PIMA INDIAN DIABETES

(Based on Logistic Regression)

REPORT OF PBL FOR

R PROGRAMMING

BACHELOR OF TECHNOLOGY

COMPUTER SCIENCE AND ENGINEERING

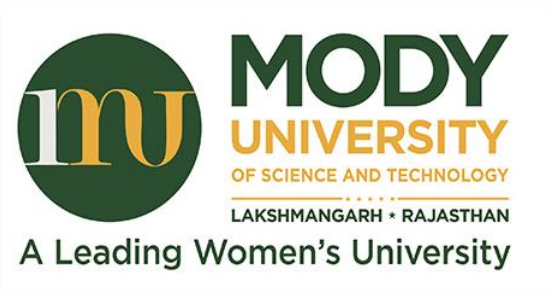
SUBMITTED BY

RIYA SINHA [170385]

UNDER THE SUPERVISION OF

DR. SS VERMA

DEPARTMENT OF {SET}



SCHOOL OF ENGINEERING AND TECHNOLOGY

MODY UNIVERSITY OF SCIENCE AND TECHNOLOGY

LAKSHMANGARH

MAY, 2019

School of Engineering and Technology

Mody University of Science and Technology,

Lakshmangarh

CERTIFICATE

This is to certify that the work contained in this report titled **PIMA INDIAN DIABETES** using data analysis is a bonafied work of **Ms. Riya Sinha** has been carried out under my supervision.

Dr . SS VERMA

Assistant Professor, CET

ACKNOWLEDGEMENT

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many of the individuals. I would like to extend our sincere gratitude to all of them.

I'm highly obliged to Dr. V.K. Jain, Dean CET, Mody University of science and technology, for the amenities provided to accomplish this main project.

I would like to thank our Head of Department, Dr. A. Senthil for his constructive criticism all the way through our project.

I'm indebted to Dr. SS Verma for his guidance and constant supervision as well as providing necessary information regarding the project and also for his support in finishing this project.

I would like to express my gratefulness towards my parents and members of Mody University for their kind co-operation and encouragement which helped me in completion of this project.

My thanks and appreciation also goes to all those people who have keenly helped me with their abilities and guidance.

ABSTRACT

This report is on "**PIMA INDIAN DIABETES**" is based on the concept of analysing data by using the concept of R programming. We have demonstrated this by plotting the graph using supervised learning problem which we need to make predictions on whether a person is to suffer the diabetes given the features in the dataset.

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Introduction to R Programming

R is a programming language and software environment for the statistical analysis, reporting and graphical representation. R was created by Robert Gentlemen and Ross Ihaka at the University of Auckland, New Zealand and is affiliated by R development core team. R is freely available under the general public licence and pre-compiled binary versions are provided for operating systems like Linux, Windows, Mac, etc. We have used R programming to extract information from large dataset of Pima group of women of Native America.

1.2 Introduction about the Project

This is a supervised learning problem which we need to make predictions on whether a person is to suffer the diabetes given the 8 features in the dataset.

We can learn following important information from the Data Set Description :

- All patients (768 Observations) in this dataset contains are females at least 21 years old of Pima Indian heritage.
- All zero values for the biological variables other than number of times pregnant should be treated as missing values.

Supervised Learning Method

Logistic Regression

Key Assumption for Logistic Regression: The independent variables should be independent from each other.

Model Assessment

We have analysed the models through the model accuracy (1 - classification error rate).

Models are trained on the 80% of data (614 Observations), and test the better model on 20% data (154 Observations).

Content

There are six main parts to my script as follows:

- Data Pre processing
- Missing Data Imputation
- Feature Engineering
- Build Models
- Comparison and Conclusion

- Model Investigation and Improvements

The Dataset

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

This dataset comprises data from **768** women with **8** physiognomies, in specific:

1. How many times she is pregnant
2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test
3. Diastolic blood pressure (mm Hg)
4. Triceps skin fold thickness (mm)
5. 2-Hour serum insulin (mu U/ml)
6. Body mass index (weight in kg/(height in m)^2)
7. Diabetes pedigree function
8. Age (in years)

CHAPTER 2

PACKAGES USED IN OUR DATASET

2.1 mice

The mice package implements a method to deal with the missing data. This package creates multiple auxiliary values for multivariate missing data.

