Polycystic Ovary Syndrome Analysis and Prediction

Shrayan Roy, Roll : MD2220
Guide : Dr. Deepayan Sarakar

Indian Statistical Institute, Delhi Centre

November 06,2022

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 describes the numerous small cysts (fluid filled sacs) that form in the ovaries.

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.

Column Names of Data Frame :

```
colnames(PCOSdata)
```

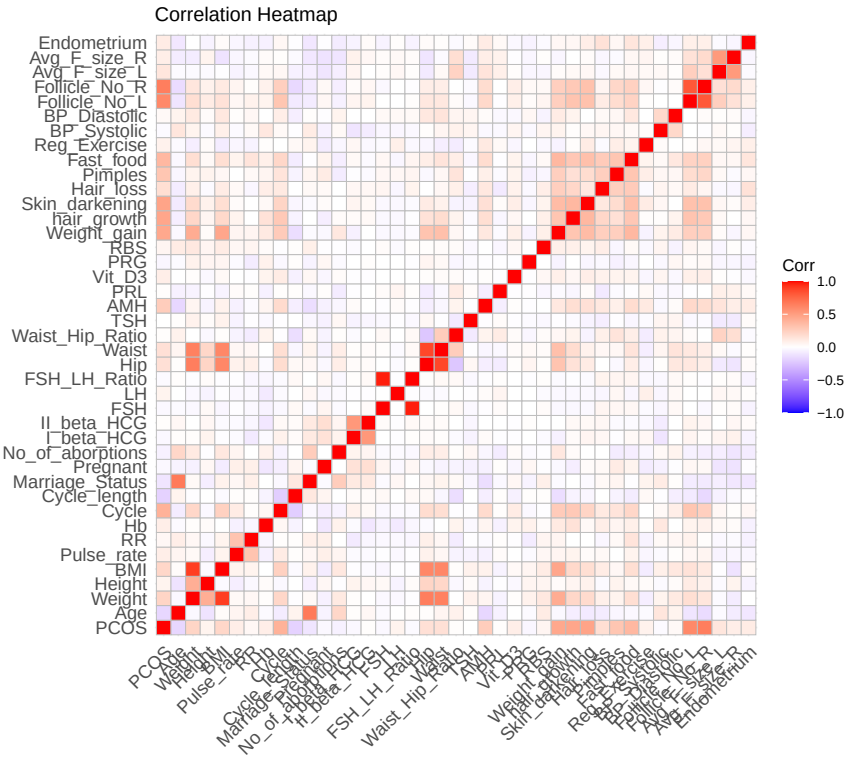
```
##  [1] "Sl..No"           "Patient_File_No"  "PCOS"             "Age"
##  [5] "Weight"           "Height"           "BMI"              "Blood_Group"
##  [9] "Pulse_rate"       "RR"               "Hb"               "Cycle"
## [13] "Cycle_length"     "Marriage_Status"  "Pregnant"         "No_of_aborptions"
## [17] "I_beta_HCG"       "II_beta_HCG"      "FSH"              "LH"
## [21] "FSH_LH_Ratio"     "Hip"              "Waist"            "Waist_Hip_Ratio"
## [25] "TSH"              "AMH"              "PRL"              "Vit_D3"
## [29] "PRG"              "RBS"              "Weight_gain"      "hair_growth"
## [33] "Skin_darkening"   "Hair_loss"        "Pimples"          "Fast_food"
## [37] "Reg_Exercise"     "BP_Systolic"      "BP_Diastolic"     "Follicle_No_L"
## [41] "Follicle_No_R"    "Avg_F_size_L"     "Avg_F_size_R"     "Endometrium"
```

Data Processing and Cleaning :

- We have deleted the NA values.
- Also, there are some unusual observations. **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)
```

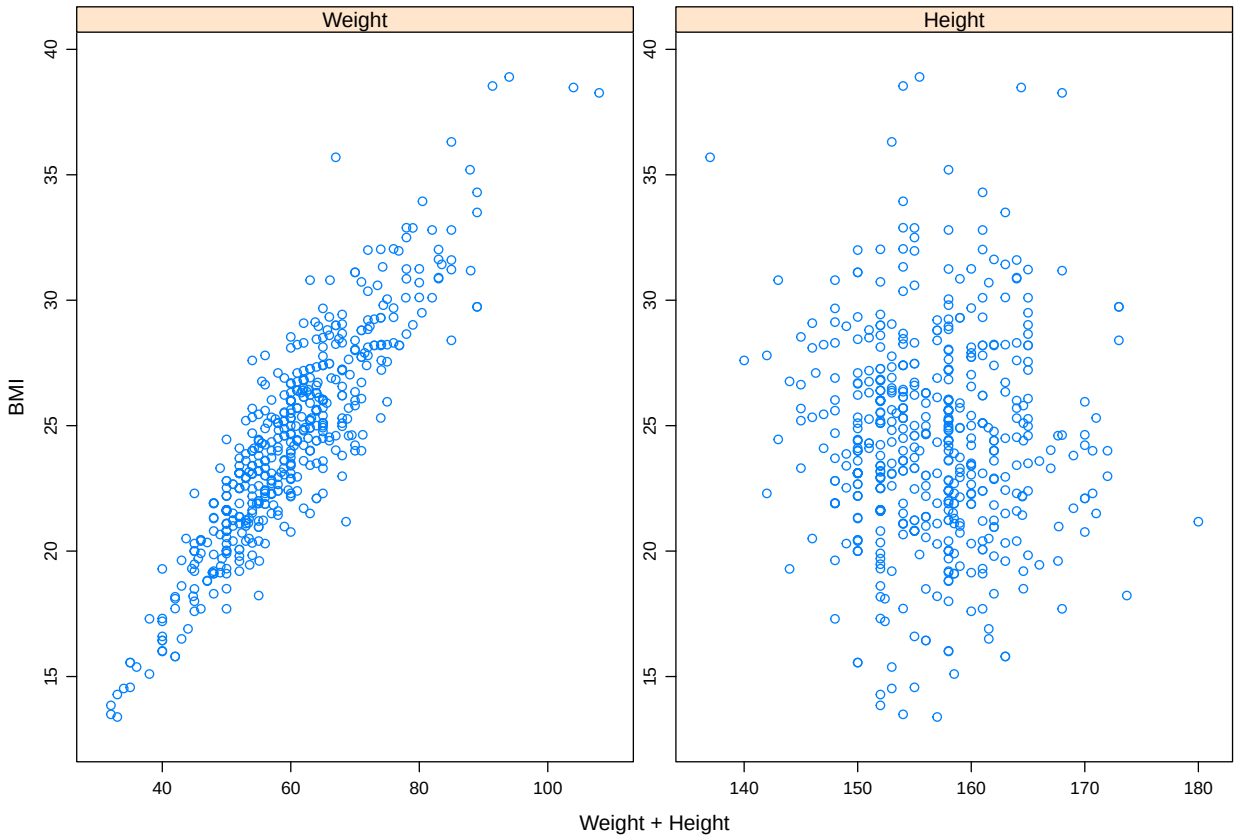
Some Preliminary Analysis :



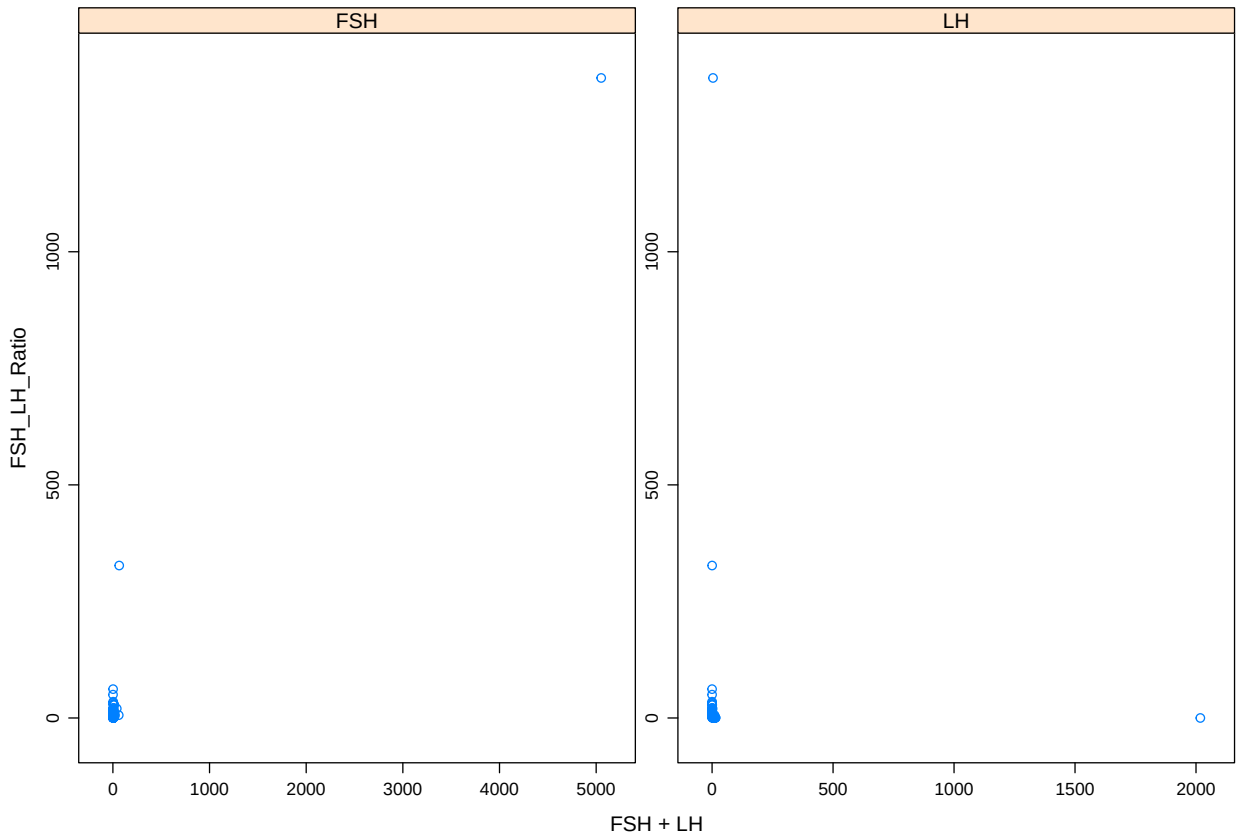
Some Preliminary Analysis : (Contd.)

- BMI & Weight are highly correlated.
- Also, (Waist, Hip), (Follicle_No_L, Follicle_No_R) are highly correlated.
- So, We prefer to delete BMI column. Because, it will introduce multicollinearity in the model.
- Similarly for FSH_LH ratio also. Also, Hip and Waist are highly correlated but they are not much correlated with Waist_Hip_Ratio. Which is very clear from the graph below.

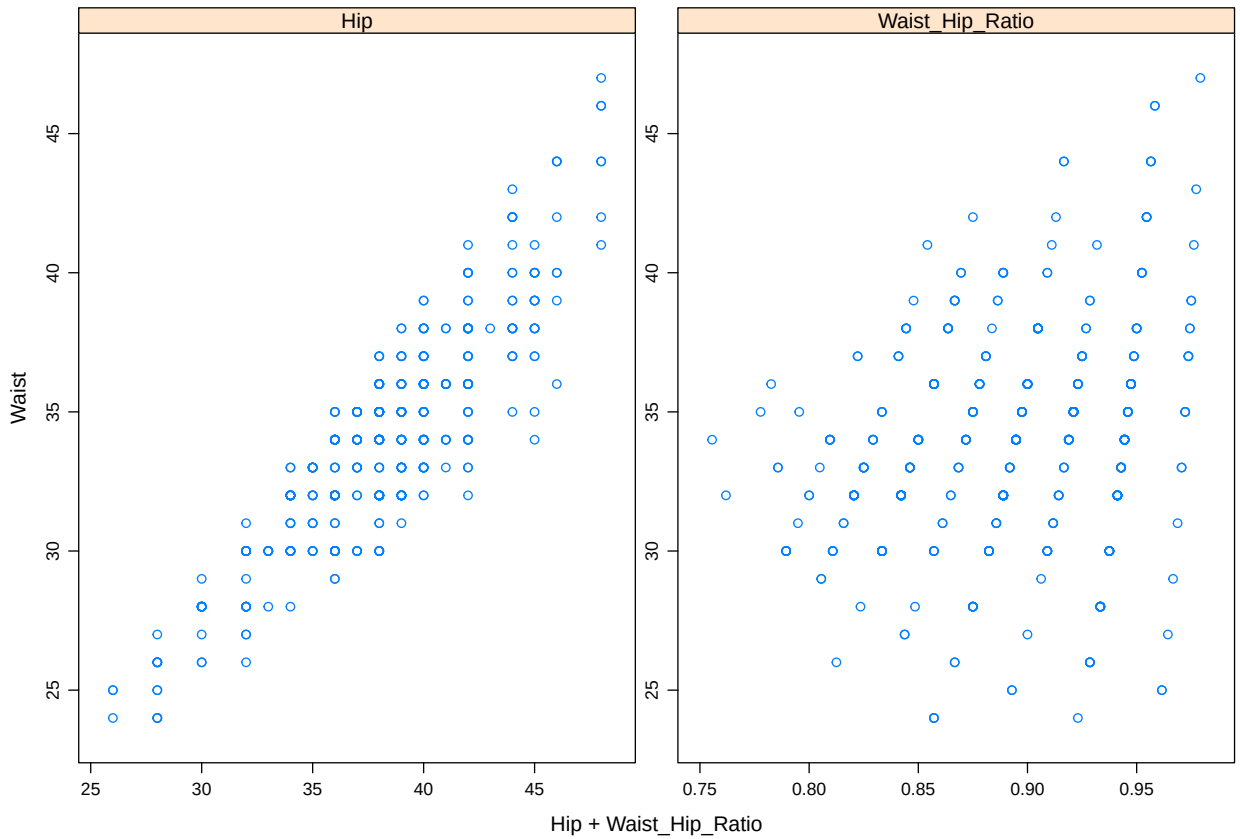
Some Preliminary Analysis : (Contd.)



Some Preliminary Analysis : (Contd.)



Some Preliminary Analysis : (Contd.)



Some Preliminary Analysis : (Contd.)

So, finally we will work with the the following data set.

