Polycystic Ovary Syndrome Analysis and Prediction

Shrayan Roy

November 6, 2022

Roll No : MD2220

Guide : Dr. Deepayan Sarkar

Introduction:

Polycystic Ovary Syndrome (PCOS) is a condition in which the ovaries produce an abnormal ammount of androgens, male sex hormones that are usually present in women in small ammounts. The name polysystic ovary syndrome decsribes the numerous small cysts (fluid filled sacs) that form in the ovaries. However, some women with disorder do not have cysts, while some women without the disorder do develop cysts.

Here, we will analyze a data to understand which are the possible influencers of PCOS. The dataset includes several variable, which are suspected to influence chances Polycystic ovary syndrome (PCOS).

Data Description:

We have used the dataset available in Kaggle. The link to the dataset `https://www.kaggle.com/datasets/prasoonkottarathil/polycystic-ovary-syndrome-pcos`. The data is collect from 10 different hospital across Kerala,India. It has 541 rows and 44 columns.

```
## 'data.frame': 541 obs. of  44 variables:
##  $ Sl..No          : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Patient_File_No : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ PCOS            : int  0 0 1 0 0 0 0 0 0 0 ...
##  $ Age             : int  28 36 33 37 25 36 34 33 32 36 ...
##  $ Weight          : num  44.6 65 68.8 65 52 74.1 64 58.5 40 52 ...
##  $ Height          : num  152 162 165 148 161 ...
##  $ BMI             : num  19.3 24.9 25.3 29.7 20.1 ...
##  $ Blood_Group     : Factor w/ 8 levels "11","12","13",..: 5 5 1 3 1 5 1 3 1 5 ...
##  $ Pulse_rate      : int  78 74 72 72 72 78 72 72 72 80 ...
##  $ RR              : int  22 20 18 20 18 28 18 20 18 20 ...
##  $ Hb              : num  10.5 11.7 11.8 12 10 ...
##  $ Cycle           : int  2 2 2 2 2 2 2 2 2 4 ...
##  $ Cycle_length    : int  5 5 5 5 5 5 5 5 5 2 ...
##  $ Marriage_Status : num  7 11 10 4 1 8 2 13 8 4 ...
##  $ Pregnant        : int  0 1 1 0 1 1 0 1 0 0 ...
##  $ No_of_aborptions: int  0 0 0 0 0 0 0 2 1 0 ...
##  $ I_beta_HCG      : num  1.99 60.8 494.08 1.99 801.45 ...
##  $ II_beta_HCG     : num  1.99 1.99 494.08 1.99 801.45 ...
##  $ FSH             : num  7.95 6.73 5.54 8.06 3.98 3.24 2.85 4.86 3.76 2.8 ...
##  $ LH              : num  3.68 1.09 0.88 2.36 0.9 1.07 0.31 3.07 3.02 1.51 ...
##  $ FSH_LH_Ratio    : num  2.16 6.17 6.3 3.42 4.42 ...
##  $ Hip             : int  36 38 40 42 37 44 39 44 39 40 ...
##  $ Waist           : int  30 32 36 36 30 38 33 38 35 38 ...
##  $ Waist_Hip_Ratio : num  0.833 0.842 0.9 0.857 0.811 ...
##  $ TSH             : num  0.68 3.16 2.54 16.41 3.57 ...
##  $ AMH            : num  2.07 1.53 6.63 1.22 2.26 6.74 3.05 1.54 1 1.61 ...
##  $ PRL            : num  45.2 20.1 10.5 36.9 30.1 ...
##  $ Vit_D3         : num  17.1 61.3 49.7 33.4 43.8 52.4 42.7 38 21.8 27.7 ...
##  $ PRG            : num  0.57 0.97 0.36 0.36 0.38 0.3 0.46 0.26 0.3 0.25 ...
##  $ RBS            : num  92 92 84 76 84 76 93 91 116 125 ...
##  $ Weight_gain    : int  0 0 0 0 0 1 0 1 0 0 ...
##  $ hair_growth    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Skin_darkening : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Hair_loss      : int  0 0 1 0 1 1 0 0 0 0 ...
##  $ Pimples        : int  0 0 1 0 0 0 0 0 0 0 ...
##  $ Fast_food      : int  1 0 1 0 0 0 0 0 0 0 ...
##  $ Reg_Exercise   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ BP_Systolic    : int  110 120 120 120 120 110 120 120 120 110 ...
##  $ BP_Diastolic   : int  80 70 80 70 80 70 80 80 80 80 ...
##  $ Follicle_No_L  : int  3 3 13 2 3 9 6 7 5 1 ...
##  $ Follicle_No_R  : int  3 5 15 2 4 6 6 6 7 1 ...
##  $ Avg_F_size_L   : num  18 15 18 15 16 16 15 15 17 14 ...
##  $ Avg_F_size_R   : num  18 14 20 14 14 20 16 18 17 17 ...
##  $ Endometrium    : num  8.5 3.7 10 7.5 7 8 6.8 7.1 4.2 2.5 ...
```

From the above description of the data, we can see the column names. Let us describe them (in brief) one by one -

- **PCOS** : It indicates whether the patient has PCOS or not. '1' indicates that the patient has PCOS and '0' doesnot have PCOS.
- **Age** : Age of the pateint in years.
- **Weight** : Weight of patient in kgs.
- **BMI** : BMI of the patient.
- **Blood_Group** : Blood Group of patient. A+ = 11, A- = 12, B+ = 13, B- = 14, O+ = 15, AB+ = 17, AB- = 18
- **Pulse_rate** : Pulse_rate of the patient in bpm.
- **RR** : Respiratory rate in breaths/min.
- **Hb** : Haemoglobin in g/dl.
- **Cycle** : '2' if patient has Regular period and '4' if patient has Irregular period.
- **Cycle_length** : Cycle length of the patient.
- **Marriage_Status**: Number of years of marriage of the patient.
- **Pregnant** : '1' if the patient is pregnant and '0' if not pregnant.

- **No_of_aborptions**: Number of aborptions of the patient.
- **I_beta_HCG , II_beta_HCG** : A positive beta human chorionic gonadotropin (HCG) level usually means that you are pregnant. Pregnancy tests detect the HCG hormone in the blood and urine. These columns are level of I_beta_HCG & II_beta_HCG of the patient in mIU/mL.
- **FSH, LH & FSH_LH_Ratio** : Follicle stimulating hormone (FSH), Luteinizing hormone (LH) levels (in mIU/mL) and their ratio of the patient.
- **Hip, Waist & Waist_Hip_Ratio** : Hip, Waist (in inch) and their ratio of the patient.
- **TSH** : Thyroid stimulating hormone of the patient in mIU/L
- **AMH, PRL** : Anti-müllerian hormone (AMH) is made in the reproductive tissues of both males and females. The role of AMH and whether levels are normal depend on your age and gender. AMH plays an important role in the development of sex organs in an unborn baby. On the otherhand, Prolactin is a hormone made by the pituitary gland, a small gland at the base of the brain. Prolactin causes the breasts to grow and make milk during pregnancy and after birth. These columns are level of AMH, PRL in ng/mL.
- **Vit_D3** : Vitamin D3 of the patient in ng/mL.
- **PRG** : Serum progesterone test result of the patient in ng/mL.
- **RBS** : Random blood sugar (RBS) measures blood glucose regardless of when you last ate. this column represents the RBS of the patient in mg/dL.
- **Weight_gain** : Whether the patient has weight gain in a specific time period accoring to the rule of the study. '1' if weight gain, '0' if no weight gain.
- **hair_growth** : Whether the patient has hair_growth in a specific time period accoring to the rule of the study. '1' if hair growth, '0' if no hair growth.
- **Skin_darkening** : Whether the patient has skin darkening. '1' if skin darkening, '0' if no skin darkening.
- **Hair_loss** : Whether the patient has hair loss in a specific time period accoring to the rule of the study. '1' if hair loss, '0' if no hair loss.
- **Pimples** : Whether the patient has pimples. '1' if pimples, '0' if no pimples.

- **Fast_food** : Whether the patient has eat fast food in a specific time period accoring to the rule of the study. '1' if eat fast food, '0' otherwise.
- **Reg_Exercise** : Whether the patient do regular exercise . '1' regular exercise, '0' otherwise.
- **Bp_Systolic , Bp_Diastolic** : Bp_Systolic and Diastolic of the patient in mmHg.
- **Follicle_No_L, Follicle_No_R** : Follicle is a small, fluid-filled sac in the ovary that contains one immature egg. These columns indicates the number of follicles of the patient in the left and right ovary.
- **Avg_F_size_L, Avg_F_size_R** : Average follicle size of the patient in the left and right ovary, measured in mm.
- **Endometrium** : The thickness layer of tissue that lines the uterus in mm.

Data Processing and Cleaning :

We need to process our data. Our data contains NA values also.

```
sum(is.na(PCOSdata))

## [1] 4
```

So, we delete those rows corresponding to NA values using na.omit() function. Also, we have some unsual rows. For example, Cycle column has value 5, which has no meaning, Vitamin D3 of a patient 0, Age , Height, Weight. After removing Them, we are left with 533 rows. Also, we encoded the Cycle column as - '0' if regular period and '1' if irregular period.

```
PCOSdata$Cycle <- ifelse(PCOSdata$Cycle == 2,0,1)
```

Some Preliminary Analysis :

We will do some preliminary analysis of the data, that we have obtained after data processing and cleaning. First we draw the correlation heatmap of the numerical variables excluding the factor Blood group.

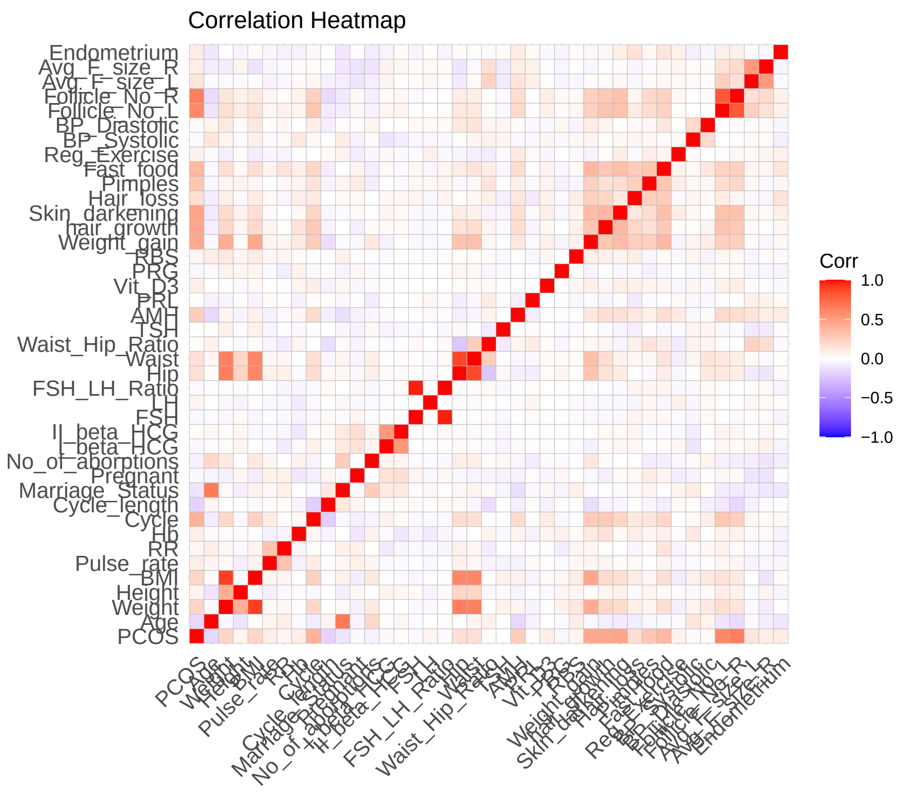
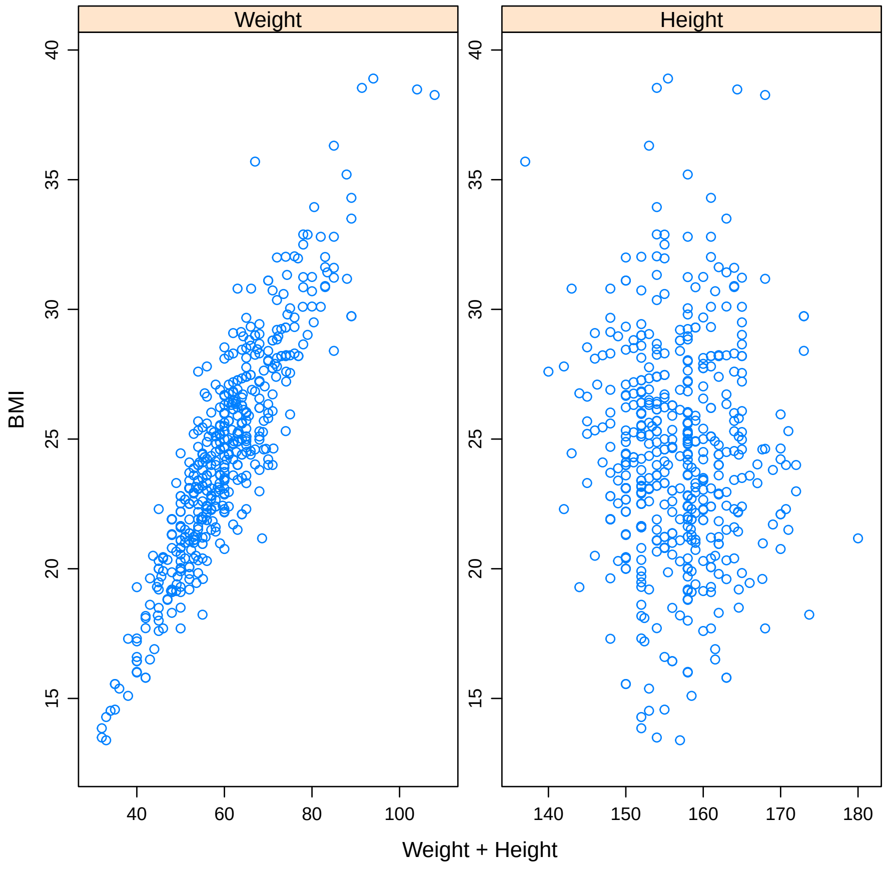
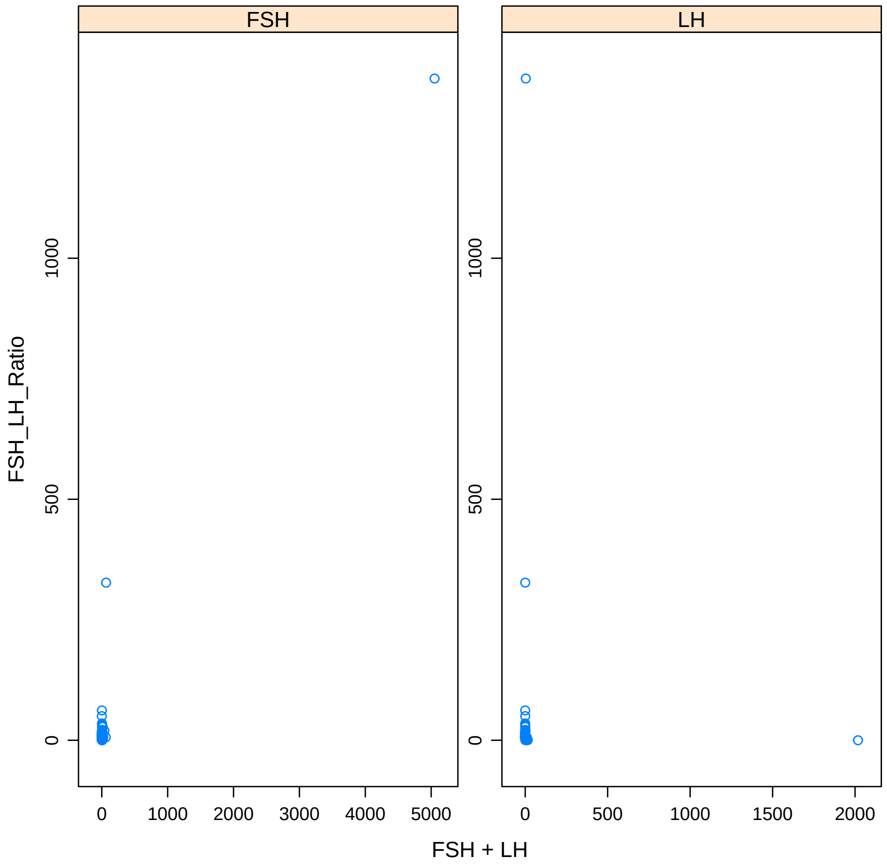
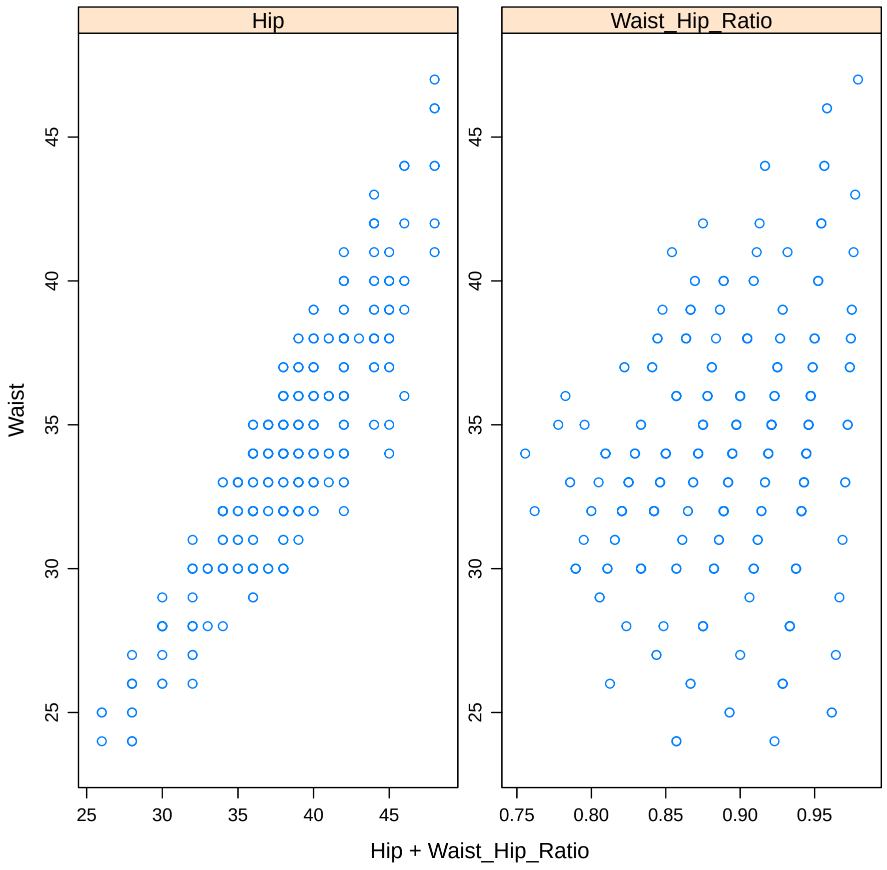


Diagram : Correlation heatmap of the remaining variables

From the above graph we can see that, the (BMI, Weight) ,(FSH, FSH_LH ratio) ,(Waist, Hip), (Follicle_No_L, Follicle_No_R) are highly correlated. Since, BMI is a deterministic function of Height, Weight. So, We prefer to delete BMI column. Because, it will introduce multicollinearity in the model. It can be seen that, if we would have add BMI in the model, then the corresponding Generalized Variance Inflation Factor(GVIF) will be high.


