1. Data cleaning and preparation
   * Combining data frames
   * Handling categorical variables
     + Mapping categorical variables to integers
     + Dummy variable creation
   * Handling missing values
2. Test-train split and scaling
3. Model Building
   * Feature elimination based on correlations
   * Feature selection using RFE (Coarse Tuning)
   * Manual feature elimination (using p-values and VIFs)
4. Model Evaluation
   * Accuracy
   * Sensitivity and Specificity
   * Optimal cut-off using ROC curve
   * Precision and Recall
5. Predictions on the test set

In this blog I will briefly explain about Logistic regression, but before getting into it let me give you some basics.

# What is Machine Learning

Machine learning (ML) is a type of artificial intelligence (AI) that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine learning algorithms use historical data as input to predict new output values.

There are different ways to train machine learning algorithms, each with their own advantages and disadvantages. To understand the pros and cons of each type of machine learning, we must first look at what kind of data they ingest. In ML, there are two kinds of data — labeled data and unlabeled data.

* Labeled data has both the input and output parameters in a completely machine-readable pattern, but requires a lot of human labor to label the data, to begin with.
* Unlabeled data only has one or none of the parameters in a machine-readable form. This negates the need for human labor but requires more complex solutions.

There are also some types of machine learning algorithms that are used in very specific use-cases, but three main methods are used today.

Machine learning is basically divided into 3 categories

1. Supervised learning
2. UnSupervised learning
3. Reinforcement learning

## Supervised learning

Supervised learning, as the name indicates, has the presence of a supervisor as a teacher. Basically supervised learning is when we teach or train the machine using data that is well labelled. Which means some data is already tagged with the correct answer. After that, the machine is provided with a new set of examples(data) so that the supervised learning algorithm analyses the training data(set of training examples) and produces a correct outcome from labelled data.

For instance, suppose you are given a basket filled with different kinds of fruits. Now the first step is to train the machine with all the different fruits one by one like this.

* If the shape of the object is rounded and has a depression at the top, is red in color, then it will be labeled as –Apple.
* If the shape of the object is a long curving cylinder having Green-Yellow color, then it will be labeled as –Banana.

Now suppose after training the data, you have given a new separate fruit, say Banana from the basket, and asked to identify it.

Since the machine has already learned the things from previous data and this time has to use it wisely. It will first classify the fruit with its shape and color and would confirm the fruit name as BANANA and put it in the Banana category. Thus the machine learns the things from training data(basket containing fruits) and then applies the knowledge to test data(new fruit).

Supervised learning is classified into two categories of algorithms:

* Classification: A classification problem is when the output variable is a category, such as “Red” or “blue” , “disease” or “no disease”.
* Regression: A regression problem is when the output variable is a real value, such as “dollars” or “weight”.

Supervised learning deals with or learns with “labeled” data. This implies that some data is already tagged with the correct answer.

Types:-

* Regression
* Classification
* Naive Bayes Classifiers
* K-NN (k nearest neighbors)
* Decision Trees
* Support Vector Machine

Advantages:-

* Supervised learning allows collecting data and produces data output from previous experiences.
* Helps to optimize performance criteria with the help of experience.
* Supervised machine learning helps to solve various types of real-world computation problems.

Disadvantages:-

* Classifying big data can be challenging.
* Training for supervised learning needs a lot of computation time. So, it requires a lot of time.

## Unsupervised learning

Unsupervised learning is the training of a machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of the machine is to group unsorted information according to similarities, patterns, and differences without any prior training of data.

Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore the machine is restricted to find the hidden structure in unlabeled data by itself.

For instance, suppose it is given an image having both dogs and cats which it has never seen.

Thus the machine has no idea about the features of dogs and cats so we can’t categorize it as ‘dogs and cats ‘. But it can categorize them according to their similarities, patterns, and differences, i.e., we can easily categorize the above picture into two parts. The first may contain all pics having dogs in them and the second part may contain all pics having cats in them. Here you didn’t learn anything before, which means no training data or examples.