```
PCOSdata <- PCOSdata[,!colnames(PCOSdata)%in%c('FSH_LH_Ratio','Hip','BMI')]
dim(PCOSdata)   #Remaining dataset dimension
```

```
## [1] 533  41
```

Lets see what proportion of patient have PCOS in our data set.

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

```
## [1] 0.3227017
```

Not, a very imbalanced data set. So, we can carry forward our analysis.

Exploratory Data Analysis :

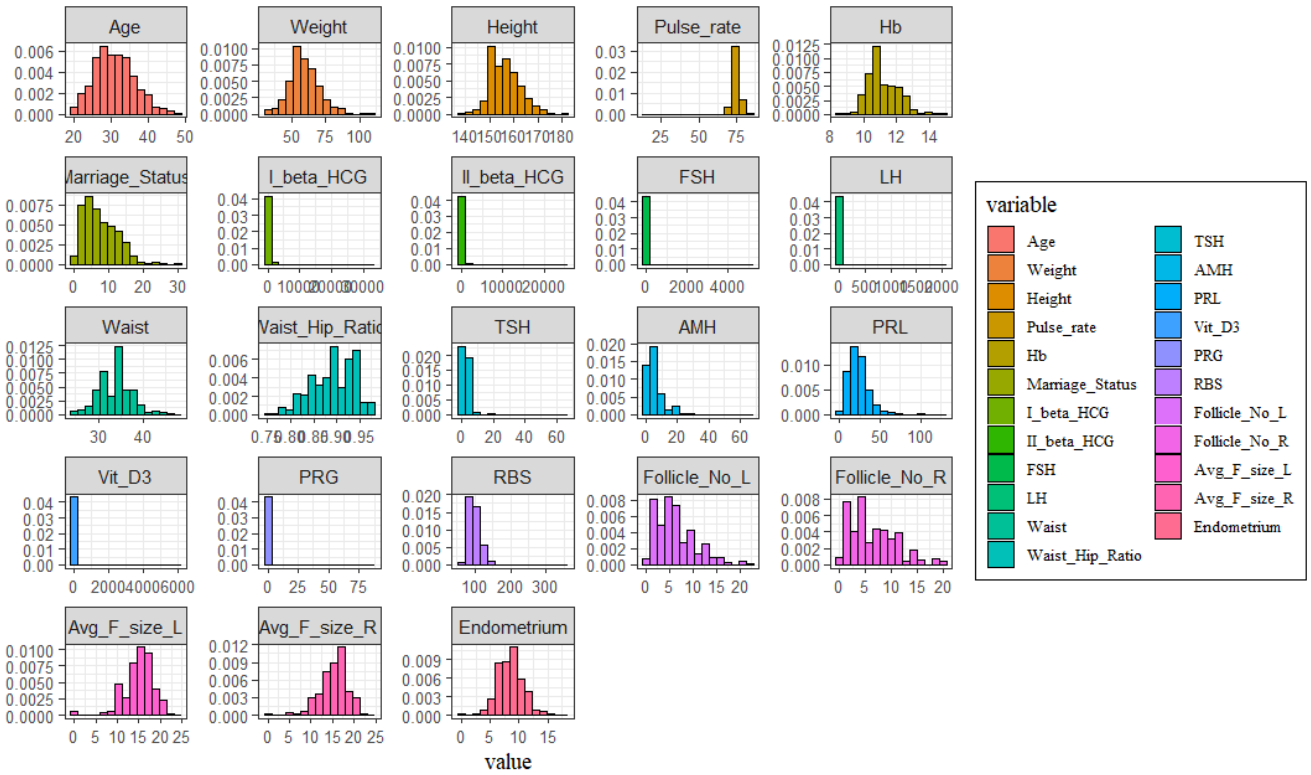


Figure 1: Histogram of Numerical Variables

Exploratory Data Analysis : (Contd.)

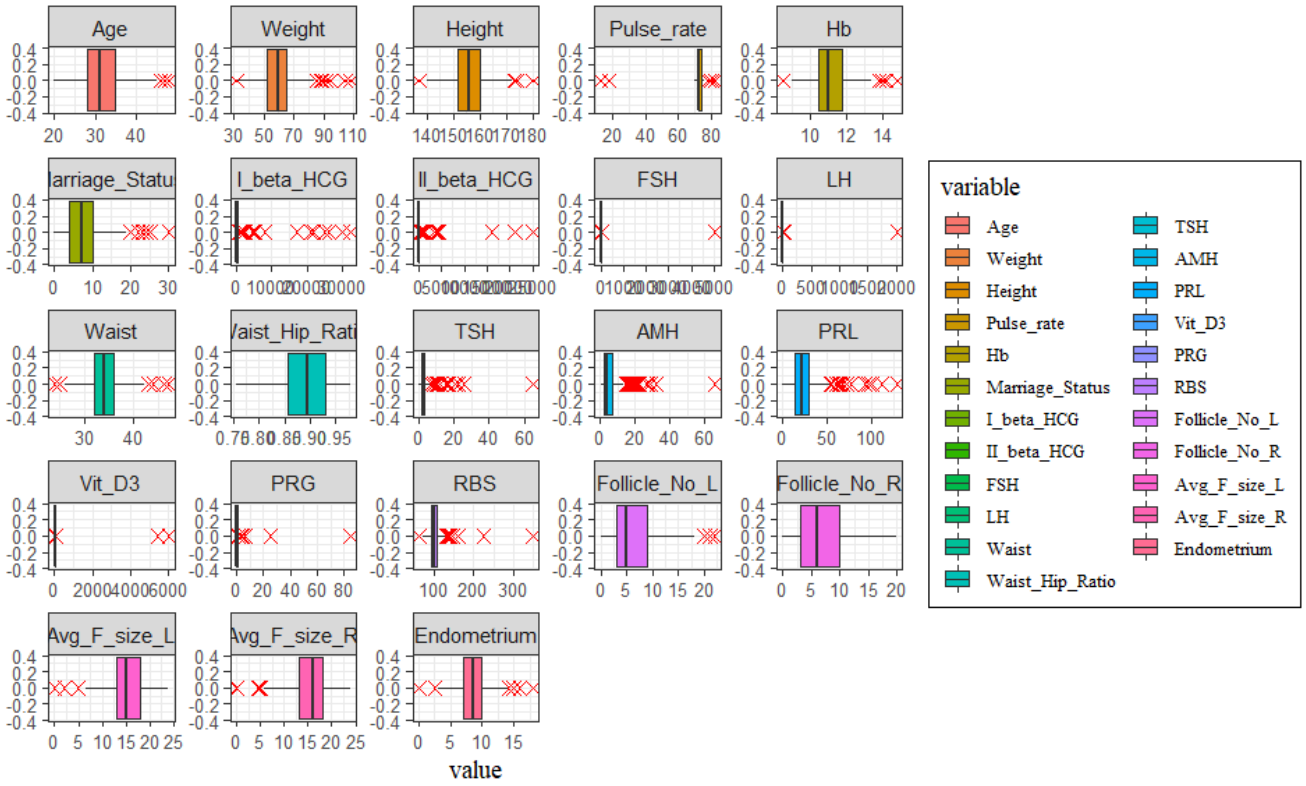


Figure 2: Boxplot of Numerical Variables

Exploratory Data Analysis : (Contd.)

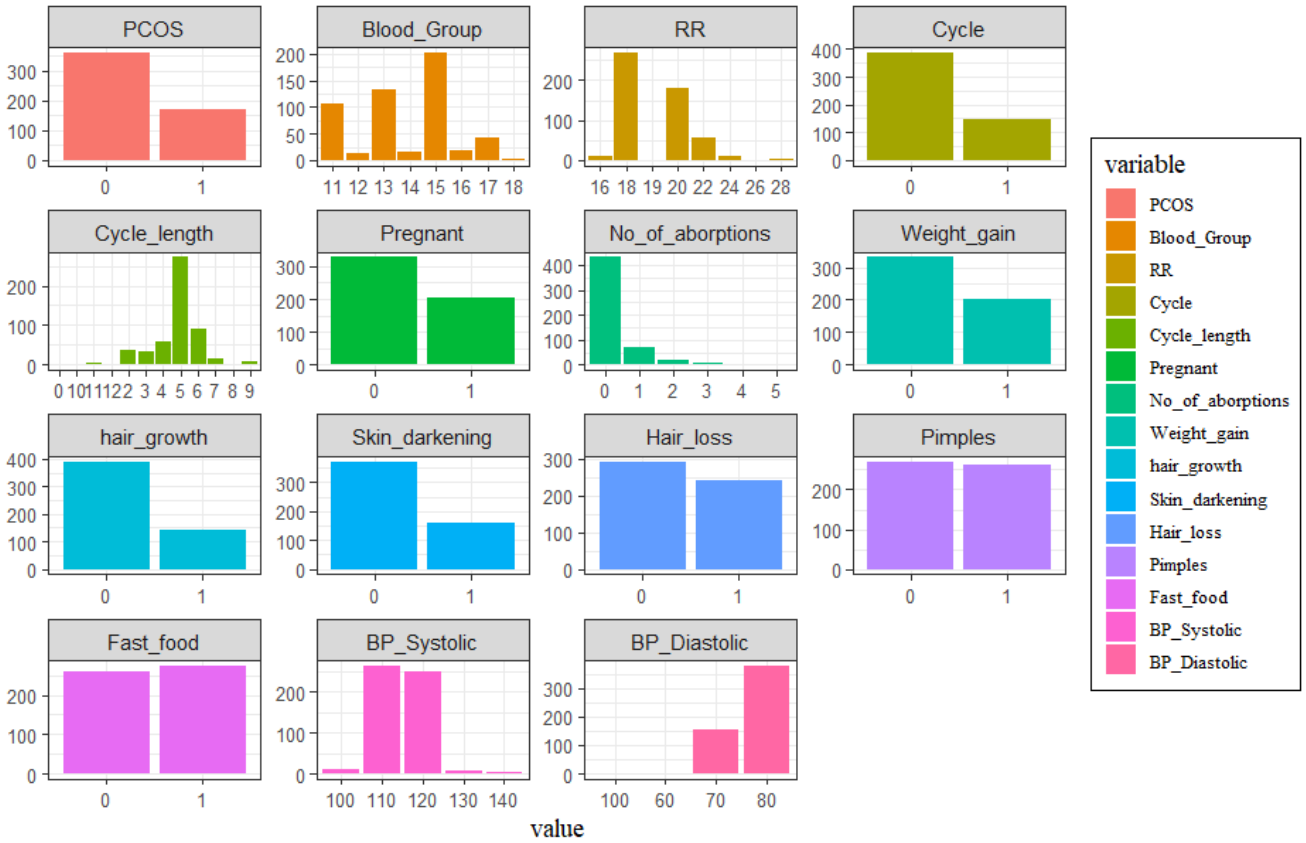


Figure 3: Barplot of Categorical Variables

Exploratory Data Analysis : (Contd.)

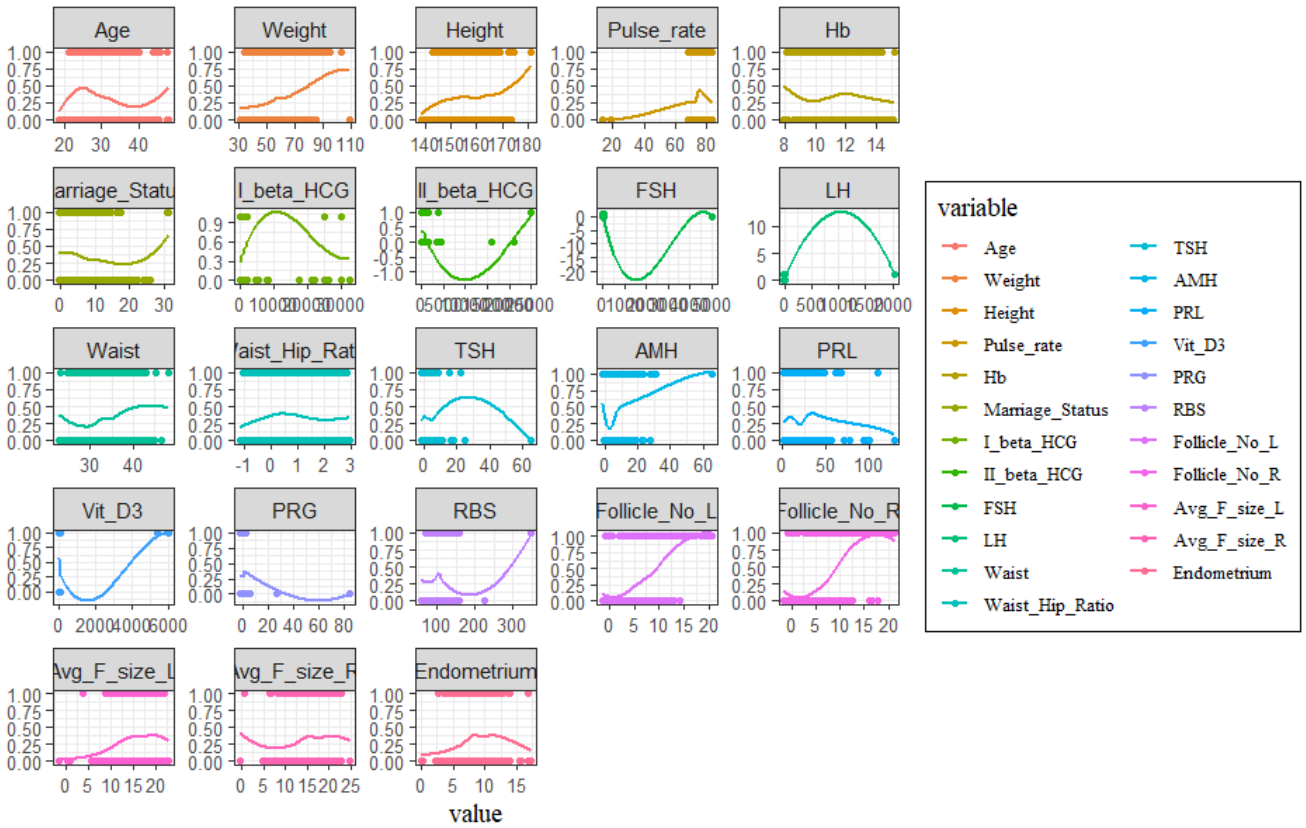


Figure 4: Scatterplot of Numerical Variables with PCOS

Exploratory Data Analysis : (Contd.)

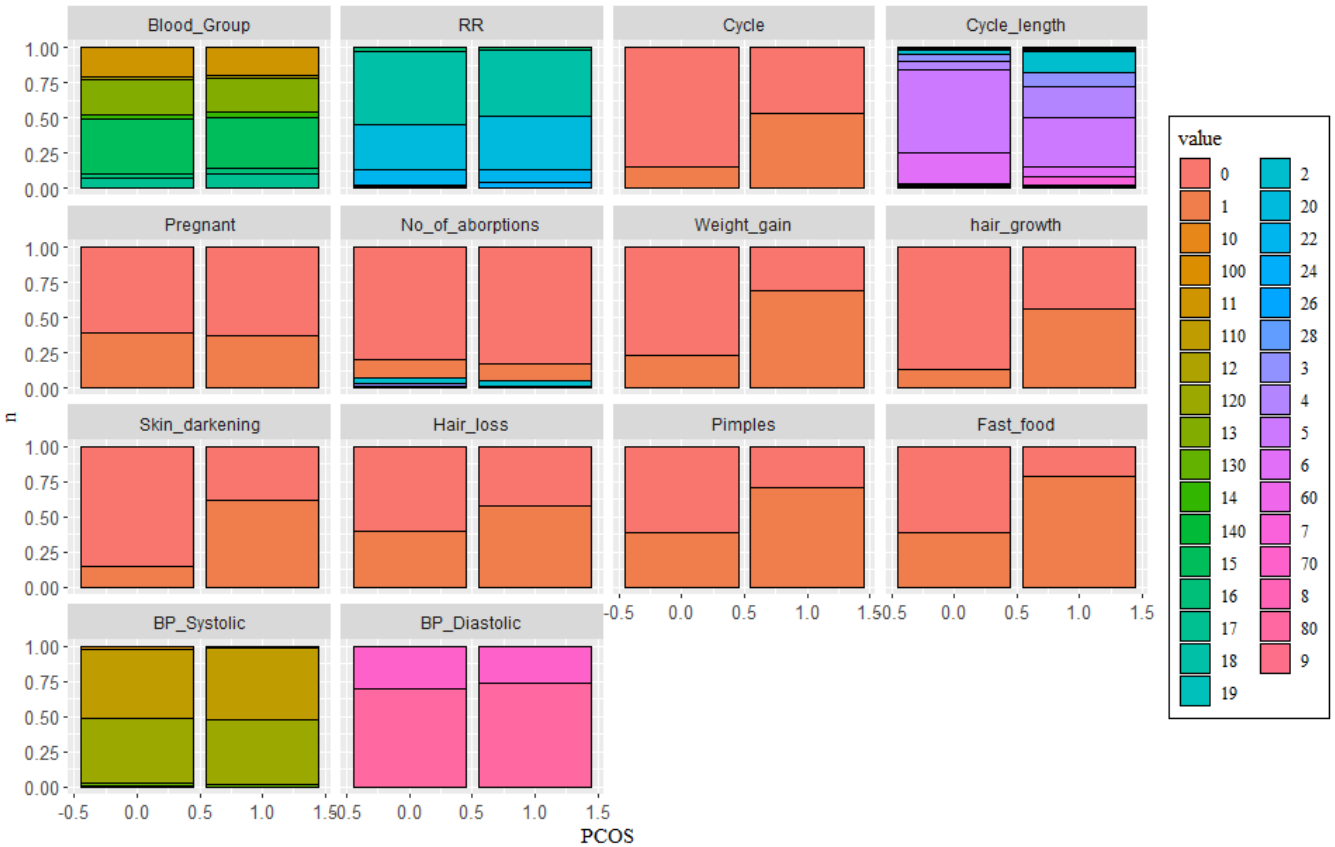
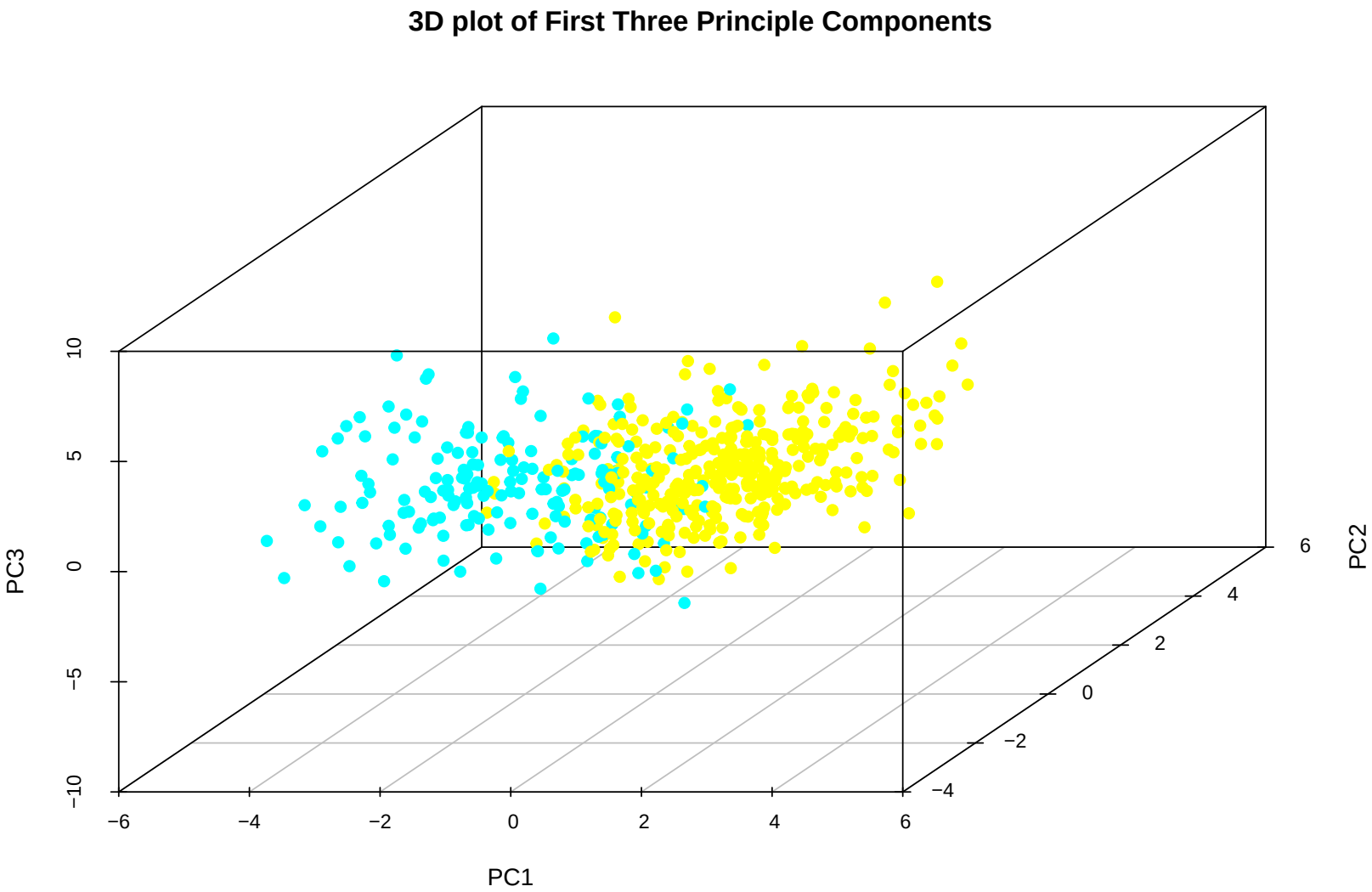


Figure 5: Stacked Barplot of Categorical Variables

Exploratory Data Analysis : (Contd.)



Exploratory Data Analysis : (Contd.)

So,from the above EDA we have learned that -

- Some of independent 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.

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 and will compare them -

