UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

COURSE NAME	**Master of Science in Business Analytics**
MODULE NAME	**Statistics for Data Analytics(MA6101)**
BATCH	**2019-2020**
SEMESTER	**Autumn**

Submitted by

Name	Student ID
Kumar Ankitesh	19059426
Fasahat Khan	19121474
Aishwarya Inamdar	19084404

MA6101: Statistics for Data Analytics Project

Problem statement:

"PimaIndiansDiabetes" (Pima Indians Diabetes Database) is a data set with 768 observations and 9 variables. Below is a description of this data set:

A population of women who were at least 21 years old, of Pima Indian heritage and living near Phoenix, Arizona, was tested for diabetes according to World Health Organization criteria. The data were collected by the US National Institute of Diabetes and Digestive and Kidney Diseases. We used the 532 complete records after dropping the (mainly missing) data on serum insulin. The structure of this data set is explained in the following table:
Format:

pregnant	Number of times pregnant
glucose	Plasma glucose concentration (glucose tolerance test)
pressure	Diastolic blood pressure (mm Hg)
triceps	Triceps skin fold thickness (mm)
insulin	2-Hour serum insulin (mu U/ml)
Mass	Body mass index (weight in kg/(height in m)\^2)
pedigree	Diabetes pedigree function
age	Age (years)
diabetes	Class variable (test for diabetes)

You will first need to install the package called "mlbench" in order to access this data set. The package should be installed, read-into R, and then the data set should be stored by the name "mydata".

1. Write the code that is required to install and read the package and to store the data set by the name “mydata”.

#Load package mlbench

#installing package “mlbench”

install.packages("mlbench")

Library(package) loads the namespace of the package with name “mlbench” and attaches it on the search list.

library(mlbench)

To load PimaIndiansDiabetes dataset

data(PimaIndiansDiabetes)

Summary represents the summary statistics of all the variables of PimaIndiansDiabetes the mean, median, max value, min value,1st quartile & 3rd quartile.

summary(PimaIndiansDiabetes)

```
> summary(PimaIndiansDiabetes)
    pregnant         glucose         pressure         triceps         insulin           mass           pedigree     
 Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00   Min.   :  0.0   Min.   : 0.00   Min.   :0.0780  
 1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00   1st Qu.:  0.0   1st Qu.:27.30   1st Qu.:0.2437  
 Median : 3.000   Median :117.0   Median : 72.00   Median :23.00   Median : 30.5   Median :32.00   Median :0.3725  
 Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54   Mean   : 79.8   Mean   :31.99   Mean   :0.4719  
 3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00   3rd Qu.:127.2   3rd Qu.:36.60   3rd Qu.:0.6262  
 Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00   Max.   :846.0   Max.   :67.10   Max.   :2.4200  
      age        diabetes  
 Min.   :21.00   neg:500  
 1st Qu.:24.00   pos:268  
 Median :29.00            
 Mean   :33.24            
 3rd Qu.:41.00            
 Max.   :81.00            
```

#Complete data frame with non-zero numbers of rows and columns can be viewed using the view() function.

view(PimaIndiansDiabetes)

Filter

	pregnant	glucose	pressure	triceps	insulin	mass	pedigree	age	diabetes
1	6	148	72	35	0	33.6	0.627	50	pos
2	1	85	66	29	0	26.6	0.351	31	neg
3	8	183	64	0	0	23.3	0.672	32	pos
4	1	89	66	23	94	28.1	0.167	21	neg
5	0	137	40	35	168	43.1	2.288	33	pos
6	5	116	74	0	0	25.6	0.201	30	neg
7	3	78	50	32	88	31.0	0.248	26	pos
8	10	115	0	0	0	35.3	0.134	29	neg
9	2	197	70	45	543	30.5	0.158	53	pos