Weight
Height
BMI
Weight + Height
40
35
30
25
20
15
40
60
80
100
140
150
160
170
180


FSH
LH
FSH_LH_Ratio
0
500
1000
0
1000
2000
3000
4000
5000
0
500
1000
1500
2000
FSH + LH



Similarly for FSH_LH ratio also. Also, Hip and Waist are highly correlated but they are not much correlated with Waist_Hip_Ratio. Which are seen from the above graphs also. So, we exclude Hip from the model.

So, finally we will work with the the following dataset.

```
#So we will delete one of the variables Waist and Hip.
#Since, Waist,Hip, Waist_hip Ratio  #carries same type of information.
PCOSdata <- PCOSdata[,!colnames(PCOSdata)%in%c('FSH_LH_Ratio','Hip','BMI')]
dim(PCOSdata)    #Remaining dataset dimension

## [1] 533  41
```

Exploratory Data Analysis :

Exploratory data analysis (EDA) is an approach of analyzing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods. It is the most important step in any data analysis. Here, we will perform EDA to get insight of the dataset.

First we will plot histogram and boxplot of the numerical variables.

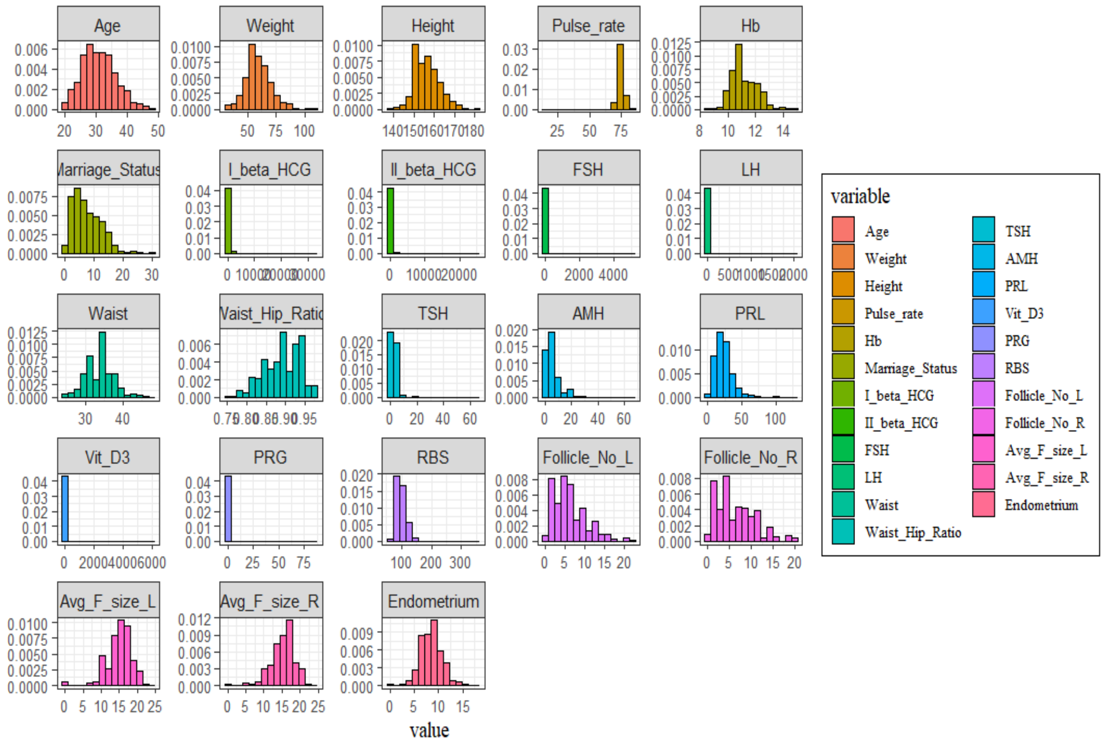


Diagram : Histogram of Numerical Variables

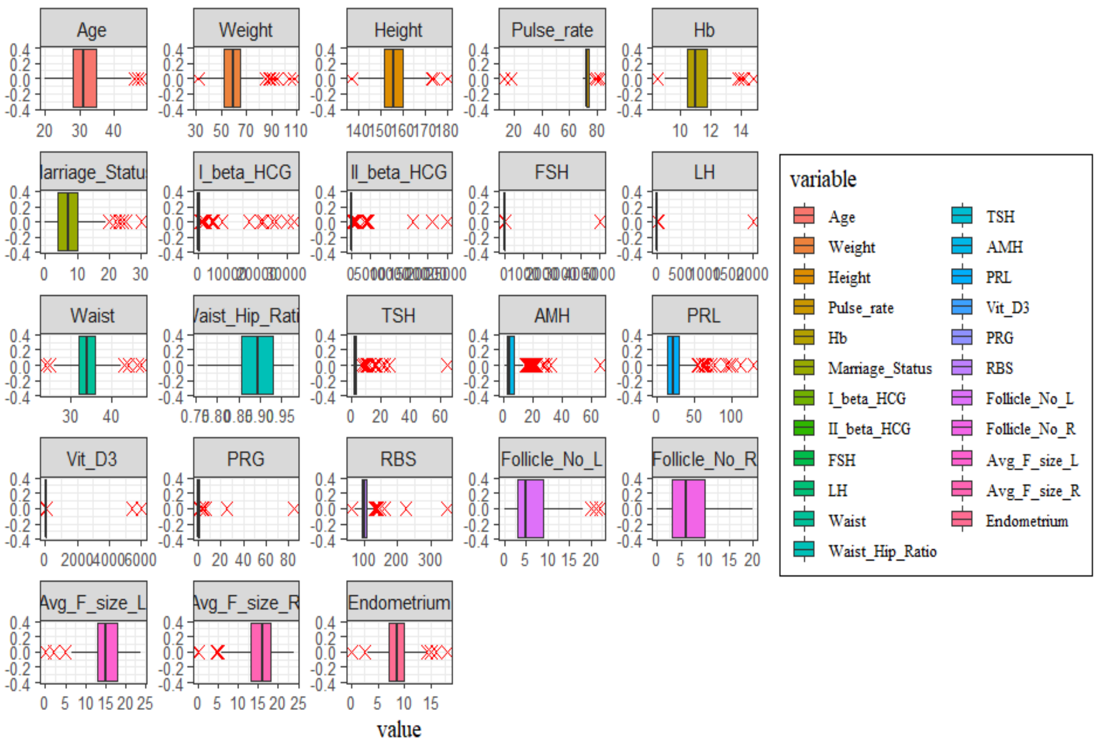


Diagram : Boxplot of Numerical Variables

So, from the above graphs we can see that there are outliers in the data. For example - in **Marriage_Status, I_beta_HCG, II_beta_HCG, TSH, AMH, PRL** there are lots of outliers. So, we have to be careful while analyzing the data. Because, these outliers can influence our analysis. Let's look at the summary of this variables.

```
##        Age            Weight            Height          Pulse_rate
##  Min.   :20.00   Min.   : 32.00   Min.   :137.0   Min.   :13.00
##  1st Qu.:28.00   1st Qu.: 52.00   1st Qu.:152.0   1st Qu.:72.00
##  Median :31.00   Median : 59.60   Median :156.0   Median :72.00
##  Mean   :31.43   Mean   : 59.74   Mean   :156.5   Mean   :73.24
##  3rd Qu.:35.00   3rd Qu.: 65.00   3rd Qu.:160.0   3rd Qu.:74.00
##  Max.   :48.00   Max.   :108.00   Max.   :180.0   Max.   :82.00
##        Hb        Marriage_Status    I_beta_HCG        II_beta_HCG
##  Min.   : 8.50   Min.   : 0.000   Min.   :    1.30   Min.   :    0.99
##  1st Qu.:10.50   1st Qu.: 4.000   1st Qu.:    1.99   1st Qu.:    1.99
##  Median :11.00   Median : 7.000   Median :   20.00   Median :    1.99
##  Mean   :11.16   Mean   : 7.705   Mean   :  672.27   Mean   :  240.81
##  3rd Qu.:11.80   3rd Qu.:10.000   3rd Qu.:  297.21   3rd Qu.:   99.69
##  Max.   :14.80   Max.   :30.000   Max.   :32460.97   Max.   :25000.00
##       FSH                LH               Waist       Waist_Hip_Ratio
##  Min.   :   0.21   Min.   :   0.02   Min.   :24.00   Min.   :0.7556
##  1st Qu.:   3.34   1st Qu.:   1.05   1st Qu.:32.00   1st Qu.:0.8571
##  Median :   4.86   Median :   2.30   Median :34.00   Median :0.8947
##  Mean   :  14.77   Mean   :   6.54   Mean   :33.84   Mean   :0.8918
##  3rd Qu.:   6.44   3rd Qu.:   3.68   3rd Qu.:36.00   3rd Qu.:0.9286
##  Max.   :5052.00   Max.   :2018.00   Max.   :47.00   Max.   :0.9792
##       TSH               AMH               PRL              Vit_D3
##  Min.   : 0.040   Min.   : 0.100   Min.   :  0.40   Min.   :   6.077
##  1st Qu.: 1.480   1st Qu.: 2.010   1st Qu.: 14.52   1st Qu.:  20.800
##  Median : 2.270   Median : 3.700   Median : 21.92   Median :  26.000
##  Mean   : 2.974   Mean   : 5.582   Mean   : 24.40   Mean   :  50.326
##  3rd Qu.: 3.570   3rd Qu.: 6.900   3rd Qu.: 29.97   3rd Qu.:  34.500
##  Max.   :65.000   Max.   :66.000   Max.   :128.24   Max.   :6014.660
##       PRG                RBS         Follicle_No_L    Follicle_No_R
##  Min.   : 0.0470   Min.   : 60.00   Min.   : 0.000   Min.   : 0.000
##  1st Qu.: 0.2500   1st Qu.: 92.00   1st Qu.: 3.000   1st Qu.: 3.000
##  Median : 0.3200   Median :100.00   Median : 5.000   Median : 6.000
##  Mean   : 0.6146   Mean   : 99.86   Mean   : 6.101   Mean   : 6.642
##  3rd Qu.: 0.4600   3rd Qu.:107.00   3rd Qu.: 9.000   3rd Qu.:10.000
##  Max.   :85.0000   Max.   :350.00   Max.   :22.000   Max.   :20.000
##   Avg_F_size_L  Avg_F_size_R    Endometrium
##  Min.   : 0    Min.   : 0.00   Min.   : 0.000
##  1st Qu.:13    1st Qu.:13.00   1st Qu.: 7.000
##  Median :15    Median :16.00   Median : 8.500
##  Mean   :15    Mean   :15.45   Mean   : 8.461
##  3rd Qu.:18    3rd Qu.:18.00   3rd Qu.: 9.800
##  Max.   :24    Max.   :24.00   Max.   :18.000
```

From the summary also, we can see that these variables have outliers. Now, we will draw barplot of the remaining variables. But before that, lets see the proportion of patient have PCOS in our dataset.

```
mean(PCOSdata[,3] == 1)

## [1] 0.3227017
```

So, 0.3227017 proportion of patient have PCOS in our dataset. Not, a very imbalanced dataset. So, we can carry forward our analysis.

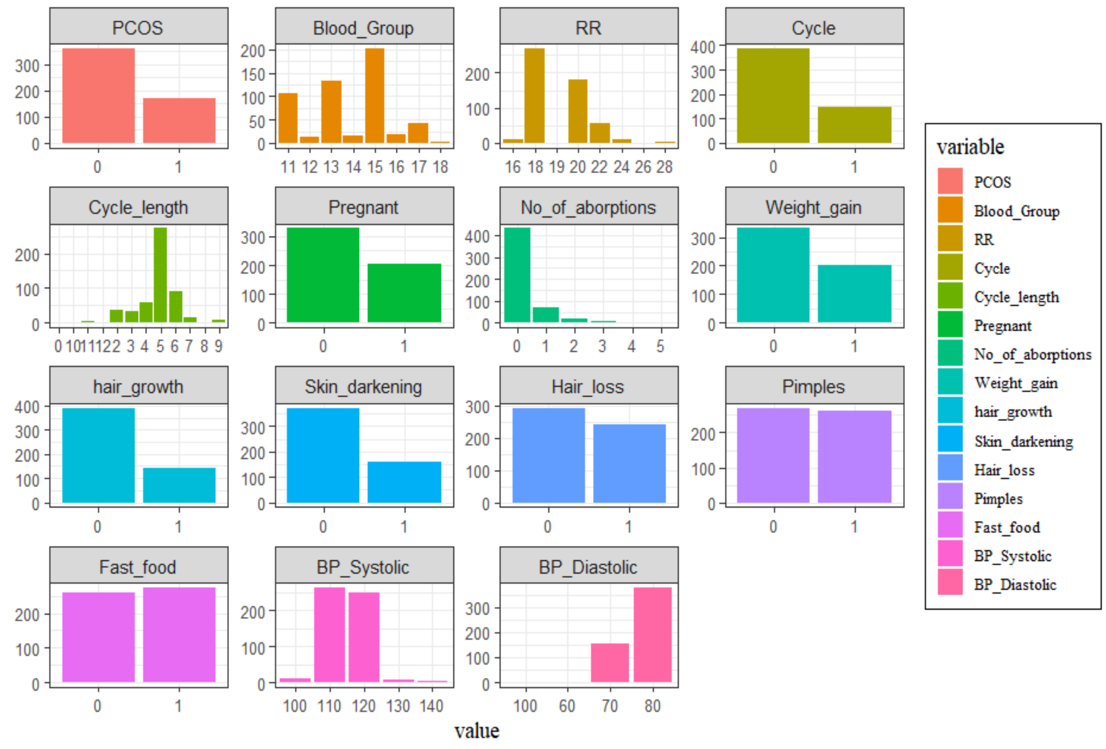


Diagram : Barplot of Categorical Variables

From the above diagram, we can see that most of the patients in our data have blood group O+, cycle length 5, No of aborptions 0.

Let's look at the pairwise scatterplot of the of the numerical variables with PCOS.

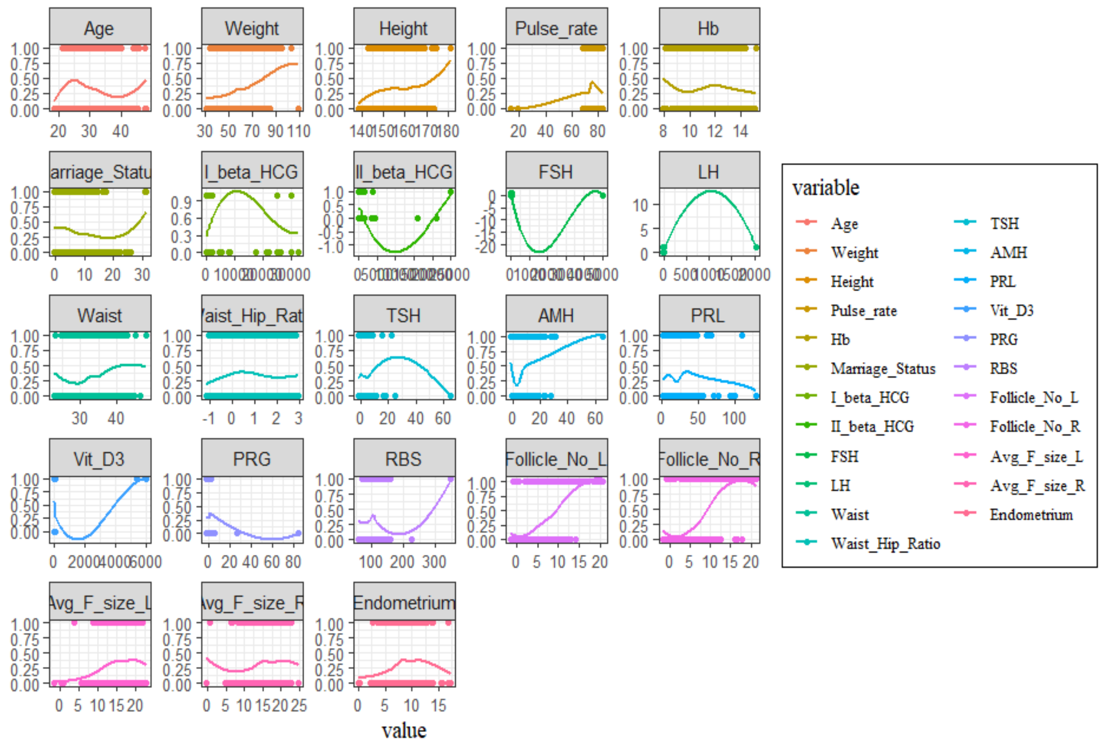


Diagram : Pairwise of Scatterplot Numerical variables with PCOS

The curvy lines is essentially a loess fit of the variables with PCOS, It indicates the effect of having PCOS. For example - Most of the patient who have PCOS have age between 20 - 35 approximately. Because, the fitted loess curve is near 1. Similarly, Most of patient who have PCOS are overweight. Also, Most of patient who have PCOS have more **Follicle_No_L** and **Follicle_No_R** than usual. Similarly, we can interpret for other variables also.

Now, we will draw a paiwise of barplot of categorical variables.