- 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)

Methods Used : (Contd.)

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

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

Brief Discussion of Methods Used :

Logistic Regression :

In statistics, the logistic model (or logit model) is a statistical model that models the probability of an event taking place by having the log-odds for the event be a linear combination of one or more independent variables.

$$Ln\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_k X_k$$

Brief Discussion of Methods Used : (Contd.)

Robust 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, $r_i = \frac{(y_i - \mu_i)}{\sqrt{V_i}}$ is the Pearson residuals & ψ_k the Huber function. $a(\mu i) = E[\psi_k(R_i)|x_i]$, 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
$F(u) = \frac{exp(u)}{(1+exp(u))}$, $V_i(\mu_i) = \mu_i(1 - \mu_i) = V_i$.
$a(\mu i) = \psi_k((1 - \mu_i)/\sqrt{V_i})\mu_i + \psi_k(-\mu i \sqrt{V_i})(1 - \mu_i)$.

Brief Discussion of Methods Used : (Contd.)

Lasso Logistic Regression :

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.

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

A solution can be found using nonlinear programming methods (Koh et al., 2007, for example).

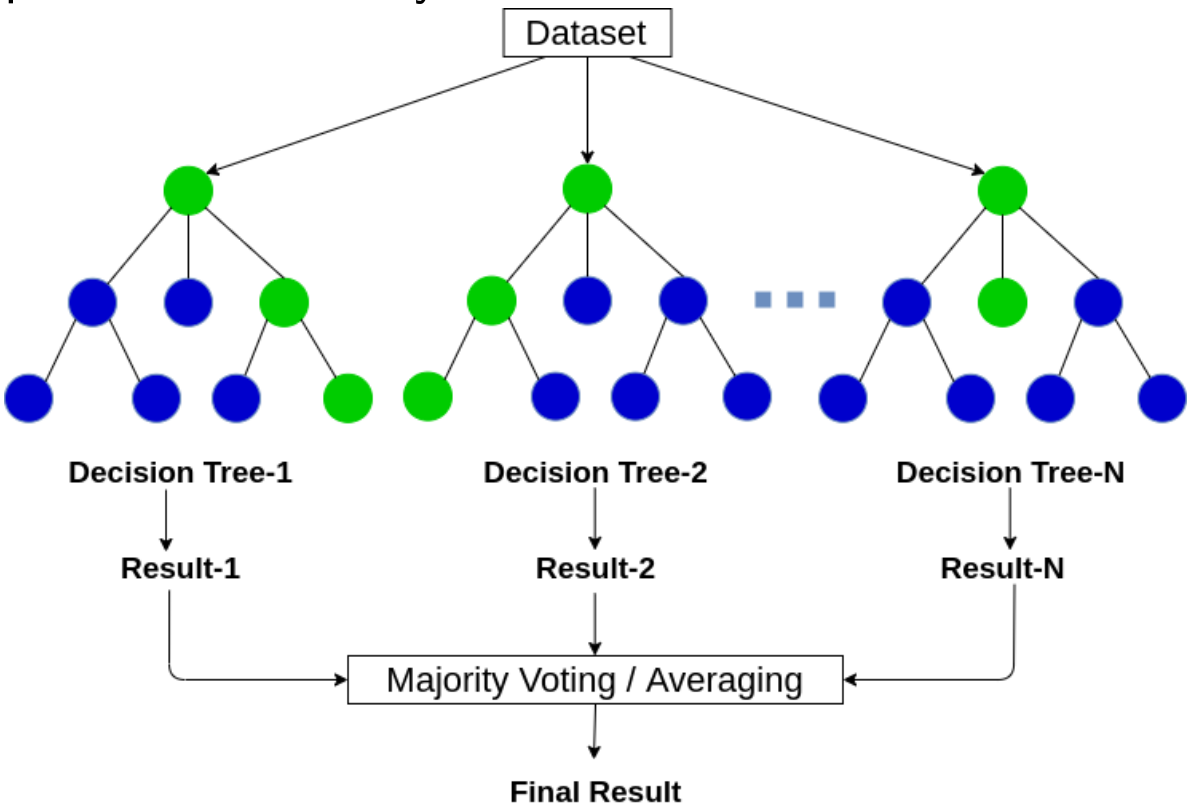
K - Nearest Neighbor Method (KNN) :