It allows the model to work on its own to discover patterns and information that was previously undetected. It mainly deals with unlabelled data.

Unsupervised learning is classified into two categories of algorithms:

1. Clustering: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
2. Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

Types of Unsupervised Learning:-

* Clustering
* Exclusive (partitioning)
* Agglomerative
* Overlapping
* Probabilistic

Clustering Types:-

* Hierarchical clustering
* K-means clustering
* Principal Component Analysis
* Singular Value Decomposition
* Independent Component Analysis

# With this background we now discuss Logistic Regression

Machine learning - > Supervised Learning ->Classification - > Logistic Regression

## [Assumptions](https://towardsdatascience.com/assumptions-of-logistic-regression-clearly-explained-44d85a22b290)

1. **The response variable is binary**

Logistic regression assumes that the response variable only takes on two possible outcomes. Some examples include:

• Yes or No

• Male or Female

• Pass or Fail

• Drafted or Not Drafted

• Malignant or Benign

How to check this assumption: Simply count how many unique outcomes occur in the response variable. If there are more than two possible outcomes, you will need to perform ordinal regression instead.

1. **Independence of observations**

Logistic regression assumes that the observations in the dataset are independent of each other. That is, the observations should not come from repeated measurements of the same individual or be related to each other in any way.

How to check this assumption: The easiest way to check this assumption is to create a plot of residuals against time (i.e. the order of the observations) and observe whether or not there is a random pattern. If there is not a random pattern, then this assumption may be violated.

1. **Absence of multicollinearity**

Logistic regression assumes that there is no severe [multicollinearity](https://www.statology.org/multicollinearity-regression/) among the [explanatory variables](https://www.statology.org/explanatory-response-variables/).

Multicollinearity occurs when two or more explanatory variables are highly correlated to each other, such that they do not provide unique or independent information in the regression model. If the degree of correlation is high enough between variables, it can cause problems when fitting and interpreting the model.

For example, suppose you want to perform logistic regression using **max vertical jump**as the response variable and the following variables as explanatory variables:

* Player height
* Player shoe size
* Hours spent practicing per day

In this case, **height** and **shoe size** are likely to be highly correlated since taller people tend to have larger shoe sizes. This means that multicollinearity is likely to be a problem if we use both of these variables in the regression.

**How to check  this assumption:**The most common way to detect multicollinearity is by using the variance inflation factor (VIF), which measures the correlation and strength of correlation between the predictor variables in a regression model. Check out [this tutorial](https://www.statology.org/multicollinearity-regression/) for an in-depth explanation of how to calculate and interpret VIF values.

1. **No strongly influential outliers**

Logistic regression assumes that there are no extreme outliers or influential observations in the dataset.

**How to check this assumption:**The most common way to test for extreme outliers and influential observations in a dataset is to calculate [Cook’s distance](https://www.statology.org/how-to-identify-influential-data-points-using-cooks-distance/) for each observation. If there are indeed outliers, you can choose to (1) remove them, (2) replace them with a value like the mean or median, or (3) simply keep them in the model but make a note about this when reporting the regression results.

1. **Linearity of independent variables and log-odds(Logit of the Response Variable)**

Logistic regression assumes that there exists a linear relationship between each explanatory variable and the logit of the response variable. Recall that the logit is defined as:

Logit(p)  = log(p / (1-p)) where p is the probability of a positive outcome.

**How to check this assumption:**The easiest way to see if this assumption is met is to use a Box-Tidwell test.

1. **Sufficiently large sample size**

Logistic regression assumes that the sample size of the dataset if large enough to draw valid conclusions from the fitted logistic regression model.

**How to check this assumption:**As a rule of thumb, you should have a minimum of 10 cases with the least frequent outcome for each explanatory variable. For example, if you have 3 explanatory variables and the expected probability of the least frequent outcome is 0.20, then you should have a sample size of at least (10\*3) / 0.20 = **150**.