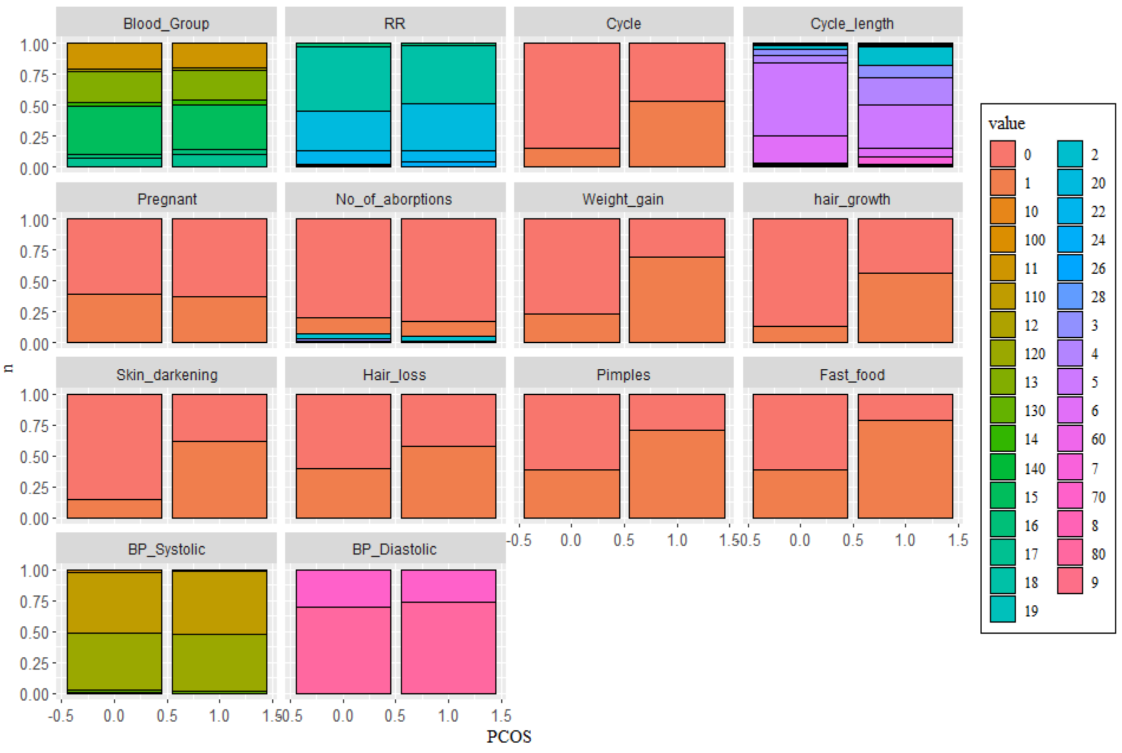


Diagram : Pairwise Barplot of Categorical Variables with PCOS

From the above barplot, we can see that most of people who have PCOS have irregular cycle. Also, most of the people who have PCOS, have **weight gain**, **hair growth**, **skin darkening**, hair loss, pimples and eat fast food. Now, we will again look at the correlation heatmap of the remaining variables.

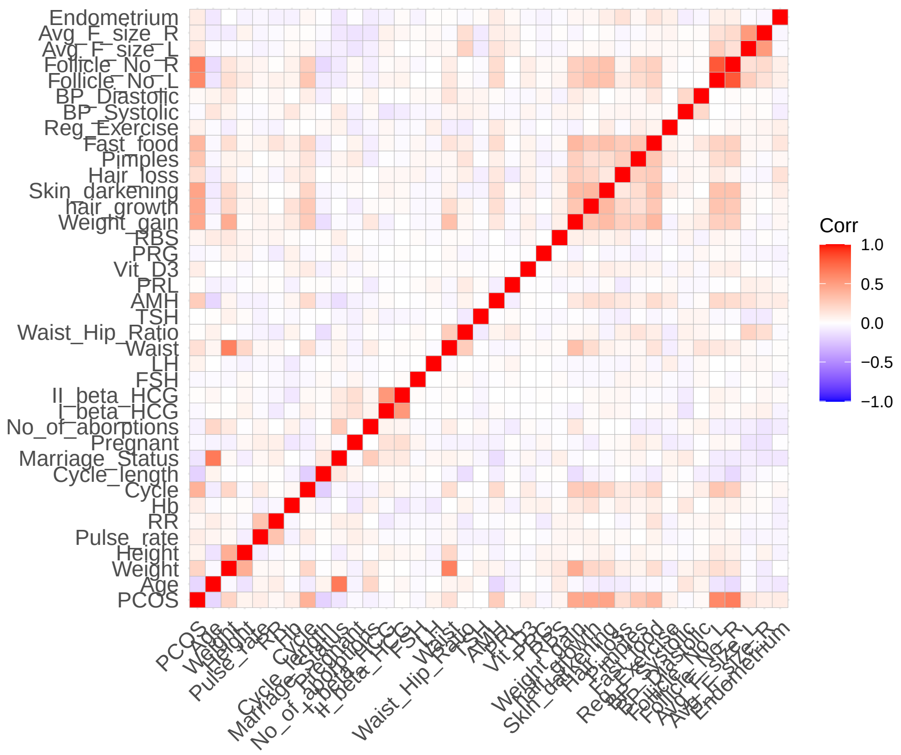


Diagram : Correlation heatmap of the remaining variables

From the above diagram, we can see that the off-diagonal elements of correlation heatmap are light coloured. Which indicates that the independent variables in our dataset are not much correlated. Which was not in our previous correlation heatmap.

Now, we will try to understand whether we can split in to two parts the feature space. That is, whether the features that the PCOS patient have and the non PCOS patient have can be separate out in a well manner. An easy way out can be plotting the Principles Components of the independent variables. This is not a very right approach. But, we can still apply this to get some insight. So, we plot first three principle components.

```
with(PCA,scatterplot3d(PC1, PC2, PC3,color = ifelse(PCOSdata[,3]==0,'yellow','cyan'),
    pch = 19,main = '3D plot of First Three Principle Components'))
```

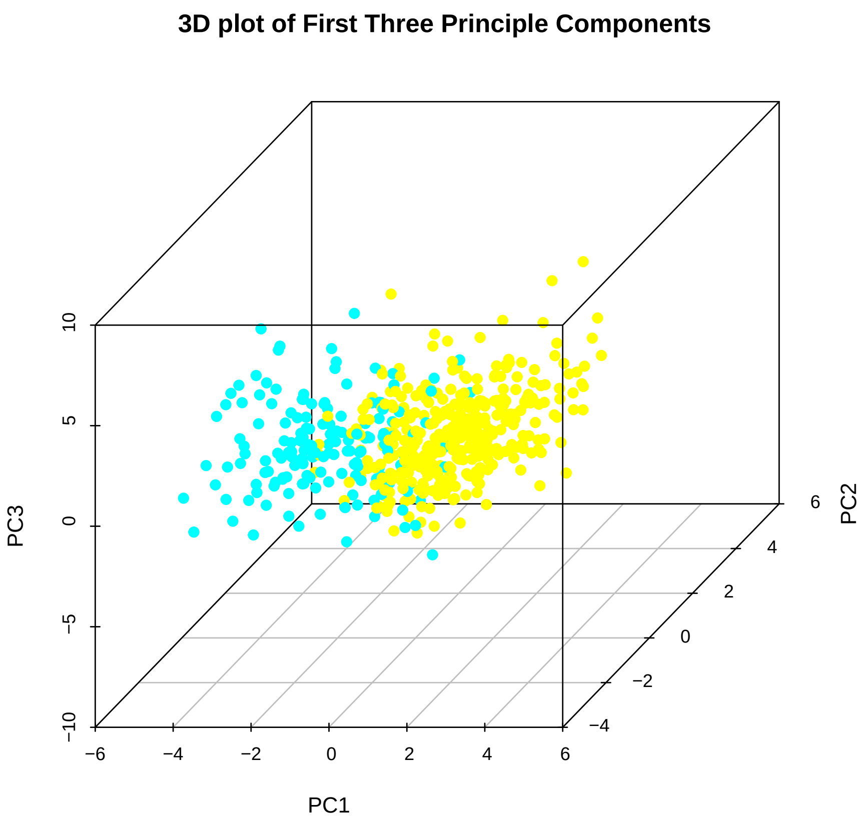


Diagram : 3d scatterplot of First Three PC

The above diagram clearly indicates that on the basis of the given variables, we can atleast make good models to predict whether patient has PCOS or not.

So, from the above EDA we have learned that -

- Some of indepent variables have outliers.
- **Age, Weight, Cycle, Weight gain, Hair growth, Skin darkening, Hair loss, Pimples, Fast Food**, **Follicle_No_L** and **Follicle_No_R** are important variables to influence chance of PCOS.

Now, we will analyze the data. But before that we will describe the methods that we will use in our analysis.

Methods Used :

Our main objective is

- To understand how the given variables influence the chance of PCOS - Inference
- Given the values of the variables, we will predict whether the patient has PCOS or not - Prediction

For that we will fit several models.

- Logistic Regression
- Robust Logistic Regression
- Lasso Logistic Regrssion
- K - Nearest Neighbour Method (KNN)
- Random Forest (RF)
- Support Vector Machine (SVM)
- Extreme Gradient Boosting (Xg Boost)

Also, we will use several Evaluation metrics to compare them.

- Accuracy
- Precision
- Specificity
- Sensitivity
- Precision
- LogLoss

We will give short description of all of them in a while. But before that we will describe some important points.

Whether a patient has PCOS or not, is essentially a classification problem. In classification problem, the possible values for the target variables are discrete, and we call these possible values "classes". Basically, we are interested in constructing a function $h(x)$ from a dataset $X = ((x_1, t_1), ..., (x_N, t_N))$ that yields prediction values x.The objective in classification is the same, except the values of t are discrete. There are three different types of classifiers.

- **Generative classifiers** that modek the joint probability distribution of the input and target variables $Pr(x, t)$.
- **Discriminative classifiers** that model the conditional probability distribution of the target given input variables $Pr(t \mid x)$.
- **Distribution-free classifiers** that do not use a probability model but directly assign input to target variables.

Logistic Regression :

We will use Logistic Regression for our purpose.

Robust Logistic Regression :

Our dataset contains lots of outliers we prefer to use robust methods. The most common approach is to use bounded influence estimators. There are several methods developed for logistic regression, like the Optimal Bias-Robust Estimator (OBRE) of K¨unsch et al. (1989), the Bianco and Yohai estimator (1996) or the Mallows-type estimator of Cantoni and Ronchetti (2001). Here, we will mainly focus on Mallows-type estimator for logistic regression.The Mallows-type estimator of Cantoni and Ronchetti (2001) is defined for the class of generalized linear models. They defined some estimating equations which nicely extend the likelihood equations. Such estimating equations can be written as -

$$g(\beta; y) = \sum_{i=1}^{n} w(xi) \frac{(\psi_k(r_i) - a(\mu_i))}{Vi(\mu i)^{1/2}} \frac{\partial \mu_i}{\partial \beta}$$

Where, $V(\mu_i)$ is the variance function, $r_i = \frac{(y_i - \mu_i)}{\sqrt{V_i}}$ the Pearson residuals and & ψ_k the Huber function . The non-random values $a(\mu_i)$ are defined as $a(\mu i)$ $= E[\psi_k(R_i)|x_i]$, and ensure the conditional Fisher-consistency of the estimating equations. For binomial or Poisson response the computation of $a(\mu i)$ is not difficult, as reported in Cantoni and Ronchetti (2001).

For our case $\mu_i = F(x_i^t \beta)$,with $F(u) = \frac{exp(u)}{(1+exp(u))}, V_i(\mu_i) = \mu_i(1 - \mu_i) = V_i$.

$$a(\mu i) = \psi_k((1 - \mu_i)/\sqrt{V_i})\mu_i + \psi_k(-\mu i\sqrt{V_i})(1 - \mu_i).$$

Solving, $g(\beta; y) = 0$, we will get estimate of β. This is how we do Robust Logistic Regression.

Lasso Logistic Regression :

We can perform Logistic Regression using Lasso. The L1 penalty used in the lasso can be used for variable selection and shrinkage with any linear regression model. For logistic regression, we would maximize :

$$L(\beta) - \lambda \sum_{j=1}^{p} |\beta_j|$$

Where, $L(\beta)$ is log likelihood in two class case with p variables. As with the lasso, we typically do not penalize the intercept term, and standardize the predictors for the penalty to be meaningful. A solution can be found using nonlinear programming methods (*Koh et al., 2007, for example*). Alternatively, using the same quadratic approximations that were used in the Newton algorithm. Chossing optimal value of lambda is done as it is in Linear Regression.

K - Nearest Neighbour Method (KNN) :

We will use K - Nearest Neighbour Method for our purpose.

Random Forest (RF) :

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. As the name suggests, "*Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.*" Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting. Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

Step 1: In Random forest n number of random records are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.

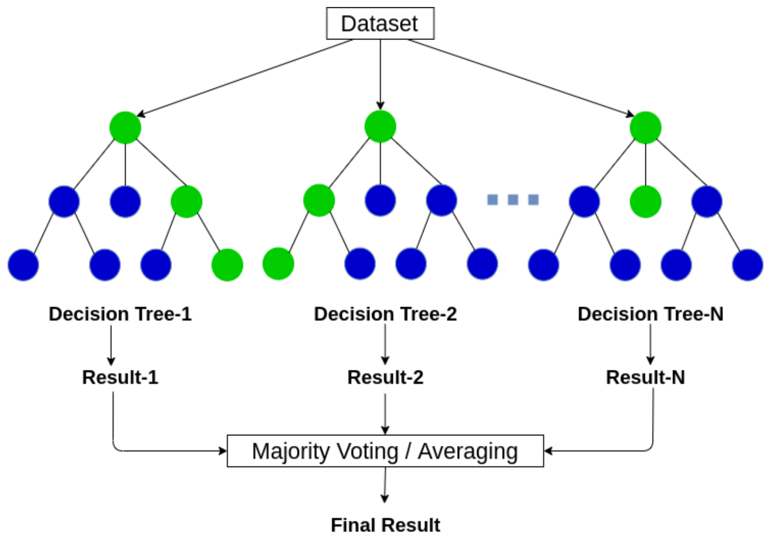


Diagram : Working Procedure of Random Forest

Support Vector Machine (SVM) :

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane. SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. SVM can be of two types. However, here we will use Linear SVM. Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

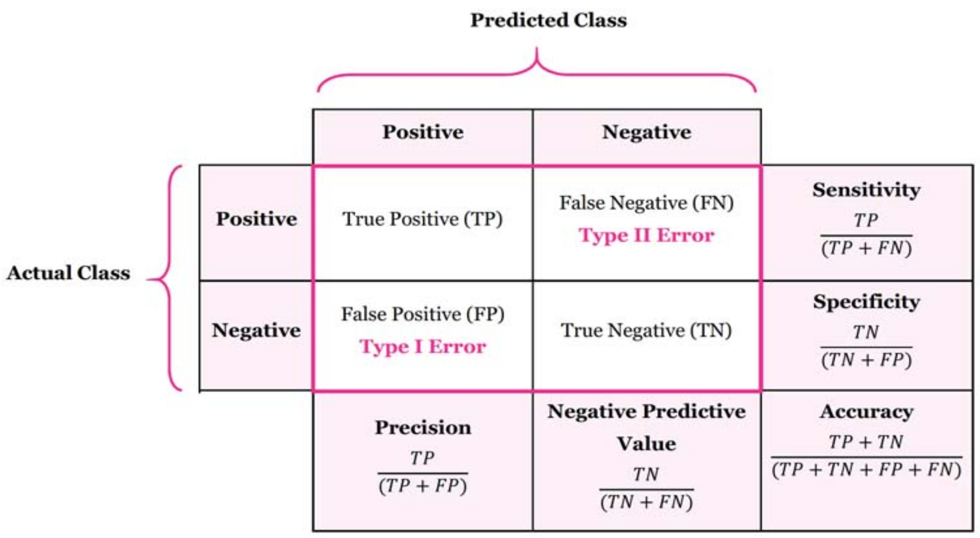
Extreme Gradient Boosting (Xg Boost) :

GBoost is an implementation of Gradient Boosted decision trees. This library was written in C++. It is a type of Software library that was designed basically to improve speed and model performance. It has recently been dominating in applied machine learning. XGBoost models majorly dominate in many Kaggle Competitions. In this algorithm, decision trees are created in sequential form. Weights play an important role in XGBoost. Weights are assigned to all the independent variables which are then fed into the decision tree which predicts results. The weight of variables predicted wrong by the tree is increased and the variables are then fed to the second decision tree. These individual classifiers/predictors then ensemble to give a strong and more precise model. It can work on regression, classification, ranking, and user-defined prediction problems.

Now, we will define the metrics which we have used.

Confusion Matrix :

A confusion matrix is a table that is used to define the performance of a classification algorithm. A confusion matrix visualizes and summarizes the performance of a classification algorithm. Suppose, on the basis of a data we have fitted a model. Then, using that model, we have obtain the fitted values of y. We denote these fitted values of y by $\hat{y}$. Then, the confusion matrix for the classification algorithm is given by -



		Predicted Class		
		Positive	**Negative**	
Actual Class	**Positive**	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

These measures has several interpretations and its own use depending upon context of the problem.

On the otherhand Log loss corresponding to a model is defined as -

$$-\frac{1}{n}\sum_{i=1}^{n}\{y_i log(\hat{\pi}_i) + (1-y_i)log(1-\hat{\pi}_i)\}$$

Where, $\hat{\pi}_i$is estimate of π_i using the model and y_i is the value of the i^{th} value of response. Note that, we can use this measure only when using our model we can find estimate of π_i.The less the value of log loss the more model is good is the sense that we have minimizing our *cross entropy loss.*

Now, we are ready to start our analysis.

Model Fitting :

First, we will split the data into two parts. Testing and training set. Basically, we are doing this to better understand the predictive power of the model that we are using.

```
#Test Data and Train Data
set.seed(seed = 987654321)
index_train <- sample(1:nrow(PCOSdata),size = floor(nrow(PCOSdata)*80/100),
                      replace = F)
PCOSdata_train <- PCOSdata[index_train,]
PCOSdata_test <- PCOSdata[-index_train,]
dim(PCOSdata_train)

## [1] 426  41
```

```
dim(PCOSdata_test)

## [1] 107  41
```

Now, we will fit our models on the training set and assess the prediction power using test set.