2.2 Random Forest

random Forest implements Breiman's random forest algorithm for the classification and regression. It can also be used in unsupervised mode for assessing closeness among data points.

2.3 ggplot2

ggplot2 is a system for declaratively creating graphics, based on The Grammar of Graphics. It provides us deep philosophy of visualizations.

2.4 glmnet

Glmnet is a package that fits a general linear model via fined maximum likelihood. The regularization path is computed for the lasso or elastic net penalty at a grid of values for the regularization parameter lambda.

2.5 readr

The objective of readr is to offer a fast and friendly way to read rectangular data (like csv, fwf, and tsv). It is aimed to flexibly analyse many types of data found in the wild, while still cleanly failing when data unexpectedly changes

CHAPTER 3

3. DATA PREPROCESSING

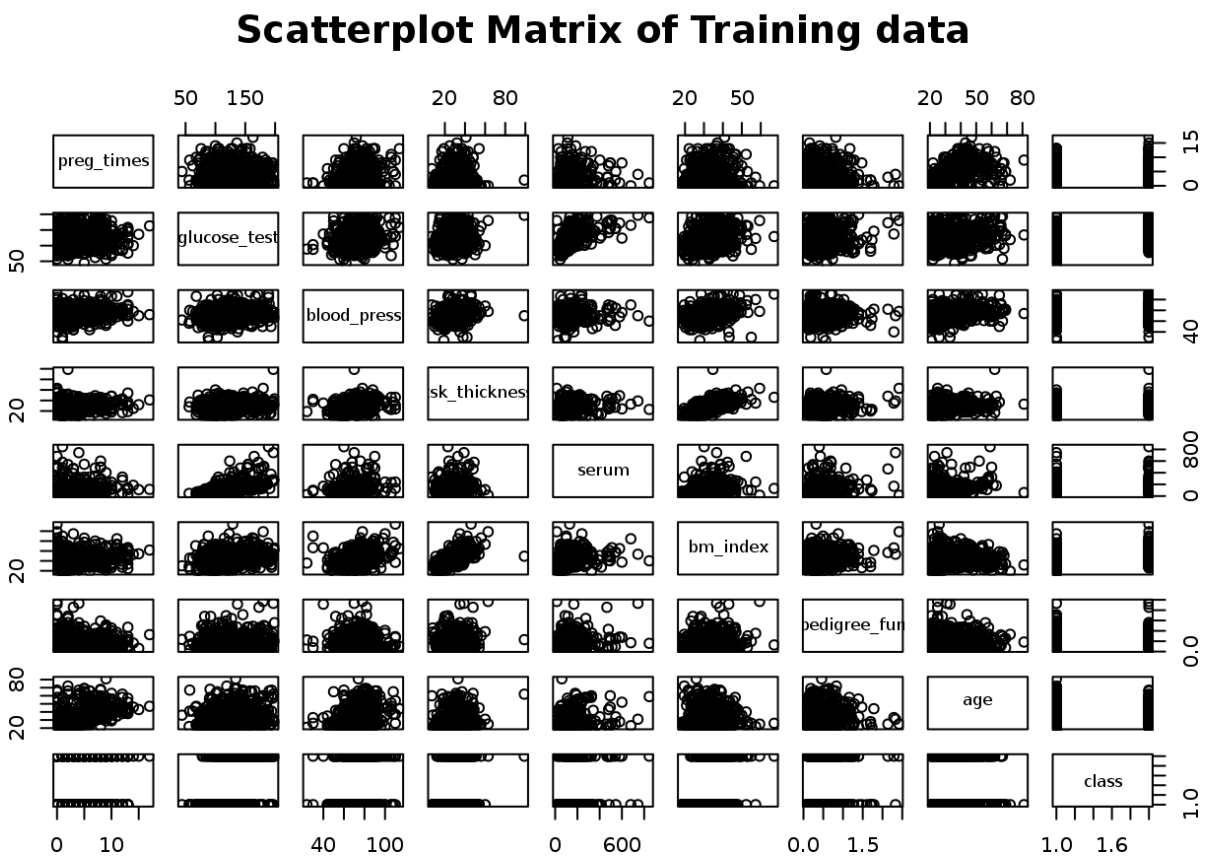
3.1 Import Data and Check Missing Values

```
library(mice)

library(randomForest)

library(ggplot2)

library(glmnet)

library(readr) # CSV file I/O, e.g. the read_csv function

# Inmport Pima Indians Diabetes Database from UCI Machine Learning Repository

#link <- "http://archive.ics.uci.edu/ml/machine-learning-databases/pima-indians-diabe
tes/pima-indians-diabetes.data"

#dataset <- read.table(link, sep = ",", strip.white=TRUE, fill = F)

dataset = read.csv("../input/diabetes.csv")


colnames(dataset) <- c("preg_times", "glucose_test", "blood_press", "tsk_thickness",

                       "serum", "bm_index", "pedigree_fun", "age", "class")

dataset$class <- as.factor(dataset$class)


# Check if there has NA in the dataset

print(all(!is.na(dataset)))

## [1] TRUE
```