## Assumptions of Logistic Regression vs. Linear Regression

In contrast to linear regression, logistic regression does not require:

* A linear relationship between the explanatory variable(s) and the response variable.
* The residuals of the model to be normally distributed.
* The residuals to have constant variance, also known as [homoscedasticity](https://www.statology.org/heteroscedasticity-regression/).

## Definition

First let’s define what is Classification

Classification: A classification problem is when the output variable is a category, such as “Red” or “blue”, “disease” or “no disease”.

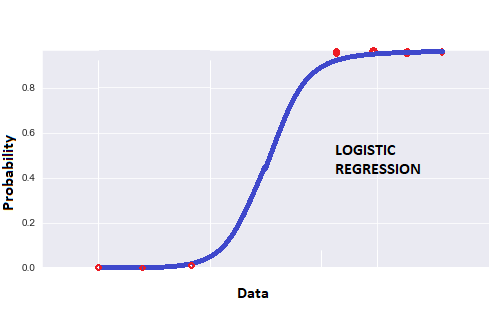
Basically, binary classification problem in which the target variable has only 2 possible values, or in other words, two classes. Some examples of binary classification are –

1. A bank wants to predict, based on some variables, whether a particular customer will default on a loan or not
2. A factory manager wants to predict, based on some variables, whether a particular machine will break down in the next month or not

Logistic Regression is a classification algorithm of Machine Learning where the output variable is categorical. It falls under the Supervised Learning method where the past data with labels is used for building the machine learning model.

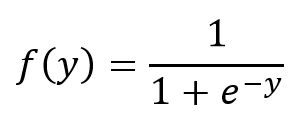
Now why the name Logistic Regression and not Logistic Classification? Essentially Logistic Regression model outputs probabilities (or log odds ratios in the logit form) that have a linear relationship with the predictor variables. When you attach a threshold to these probability values, it classifies the outcomes as 1 or 0 (Binomial Logistic Regression). Hence even if Logistic Regression is a classification algorithm, it has the word regression in it.

## Sigmoid Curve and Log odds



The S-shaped curve shown in the figure above is a sigmoid curve. Logistic regression function is also called sigmoid function. The expression for logistic regression function is :

Lets start with Binary classification which is the problem we are trying to solve using Logistic Regression.



Where:

**y = β0 + β1x (**in case of univariate Logistic regression)

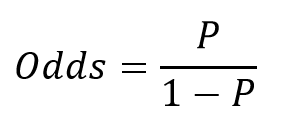
**y = β0 + β1x1 + β2x2 … +βnxn** (in case of multivariate logistic regression)

Univariate Logistic Regression means the output variable is predicted using only one predictor variable, while Multivariate Logistic Regression means output variable is predicted using multiple predictor variables.

The logistic regression function converts the values of **logits** also called **log-odds**that range from **−∞ to +∞** to a range between**0 and 1**.

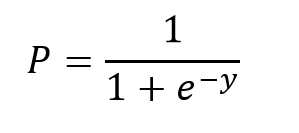
Now let us try to simply what we said. Let **P** be the probability of occurrence of an event. So probability the event will not occur is **1-P.**

**Odds** is defined as the ratio of the probability of occurrence of a particular event to the probability of the event not occurring.



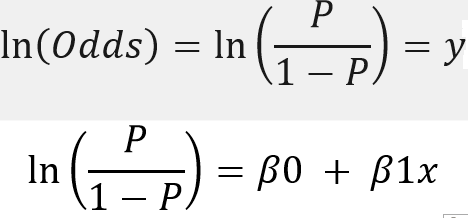
Expression for Odds

We know that logistic regression function gives us probability value. So we can write :