Base line Logistic Regression :

We have fitted logistic regression on the whole training dataset.

```
full.model <- glm(PCOS ~ . ,data = PCOSdata_train[,-c(1,2)],
                   family = binomial(link = "logit"))
summary(full.model)

##
## Call:
## glm(formula = PCOS ~ ., family = binomial(link = "logit"), data = PCOSdata_train[,
##     -c(1, 2)])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9295  -0.2144  -0.0551   0.0599   3.4796
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -1.830e+01  1.521e+01  -1.203 0.229041
## Age                 1.138e-02  6.272e-02   0.181 0.856067
## Weight              2.832e-02  3.940e-02   0.719 0.472208
## Height              3.577e-02  5.247e-02   0.682 0.495488
## Blood_Group12      -1.498e+00  1.380e+00  -1.085 0.277810
## Blood_Group13      -1.992e-01  8.008e-01  -0.249 0.803585
## Blood_Group14       2.095e+00  1.289e+00   1.625 0.104094
## Blood_Group15      -3.632e-01  6.978e-01  -0.521 0.602686
## Blood_Group16      -1.716e-02  1.380e+00  -0.012 0.990082
## Blood_Group17      -3.525e-01  1.098e+00  -0.321 0.748074
## Blood_Group18      -2.786e+00  6.624e+00  -0.421 0.674016
## Pulse_rate          2.262e-01  1.128e-01   2.005 0.044973 *
## RR                 -2.191e-01  1.685e-01  -1.300 0.193470
## Hb                 -1.528e-01  3.218e-01  -0.475 0.634928
## Cycle               7.020e-01  6.115e-01   1.148 0.250932
## Cycle_length       -2.198e-01  1.763e-01  -1.247 0.212359
## Marriage_Status    -1.144e-01  7.497e-02  -1.526 0.126974
## Pregnant           -6.143e-01  4.999e-01  -1.229 0.219169
## No_of_aborptions   -1.389e-01  4.517e-01  -0.308 0.758461
## I_beta_HCG         -3.935e-05  5.200e-05  -0.757 0.449298
## II_beta_HCG       1.221e-04  3.756e-04   0.325 0.745130
## FSH               7.335e-03  7.894e-02   0.093 0.925972
## LH                4.012e-02  1.046e-01   0.384 0.701298
## Waist            -1.766e-02  9.135e-02  -0.193 0.846731
## Waist_Hip_Ratio  -7.989e+00  5.803e+00  -1.377 0.168573
## TSH               1.047e-01  5.646e-02   1.855 0.063588 .
## AMH               1.412e-02  4.031e-02   0.350 0.726007
## PRL               7.295e-03  1.907e-02   0.383 0.702057
## Vit_D3            5.695e-04  1.849e-03   0.308 0.758063
## PRG              -7.737e-02  8.441e-01  -0.092 0.926968
## RBS               1.033e-02  1.948e-02   0.530 0.595948
## Weight_gain       2.009e+00  6.465e-01   3.107 0.001889 **
## hair_growth       1.965e+00  5.722e-01   3.433 0.000596 ***
## Skin_darkening    1.228e+00  5.109e-01   2.404 0.016227 *
## Hair_loss         2.457e-01  5.111e-01   0.481 0.630639
## Pimples           1.167e+00  5.345e-01   2.184 0.028957 *
## Fast_food         1.205e+00  5.715e-01   2.108 0.035030 *
## Reg_Exercise      6.733e-01  5.707e-01   1.180 0.238134
## BP_Systolic      -5.568e-02  4.166e-02  -1.337 0.181352
## BP_Diastolic      2.428e-02  5.951e-02   0.408 0.683261
## Follicle_No_L     9.570e-02  1.041e-01   0.919 0.357852
## Follicle_No_R     5.256e-01  1.072e-01   4.902 9.47e-07 ***
## Avg_F_size_L      2.177e-01  1.116e-01   1.951 0.051046 .
## Avg_F_size_R      5.133e-03  1.017e-01   0.050 0.959749
## Endometrium       2.563e-02  1.152e-01   0.223 0.823913
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 535.11  on 425  degrees of freedom
## Residual deviance: 152.44  on 381  degrees of freedom
## AIC: 242.44
##
## Number of Fisher Scoring iterations: 10
```

So, most of the variables are coming out to be insignificant using Wald's test. Also, Value of AIC is 242.44. Let's look at the VIF's of the model.

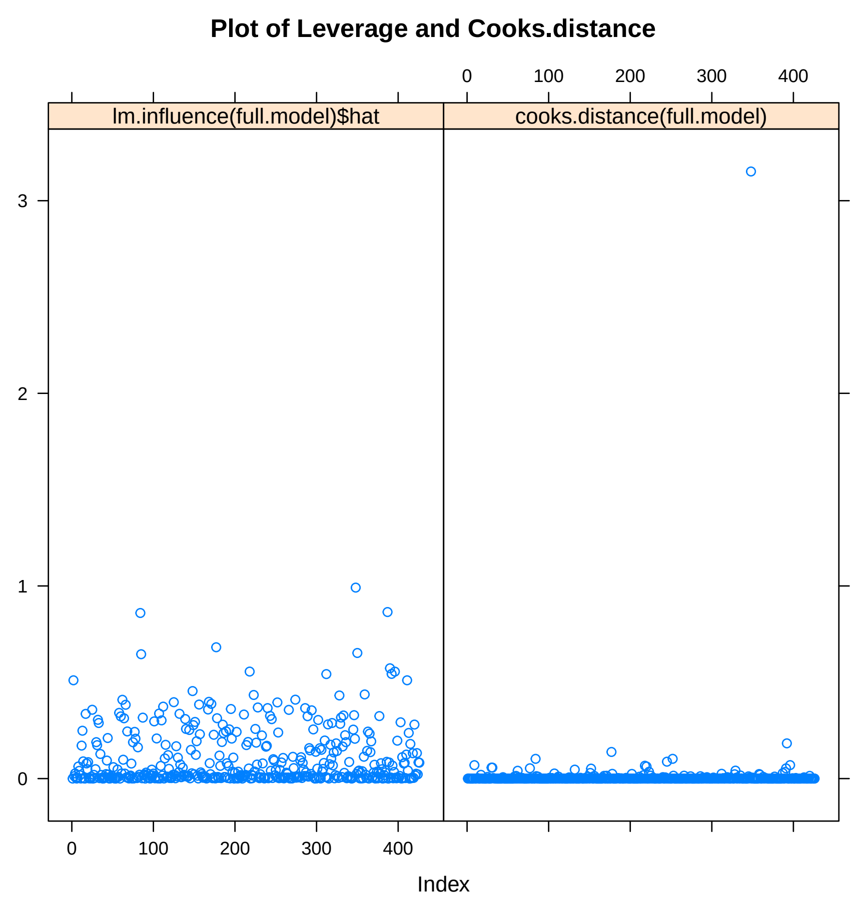
```
car::vif(full.model)
```

```
##                      GVIF Df GVIF^(1/(2*Df))
## Age              2.704900  1        1.644658
## Weight           3.357882  1        1.832452
## Height           1.970352  1        1.403692
## Blood_Group      5.873210  7        1.134801
## Pulse_rate       2.095365  1        1.447538
## RR               1.921325  1        1.386119
## Hb               1.521245  1        1.233388
## Cycle            1.905064  1        1.380241
## Cycle_length     1.498129  1        1.223981
## Marriage_Status  2.658606  1        1.630523
## Pregnant         1.386451  1        1.177477
## No_of_aborptions 1.413759  1        1.189016
## I_beta_HCG       1.544241  1        1.242675
## II_beta_HCG      1.138964  1        1.067223
## FSH              1.367357  1        1.169340
## LH               1.726943  1        1.314132
## Waist            2.539746  1        1.593658
## Waist_Hip_Ratio  1.696019  1        1.302313
## TSH              1.293925  1        1.137508
## AMH              1.413353  1        1.188845
## PRL              1.305584  1        1.142622
## Vit_D3           1.043670  1        1.021602
## PRG              1.255193  1        1.120354
## RBS              1.325575  1        1.151336
## Weight_gain      2.373626  1        1.540658
## hair_growth      1.620272  1        1.272899
## Skin_darkening   1.373928  1        1.172147
## Hair_loss        1.468470  1        1.211804
## Pimples          1.605456  1        1.267066
## Fast_food        1.760511  1        1.326842
## Reg_Exercise     1.485827  1        1.218945
## BP_Systolic      1.445363  1        1.202233
## BP_Diastolic     1.708960  1        1.307272
## Follicle_No_L    2.893874  1        1.701139
## Follicle_No_R    3.452757  1        1.858160
## Avg_F_size_L     2.928776  1        1.711367
## Avg_F_size_R     2.338115  1        1.529089
## Endometrium      1.347443  1        1.160794
```

All values are quite small. Hence, we can safely say that there is no multicollinearity in data. But it would not be the case if we fit in the presence of **FSH_LH_Ratio, Hip, BMI**. Let's look at it.

```
##                        GVIF Df GVIF^(1/(2*Df))
## Age                2.509263  1        1.584065
## Weight           274.388287  1       16.564670
## Height            60.199714  1        7.758847
## BMI              229.941188  1       15.163812
## Blood_Group        4.283499  7        1.109503
## Pulse_rate         2.075148  1        1.440537
## RR                  1.775090  1        1.332325
## Hb                  1.520208  1        1.232967
## Cycle               1.743661  1        1.320478
## Cycle_length        1.363744  1        1.167794
## Marriage_Status     2.702846  1        1.644033
## Pregnant            1.282811  1        1.132612
## No_of_aborptions    1.384124  1        1.176488
## I_beta_HCG          1.681849  1        1.296861
## II_beta_HCG         1.500314  1        1.224873
## FSH                 1.415675  1        1.189821
## LH                  1.954476  1        1.398026
## FSH_LH_Ratio        1.682267  1        1.297022
## Hip               451.810684  1       21.255839
## Waist             490.843851  1       22.154996
## Waist_Hip_Ratio   112.709317  1       10.616464
## TSH                 1.227407  1        1.107884
## AMH                 1.421770  1        1.192380
## PRL                 1.213530  1        1.101603
## Vit_D3              1.058341  1        1.028757
## PRG                 1.162071  1        1.077994
## RBS                 1.248215  1        1.117236
## Weight_gain         1.913270  1        1.383210
## hair_growth         1.604492  1        1.266686
## Skin_darkening      1.533723  1        1.238436
## Hair_loss           1.515139  1        1.230910
## Pimples             1.441166  1        1.200486
## Fast_food           1.442601  1        1.201083
## Reg_Exercise        1.541070  1        1.241398
## BP_Systolic         1.408424  1        1.186771
## BP_Diastolic        1.382148  1        1.175648
## Follicle_No_L       2.196717  1        1.482133
## Follicle_No_R       2.768927  1        1.664009
## Avg_F_size_L        2.331722  1        1.526998
## Avg_F_size_R        2.194684  1        1.481446
## Endometrium         1.319003  1        1.148479
```

That’s why we prefer to delete columns and by doing this we will not loss any information as such. Let’s look at the diagnostic plots of the model.



Let's see how many $h_i$values are greater than the usual threshold.

```
sum(lm.influence(full.model)$hat > 2*(ncol(PCOSdata_train) - 2 + 1)/nrow(PCOSdata))

## [1] 113
```

So many points are leverage points !. Let's see which are influential points. If we look at the plot of cooks distance, then we observe that only one point is essentially far away from rest of the points.

```
sum(cooks.distance(full.model) > 1)

## [1] 1
```

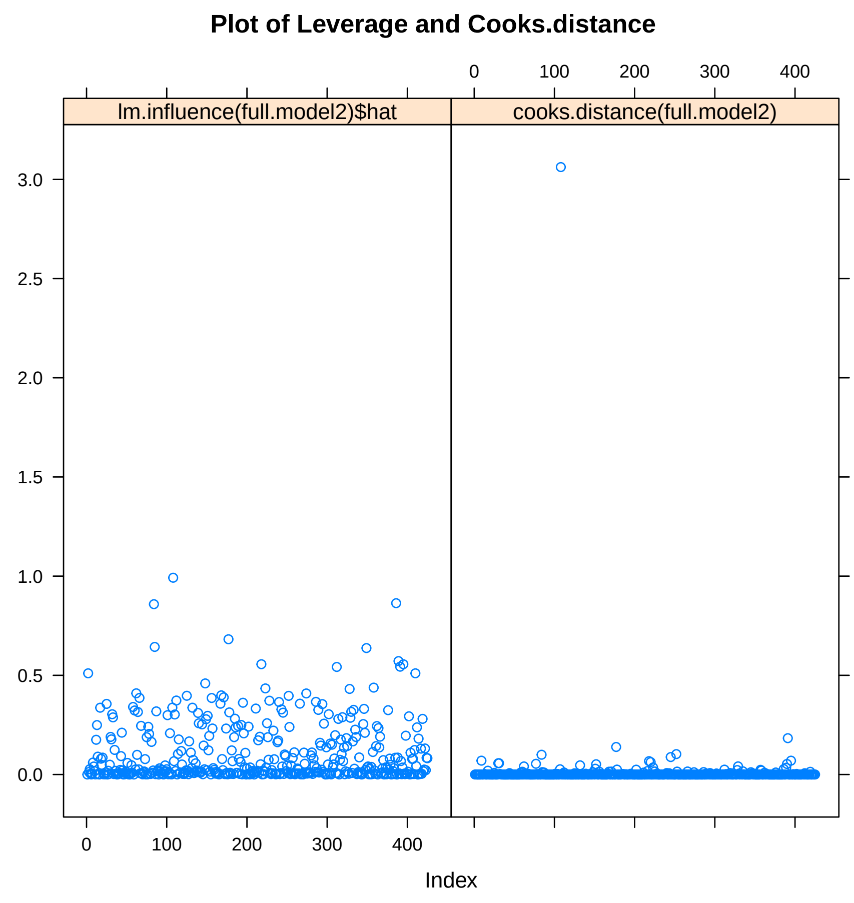
Let's see what is that point.

```
PCOSdata_train[(cooks.distance(full.model) > 1),]

##     Sl..No Patient_File_No PCOS Age Weight Height Blood_Group Pulse_rate RR
## 196    196             196    1  35     60  153.4          13         72 20
##       Hb Cycle Cycle_length Marriage_Status Pregnant No_of_aborptions
## 196 13.2     1            4              14        0                1
##     I_beta_HCG II_beta_HCG FSH   LH Waist Waist_Hip_Ratio  TSH  AMH   PRL
## 196       1.99        1.99  22 3.39    35       0.9210526 2.42 6.65 16.34
##     Vit_D3  PRG RBS Weight_gain hair_growth Skin_darkening Hair_loss Pimples
## 196 5418.6 0.31 100           1           0              1         1       1
##     Fast_food Reg_Exercise BP_Systolic BP_Diastolic Follicle_No_L Follicle_No_R
## 196         1            0         120           80             8            10
##     Avg_F_size_L Avg_F_size_R Endometrium
## 196           15           13         5.5
```

Notice the value of Vit D3 is really strange and not very meaningful. So, we delete this point from our training set.

```
#Remove that influential observation
PCOSdata_train <- PCOSdata_train[-which(cooks.distance(full.model) > 1),]

#Re-Fit model Logistic Regression Using all Variable
full.model2 <- glm(PCOS ~ . ,data=PCOSdata_train[,-c(1,2)],
                  family = binomial(link = "logit"))
```

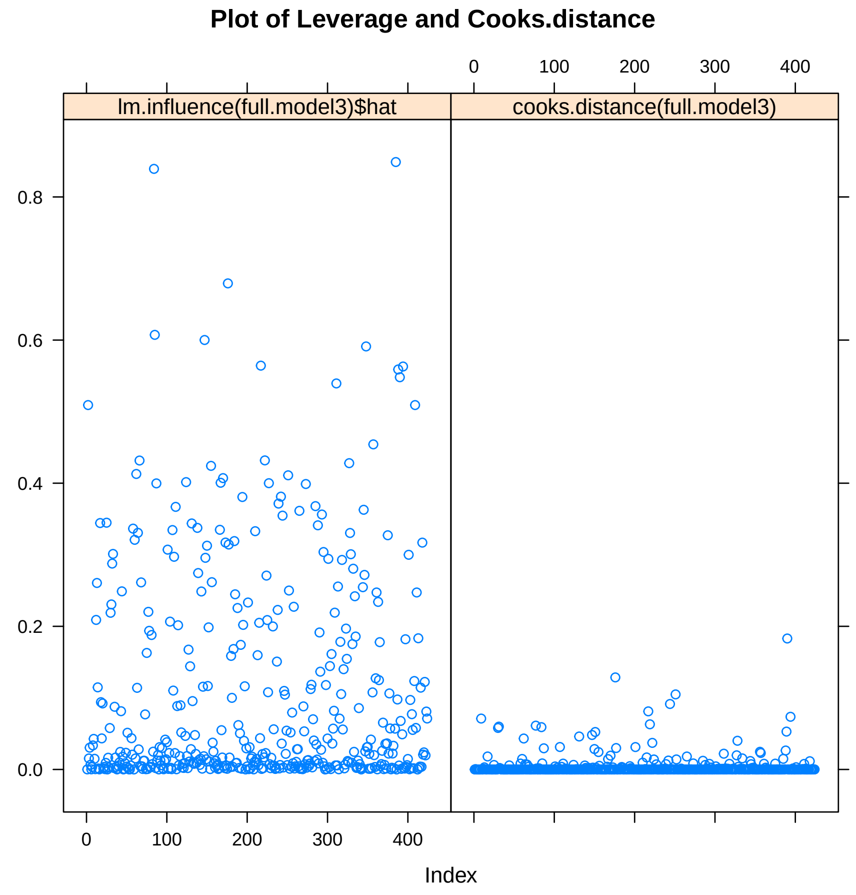
Let's again look at the model diagnostic plots.



If we look at the plot of cooks distance, then we observe that again one point is essentially far away from rest of the points. We look at that observation.

```
PCOSdata_train[(cooks.distance(full.model2) > 1),]

##     Sl..No Patient_File_No PCOS Age Weight Height Blood_Group Pulse_rate RR Hb
## 192    192             192    1  29     63    153          17         74 18 11
##     Cycle Cycle_length Marriage_Status Pregnant No_of_aborptions I_beta_HCG
## 192     1            3               8        0                0       3.99
##     II_beta_HCG  FSH   LH Waist Waist_Hip_Ratio  TSH  AMH   PRL  Vit_D3  PRG
## 192        3.99 3.63 1.02    35       0.8974359 2.66 6.41 29.08 6014.66 0.25
##     RBS Weight_gain hair_growth Skin_darkening Hair_loss Pimples Fast_food
## 192 123           1           1              1         1       1         1
##     Reg_Exercise BP_Systolic BP_Diastolic Follicle_No_L Follicle_No_R
## 192            0         120           70            14            16
##     Avg_F_size_L Avg_F_size_R Endometrium
## 192                16                17                9
```

Notice the value of Vit D3 is really strange and not very meaningful. So, we delete this point from our training set. Then, we will again refit our model and look at the diagnostic plot. This time, there is no unusual point in the cooks distance plot. So, we keep this model as our as final model. We name it in R by final_full.model



Now, we have identified the influential points, which may influence our analysis and removed them. But, still we are stuck at the problem of identifying the important variables or the best model for prediction. We have so many variables. So, we need to somehow find the best set of variables for prediction. For that we will use **Step-wise Regression**.

First, we will do **Forward Selection.** It includes following steps -

- Find best one-variable model
- Find best two-variable model by adding another variable and so on.

- That is, do not look at all two-variable models; only ones that contain the best one-variable model.