3.2 Analyze the Dataset

```
# Set a random seed

set.seed(123)

# Split the dataset: 80% for trainging and 20% for testing

# trainging: dataset[train,]

# testing: dataset[-(train),]

train <- sample(nrow(dataset), round(0.8*nrow(dataset)), replace = FALSE)
```

```
# Show scatterplot matrix on the training data

pairs(~.,data=dataset[train,], main="Scatterplot Matrix of Training data")
```



The scatterplot matrix above shows:

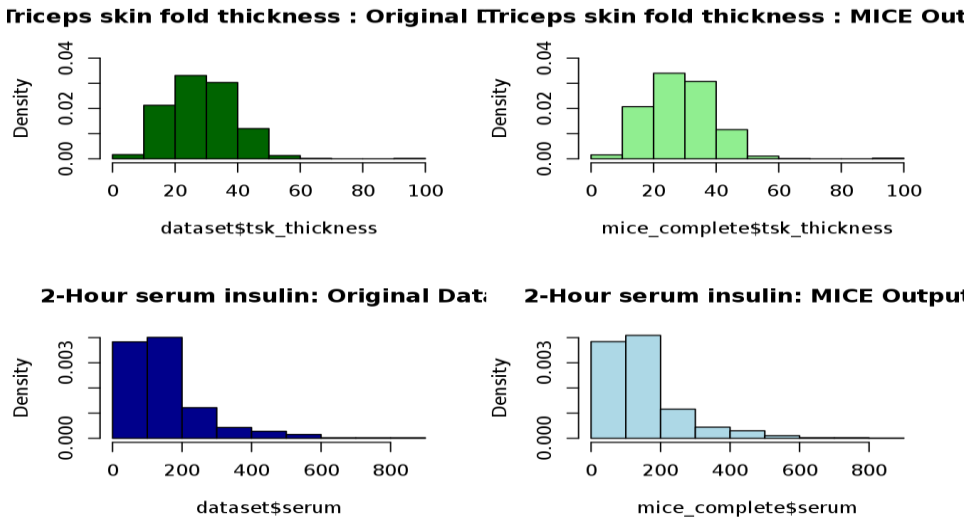
- No obvious high correlation between independent variables.
- No obvious relationship between diastolic blood pressure and diabetes.
- No obvious relationship between age and diabetes.

4. MISSING DATA IMPUTATION

Because of the small size of the dataset, I want to obtain as much as information from it, so I will not delete either entire observations (rows) or variables (columns) containing missing values right now. Two options:

- Replacing missing data with sensible values (mean or median) given the distribution of the data.
- Replacing missing data with prediction (Mutiple Imputation).

```
mice_complete <- complete(mice_mod)

# Show distributions for tsk_thickness and serum

par(mfrow=c(2,2))

hist(dataset$tsk_thickness, freq=F, main='Triceps skin fold thickness : Original Data
',

     col='darkgreen', ylim=c(0,0.04))

hist(mice_complete$tsk_thickness, freq=F, main='Triceps skin fold thickness : MICE Ou
tput',

     col='lightgreen', ylim=c(0,0.04))

hist(dataset$serum, freq=F, main='2-Hour serum insulin: Original Data',

     col='darkblue', ylim=c(0,0.004))

hist(mice_complete$serum, freq=F, main='2-Hour serum insulin: MICE Output',

 col='lightblue', ylim=c(0,0.004))
```

