It is calculated as,
- ***Recall = TP/ (TP+FN), also known as sensitivity.***

f1-score

- It is just the harmonic mean of precision and recall:
- It is calculated as,
- ***f1-score=2*((precision*recall)/(precision+recall))***

Recall & Precision

- If we try to only optimize recall, algorithm will predict most examples to belong to the positive class, but that will result in many false positives and, hence, low precision.
- Also, if you try to optimize precision, the model will predict very few examples as positive results (the ones which highest probability), but recall will be very low.
- So it can be insightful to balance out and consider both and see the result.

The Logistic Equation

$$Z_i = ln\left(\frac{P_i}{1-P_i}\right) = \alpha + \beta_1 x_1 + .. + \beta_n x_n$$

To implement this we use glm() function by setting the family argument as “binomial”

Building Logistic Regression model in R

- There is a problem to model the data because except Id, all the other columns are factors.
- When we build a logistic model with factor variables as features, it converts each level in the factor into a dummy binary variable of 1's and 0's.
- Cell shape is a factor with 10 levels. When we use glm to model Class as a function of cell shape, the cell shape will be split into 9 different binary categorical variables before building the model.

Conversion of factors to numeric

```
# convert factors to numeric
for(i in 1:9) {
bc[, i] <- as.numeric(as.character(bc[, i]))
}
```

Encode the response variable

Encode the response variable into a factor variable of 1's and 0's.
So whenever the Class is malignant, it will be 1 else it will be 0.
Then, I am converting it into a factor.

```
bc$Class <- ifelse(bc$Class == "malignant", 1, 0)
bc$Class <- factor(bc$Class, levels = c(0, 1))
```

The response variable class is now a factor variable and all other columns are numeric.
All the classes of the columns are set

Dealing with class imbalance

- Before building the logistic regressor, you need to randomly split the data into training and test samples.
- Since the response variable is a binary categorical variable, we need to make sure the training data has approximately equal proportion of classes.
- The classes 'benign' and 'malignant' are split approximately in 1:2 ratio.
- Clearly there is a class imbalance.
- So, before building the logit model, we need to build the samples such that both the 1's and 0's are in approximately equal proportions.

How to handle Class Imbalance with Upsampling and Downsampling

- In Down sampling, the majority class is randomly down sampled to be of the same size as the smaller class.
- That means, when creating the training dataset, the rows with the benign Class will be picked fewer times during the random sampling.
- But in case of Hybrid sampling, artificial data points are generated and are systematically added around the minority class.

There is approximately 2 times more benign samples. So lets downsample it using the `downSample` function from `caret` package.

Training and Testing data

By setting `p=.70` I have chosen 70% of the rows to go inside `trainData` and the remaining 30% to go to `testData`

table(trainData$Class)
There is approximately 2 times more benign samples .
Downsample by using downsample package

Assumptions

- The response variable must follow a binomial distribution.
- Logistic Regression assumes a linear relationship between the independent variables and the link function (logit).
- The dependent variable should have mutually exclusive and exhaustive categories.

How does Logistic Regression work?

- Logistic Regression assumes that the dependent (or response) variable follows a binomial distribution.
- Binomial distribution can be identified by the following characteristics:
- There must be a fixed number of trials denoted by **n**, i.e. in the data set, there must be a fixed number of rows.
- Each trial can have only two outcomes; i.e., the response variable can have only two unique categories.
- The outcome of each trial must be independent of each other; i.e., the unique levels of the response variable must be independent of each other.
- The probability of success (p) and failure (q) should be the same for each trial.

In Logistic Regression:

- calculate probabilities and it always lie between 0 and 1.
- The response value must be positive.

Logistic Regression Interpretation

- A unit increase in variable x results in multiplying the odds ratio by **ε to power β**.
- The regression coefficients explain the change in log(odds) in the response for a unit change in predictor.
- Since the relationship between p(X) and X is not straight line, a unit change in input feature doesn't really affect the model output directly but it affects the odds ratio.

Evaluation metrics for Logistic Regression

1. Akaike Information Criteria (AIC)
2. Null Deviance and Residual Deviance
3. Confusion Matrix
4. Receiver Operator Characteristic (ROC)

Akaike Information Criteria (AIC

- Look at AIC as counterpart of adjusted r square in multiple regression. It's an important indicator of model fit.
- It follows the rule: Smaller the better.
- AIC penalizes increasing number of coefficients in the model.
- Adding more variables to the model wouldn't let AIC increase.
- It helps to avoid overfitting.
- Looking at the AIC metric of one model wouldn't really help. It is more useful in comparing models (model selection).
- Build 2 or 3 Logistic Regression models and compare their AIC.
- The model with the lowest AIC will be relatively better.

Null Deviance and Residual Deviance

- Null deviance is calculated from the model with no features, i.e.,only intercept.
- Residual deviance is calculated from the model having all the features.
- On comparison with Linear Regression, think of residual deviance as residual sum of square (RSS) and null deviance as total sum of squares (TSS). The larger the difference between null and residual deviance, better the model.
- We can use these metrics to compared multiple models: whichever model has a lower null deviance, means that the model explains deviance pretty well, and is a better model.
- Lower the residual deviance, better the model.
- Practically, AIC is always given preference above deviance to evaluate model fit.

3.Confusion Matrix

- Confusion matrix is the most crucial metric commonly used to evaluate classification models.
- The skeleton of a confusion matrix looks like this:
- **Accuracy** - It determines the overall predicted accuracy of the model. It is calculated as Accuracy = (True Positives + True Negatives)/(True Positives + True Negatives + False Positives + False Negatives)
- **Precision:** It indicates how many values, out of all the predicted positive values, are actually positive. It is formulated as:(TP / TP + FP).
- **F Score:** F score is the harmonic mean of precision and recall. It lies between 0 and 1. Higher the value, better the model. It is formulated as 2((precision*recall) / (precision+recall))

4.Receiver Operator Characteristic (ROC)

- ROC determines the accuracy of a classification model at a user defined threshold value.
- It determines the model's accuracy using Area Under Curve (AUC). The area under the curve (AUC), also referred to as index of accuracy (represents the performance of the ROC curve.
- Higher the area, better the model. ROC is plotted between True Positive Rate (Y axis) and False Positive Rate (X Axis).
- In this plot, our aim is to push the red curve (shown below) toward 1 (left corner) and maximize the area under curve. Higher the curve, better the model.
- The yellow line represents the ROC curve at 0.5 threshold. At this point, sensitivity = specificity.