- Given , find nearest neighbours.
- Classify as modal (most common) class among these observations.
- Can use different distance metrics depending upon type of data.

Brief Discussion of Methods Used : (Contd.)

Random Forest :

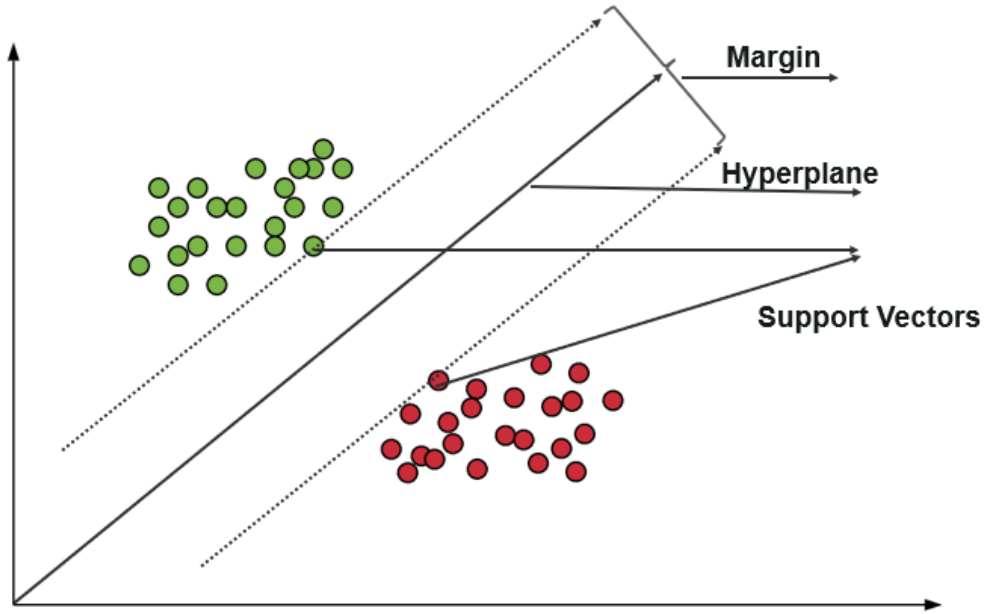
- A classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.



Brief Discussion of Methods Used : (Contd.)

Support Vector Machine (SVM) :

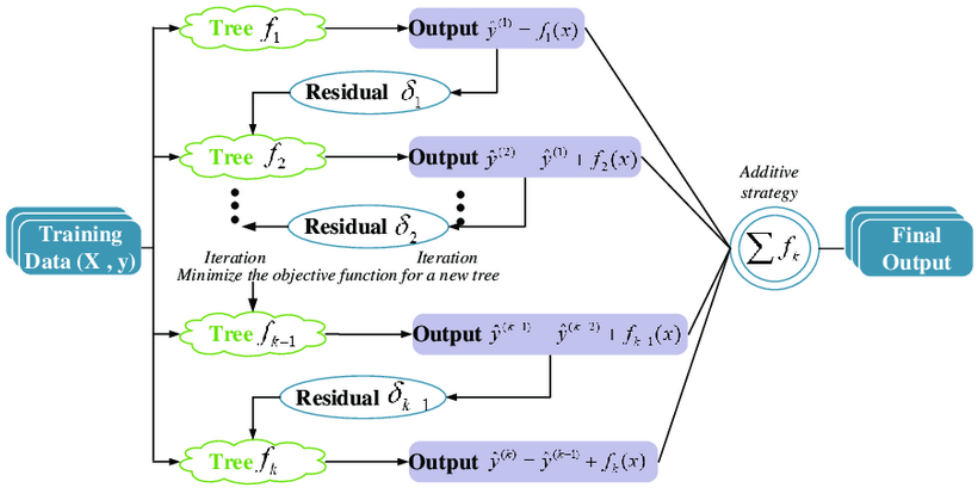
- The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future.
- Chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors.



Brief Discussion of Methods Used : (Contd.)

Extreme Gradient Boosting:

- 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.
- 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.



Confusion Matrix & Evaluation Metrics:

- A table that is used to define the performance of a classification algorithm.
- It visualizes and summarizes the performance of a classification algorithm.

		Predicted Class: Positive	Predicted Class: 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** $\frac{TP+TN}{(TP+TN+FP+FN)}$

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

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

Model Fitting :

Split the data into two parts. Testing and training set. Use Test data to fit model and Train data to test the model.