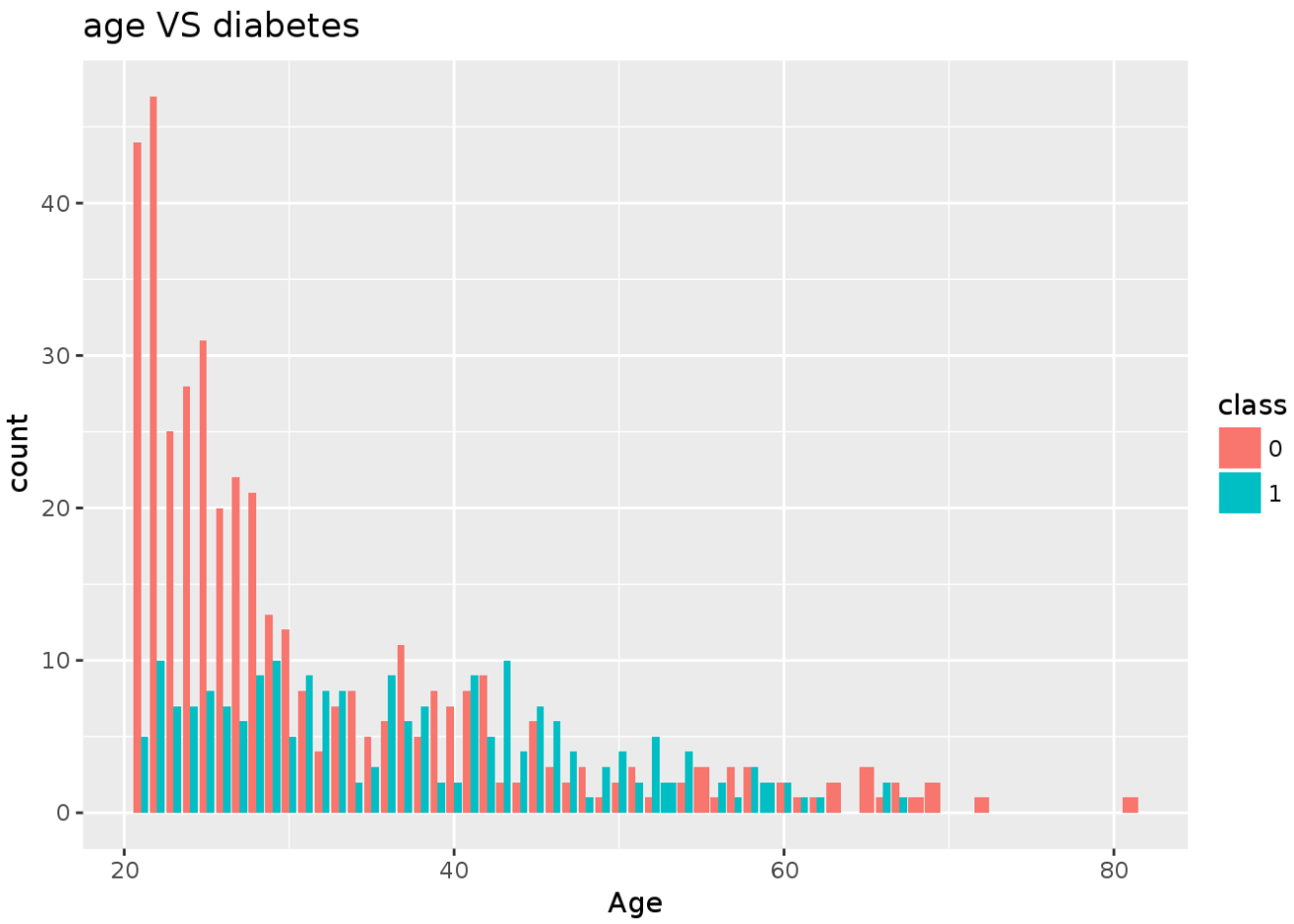


CHAPTER 5

5. FEATURE ENGINEERING

5.1 Try to Create New Variable

```
# Visualize the relationship between age and diabetes on training data

 ggplot(data=dataset[train,], aes(x = age, fill = class)) +

   geom_bar(stat='count', position='dodge') +

   ggtitle("age VS diabetes") +

   labs(x = 'Age')
```



Clearly, we can see that there's a age penalty to diabetes on the age large than 30. Initially, I try to coll apse this variable into two levels which probably will provide more insights. But, after testing the meth od on the training data, the result is even worse than before, so right now, I stop here.