Let's do forward selection on our data.

```
library(MASS)
#Forward regression
lm.forward <- stepAIC(glm(PCOS ~. ,data = PCOSdata_train[,-c(1,2)],
              family = binomial(link = 'logit')),direction = 'forward',
              trace = 0)
variable.names(lm.forward)  #which variables are selected

##  [1] "(Intercept)"      "Age"              "Weight"           "Height"
##  [5] "Blood_Group12"    "Blood_Group13"    "Blood_Group14"    "Blood_Group15"
##  [9] "Blood_Group16"    "Blood_Group17"    "Blood_Group18"    "Pulse_rate"
## [13] "RR"               "Hb"               "Cycle"            "Cycle_length"
## [17] "Marriage_Status"  "Pregnant"         "No_of_aborptions" "I_beta_HCG"
## [21] "II_beta_HCG"      "FSH"              "LH"               "Waist"
## [25] "Waist_Hip_Ratio"  "TSH"              "AMH"              "PRL"
## [29] "Vit_D3"           "PRG"              "RBS"              "Weight_gain"
## [33] "hair_growth"      "Skin_darkening"   "Hair_loss"        "Pimples"
## [37] "Fast_food"        "Reg_Exercise"     "BP_Systolic"      "BP_Diastolic"
## [41] "Follicle_No_L"    "Follicle_No_R"    "Avg_F_size_L"     "Avg_F_size_R"
## [45] "Endometrium"
```

Unfortunately all variables are selected. Now, we will do **Sequential Replacement**. Which is basically adding and droping variables at each step.

```
#Sequential Replacement
lm.seq <- stepAIC(glm(PCOS ~. ,data = PCOSdata_train[,-c(1,2)],
           family = binomial(link = 'logit')),direction = 'both',
           trace = 0)
variable.names(lm.seq)  #which variables are selected

##  [1] "(Intercept)"     "Cycle"           "Marriage_Status" "LH"
##  [5] "Weight_gain"     "hair_growth"     "Skin_darkening"  "Pimples"
##  [9] "Fast_food"       "Follicle_No_L"   "Follicle_No_R"   "Avg_F_size_L"

length(variable.names(lm.seq)) - 1  #Number of variables selected

## [1] 11
```

All the selected variables are very much meaningful from EDA. Now, if we look at the summary of this model -

```
summary(lm.seq)

##
## Call:
## glm(formula = PCOS ~ Cycle + Marriage_Status + LH + Weight_gain +
##     hair_growth + Skin_darkening + Pimples + Fast_food + Follicle_No_L +
##     Follicle_No_R + Avg_F_size_L, family = binomial(link = "logit"),
##     data = PCOSdata_train[, -c(1, 2)])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9752  -0.2651  -0.0800   0.0942   3.5610
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -9.17535    1.45075  -6.325 2.54e-10 ***
## Cycle            1.03420    0.44620   2.318 0.020460 *
## Marriage_Status -0.12094    0.04827  -2.505 0.012235 *
## LH               0.08085    0.08404   0.962 0.336039
## Weight_gain      1.67286    0.44401   3.768 0.000165 ***
## hair_growth      1.67073    0.48101   3.473 0.000514 ***
## Skin_darkening   1.36360    0.43051   3.167 0.001538 **
## Pimples          0.81507    0.43642   1.868 0.061816 .
## Fast_food        1.02312    0.47090   2.173 0.029803 *
## Follicle_No_L    0.12420    0.07560   1.643 0.100401
## Follicle_No_R    0.44813    0.07643   5.863 4.55e-09 ***
## Avg_F_size_L     0.10643    0.06953   1.531 0.125847
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 530.56  on 423  degrees of freedom
## Residual deviance: 171.82  on 412  degrees of freedom
## AIC: 195.82
##
## Number of Fisher Scoring iterations: 9
```

We observe that, the AIC is 195.82, which is less than the AIC of full.model. Which is a good indication.

Lasso Logistic :

Now, lets look at another method of variable selection. Which is Lasso logistic. We will be required *glmnet* package for this. Let's fit the model. Using cross-validation we have find out the optimum values of lambda.

```r
#Logistic Lasso Regression
X.mat_train <- model.matrix(PCOS ~ . -1,data=PCOSdata_train[,-c(1,2)])
cv.lasso <- glmnet::cv.glmnet(X.mat_train,PCOSdata_train[,3],
             family = "binomial",alpha = 1)
s.cv <- c(lambda.min = cv.lasso$lambda.min,
              lambda.1se = cv.lasso$lambda.1se)
s.cv

##  lambda.min  lambda.1se
## 0.008098454 0.039379553
```

```r
plot(cv.lasso)
```

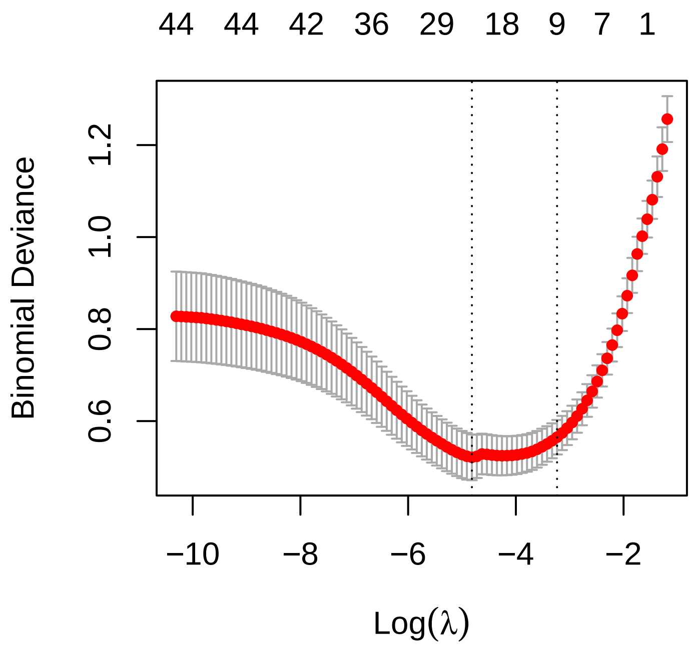


Diagram : Result of CV Lasso

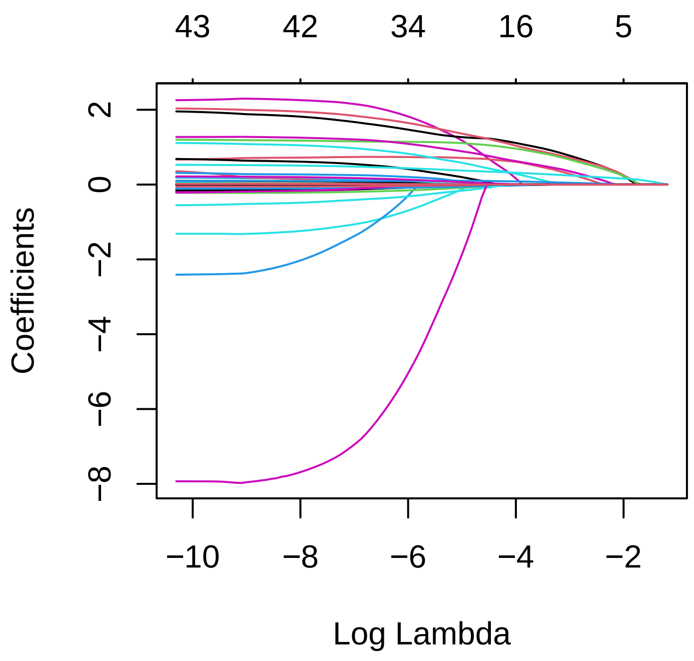


Diagram : Shrinkage of Coefficients Using Lasso

From the above plot we can see the convergence of coefficients for various values of lambda. As lambda increases the coefficients becomes zero. Let's look which variables have non-zero coefficients for both *cv.lasso$lambda.min* and *cv.lasso$lambda.1se*.

```
(lasso.min.variables <- rownames(lasso.coef)[lasso.coef[,1] != 0])

##  [1] "(Intercept)"     "Age"             "Weight"          "Blood_Group12"
##  [5] "Blood_Group14"   "Pulse_rate"      "Cycle"           "Cycle_length"
##  [9] "Marriage_Status" "Pregnant"        "LH"              "Waist_Hip_Ratio"
## [13] "AMH"             "Vit_D3"          "Weight_gain"     "hair_growth"
## [17] "Skin_darkening"  "Hair_loss"       "Pimples"         "Fast_food"
## [21] "Reg_Exercise"    "BP_Systolic"     "Follicle_No_L"   "Follicle_No_R"
## [25] "Avg_F_size_L"

(lasso.1se.variables <- rownames(lasso.coef)[lasso.coef[,2] != 0])

##  [1] "(Intercept)"    "Cycle"          "LH"             "Weight_gain"
##  [5] "hair_growth"    "Skin_darkening" "Pimples"        "Fast_food"
##  [9] "Follicle_No_L"  "Follicle_No_R"
```

Using *cv.lasso$lambda.min* we get some variables and obviously with *cv.lasso$lambda.1se* we get less number of variables than *cv.lasso$lambda.min*. Also, the variables selected are very much consistent with our observations from EDA. That's why EDA is the stepping stone in data analysis. We can refit our model using the variables selected from lasso. That may give in some sense satisfactory fit.

```
#Fit a glm using selected variables
sv_fm.min.lasso <- glm(PCOS ~ .,
data = as.data.frame(cbind(PCOS = PCOSdata_train[,3],X.mat_train[,colnames(X.mat_train)%in%c
summary(sv_fm.min.lasso)   #summary output

##
## Call:
## glm(formula = PCOS ~ ., family = binomial(link = "logit"), data = as.data.frame(cbind(PCO
##     3], X.mat_train[, colnames(X.mat_train) %in% c("PCOS", lasso.min.variables)])))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7003  -0.2431  -0.0625   0.0728   3.4018
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -9.904364   8.467021  -1.170 0.242098
## Age              0.009086   0.055960   0.162 0.871011
## Weight           0.032213   0.026084   1.235 0.216841
## Blood_Group12   -0.927603   1.189041  -0.780 0.435316
## Blood_Group14    2.311932   1.048075   2.206 0.027392 *
## Pulse_rate       0.141885   0.088449   1.604 0.108684
## Cycle            0.752023   0.532600   1.412 0.157955
## Cycle_length    -0.170308   0.161511  -1.054 0.291672
## Marriage_Status -0.137348   0.069197  -1.985 0.047157 *
## Pregnant        -0.442093   0.457813  -0.966 0.334213
## LH               0.057761   0.093530   0.618 0.536864
## Waist_Hip_Ratio -6.995900   5.043510  -1.387 0.165408
## AMH              0.026691   0.039276   0.680 0.496783
## Vit_D3          -0.022558   0.018434  -1.224 0.221049
## Weight_gain      1.667629   0.542309   3.075 0.002105 **
## hair_growth      1.860674   0.536648   3.467 0.000526 ***
## Skin_darkening   1.251958   0.473257   2.645 0.008159 **
## Hair_loss        0.325307   0.482859   0.674 0.500496
## Pimples          1.078325   0.496193   2.173 0.029765 *
## Fast_food        1.099803   0.524116   2.098 0.035870 *
## Reg_Exercise     0.580879   0.519318   1.119 0.263335
## BP_Systolic     -0.048876   0.037278  -1.311 0.189813
## Follicle_No_L    0.076559   0.087746   0.873 0.382929
## Follicle_No_R    0.508372   0.092716   5.483 4.18e-08 ***
## Avg_F_size_L     0.171536   0.080222   2.138 0.032494 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 530.56  on 423  degrees of freedom
## Residual deviance: 155.93  on 399  degrees of freedom
## AIC: 205.93
##
## Number of Fisher Scoring iterations: 9
```

```
sv_fm.1se.lasso <- glm(PCOS ~ .,
                     data = as.data.frame(cbind(PCOS = PCOSdata_train[,3],X.mat_train[,col
                     family = binomial(link = 'logit'))
summary(sv_fm.1se.lasso)  #summary output
```

```
##
## Call:
## glm(formula = PCOS ~ ., family = binomial(link = "logit"), data = as.data.frame(cbind(PC
##     3], X.mat_train[, colnames(X.mat_train) %in% c("PCOS", lasso.1se.variables)])))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1276  -0.2935  -0.0986   0.1024   3.6994
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -8.22121    0.89471  -9.189  < 2e-16 ***
## Cycle           0.93154    0.42645   2.184 0.028931 *
## LH              0.08246    0.08219   1.003 0.315755
## Weight_gain     1.74199    0.42773   4.073 4.65e-05 ***
## hair_growth     1.49175    0.45190   3.301 0.000963 ***
## Skin_darkening  1.50360    0.41467   3.626 0.000288 ***
## Pimples         0.59468    0.41611   1.429 0.152968
## Fast_food       0.88943    0.45167   1.969 0.048930 *
## Follicle_No_L   0.13607    0.06991   1.946 0.051624 .
## Follicle_No_R   0.44343    0.07407   5.987 2.14e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 530.56  on 423  degrees of freedom
## Residual deviance: 181.35  on 414  degrees of freedom
## AIC: 201.35
```

```
##
## Number of Fisher Scoring iterations: 8
```

The AIC for Logistic model using variables selected from Lasso Regression using $\lambda=$ *cv.lasso$lambda.min* is 206.95, while that for $\lambda=$ *cv.lasso$lambda.1se* is 201.35. Which is slightly higher than that of logistic model obtained using sequential selection.

Robust Logistic Regression :

Since, Robust methods are consistent even if there are influential observations. We will fit robust regression on the data including influential observations to compare its efficacy.

```
library(robustbase)
#Robust Logistic Regression
robust.glm <- glmrob(PCOS ~ .,data = PCOSdata_train.1[,-c(1,2)],
                      family = binomial,method = "Mqle",
                      control = glmrobMqle.control(tcc = 16))
summary(robust.glm)

##
## Call:  glmrob(formula = PCOS ~ ., family = binomial, data = PCOSdata_train.1[,      -c(1,
##
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -2.119e+01  1.579e+01  -1.341 0.179766
## Age                1.802e-02  6.463e-02   0.279 0.780452
## Weight             1.025e-02  4.039e-02   0.254 0.799622
## Height             4.340e-02  5.468e-02   0.794 0.427355
## Blood_Group12     -1.862e+00  1.426e+00  -1.305 0.191802
## Blood_Group13     -4.099e-01  8.375e-01  -0.489 0.624578
## Blood_Group14      2.209e+00  1.346e+00   1.641 0.100857
## Blood_Group15     -4.165e-01  7.240e-01  -0.575 0.565121
## Blood_Group16      2.176e-01  1.390e+00   0.157 0.875603
## Blood_Group17     -4.013e-01  1.153e+00  -0.348 0.727894
## Blood_Group18     -2.772e+00  8.519e+00  -0.325 0.744850
## Pulse_rate         2.559e-01  1.180e-01   2.169 0.030120 *
## RR                -2.457e-01  1.745e-01  -1.408 0.159120
## Hb                -1.552e-01  3.333e-01  -0.466 0.641401
## Cycle              7.432e-01  6.427e-01   1.156 0.247568
## Cycle_length      -2.609e-01  1.840e-01  -1.418 0.156170
## Marriage_Status   -1.139e-01  7.703e-02  -1.479 0.139128
## Pregnant          -6.026e-01  5.177e-01  -1.164 0.244440
## No_of_aborptions -1.429e-01  4.710e-01  -0.303 0.761644
## I_beta_HCG       -3.974e-05  5.310e-05  -0.749 0.454146
## II_beta_HCG       1.732e-04  3.687e-04   0.470 0.638580
## FSH               1.340e-02  8.234e-02   0.163 0.870744
## LH                5.742e-02  1.083e-01   0.530 0.596099
## Waist            -1.033e-02  9.546e-02  -0.108 0.913826
## Waist_Hip_Ratio  -9.036e+00  6.067e+00  -1.489 0.136359
## TSH               1.104e-01  6.601e-02   1.673 0.094337 .
## AMH              -1.413e-03  3.766e-02  -0.038 0.970078
## PRL               1.020e-02  1.983e-02   0.514 0.606980
## Vit_D3            5.684e-04  1.880e-03   0.302 0.762393
## PRG              -7.098e-02  8.778e-01  -0.081 0.935555
## RBS               1.284e-02  2.024e-02   0.634 0.525774
## Weight_gain       2.395e+00  7.032e-01   3.406 0.000659 ***
## hair_growth       2.129e+00  6.023e-01   3.534 0.000409 ***
## Skin_darkening    1.267e+00  5.319e-01   2.382 0.017201 *
## Hair_loss         1.332e-01  5.314e-01   0.251 0.802109
## Pimples           1.075e+00  5.608e-01   1.916 0.055351 .
## Fast_food         1.399e+00  6.081e-01   2.300 0.021451 *
## Reg_Exercise      7.879e-01  6.054e-01   1.301 0.193154
## BP_Systolic      -5.468e-02  4.325e-02  -1.264 0.206066
## BP_Diastolic      2.831e-02  6.223e-02   0.455 0.649202
## Follicle_No_L     9.908e-02  1.085e-01   0.913 0.361272
## Follicle_No_R     5.622e-01  1.145e-01   4.911 9.05e-07 ***
## Avg_F_size_L      2.397e-01  1.165e-01   2.058 0.039546 *
## Avg_F_size_R      4.390e-03  1.052e-01   0.042 0.966706
## Endometrium       4.286e-02  1.194e-01   0.359 0.719528
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Robustness weights w.r * w.x:
##  425 weights are ~= 1. The remaining one are
##    391
## 0.3284
##
## Number of observations: 426
## Fitted by method 'Mqle'  (in 12 iterations)
##
## (Dispersion parameter for binomial family taken to be 1)
##
## No deviance values available
## Algorithmic parameters:
##   acc
## 1e-04
## maxit   tcc
##    50    16
## test.acc
##   "coef"
```

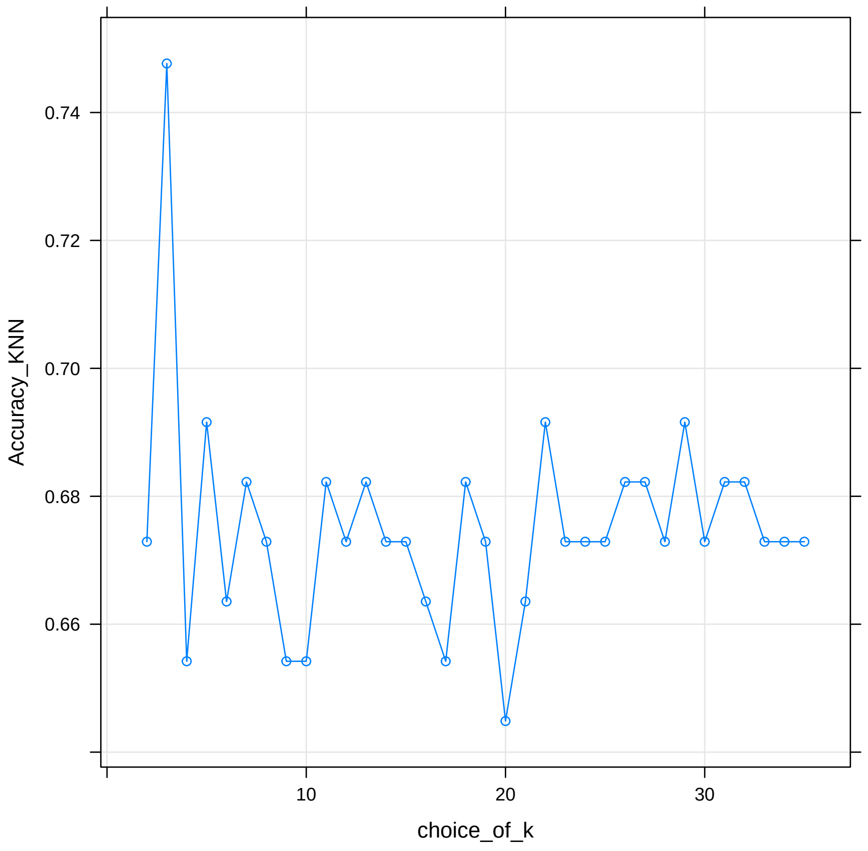
Note : method = "Mqle" fits a generalized linear model using Mallows or Huber type robust estimators, as described in Cantoni and Ronchetti (2001) and Cantoni and Ronchetti (2006). In contrast to the implementation described in Cantoni (2004), the pure influence algorithm is implemented.