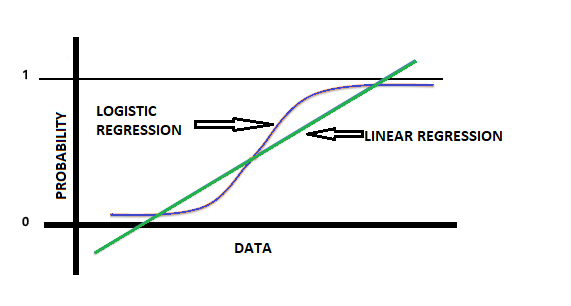


Simplified expression of Logistic regression function

Now since we mentioned **log odds**, let us take the natural log of both sides of the **Odds** equation and substitute the value of **P.**



Thus we get a more simplified form of logistic regression function equation and we can say that **log odds** has linear relationship with the predictor variable **x**.



Thus we get a more simplified form of logistic regression function equation and we can say that **log odds** has linear relationship with the predictor variable **x**.

When it comes to binomial classification (0/1), we need to create a boundary between the values that are classified as 0 or 1. In linear regression, we know that the output is a continuous variable, so drawing a straight line to create this boundary seems infeasible as the values may go from −∞ to +∞.

Since the logistic regression model outputs probabilities with sigmoid function, that can be mapped to 0 or 1, it is preferred over linear regression in case of classification.

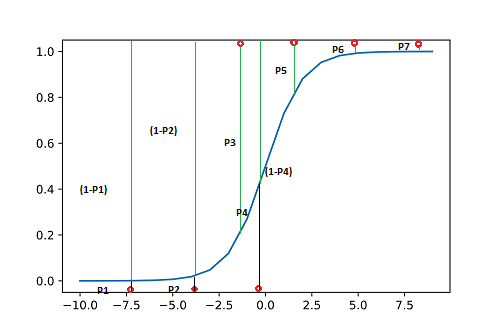
## Maximum Likelihood Estimation

In order that our model predicts output variable as 0 or 1, we need to find the best fit sigmoid curve, that gives the optimum values of beta co-efficients. That is we need to create an efficient boundary between the 0 and 1 values.

Now a cost function tells you how close your values are from actual. So here we need a cost function which maximizes the likelihood of getting desired output values. Such a cost function is called as **Maximum Likelihood Estimation (MLE)**function.

Let this cost function be represented as P(Y ; z). We have some sample data points represented by **Y**. So **Y** represents ‘n’ observations say Y1, Y2….Yn. We need to find this unknown parameter **z,** such that the probability of observing **Y**is maximized. **z**is the bias which decides whether the variable will take a value of 0 or 1.

Remember this is a supervised learning algorithm. So we will have actual observations and predicted observations. Our cost function should be such that it maximizes the probability of predicted values being close to the actual ones.



Sigmoid curve and probabilities

From the above figure, we can see the points being classified as 0 or 1 and the respective probabilities associated with them. There are 7 points and seven associated probabilities P1 to P7.

For points to be 0, we need the probabilities P1, P2 and P4 to be as minimum as possible and for points to be 1, we need the probabilities P3, P5, P6 and P7 to be as high as possible, for correct classification. We can also say that (1-P1), (1-P2), P3, (1-P4), P5, P6 and P7 should be as high as possible.

The joint probability is nothing but the product of probabilities. So the product :[ **(1-P1)\*(1-P2)\* P3\*(1-P4)\*P5\*P6\*P7 ]**should be maximum.

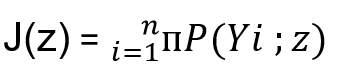
This joint probability function is nothing but our cost function which should be maximized in order to get a best fit sigmoid curve. Or we can say predicted values to be close to the actual values.

This process, where you vary the betas, until you find the best fit curve for probability of diabetes**, is called logistic regression.**

Now coming to our cost function, let **J(z)** be a function of **z** such that

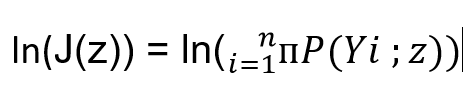
**J(z) = P(Y ; z) = P(Y1, Y2 … Yn ; z)**

The assumption here is that all Y are independent.