```
dim(PCOSdata_train)
```

```
## [1] 426  41
```

```
dim(PCOSdata_test)
```

```
## [1] 107  41
```

Fitting Base line Logistic Regression Model

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

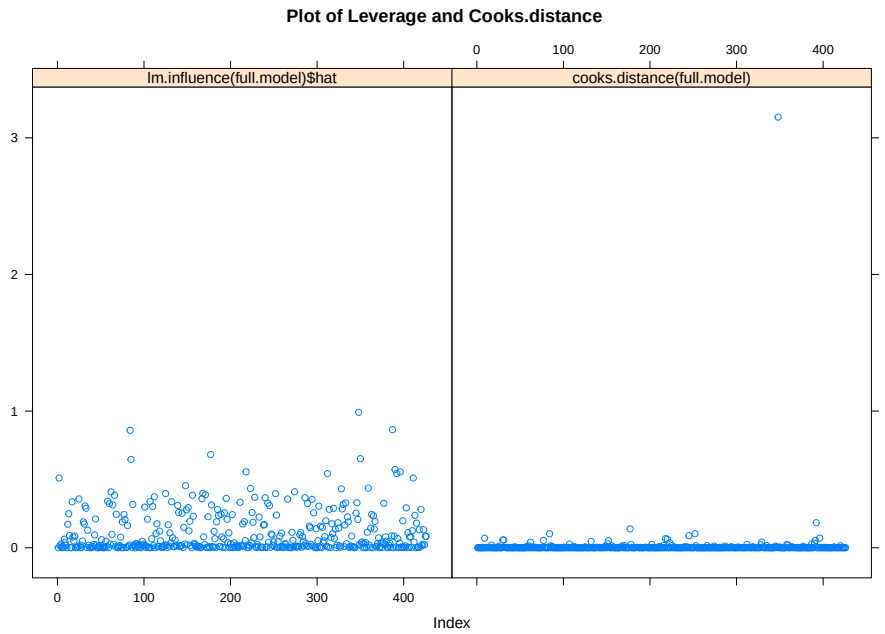
```
##
## Call:
## glm(formula = PCOS ~ ., family = binomial(link = "logit"), data = PCOSdata_train[,
##     -c(1, 2)])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9295  -0.2144  -0.0551   0.0599   3.4796
##
## Coefficients:
##                    Estimate Std. Error 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
```

Variance Inflation Factor of Full model

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

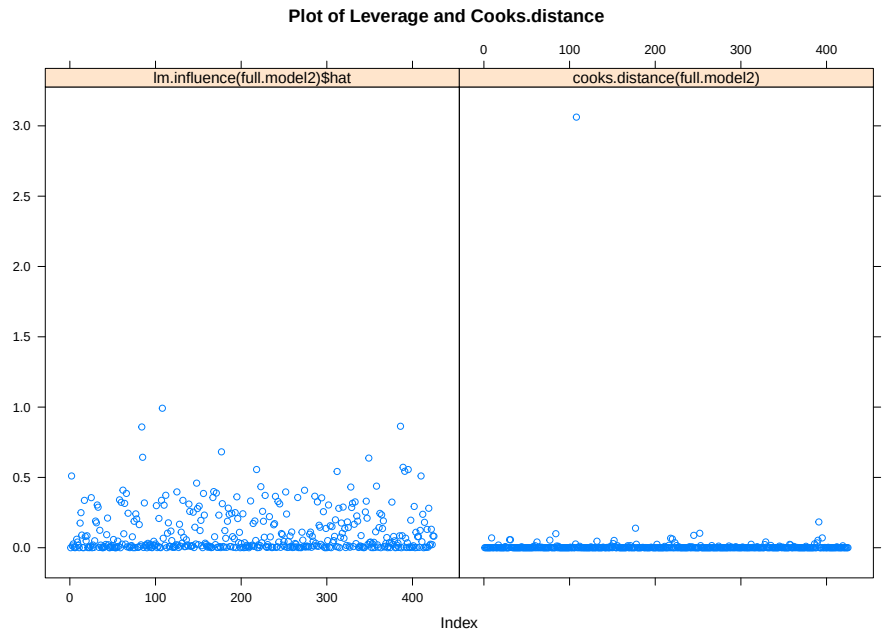
```
##                       GVIF Df GVIF^(1/(2*Df))
## Age               2.704900  1        1.644658
## Weight            3.357882  1        1.832452
## Height            1.970352  1        1.403692
## Blood_Group       5.873210  7        1.134801
## Pulse_rate        2.095365  1        1.447538
## 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
```

```
##                       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
```



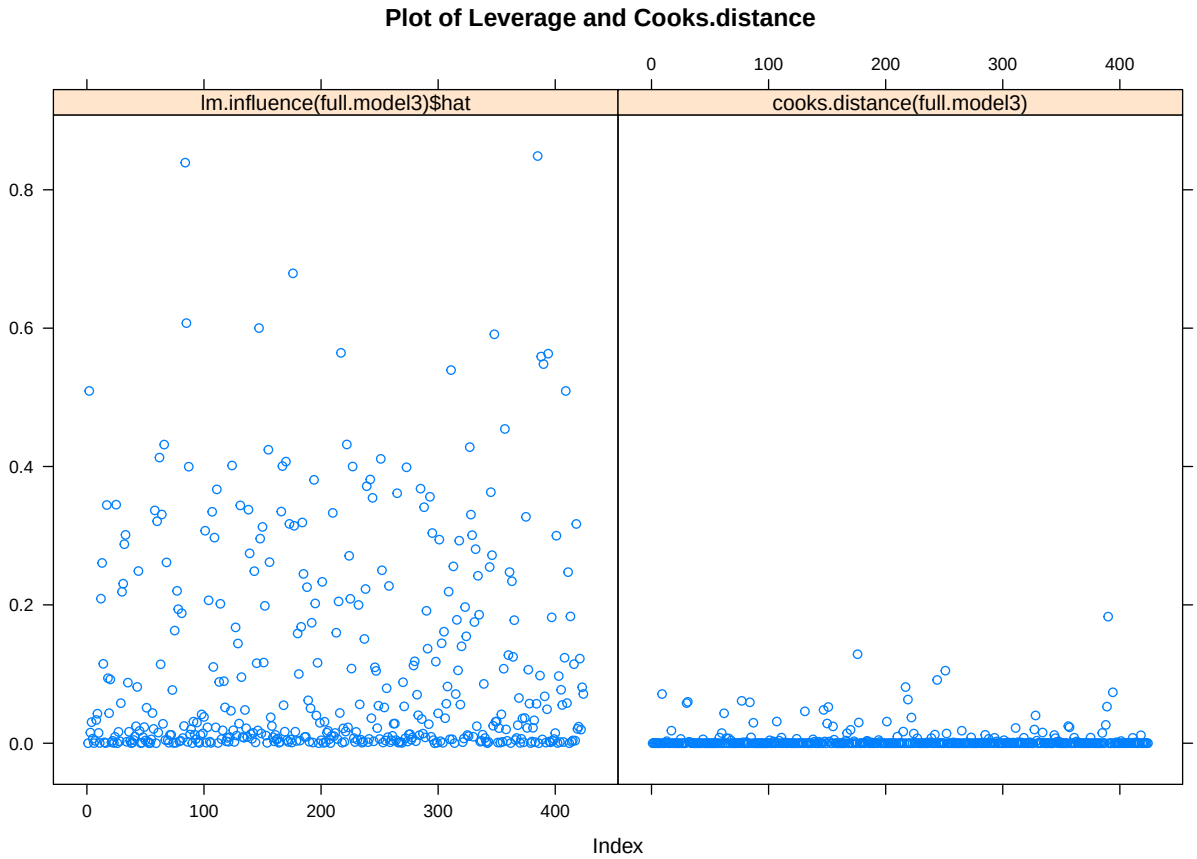

```
PCOSdata_train[(cooks.distance(full.model) > 1),]
```

```
##     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
```




```
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
```



Our Final Model is -

```
final_full.model <- full.model3   #This is our final model
```

Finding Best Model - Forward Selection :

```
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"
```

- All Variables are selected by forward selection !

Finding Best Model - Sequential Selection