From the output we can see that, 425 weights are approximately 1 and one is 0.3284. Also, glmrobMqle.control function is used to control the parameters of Huber psi function. The outputs can be used for inference purpose. Although we will not use them.

K Nearest Neighbor :

We will use K Nearest neighbour method for classification. However, here in true sense no model exists.

```
#KNN
library(class)
choice_of_k <- 2:35
Accuracy_KNN <- NULL
for(i in 1:length(choice_of_k)){
 KNN.model.sim <- knn(train = PCOSdata_numeric[index_train,-1],
                      cl=PCOSdata_train.1$PCOS,
                      test = PCOSdata_numeric[-index_train,-1],
                      k=choice_of_k[i])
 Accuracy_KNN <- c(Accuracy_KNN,mean(PCOSdata_test[,3]==(as.numeric(KNN.model.sim)-1)))
}

lattice::xyplot(Accuracy_KNN ~ choice_of_k,grid = T,type = c('p','l'))
```



From the above grpah, we can clearly see that based on accuracy metric k = 3 is optimal. So, we fit using k = 3.

```
KNN.model <- knn(train = PCOSdata_numeric[index_train,-1],
                      cl = PCOSdata_train.1$PCOS,
                        test = PCOSdata_numeric[-index_train,-1],k = 3)
```

Random Forest :

Now we will fit Random Forest on the data.

```
set.seed(seed = 987654321)
#Random_Forest
library(randomForest)
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package:  'randomForest'
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
randomForest.model <- randomForest(x = PCOSdata_train[,-c(1,2,3)],
                    y = as.factor(PCOSdata_train[,3]),ntree = 700)
RF_pred <- predict(randomForest.model, newdata = PCOSdata_test[,-c(1,2,3)])

randomForest.model  #summary of random forest
```

```
##
## Call:
##  randomForest(x = PCOSdata_train[, -c(1, 2, 3)], y = as.factor(PCOSdata_train[,      3])
##                Type of random forest: classification
##                      Number of trees: 700
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 9.67%
## Confusion matrix:
##     0   1 class.error
## 0 277  12  0.04152249
## 1  29 106  0.21481481
```

```r
plot(randomForest.model,main = 'Error Rate Plot of Random Forest')
```

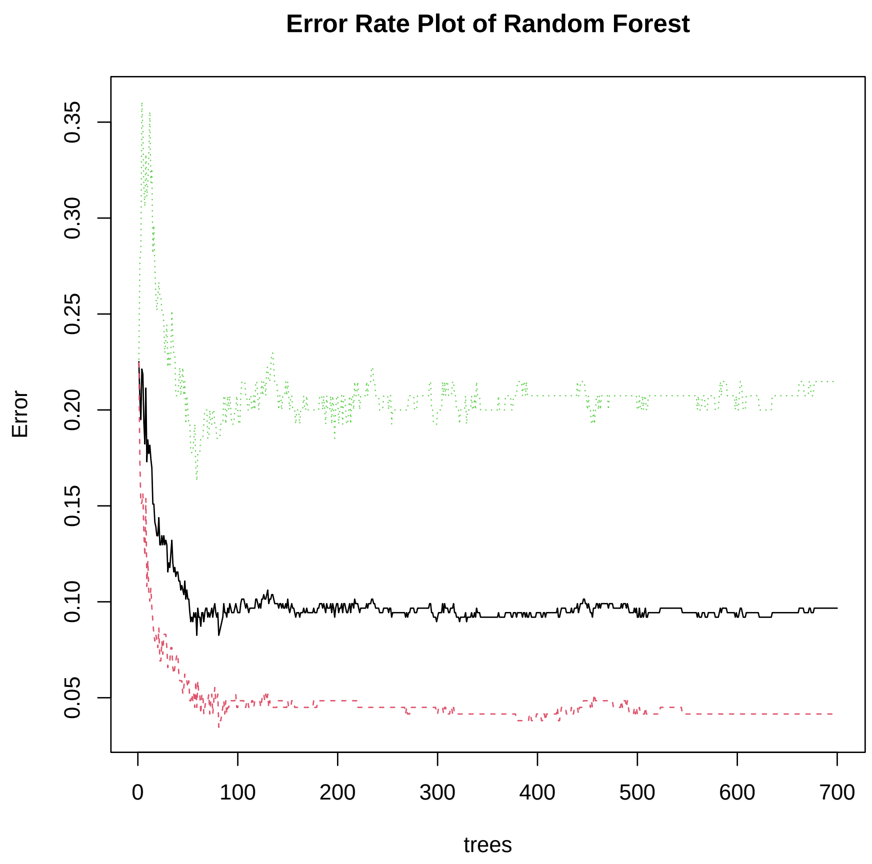


Diagram : Error Rate plot of Random Forest.

From the above plot, we can see that error rate as a function of number of trees. R itself chosses the ntree 700. Out of Bag error rate is 8.25%. Which is moderate. Also, we draw the Variable importance plot.

```
varImpPlot(randomForest.model,col = 'red',pch = 19,
             main = 'Variable Importance Plot of Random Forest')
```

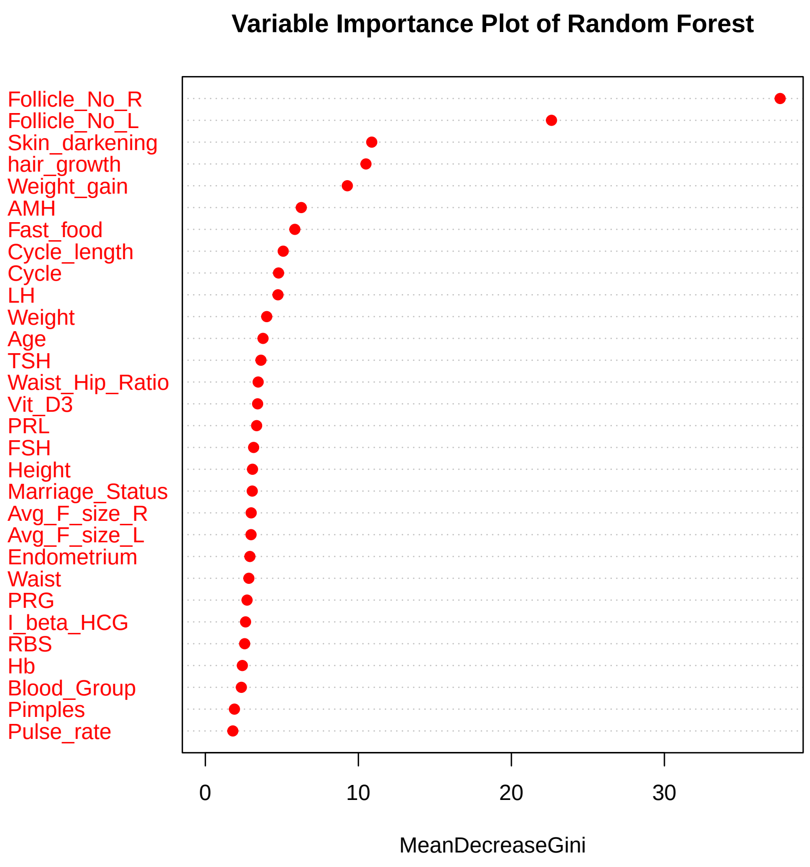


Diagram : Variable Importance Plot of Random Forest.

From this plot we can understand, which variables are important for our RF classifier. Notice that the important variables are selected as significant variable in our sequential logistic model and lasso logistic model.

Support Vector Machine :

Our next target is to fit a support vector machine on data.

```
library(e1071)
#Support vector machine
svm.model <- svm(as.factor(PCOS) ~ ., data = PCOSdata_train[,-c(1,2)],
                 kernel = "linear")

svm.model #model internals
```

```
##
## Call:
## svm(formula = as.factor(PCOS) ~ ., data = PCOSdata_train[, -c(1,
##     2)], kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  97
```

Extreme Gradient Boosting :

Lastly, we will fit XgBoost in our model.

```
library(xgboost)
#XgBoost Algorithm
X_Train.matrix <- data.matrix(PCOSdata_train[,-c(1,2,3)])
y_Train <- PCOSdata_train[,3]
XgBoost.model <- xgboost(data = X_Train.matrix,label = y_Train,
                    objective = "binary:logistic",nrounds = 25)
```

```
## [1] train-logloss:0.503655
## [2] train-logloss:0.385544
## [3] train-logloss:0.305631
## [4] train-logloss:0.246442
## [5] train-logloss:0.203010
## [6] train-logloss:0.170852
## [7] train-logloss:0.146480
## [8] train-logloss:0.126655
## [9] train-logloss:0.110025
## [10] train-logloss:0.097812
## [11] train-logloss:0.086451
## [12] train-logloss:0.078623
## [13] train-logloss:0.071057
## [14] train-logloss:0.065848
## [15] train-logloss:0.059997
## [16] train-logloss:0.055736
## [17] train-logloss:0.050873
## [18] train-logloss:0.046953
## [19] train-logloss:0.043640
## [20] train-logloss:0.041431
## [21] train-logloss:0.038915
## [22] train-logloss:0.036859
## [23] train-logloss:0.035131
## [24] train-logloss:0.033398
## [25] train-logloss:0.031737
```

Thus, we are done fitting models. Now, we will go for evaluation of these fitted models.

Prediction & Evaluation Metrics :

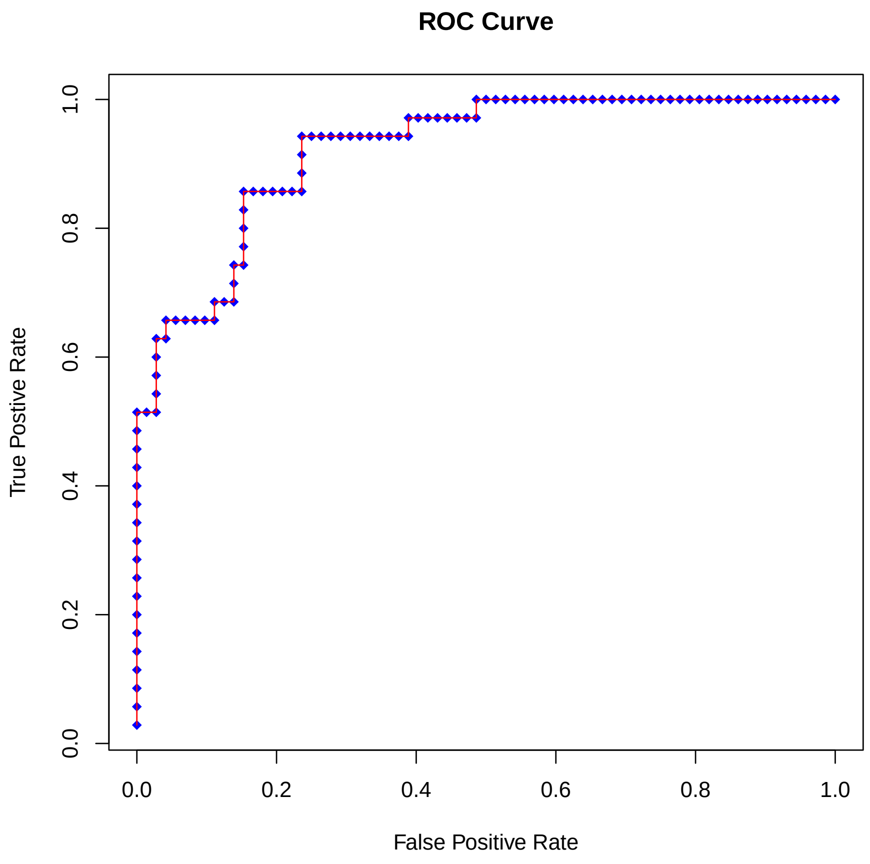
ROC Curve :

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. Here we plot TPR against FPR. Where, $TPR = \frac{TP}{TP+FN}$ & $FPR = \frac{FP}{FP+TN}$. The optimal cut off would be where TPR is high and FPR is low. Because, TPR is high means the model predicts Positive cases well and FPR is low or equivalently 1 - FPR is high means that the model predicts negative cases well. A good model should predict both the cases in a well manner. I have written a user defined function to draw ROC Curve and also to find this optimal threshold.

```
#ROC Curve optimal Threshold
myROC <- function(Y,pi.hat,plot.R = F){
   #Thresholds
   my.threshold <- unique(sort(pi.hat,decreasing = T))
   #Finding TPR and FPR for each threshold
   TPR <- NULL; FPR <- NULL
   for(i in 1:length(my.threshold)){
      Y.hat <- ifelse(my.threshold[i] > pi.hat,0,1)
      TPR <- c(TPR,sum((Y.hat == 1) & (Y == 1))/sum(Y == 1))#storing TPR
     FPR <- c(FPR,sum((Y.hat == 1) & (Y == 0))/sum(Y == 0))#storing FPR
   }
   #Storing final threshold
    final.threshold <- my.threshold[which.max(TPR*(1-FPR))]
   #Visualize
    if(plot.R){
      plot(FPR,TPR,type = 'p',pch = 18,main = 'ROC Curve',col = 'blue',
           xlab = 'False Positive Rate',ylab = 'True Postive Rate')
       lines(FPR,TPR,col = 'red')      #mtext(summary(model)[1],side = 3)
     }
   #Return
   return(final.threshold)
}
```

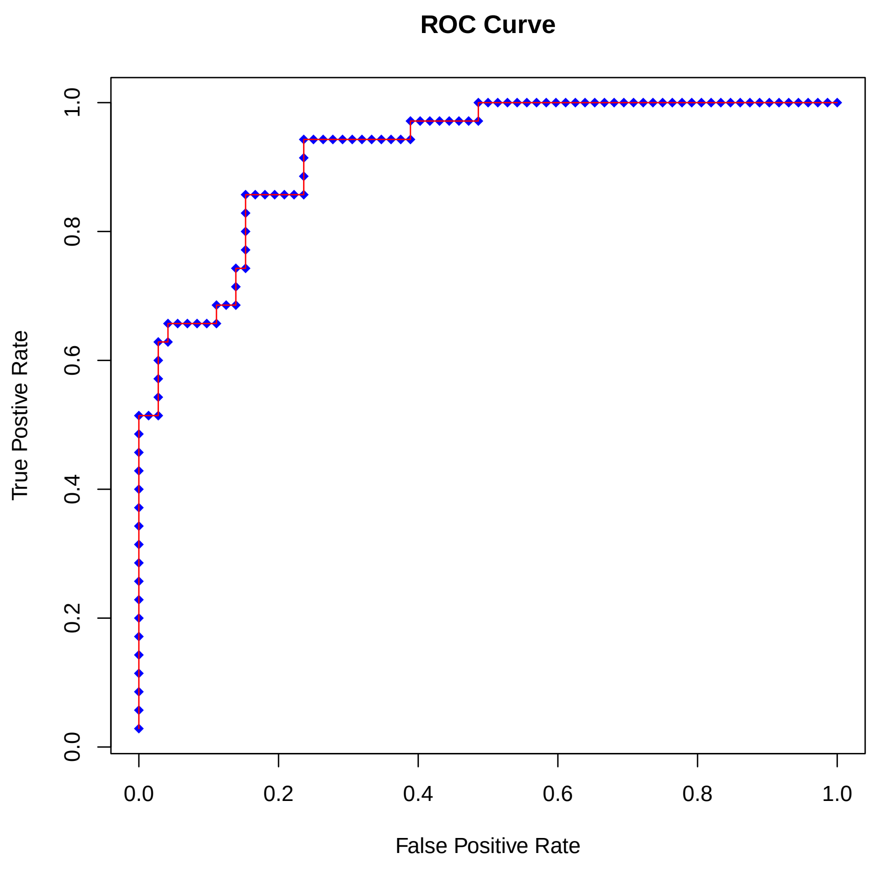
```
##
## Attaching package:  'MLmetrics'
## The following object is masked from 'package:base':
##
##     Recall
```

```
#ROC Curves  and optimal thresholds

#final_full_model
myROC(Y = PCOSdata_test[,3],       pi.hat = predict(final_full.model,
     newdata = PCOSdata_test[,-c(1,2)],type = 'response'),plot.R = T)
```



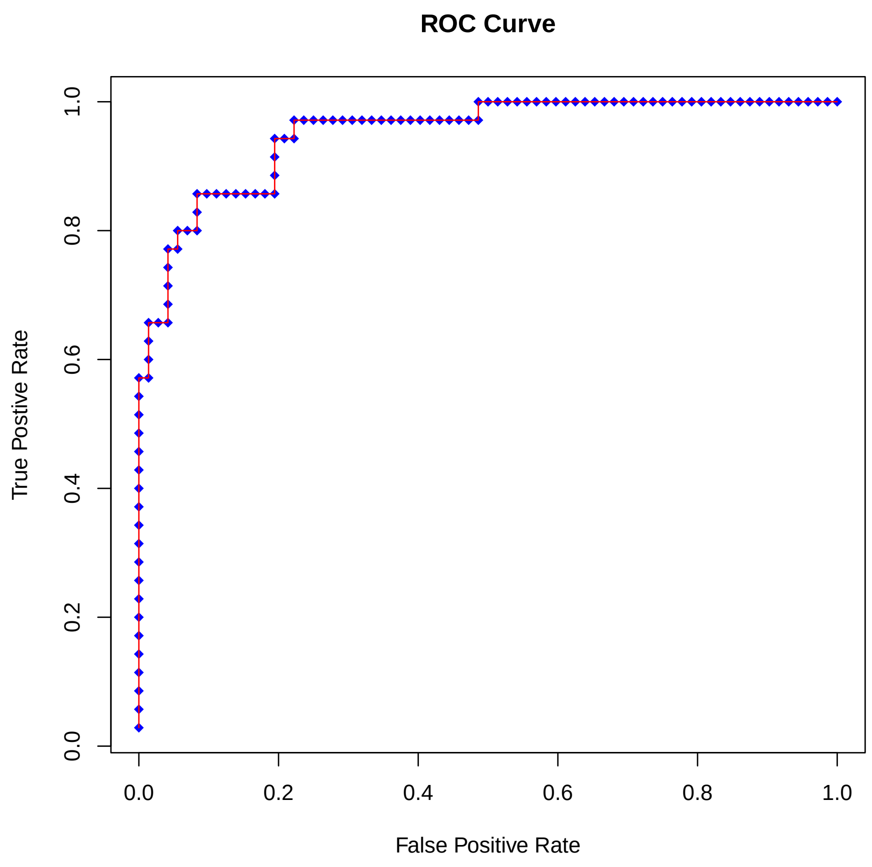
```
## [1] 0.3237803
```

```
#lm.forward
myROC(Y = PCOSdata_test[,3], pi.hat = predict(lm.forward,
     newdata = PCOSdata_test[,-c(1,2)],type = 'response'),plot.R = T)
```