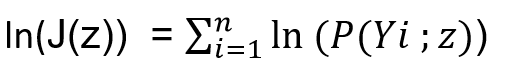
Cost function is product of all probabilities P(Yi)

Taking natural log on both sides :



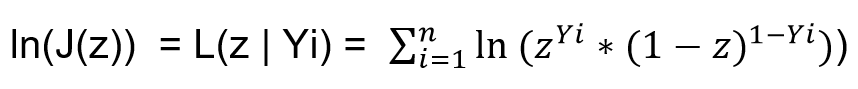
Taking log of both sides

Since log of product becomes summation :

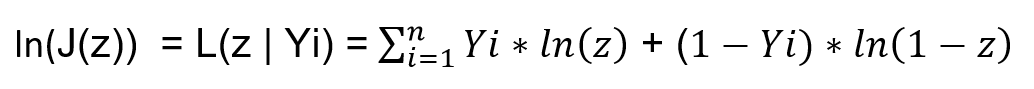


log of product is summation

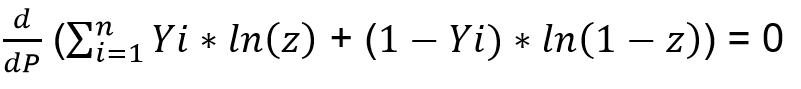
J(z) can also be written as L(z|Yi) (L for Likelihood). For a given value of **z** and observed sample Yi , this function gives the probability of observing the sample values. So if Yi=1 the expression becomes **z** and if Yi is 0 the expression becomes **1-z**:



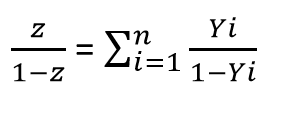
Solving this equation further we get :



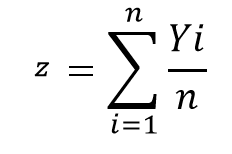
Differentiating this equation with respect to **z** and setting the derivative to zero, we calculate the maxima using closed form solution:



Solving further, this equation becomes:



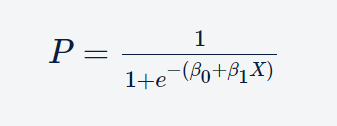
The right side term represents the ratio of number of 1s to number of 0s. Thus the function achieves a maximum at :



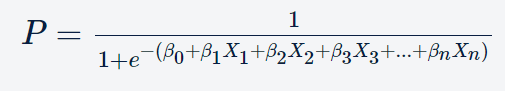
## Multivariate Logistic Regression (Model Building)

Just like when you’re building a model using linear regression, one independent variable might not be enough to capture all the uncertainties of the target variable in logistic regression as well. So, in order to make good and accurate predictions, you need multiple variables and that is what we’ll study in this session.

Before starting with multivariate logistic regression, the first question that arises is, “Do you need any extensions while moving from univariate to multivariate logistic regression?” *Recall* the equation used in the case of univariate logistic regression was:



The above equation has only one feature variable X, for which the coefficient is β1. Now, if you have multiple features, say n, you can simply extend this equation with ‘n’ feature variables and ‘n’ corresponding coefficients such that the equation now becomes:



### Data cleaning and preparation

Before you jump into the actual model building, you first need to clean and prepare your data.

#### Merging

#### Dummy variables (Dummy variable trap)

### *Over Sampling using SMOTE*

#### Missing Values treatment

#### Outliers’ treatment

### Splitting Data into Test, train and Validation set

### Feature Scaling

### Handle class imbalance

### Model Building

#### The first step in model building is to check the correlations between features to get an idea about how the different independent variables are correlated.