5.2 Feature Selection

1. For random forest, it will perform feature selection when we apply the algorithm, because the Gini Impurity method will only choose the variables have significant impact to the result.
2. For logistic regression, we fit a model via penalized maximum likelihood (regularization), which will also do feature selection for us

CHAPTER 6

6. BUILD MODELS

6.1 Normalize the Training Data

```
# Normalize training data
scale_training <- as.data.frame(scale(dataset[train, -9],
                                      center = TRUE, scale = TRUE))
scale_training$class <- dataset[train, "class"]
str(scale_training)
## 'data.frame':    614 obs. of  9 variables:
##  $ preg_times   : num  -1.1502 0.0412 -0.2566 0.637 -0.8524 ...
```

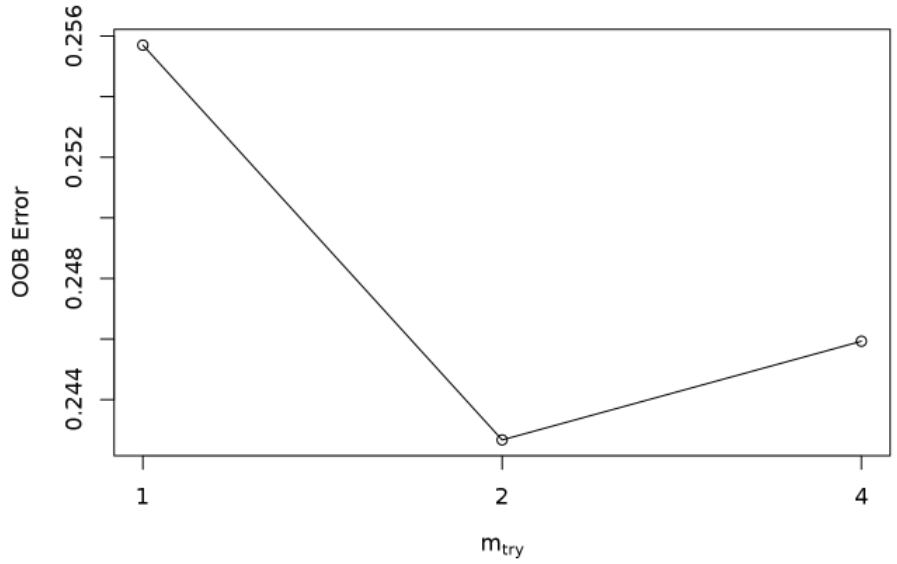
```
##  $ glucose_test : num  1.851 2.048 -0.252 2.443 -0.416 ...
##  $ blood_press  : num  -1.014 -0.011 -1.85 -0.178 -1.014 ...
##  $ tsk_thickness: num  -0.0405 -0.1352 -1.8392 0.6221 1.5688 ...
##  $ serum        : num  2.784 0.453 -0.571 -0.443 0.223 ...
##  $ bm_index     : num  0.277 -0.598 -0.442 -0.245 0.405 ...
##  $ pedigree_fun : num  1.791 -0.782 0.457 -0.435 -0.175 ...
##  $ age          : num  -1.04 0.248 -0.697 -0.182 -0.783 ...
##  $ class        : Factor w/ 2 levels "0","1": 2 2 1 2 1 1 2 1 2 1 ...
```