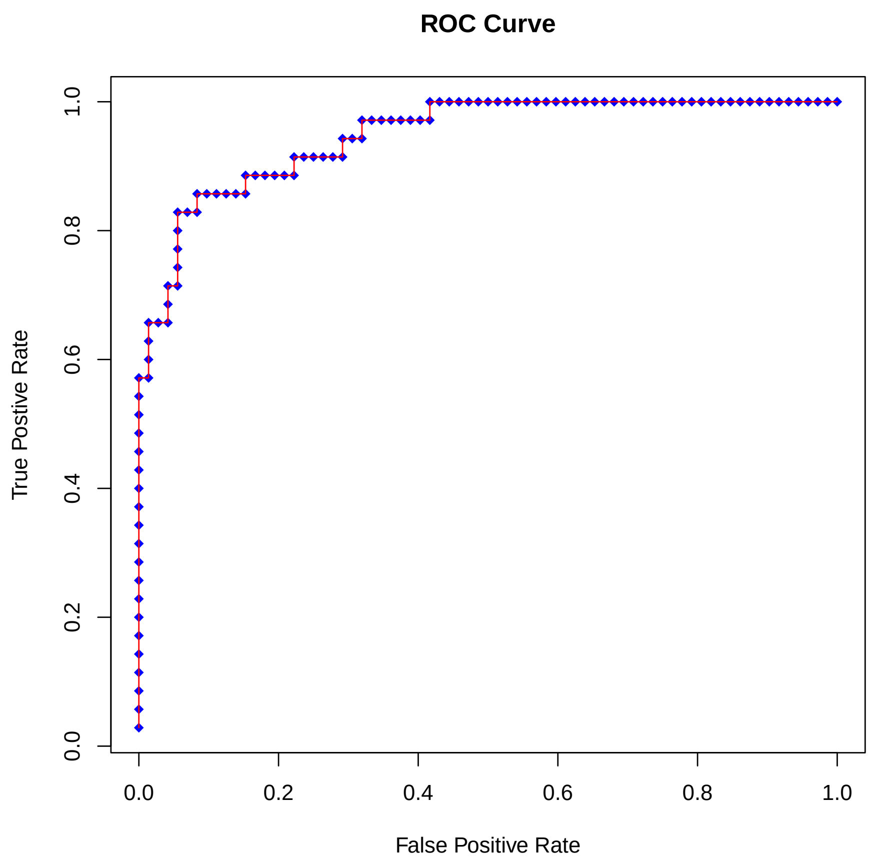
```
## [1] 0.3237803
```

```
#lm.seq
myROC(Y = PCOSdata_test[,3], pi.hat = predict(lm.seq,
     newdata = PCOSdata_test[,-c(1,2)],type = 'response'),plot.R = T)
```



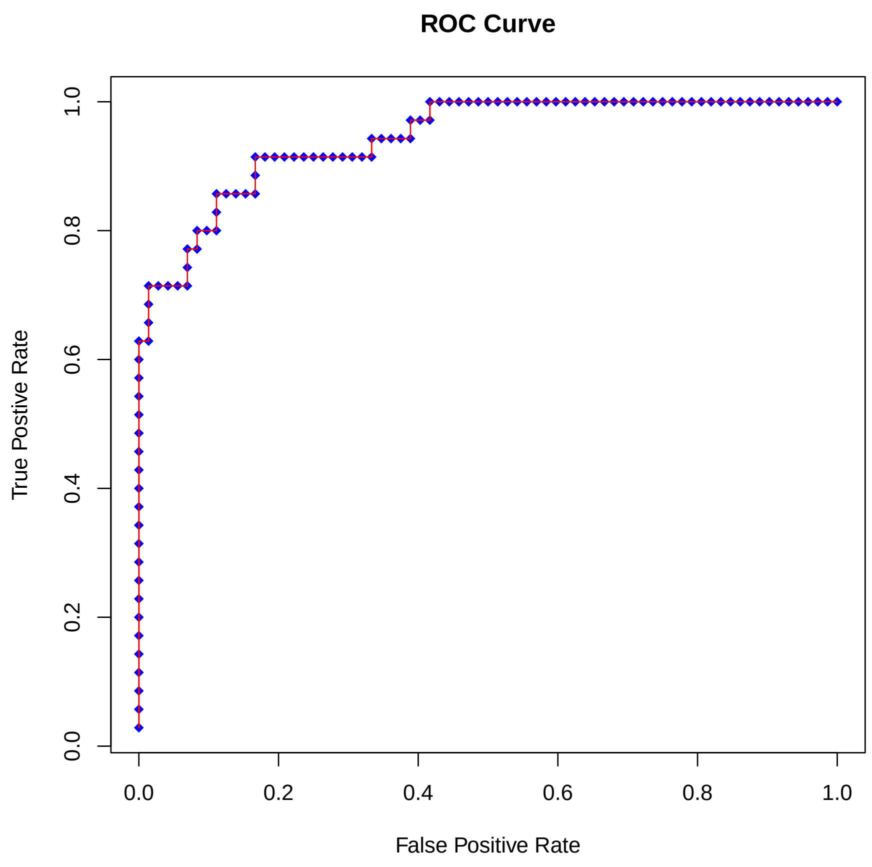
```
## [1] 0.4174999
```

```
#fm.min.lasso
myROC(Y = PCOSdata_test[,3],pi.hat = predict(fm.min.lasso,
      newx = X.mat_test,type = 'response'),plot.R = T)
```



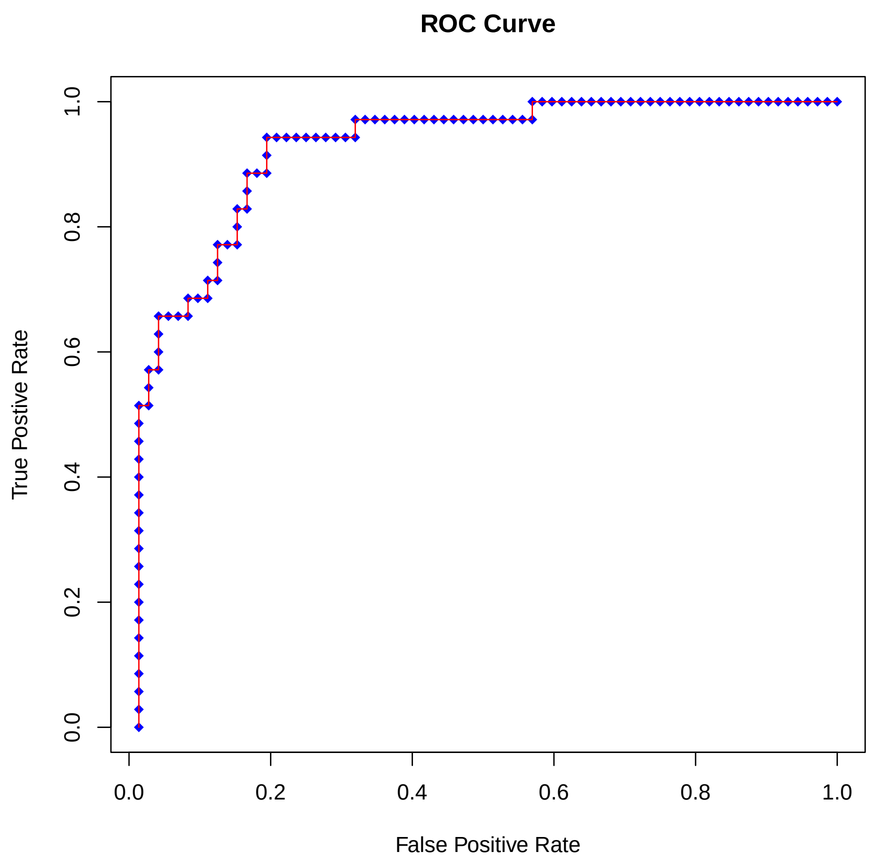
```
## [1] 0.4120581
```

```
#fm.1se.lasso
myROC(Y = PCOSdata_test[,3],pi.hat = predict(fm.1se.lasso,
      newx = X.mat_test,type = 'response'),plot.R = T)
```



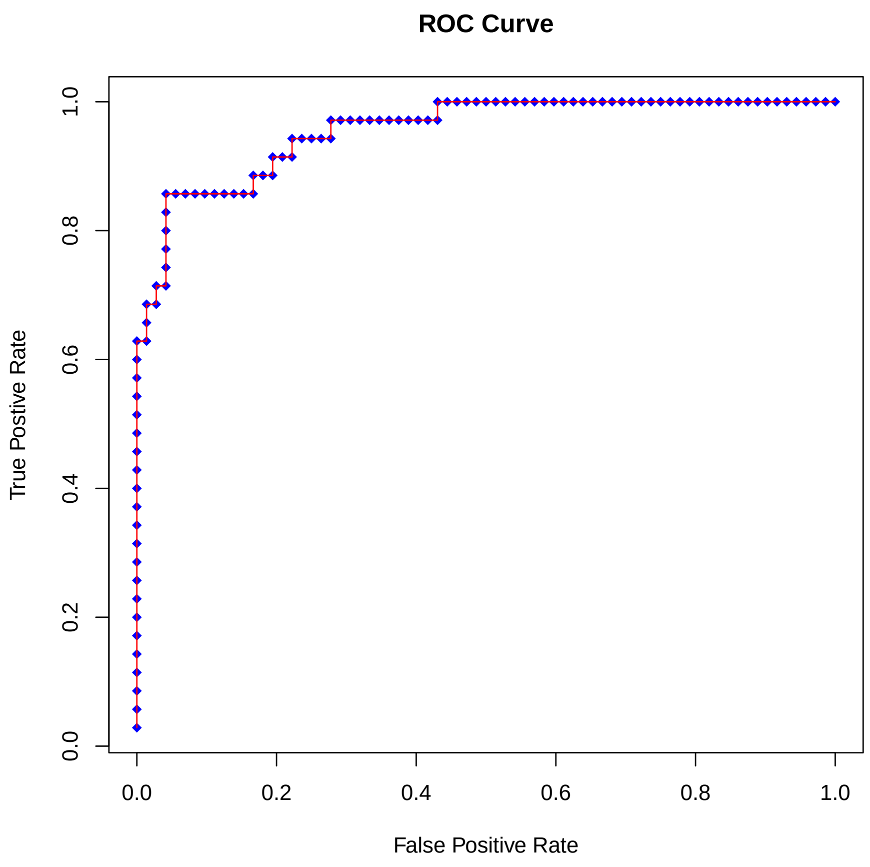
```
## [1] 0.3919745
```

```
#robust glm
myROC(Y = PCOSdata_test[,3],pi.hat = predict(robust.glm,
      newdata = PCOSdata_test[,-c(1,2)],type = 'response'),plot.R = T)
```



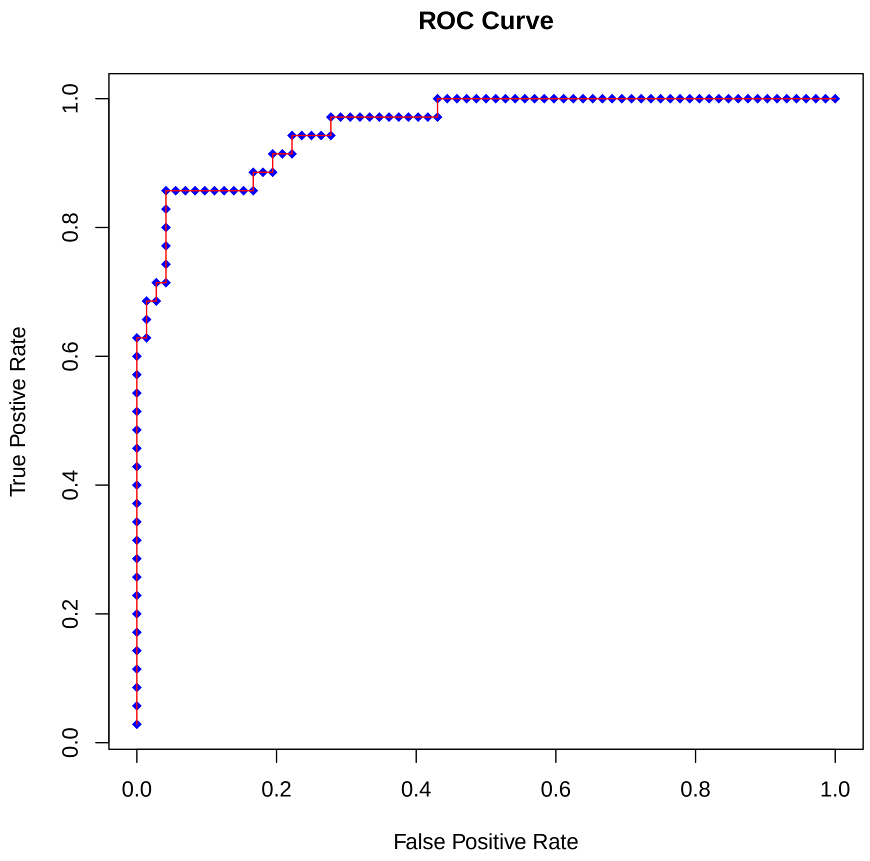
```
## [1] 0.1557939
```

```
#sv_fm.min.lasso
myROC(Y = PCOSdata_test[,3],pi.hat = predict(sv_fm.1se.lasso,newdata =
      as.data.frame(cbind(PCOS = PCOSdata_test[,3],
       X.mat_test[,colnames(X.mat_test)%in%c('PCOS',lasso.1se.variables)])),
       type = 'response'),plot.R = T)
```



```
## [1] 0.3942447
```

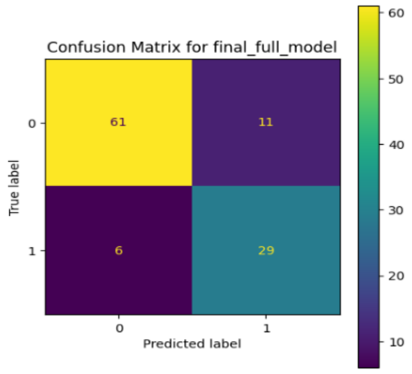
```
#sv_fm.1se.lasso
myROC(Y = PCOSdata_test[,3],pi.hat = predict(sv_fm.1se.lasso,newdata =
     as.data.frame(cbind(PCOS = PCOSdata_test[,3],
     X.mat_test[,colnames(X.mat_test)%in%c('PCOS',lasso.1se.variables)])),
     type = 'response'),plot.R = T)
```

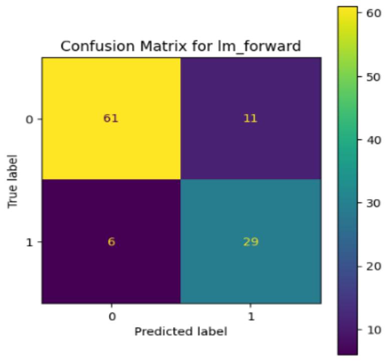


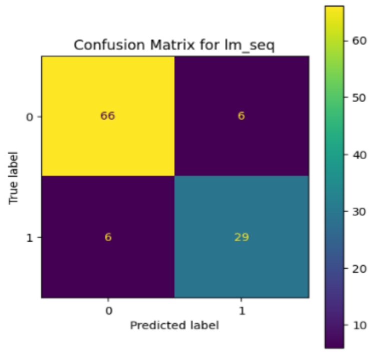
```
## [1] 0.3942447
```

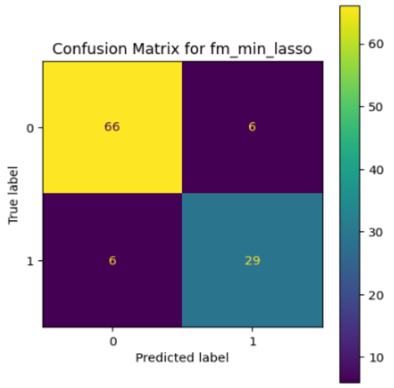
Confusion Matrix :

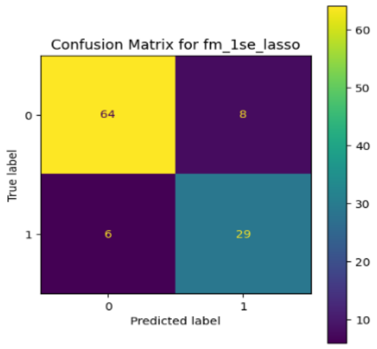
Now, we will draw the confusion matrices on the basis of these optimal values.