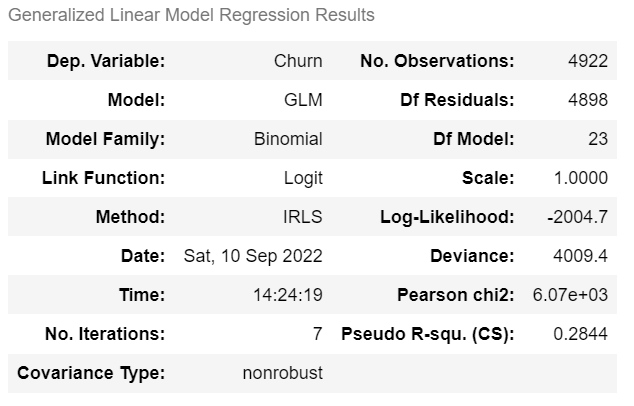
#### Dropping highly correlated dummy variables

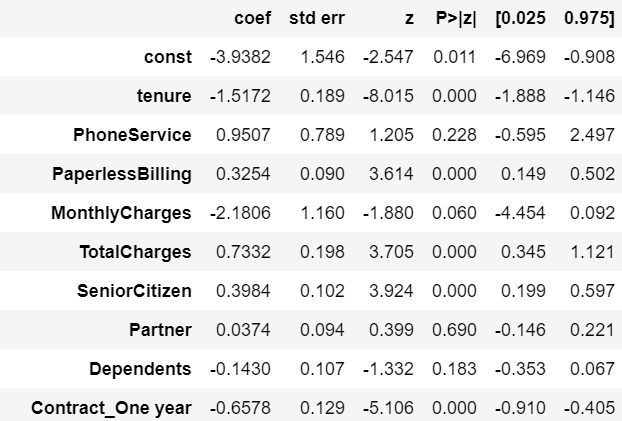
#### Run the model

logm1 = sm.GLM(y\_train,(sm.add\_constant(X\_train)), family = sm.families.Binomial())

#### Model Summary

logm1.fit().summary()





##### Log-likelihood

The **log-likelihood value** of a regression model is a way to measure the goodness of fit for a model. The higher the value of the log-likelihood, the better a model fits a dataset.

The log-likelihood value for a given model can range from negative infinity to positive infinity. The actual log-likelihood value for a given model is mostly meaningless, but **it’s useful for comparing two or more models**.

In practice, we often fit several regression models to a dataset and choose the model with the highest log-likelihood value as the model that fits the data best. When calculating log-likelihood values, it’s important to note that adding more predictor variables to a model will almost always increase the log-likelihood value even if the additional predictor variables aren’t statistically significant.

This means you should only compare the log-likelihood values between two regression models if each model has the same number of predictor variables.

To compare models with different numbers of predictor variables, you can perform a [likelihood-ratio test](https://www.statology.org/likelihood-ratio-test-in-r/) to compare the goodness of fit of two nested regression models.

##### Deviance

##### Chi2

##### p-Value

Variables whose p-values are high imply that that variable is statistically insignificant. So, we need to eliminate some of the variables in order to build a better model

### RFE

After model building, we can see from the summary that there are many variables whose p-values are high, implying that that variable is statistically insignificant. So we need to eliminate some of the variables in order to build a better model.

We'll first eliminate a few features using Recursive Feature Elimination (RFE), and once we have reached a small set of variables to work with, we can then use manual feature elimination (i.e. manually eliminating features based on observing the p-values and VIFs)

RFE will not work with statsmodels ( glm) so we will use

from sklearn.linear\_model import LogisticRegression

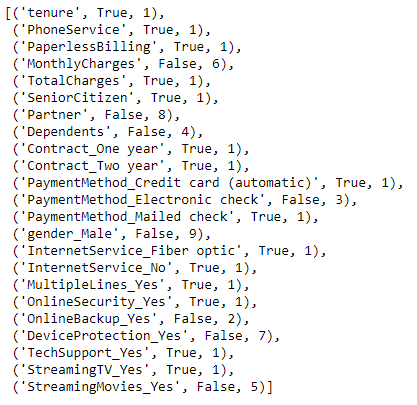
logreg = LogisticRegression()

Then you run an RFE on the dataset using the same command as you did in linear regression. In this case, we choose to select 15 features first (15 is, of course, an arbitrary number).

from sklearn.feature\_selection import RFE

rfe = RFE(logreg, 15) # running RFE with 15 variables as output

rfe = rfe.fit(X\_train, y\_train)



RFE Output

RFE will eliminated certain features from which we will create a new training set, target variable will same.