```
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
```

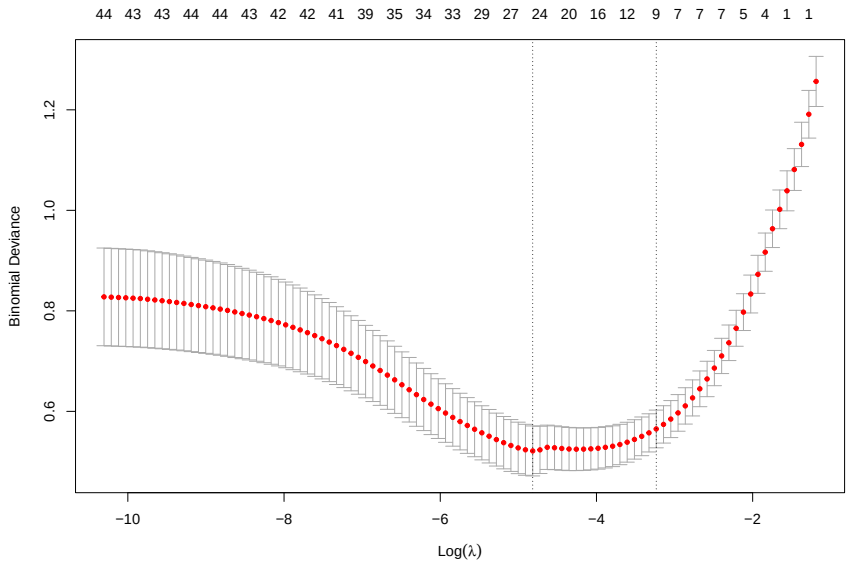
- All the selected variables are very meaningful from EDA.
- We observe that, the AIC is 195.82, which is less than the AIC of full.model. Which is a good indication.

Variable Selection : Logistic Lasso

```
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
```

We will use both values of λ.



Selected Variables for Both Lambda

```
(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"
```

- We can refit our model using the variables selected from lasso.
- That may give in some sense satisfactory fit.

Fitting Logistic Regression Using the Lasso Selected Variables

```
sv_fm.min.lasso
```

```
##
## Call:  glm(formula = PCOS ~ ., family = binomial(link = "logit"), data = as.data.frame(cbind(PCOS = PC
##     3], X.mat_train[, colnames(X.mat_train) %in% c("PCOS", lasso.min.variables)])))
##
## Coefficients:
##     (Intercept)              Age           Weight    Blood_Group12
##       -9.904364         0.009086         0.032213        -0.927603
##   Blood_Group14       Pulse_rate            Cycle     Cycle_length
##        2.311932         0.141885         0.752023        -0.170308
## Marriage_Status         Pregnant               LH  Waist_Hip_Ratio
##       -0.137348        -0.442093         0.057761        -6.995900
##              AMH           Vit_D3      Weight_gain      hair_growth
##        0.026691        -0.022558         1.667629         1.860674
##  Skin_darkening        Hair_loss          Pimples        Fast_food
##        1.251958         0.325307         1.078325         1.099803
##    Reg_Exercise      BP_Systolic    Follicle_No_L    Follicle_No_R
##        0.580879        -0.048876         0.076559         0.508372
##    Avg_F_size_L
##        0.171536
##
## Degrees of Freedom: 423 Total (i.e. Null);  399 Residual
## Null Deviance:       530.6
## Residual Deviance: 155.9     AIC: 205.9
```

```
sv_fm.1se.lasso
```

```
##
## Call:  glm(formula = PCOS ~ ., family = binomial(link = "logit"), data = as.data.frame(cbind(PCOS = PC
##     3], X.mat_train[, colnames(X.mat_train) %in% c("PCOS", lasso.1se.variables)])))
##
## Coefficients:
##    (Intercept)           Cycle              LH     Weight_gain     hair_growth
##       -8.22121         0.93154         0.08246         1.74199         1.49175
## Skin_darkening         Pimples       Fast_food   Follicle_No_L   Follicle_No_R
##        1.50360         0.59468         0.88943         0.13607         0.44343
##
## Degrees of Freedom: 423 Total (i.e. Null);  414 Residual
## Null Deviance:       530.6
## Residual Deviance: 181.3     AIC: 201.3
```

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

Robust Logistic Regression

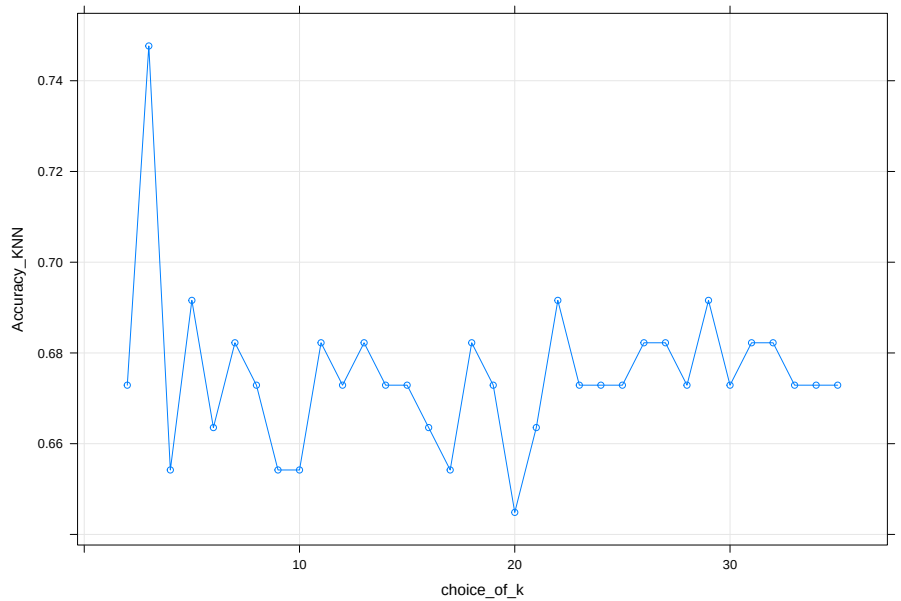
```
robust.glm <- glmrob(PCOS ~ .,data = PCOSdata_train.1[,-c(1,2)],
                     family = binomial,method = "Mqle",
                     control = glmrobMqle.control(tcc = 16))
summary(robust.glm)
```

```
## 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
```

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.

K Nearest Neighbor :

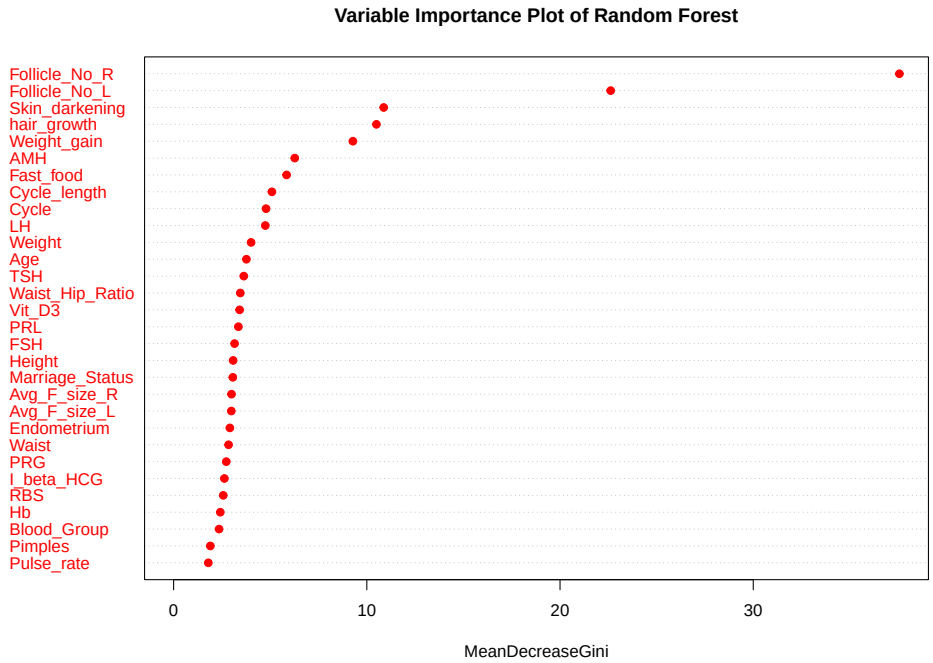
```
choice_of_k <- 2:35
Accuracy_KNN <- NULL
for(i in 1:length(choice_of_k)){
 KNN.model.sim <- knn(train = PCOSdata_numeric[index_train,-1],
                      cl=PCOSdata_train.1$PCOS,
                      test = PCOSdata_numeric[-index_train,-1],
                      k=choice_of_k[i])
 Accuracy_KNN <- 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'))
```



Random Forest :

```
randomForest.model  #summary of random forest
```

```
##
## Call:
##  randomForest(x = PCOSdata_train[, -c(1, 2, 3)], y = as.factor(PCOSdata_train[,      3]), ntree = 700)
##                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
```



- 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 :

```
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 :

```
#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
```

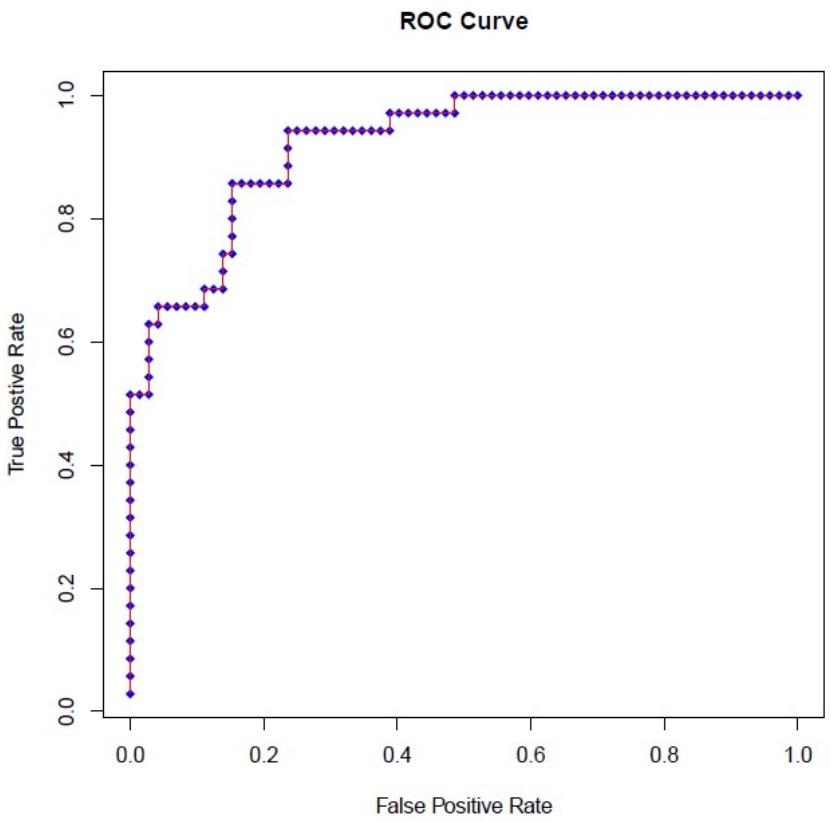
Prediction & Evaluation Metrics :

ROC Curve :

- 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.