6.2 Random Forest

6.2.1 Find the Optimal Subset for Random Forest

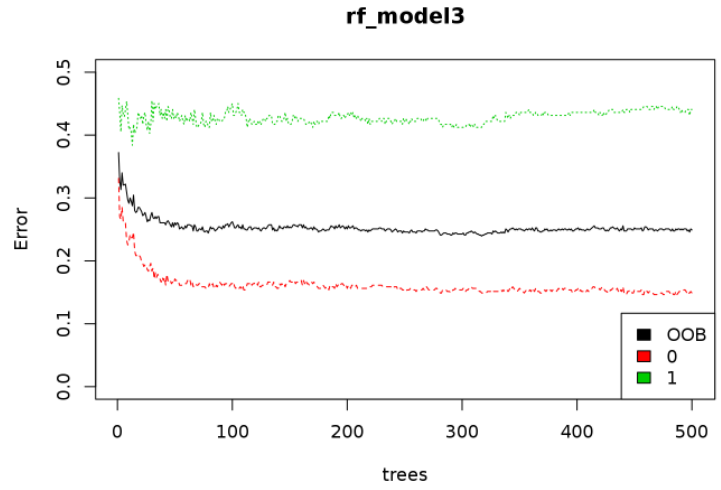
```
bestmtry <- tuneRF(scale_training[, c(-9)],scale_training$class, ntreeTry=300,
                   stepFactor=2,improve=0.05, trace=TRUE, plot=TRUE, dobest=FALSE)
```

```
## mtry = 2  OOB error = 24.27%
## Searching left ...
## mtry = 1     OOB error = 25.57%
## -0.05369128 0.05
## Searching right ...
## mtry = 4     OOB error = 24.59%
## -0.01342282 0.05
```



The result chooses 2 as the optimal number of mtry for each tree. Thus, we use mtry = 2 in the following section.

```
rf_model3 <- randomForest(class ~ ., data = scale_training, ntree=500, mtry=2)

# Output classfiaction error rate

plot(rf_model3,ylim = c(0, 0.5))

legend('bottomright', colnames(rf_model3$err.rate), col=1:3, fill=1:3)
```

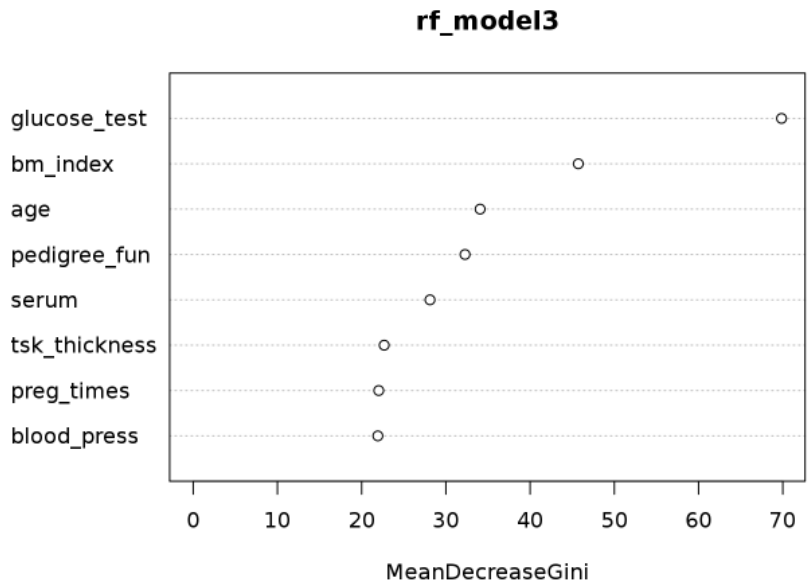



```
print(rf_model3$err.rate[nrow(rf_model3$err.rate),])

##       OOB         0         1
```

```
## 0.2491857 0.1488834 0.4407583
```

```
# Output variables inportance graph
```

```
varImpPlot(rf_model3)
```



As shown in the Variable Inportance graph above, random forest uses all the features in its algorithm. And the Plasma glucose concentration and Body mass index are sigificant for identifying the diabetes.