We decided to go ahead with this model but since we are also interested in the statistics, we take the columns selected by RFE and use them to build a model using statsmodels using:

// Same thing what we have used before to build the model

X\_train\_sm = sm.add\_constant(X\_train[col])

logm2 = sm.GLM(y\_train,X\_train\_sm, family = sm.families.Binomial())

res = logm2.fit()

Here, you use the GLM (Generalized Linear Models) method of the library statsmodels. 'Binomial()' in the 'family' argument tells statsmodels that it needs to fit a logit curve to a binomial data (i.e. in which the target will have just two classes, here 'Churn' and 'Non-Churn').

Now, recall that the logistic regression curve gives you the probabilities of churning and not churning. You can get these probabilities by simply using the 'predict' function as shown in the notebook.

Since the logistic curve gives you just the probabilities and not the actual classification of 'Churn' and 'Non-Churn', you need to find a threshold probability to classify customers as 'churn' and 'non-churn'. Here, we choose 0.5 as an arbitrary cutoff wherein if the probability of a particular customer churning is less than 0.5, you'd classify it as 'Non-Churn' and if it's greater than 0.5, you'd classify it as 'Churn'. The choice of 0.5 is completely arbitrary at this stage and you'll learn how to find the optimal cutoff in **'Model Evaluation'**, but for now, we'll move forward with 0.5 as the cutoff.

### Model Evaluation

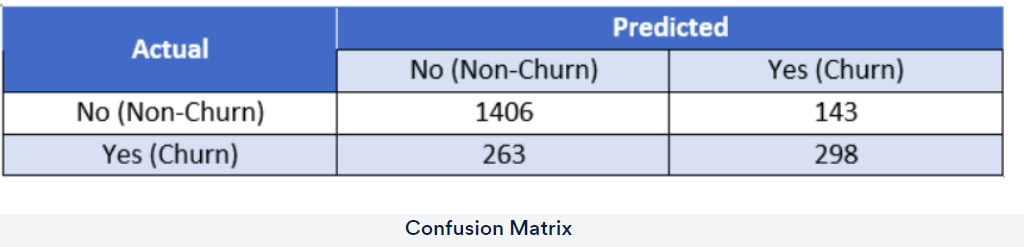
#### Confusion Matrix and Accuracy

You chose a cutoff of 0.5 in order to classify the customers into 'Churn' and 'Non-Churn'. Now, since you're classifying the customers into two classes, you'll obviously have some errors. The classes of errors that would be there are:

'Churn' customers being (incorrectly) classified as 'Non-Churn'

'Non-Churn' customers being (incorrectly) classified as 'Churn'

To capture these errors, and to evaluate how well the model is, you'll use something known as the 'Confusion Matrix'. A typical confusion matrix would look like the following:



This table shows a comparison of the predicted and actual labels. The actual labels are along the vertical axis, while the predicted labels are along the horizontal axis. Thus, the second row and first column (263) is the number of customers who have actually ‘churned’ but the model has predicted them as non-churn.

Similarly, the cell at second row, the second column (298) is the number of customers who are actually ‘churn’ and also predicted as ‘churn’.