Optimal Thresholds and ROC Curve :

ROC Curve for final_model.




```
## [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 = F)
```

```
## [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 = F)
```

```
## [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 = F)
```

```
## [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 = F)
```

```
## [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 = F)
```

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

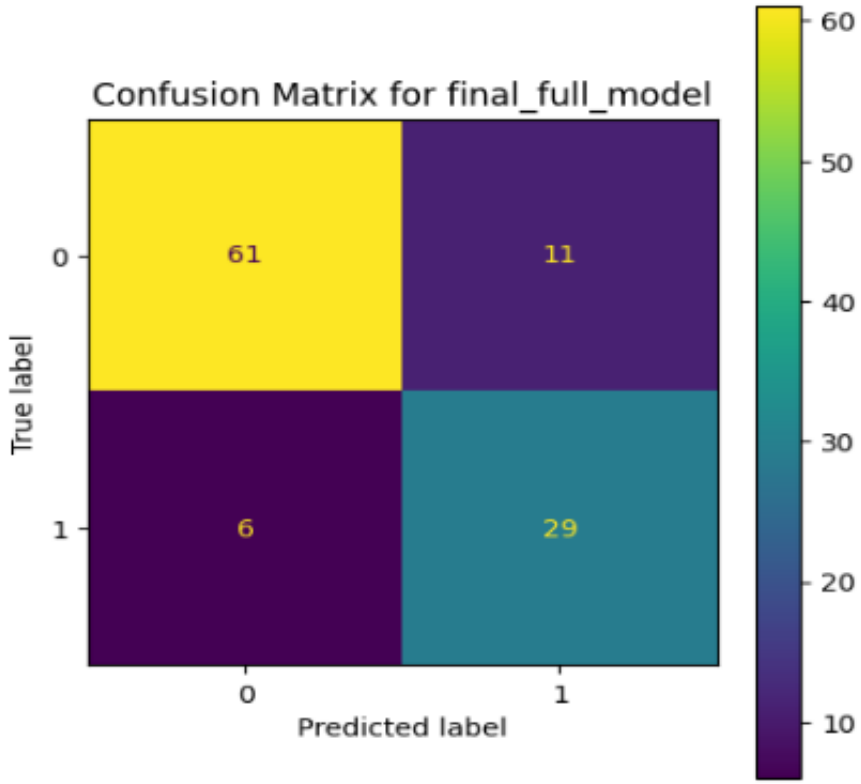
```r
#sv_fm.min.lasso
myROC(Y = PCOSdata_test[,3],pi.hat = predict(sv_fm.1se.lasso,newdata =
     as.data.frame(cbind(PCOS = PCOSdata_test[,3],
      X.mat_test[,colnames(X.mat_test)%in%c('PCOS',lasso.1se.variables)])),
      type = 'response'),plot.R = F)
```

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

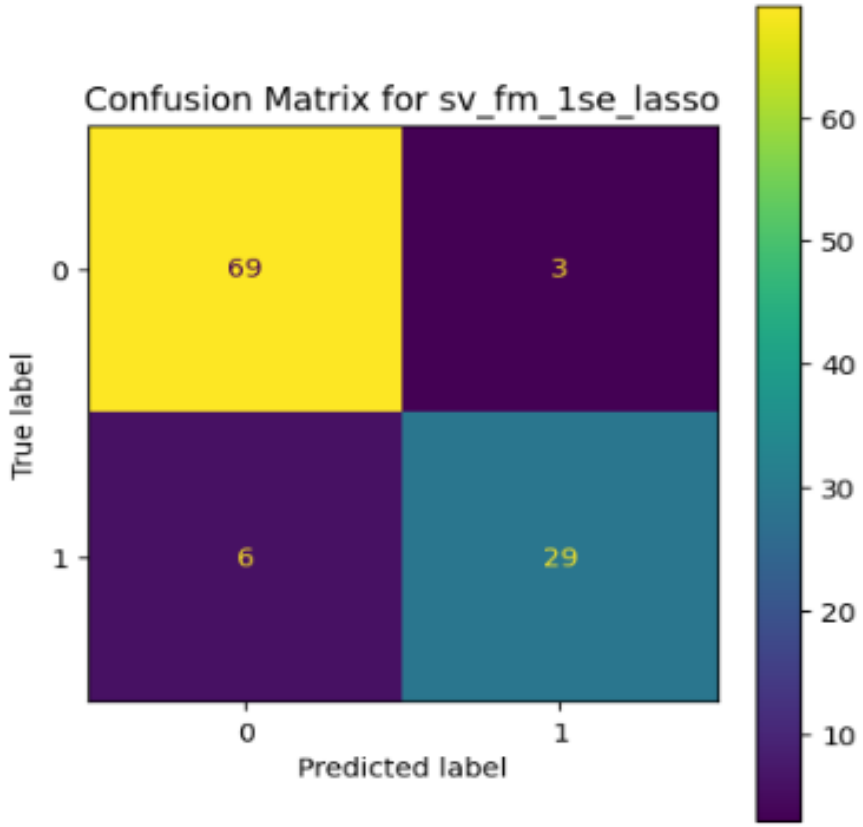
```r
#sv_fm.1se.lasso
myROC(Y = PCOSdata_test[,3],pi.hat = predict(sv_fm.1se.lasso,newdata =
     as.data.frame(cbind(PCOS = PCOSdata_test[,3],
     X.mat_test[,colnames(X.mat_test)%in%c('PCOS',lasso.1se.variables)])),
     type = 'response'),plot.R = F)
```

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

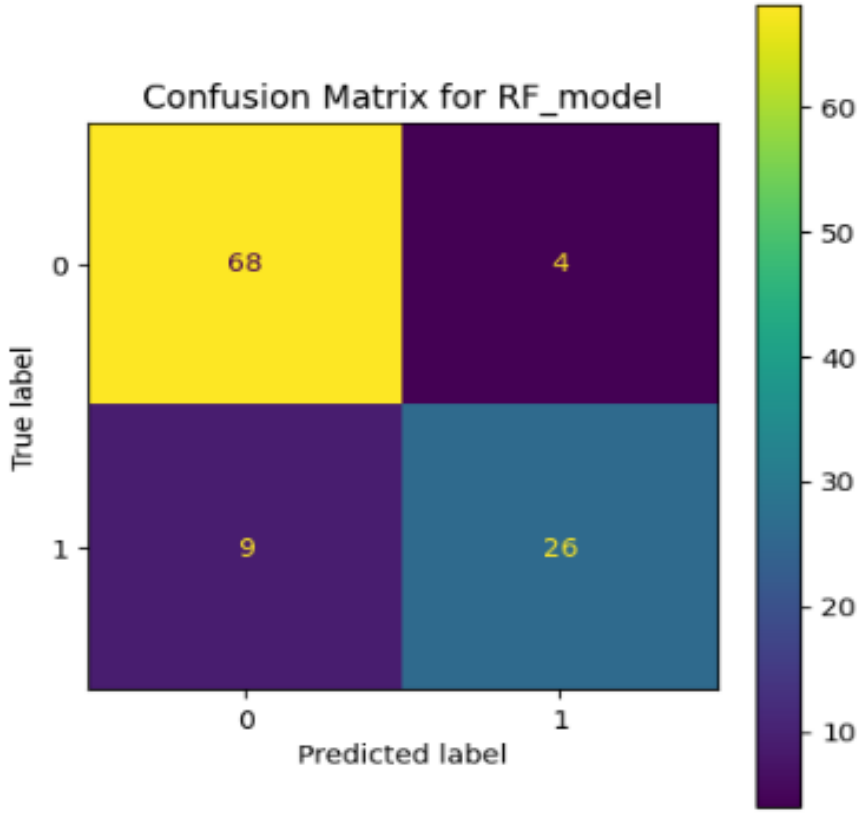
Confusion Matrix :



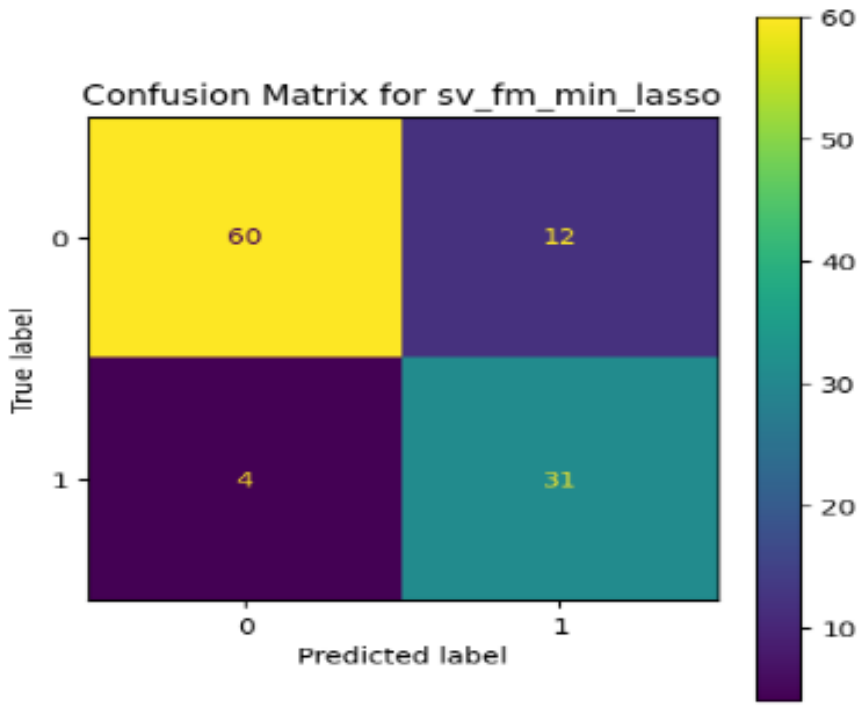
Confusion Matrix :