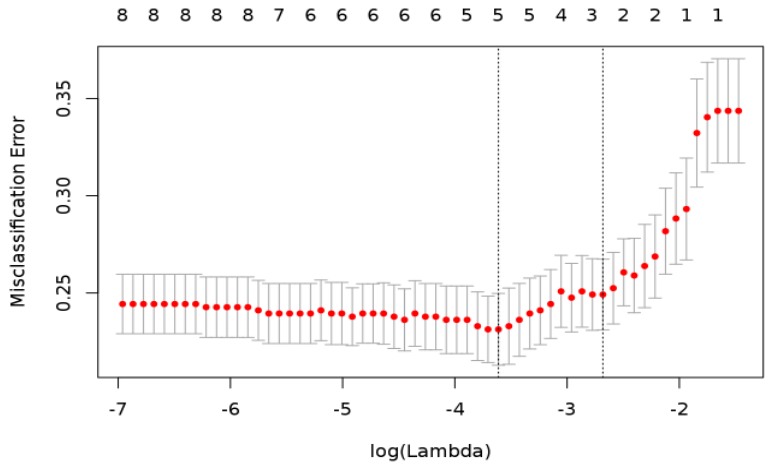
6.3 Logistic Regression

Using 10 folds cross-validation to find the best Logistic Regression while avoiding the overfitting by regularization.

```
cvfit = cv.glmnet(as.matrix(scale_training[, c(-9)]), scale_training$class,
                  family = "binomial", type.measure = "class")
```

```
# Show the trend of using different value of the penalized parameter (lambda)
```

```
plot(cvfit)
```




```
# Show the cefficients of the best model

coef(cvfit, s = "lambda.min")

## 9 x 1 sparse Matrix of class "dgCMatrix"

##                          1

## (Intercept)   -0.79454616

## preg_times     0.23513025

## glucose_test   0.89863688

## blood_press    .

## tsk_thickness  .

## serum          .

## bm_index       0.42152241

## pedigree_fun   0.09236745

## age            0.04708040
```

As shown in the coefficient table above, the best logical regression only uses 5 the features in its algorithm after regularization. Also, it shows Plasma glucose concentration and Body mass index have significant impact on identifying the diabetes.

```
lg_p = predict(cvfit, newx = as.matrix(scale_training[, c(-9)]),

            s = "lambda.min", type = "class")


# Show confusion matrix
```

```
(lg_result <- table(lg_p, scale_training$class))
##
## lg_p   0   1
##    0 363 100
##    1  40 111
# Overall error rate and accuracy
overall_accuracy <- (lg_result[1] +  lg_result[4]) / sum(lg_result)
overall_error <- 1 - overall_accuracy
# Error rate in class 0
error_c0 <- lg_result[2] / (lg_result[1] +  lg_result[2])
# Error rate in class 1
error_c1 <- lg_result[3] / (lg_result[3] +  lg_result[4])
```

7. COMPARISON AND CONCLUSION

7.1 Model Comparison

Accuracy for logistic regression:

```
overall_accuracy
```

```
## [1] 0.771987
```

Accuracy for random forest:

```
1 - rf_model3$err.rate[nrow(rf_model3$err.rate), 1]
```

```
##       OOB
## 0.7508143
```

The results above suggests that logistic regression performed better than random forest.

7.2 Test Error Rate for Logistic Regression

```
# Normalize the testing dataset
scale_testing <- as.data.frame(scale(dataset[-(train), -9],
                                     center = TRUE, scale = TRUE))
scale_testing$class <- dataset[-(train), "class"]
# Test error for LG
lg_test = predict(cvfit, newx = as.matrix(scale_testing[, c(-9)]),
                  s = "lambda.min", type = "class")
# Show confusion matrix
(lg_result <- table(lg_test, scale_testing$class))
```

```
##
## lg_test  0  1
##       0 88 27
##       1  9 30
```

```
# Obtain test error and accuracy
test_accuracy <- (lg_result[1] +  lg_result[4]) / sum(lg_result)
```

```
test_error <- 1 - overall_accuracy


# Error rate in class 0

(test_c0 <- lg_result[2] / (lg_result[1] +  lg_result[2]))

## [1] 0.09278351

# Error rate in class 1

(test_c1 <- lg_result[3] / (lg_result[3] +  lg_result[4]))

## [1] 0.4736842
```

The results below show the test error and accuracy for logistic regression.

```
print(paste("overall test error: ", overall_error))

## [1] "overall test error:  0.228013029315961"

print(paste("overall accuracy: ", overall_accuracy))

## [1] "overall accuracy:  0.771986970684039"
```

The test error rate is not significant different than the estimate test error, which is obtained from the 10 folds cross-validation. So, we are confident about the logistic regression model is not over fitting.

Reference

i. R documentation
ii. www.quora.in
iii. https://www.techopedia.com
iv .https://www.geeksforgeeks.org
v. https://www.kaggle.com