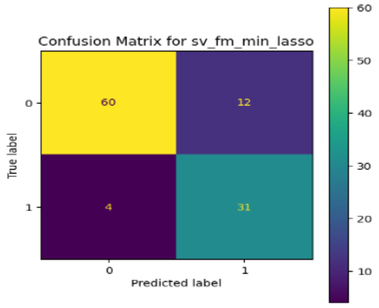

Confusion Matrix for final_full_model
61
11
6
29
True label
Predicted label
0
1
60
50
40
30
20
10

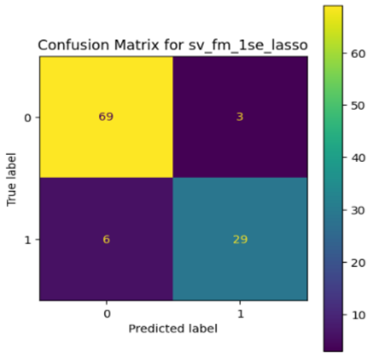

Confusion Matrix for lm_forward
61
11
6
29
True label
Predicted label
0
1
60
50
40
30
20
10

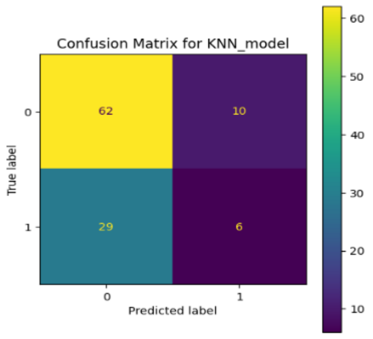

Confusion Matrix for lm_seq
66
6
6
29
True label
Predicted label
0
1
60
50
40
30
20
10

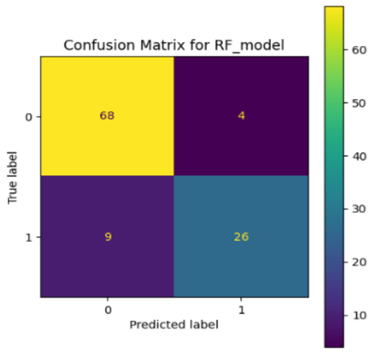

Confusion Matrix for fm_min_lasso
66
6
6
29
True label
Predicted label
0
1
60
50
40
30
20
10

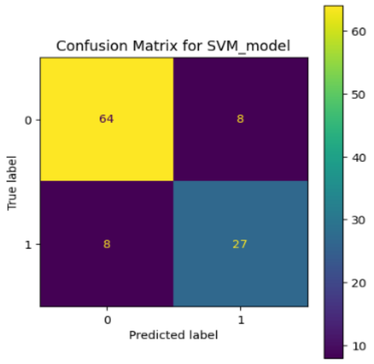

Confusion Matrix for fm_1se_lasso
64
8
6
29
True label
Predicted label
0
1
60
50
40
30
20
10

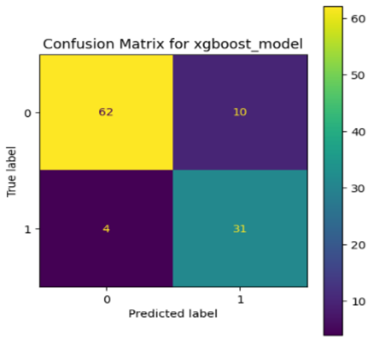

Confusion Matrix for sv_fm_min_lasso
60
12
4
31
True label
Predicted label
0
1
60
50
40
30
20
10











We have created a user defined function to find out different evaluation metrics on the basis of these Confusion Matrix. We will use to find out different metrics and will compare our models.

```
#Evaluation_Metrics
My_Evaluation_Metric <- function(y,y_pred,pred_prob = NULL){
       c(Accuracy = mean(y == y_pred),
         Specificty = sum(y == 0 & y_pred == 0)/sum(y == 0),
         Sensitivity = sum(y == 1 & y_pred == 1)/sum(y == 1),
         Precision = sum(y == 1 & y_pred == 1)/sum(y_pred == 1),
 LogLoss = ifelse(!is.null(pred_prob),MLmetrics::LogLoss(pred_prob,y),NA))
}
```

I have created separate data frames to store predicted values of y for different models (using optimum value of lambda & the usual value of lambda $= 0.5$) and also for predicted probabilities. Using them, i will calculate the metrics.

```
#== Evaluation Metrics Calculation for Optimum models ==

My_Evaluation_Metric(PCOSdata_test[,3],
     prediction_Df_ALL.opt[,1],Pred_Prob[,1]) #final_full.model

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8411215   0.8472222   0.8285714   0.7250000   0.3958885

My_Evaluation_Metric(PCOSdata_test[,3],
     prediction_Df_ALL.opt[,2],Pred_Prob[,2]) #fm.min.lasso

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8878505   0.9166667   0.8285714   0.8285714   0.2646730

My_Evaluation_Metric(PCOSdata_test[,3],
     prediction_Df_ALL.opt[,3],Pred_Prob[,3]) #fm.1se.lasso

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8691589   0.8888889   0.8285714   0.7837838   0.2932286

My_Evaluation_Metric(PCOSdata_test[,3],
     prediction_Df_ALL.opt[,4],Pred_Prob[,4]) #robust.glm

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8411215   0.8055556   0.9142857   0.6956522   0.6801102

My_Evaluation_Metric(PCOSdata_test[,3],
     prediction_Df_ALL.opt[,5],Pred_Prob[,5]) #lm.forward

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8411215   0.8472222   0.8285714   0.7250000   0.3958885
```

```
My_Evaluation_Metric(PCOSdata_test[,3],
     prediction_Df_ALL.opt[,6],Pred_Prob[,6]) #lm.seq

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8878505   0.9166667   0.8285714   0.8285714   0.2618273

My_Evaluation_Metric(PCOSdata_test[,3],
     prediction_Df_ALL.opt[,7],Pred_Prob[,7]) #sv_fm.min.lasso

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8504673   0.8333333   0.8857143   0.7209302   0.3134082

My_Evaluation_Metric(PCOSdata_test[,3],
     prediction_Df_ALL.opt[,8],Pred_Prob[,8]) #sv_fm.1se.lasso

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.9158879   0.9583333   0.8285714   0.9062500   0.2464480

My_Evaluation_Metric(PCOSdata_test[,3],
     prediction_Df_ALL.opt[,10]) #KNN.model

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.7476636   0.9166667   0.4000000   0.7000000          NA

My_Evaluation_Metric(PCOSdata_test[,3],
     prediction_Df_ALL.opt[,11]) #RF.model

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8785047   0.9444444   0.7428571   0.8666667          NA

My_Evaluation_Metric(PCOSdata_test[,3],
     prediction_Df_ALL.opt[,12]) #svm.model

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8504673   0.8888889   0.7714286   0.7714286          NA

My_Evaluation_Metric(PCOSdata_test[,3],
     prediction_Df_ALL.opt[,9],Pred_Prob[,9]) #xgboost.model

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8691589   0.8611111   0.8857143   0.7560976   0.2831856
```

From the above outputs we can say that interms of Accuracy , Specificity, Precision, Log loss *sv_fm.1se.lasso* is best, while interms of Sensitivity *robust.glm* is best. Also, among the popular machine learning classifiers, Random Forest is best in terms of Accuracy, Specificity and Precision. While interms of Sensitivity XgBoost is best.

```
#==== Evaluation Metrics Calculation for Usual models ====

My_Evaluation_Metric(PCOSdata_test[,3],
    prediction_Df_ALL.Usual[,1],Pred_Prob[,1]) #final_full.model

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8224299   0.8611111   0.7428571   0.7222222   0.3958885

My_Evaluation_Metric(PCOSdata_test[,3],
    prediction_Df_ALL.Usual[,2],Pred_Prob[,2]) #fm.min.lasso

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8691589   0.9444444   0.7142857   0.8620690   0.2646730

My_Evaluation_Metric(PCOSdata_test[,3],
    prediction_Df_ALL.Usual[,3],Pred_Prob[,3]) #fm.1se.lasso

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8691589   0.9444444   0.7142857   0.8620690   0.2932286

My_Evaluation_Metric(PCOSdata_test[,3],
    prediction_Df_ALL.Usual[,4],Pred_Prob[,4]) #robust.glm

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8411215   0.8750000   0.7714286   0.7500000   0.6801102

My_Evaluation_Metric(PCOSdata_test[,3],
    prediction_Df_ALL.Usual[,5],Pred_Prob[,5]) #lm.forward

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8224299   0.8611111   0.7428571   0.7222222   0.3958885

My_Evaluation_Metric(PCOSdata_test[,3],
    prediction_Df_ALL.Usual[,6],Pred_Prob[,6]) #lm.seq

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8971963   0.9583333   0.7714286   0.9000000   0.2618273

My_Evaluation_Metric(PCOSdata_test[,3],
    prediction_Df_ALL.Usual[,7],Pred_Prob[,7]) #sv_fm.min.lasso

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8411215   0.9027778   0.7142857   0.7812500   0.3134082

My_Evaluation_Metric(PCOSdata_test[,3],
    prediction_Df_ALL.Usual[,8],Pred_Prob[,8]) #sv_fm.1se.lasso

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8878505   0.9583333   0.7428571   0.8965517   0.2464480

My_Evaluation_Metric(PCOSdata_test[,3],
    prediction_Df_ALL.opt[,9],Pred_Prob[,9]) #xgboost.model

##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.8691589   0.8611111   0.8857143   0.7560976   0.2831856
```

From the above outputs we see that, interms of Accuracy, Precision and Sensitivity lm.sq is best. Thus, looking at the above two chunks of output we can say that *sv_fm.1se.lasso* and *lm.seq* performs than others.

Inferring About Significance of Predictors:

So far we are only concerned with fitting different models and evaluating them interms of some metrics. We will do our inference on the basis one of the model. Since, inference is meaning full for parametric models. So, we will restrict ourself to Logistic type models. Interms of evaluation metrics sv_fm.1se.lasso is better than other (when optimum threshold is chossen). But, very less number of variables are chossen. So, we will instead use sv_fm.min.lasso for inference.

But, the problem is that we cannnot make inference on the basis of summary output of the model. Because, we have directly fitted the model. Instead, we have first fitted Lasso and then we have fitted glm using the variables having non-zero coefficients from the lasso model. So, we have fitted 2 models, to get the summary output. The Wald test's efficiancy may reduce by that. So, we need to do simulation study for that.

Suppose, we want to test the hypothesis $H_{0j} : \beta_j = 0$ vs. $H_{1j} : \beta_j \neq 0$.
For that, we will use the following steps -

- Step 1 : We will calculate the linear predictors η_i's on the basis of $\hat{\beta}_{LASSO.min}$.
- Step 2 : Now, we will calculate $\hat{\pi}_i$ using `plogis(`η_i`).`
- Step 3 : Using these $\hat{\pi}_i$, we will generate random sample from Bernoulli Distribution.
- Step 4 : We will fit logistic lasso model on this using `lambda.min`
- Step 5 : Then, using the variables having non-zero coefficients, we will again fit a glm and will collect the z values.
- Step 6 : Repeat steps 3 to 6 , R times.
- Step 7 : Use Sample Quantiles as cutoff points.

We have written, the following function to made easy our task.

```r
my.Simulation <- function(var_name){

  set.seed(1234)
  beta_hat <- as.matrix(coef(fm.min.lasso))  #beta_hat_lasso
  beta_hat[rownames(beta_hat)%in%c(var_name),] <- 0
  eta.x <- as.vector(X.mat_train%*%beta_hat) #linear predictor
  pi.hat <- plogis(eta.x)   #pi.hat
  sim_Beta_hat <- NULL
  sim_p_val <- NULL
```

```
 for(i in 1:100){ #Loop starts
      y <- rbinom(nrow(PCOSdata_train),size = 1,prob = pi.hat)  #generating y value
      cv.sim <- glmnet::cv.glmnet(X.mat_train,y,
                   family = "binomial",alpha = 1)
      sim.model <- glmnet::glmnet(X.mat_train,y,alpha = 1,
           family = "binomial",lambda = cv.sim$lambda.min)
      sim.coef <- as.matrix(round(coef(sim.model), 4))
      sim.min.variables <- rownames(sim.coef)[sim.coef[,1] != 0]
     lasso.df <- X.mat_train.df[,colnames(X.mat_train.df)[-1]%in%sim.min.variables]

  if(ncol(lasso.df) != 0){
     glm.model <- glm(y ~. ,data = lasso.df,family = binomial(link = 'logit'))
     sim_Beta_hat <- rbind(sim_Beta_hat,ifelse(colnames(X.mat_train)%in%sim.min.variables,
        as.vector(summary(glm.model)$coef[,3]),0))
        sim_p_val <- rbind(sim_p_val,ifelse(colnames(X.mat_train)%in%sim.min.variables,
      as.vector(summary(glm.model)$coef[,4]),0.5))
  }else{
     sim_Beta_hat <- rbind(sim_Beta_hat,rep(0,ncol(X.mat_train)))
     sim_p_val <- rbind(sim_p_val,rep(0.5,ncol(X.mat_train)))
   }

 }
 colnames(sim_Beta_hat) <- colnames(X.mat_train)
 return(c(dec_Level  = mean(sim_p_val[,colnames(sim_Beta_hat)%in%c(var_name)] <= 0.05),
   quantile(sim_Beta_hat[,colnames(sim_Beta_hat)%in%c(var_name)],c(0.025,0.975))))
}
```

Now, we will use this function to calculate check whether the individuals variables are significant

```
simulation_results <-  NULL
for(i in 2:length(lasso.1se.variables)){
  simulation_results <- rbind(simulation_results,my.Simulation(lasso.min.variables[i]))
}
sim_result <- cbind(simulation_results,z_value = summary(sv_fm.min.lasso)$coef[-1,3],
                   variable = lasso.min.variables[-1])
sim_result
```

```
##    dec_Level     X2.50.   X97.50.    z_value        variable
## 1       0.01 -1.7818821 0.9071952  0.1623741             Age
## 2       0.01 -1.1049443 0.6606060  1.2349724          Weight
## 3       0.05 -1.8761988 1.3310978 -0.7801271   Blood_Group12
## 4       0.08 -2.0043019 3.2155131  2.2058835   Blood_Group14
## 5       0.05  0.0000000 3.0184336  1.6041382      Pulse_rate
## 6        0.09 -1.0980998 3.6151661  4.4119839           Cycle
## 7        0.08 -0.5884618 4.0055514 -1.0544612    Cycle_length
## 8        0.05 -0.5257046 3.0324348 -1.9848904 Marriage_Status
## 9        0.11 -0.5077363 5.8576312 -0.9656632        Pregnant
## 10       0.50  0.0000000 0.0000000  0.6175617              LH
## 11       0.04 -1.6066168 1.8064914 -1.3871093 Waist_Hip_Ratio
## 12       0.11 -1.7253108 3.9751907  0.6795608             AMH
## 13       0.07 -1.4232037 4.6491981 -1.2237444          Vit_D3
## 14       0.11 -1.3492097 2.4842585  3.0750523     Weight_gain
## 15       0.11 -1.3286164 2.9101493  3.4672132     hair_growth
## 16       0.14 -1.3483282 1.6492483  2.6454082  Skin_darkening
## 17       0.08 -0.9866333 4.0338080  0.6737102       Hair_loss
## 18       0.06 -0.8093968 1.9362940  2.1731985         Pimples
## 19       0.09 -1.1997848 1.2844597  2.0983972       Fast_food
## 20       0.07 -1.5624685 4.5793808  5.1185432    Reg_Exercise
## 21       0.06 -1.6580010 2.3462110 -1.3111333     BP_Systolic
## 22       0.12 -1.3296746 5.1700153  0.8725122    Follicle_No_L
## 23       0.10 -0.9776205 3.6917251  5.4831078    Follicle_No_R
## 24       0.04 -0.8285820 2.0048661  2.1382799     Avg_F_size_L
```

Now, we can see which variables are significant.

```
##  [1] "Weight"         "Cycle"          "Cycle_length"    "Marriage_Status"
##  [5] "Pregnant"       "LH"             "Weight_gain"     "hair_growth"
##  [9] "Skin_darkening" "Pimples"        "Fast_food"       "Reg_Exercise"
## [13] "Follicle_No_R"  "Avg_F_size_L"
```

In addition we can look at the individual densityplot of z statistics.

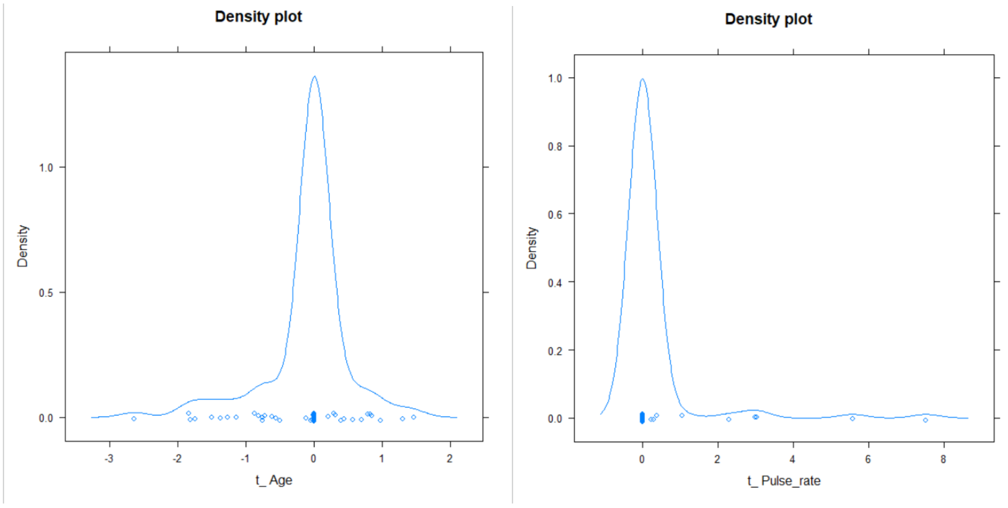


Diagram : Density plot of z statistics for these scenario.

Which suggests that, the symmetricity of z statistics in large sample is destroyed.

Interpretation :

We can interpret the coefficient estimates for those variables which are significant.

```
##                         [,1]
## Weight            0.03221268
## Cycle             0.75202308
## Cycle_length     -0.17030756
## Marriage_Status -0.13734772
## Pregnant         -0.44209318
## LH                0.05776080
## Weight_gain       1.66762876
## hair_growth       1.86067373
## Skin_darkening    1.25195762
## Pimples           1.07832481
## Fast_food         1.09980341
## Reg_Exercise      0.58087920
## Follicle_No_R     0.50837227
## Avg_F_size_L      0.17153630
```

Since, we don't know whether the estimators are unbiased or not. We cannot interpret the coefficients in usual manner. But we can roughly say that -

- As weight increases chances of PCOS increases on an average.
- If a female has irregular cycle, then chances of PCOS increases on an average.
- If Cycle length decreases, then chances of PCOS increases on an average.
- If year of Marriage increases, then chances of PCOS increases on an average.
- If a female is Pregnant, then chances of PCOS increases on an average.
- If LH increases, then chances of PCOS increases on an average.
- If a female has weight gain , then chances of PCOS increases.
- If a female has hair growth, then chances of PCOS increases.
- If a female has skin darkening, then chances of PCOS increases.
- If a female has pimples, then chances of PCOS increases.

- If a female eat fast foods, then chances of PCOS increases.
- If Follicle_No_R increases, then chances of PCOS increases.
- If Avg_F_size_L increases, then chances of PCOS increases.

Conclusion :

From the above analysis of the data we get that Weight, Type of Period Cycle, Cycle Length, Year of Marriage, LH level, Prgenant or not, Weight gain or not, Hair growth or not, Skin darkening or not, have pimples or not, eat fast foods or not, Do Reg Excercise or not , Follicle No R and Avg_F_size_L are important variables to influence chances of PCOS. Also, we get good a model -

```
##    (Intercept)          Cycle             LH    Weight_gain    hair_growth
##     -8.2212149      0.9315411      0.0824581      1.7419949      1.4917546
## Skin_darkening        Pimples      Fast_food  Follicle_No_L  Follicle_No_R
##      1.5035996      0.5946759      0.8894254      0.1360667      0.4434274
```

With evaluation metrics -

```
##    Accuracy  Specificty Sensitivity   Precision     LogLoss
##   0.9158879   0.9583333   0.8285714   0.9062500   0.2464480
```

References :

1. Random Forest
2. Support Vector Machine
3. Xg Boost
4. Robust GLM
5. Lasso Regresion