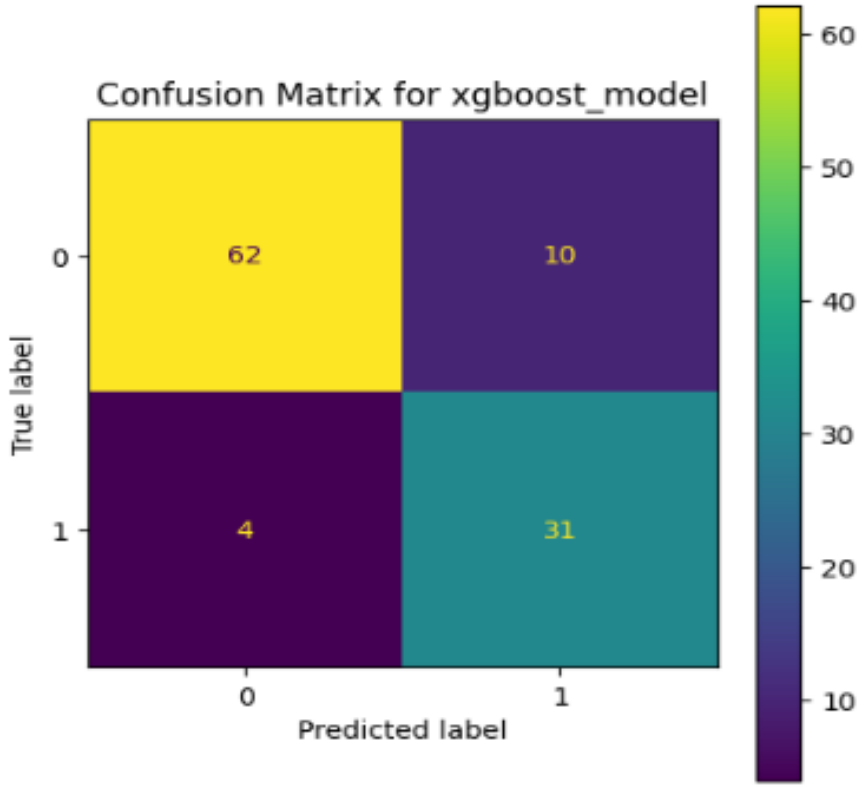
Confusion Matrix :



Confusion Matrix :



Confusion Matrix :



Evaluation Metrics Calculation :

```
#== 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[,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
```

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

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

Using 0.5 as Cutoff :

```
#==== 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]) #su_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]) #su_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
```

Observations :

1. Using "Optimum" Thresholds 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.
2. Using 0.5 as threshold 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:

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 $\eta_i's$ on the basis of $\hat{\beta}_{LASSO.min}$.
- Step 2 : Now, we will calculate $\hat{\pi}_i$ using $plogis(\eta_i)$.
- 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.

Simulation Output

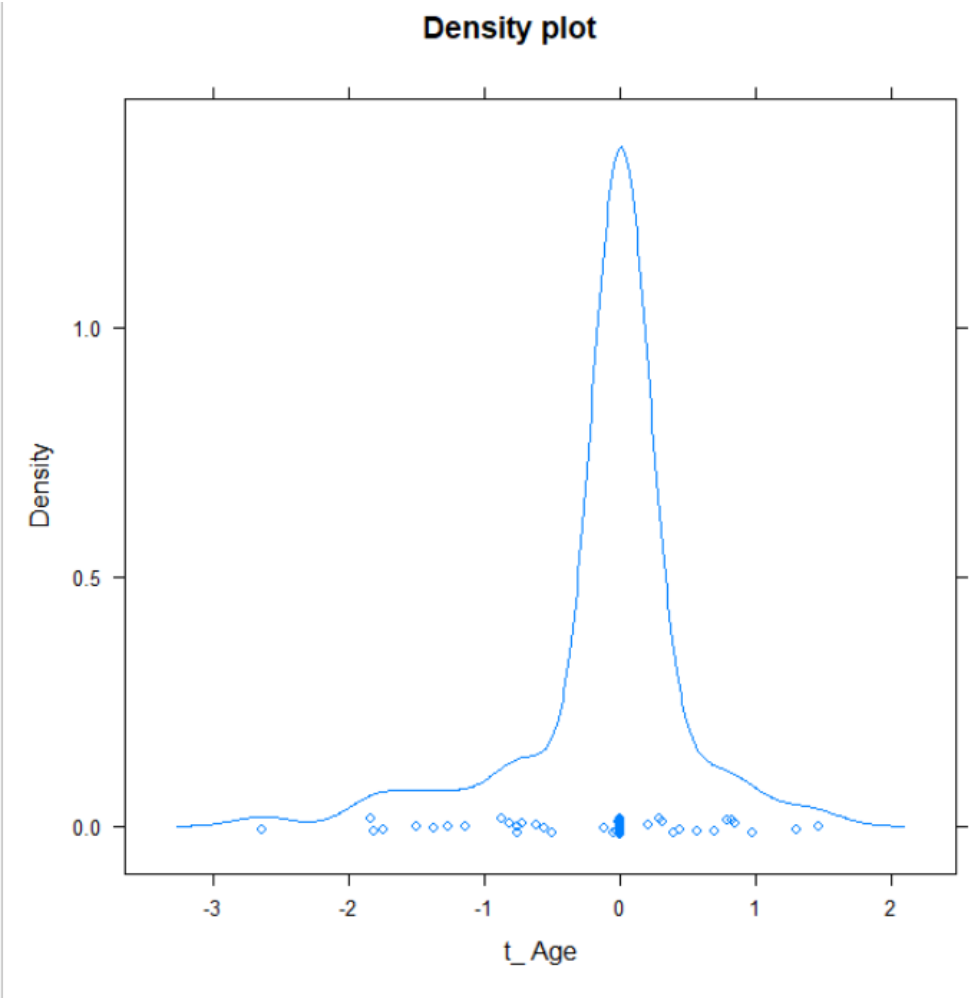
	dec_Level	X2.50.	X97.50.	z_value	variable
1	0.01	-1.7818821	0.9071952	0.1623741	Age
2	0.01	-1.1049443	0.6606060	1.2349724	Weight
3	0.05	-1.8761988	1.3310978	-0.7801271	Blood_Group12
4	0.08	-2.0043019	3.2155131	2.2058835	Blood_Group14
5	0.05	0.0000000	3.0184336	1.6041382	Pulse_rate
6	0.09	-1.0980998	3.6151661	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

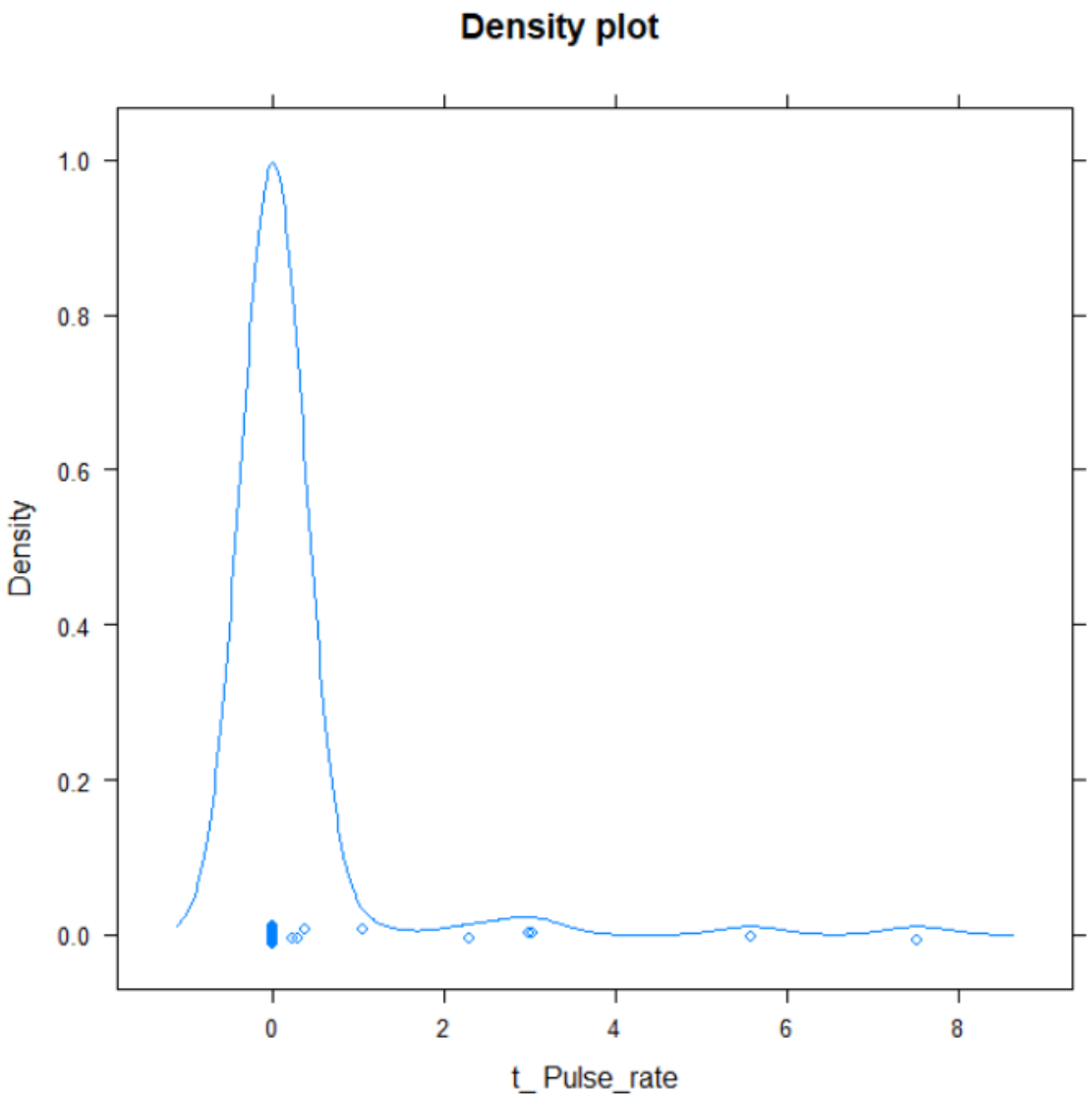
Significant Variables

Using the simulated Cutoffs, we get the following variables as 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"
```

Density plot of z statistics




Density plot
Density
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8
t_ Pulse_rate

Significant Predictors :

- Significant Predictors and corresponding estimated coefficients.

```
##                       [,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
```

Interpretation :

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 -

```
coef(sv_fm.1se.lasso)
```

```
##    (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
```

Acknowledgment:

- I would like to offer my heartiest gratitude to **Dr. Deepayan Sarkar** for his constant support and guidance throughout my project work.
- Also, I want to express my thanks to my parents for their constant encouragement and my friends for sharing their insightful ideas that kept me motivated in my project work.

References :

1. [https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/||Random Forest]
2. [https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm||Support Vector Machine]
3. [https://machinelearningmastery.com/extreme-gradient-boosting-ensemble-in-python/||Xg Boost]
4. [https://www.researchgate.net/publication/228906268_An_introduction_to_robust_estimation_with_R_functions||Robus t GLM]
5. [https://towardsdatascience.com/build-better-regression-models-with-lasso-271ce0f22bd||Lasso Regresion]