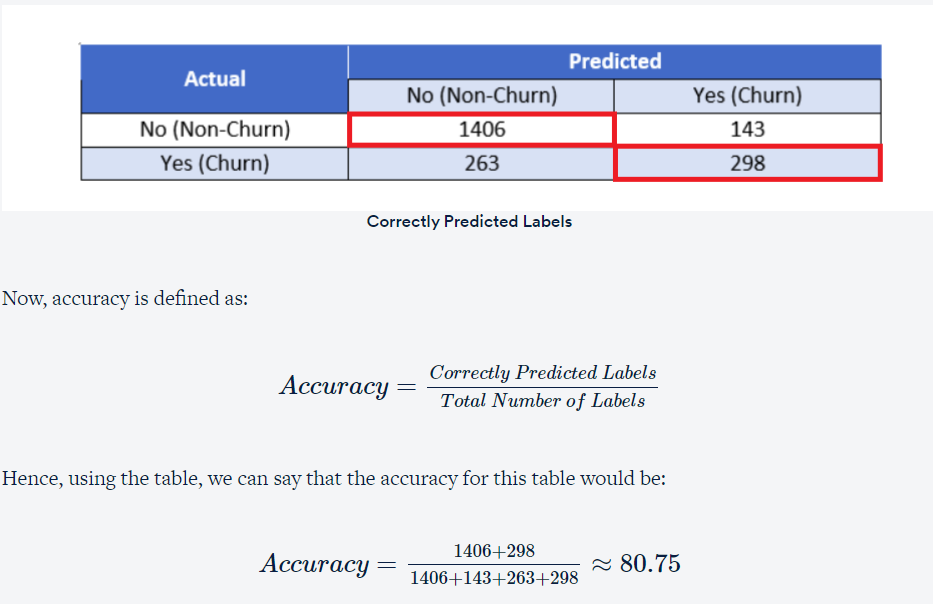
Note that this is an example table and not what you'll get in Python for the model you've built so far. It is just used an example to illustrate the concept.

Now, the simplest model evaluation metric for classification models is accuracy - it is the percentage of correctly predicted labels. So what would the correctly predicted labels be? They would be:

'Churn' customers being actually identified as churn

'Non-churn' customers being actually identified as non-churn.

As you can see from the table above, the correctly predicted labels are contained in the first row and first column, and the last row and last column as can be seen highlighted in the table below:



The steps you need to calculate accuracy are:

* Create the confusion matrix
* Calculate the accuracy by applying the 'accuracy\_score' function to the above matrix

The code used to do this was:

# Create confusion matrix

confusion = metrics.confusion\_matrix(y\_train\_pred\_final.Churn, y\_train\_pred\_final.predicted)

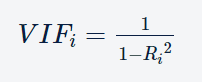
# Calculate accuracy

print(metrics.accuracy\_score(y\_train\_pred\_final.Churn, y\_train\_pred\_final.predicted))

So far you have only selected features based on RFE. Further elimination of features using the p-values and VIFs manually is yet to be done. You'll do that in the next section.

### Manual Feature elimination using p-values and VIF

Recall that you had used RFE to select 15 features. But as you saw in the pairwise correlations, there are high values of correlations present between the 15 features, i.e. there is still some multicollinearity among the features. So you definitely need to check the VIFs as well to further eliminate the redundant variables. Recall that VIF calculates how well one independent variable is explained by all the other independent variables combined. And its formula is given as:



where 'i' refers to the ith variable which is being represented as a combination of rest of the independent variables.

Variance inflation factor (VIF) is a measure of the amount of multicollinearity in a set of multiple regression variables. Mathematically, the VIF for a regression model variable is equal to the ratio of the overall model variance to the variance of a model that includes only that single independent variable. This ratio is calculated for each independent variable. A high VIF indicates that the associated independent variable is highly collinear with the other variables in the model.

* A variance inflation factor (VIF) provides a measure of multicollinearity among the independent variables in a multiple regression model.
* Detecting multicollinearity is important because while multicollinearity does not reduce the explanatory power of the model, it does reduce the statistical significance of the independent variables.
* A large variance inflation factor (VIF) on an independent variable indicates a highly collinear relationship to the other variables that should be considered or adjusted for in the structure of the model and selection of independent variables.