#Dataset stored in mydata

The dataset in the variable PimaIndiansDiabetes is loaded in “mydata”

mydata <-PimaIndiansDiabetes

mydata 768 obs. of 9 variables

	pregnant	glucose	pressure	triceps	insulin	mass	pedigree	age	diabetes
1	6	148	72	35	0	33.6	0.627	50	pos
2	1	85	66	29	0	26.6	0.351	31	neg
3	8	183	64	0	0	23.3	0.672	32	pos
4	1	89	66	23	94	28.1	0.167	21	neg
5	0	137	40	35	168	43.1	2.288	33	pos
6	5	116	74	0	0	25.6	0.201	30	neg
7	3	78	50	32	88	31.0	0.248	26	pos
8	10	115	0	0	0	35.3	0.134	29	neg
9	2	197	70	45	543	30.5	0.158	53	pos

2. Present the variables descriptively, i.e., present mean (sd) or number (%) for each variable appropriately.

#Summary represents the summary statistics of all the variables of mydata the mean, median, max value, min value,1st quartile & 3rd quartile.

summary(mydata)

```
> summary(mydata)
    pregnant         glucose         pressure         triceps         insulin           mass      
 Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00   Min.   :  0.0   Min.   : 0.00  
 1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00   1st Qu.:  0.0   1st Qu.:27.30  
 Median : 3.000   Median :117.0   Median : 72.00   Median :23.00   Median : 30.5   Median :32.00  
 Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54   Mean   : 79.8   Mean   :31.99  
 3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00   3rd Qu.:127.2   3rd Qu.:36.60  
 Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00   Max.   :846.0   Max.   :67.10  
    pedigree           age         diabetes  
 Min.   :0.0780   Min.   :21.00   neg:500  
 1st Qu.:0.2437   1st Qu.:24.00   pos:268  
 Median :0.3725   Median :29.00            
 Mean   :0.4719   Mean   :33.24            
 3rd Qu.:0.6262   3rd Qu.:41.00            
 Max.   :2.4200   Max.   :81.00            
```

str() represents the respective datatypes of all the variables of the argument dataset.

str(mydata)

```
> str(mydata)
'data.frame':   768 obs. of  9 variables:
 $ pregnant: int  6 1 8 1 0 5 3 10 2 8 ...
 $ glucose : int  148 85 183 89 137 116 78 115 197 125 ...
 $ pressure: int  72 66 64 66 40 74 50 0 70 96 ...
 $ triceps : int  35 29 0 23 35 0 32 0 45 0 ...
 $ insulin : int  0 0 0 94 168 0 88 0 543 0 ...
 $ mass    : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
 $ pedigree: num  0.627 0.351 0.672 0.167 2.288 ...
 $ age     : int  50 31 32 21 33 30 26 29 53 54 ...
 $ diabetes: Factor w/ 2 levels "neg","pos": 2 1 2 1 2 1 2 1 2 2 ...
```

Dim(mydata) function provides the dimensions of the data frame ‘mydata’

Dim(mydata)

```
> dim(mydata)
[1] 768   9
```

split(mydata, mydata$diabetes) splits the ‘mydata’ data frame by the factor diabetes into positive and negative and loads it in ‘diabetic’ variable.

Diabatic <- split(mydata, mydata$diabetes)

diabatic	list [2]	List of length 2
1	list [500 x 9] (S3: data.frame)	A data.frame with 500 rows and 9 columns
2	list [268 x 9] (S3: data.frame)	A data.frame with 268 rows and 9 columns

sd(diabaticposglucose) and sd(diabaticnegglucose) compute the standard deviations for diabetic people with positive and negative glucose and load the values in sd_pos and sd_neg variables respectively.

sd_pos <- sd(diabaticposglucose)

sd_neg <- sd(diabaticnegglucose)

sd_neg	26.1411997553536
sd_pos	31.9396220580072

Printing the data in sd_pos and sd_neg

sd_pos

sd_neg

```
> sd_pos
[1] 31.93962
> sd_neg
[1] 26.1412
```

summary(diabaticposglucose) and summary(diabaticnegglucose) represent the summary statistics of all the variables of sd_pos and sd_neg repectively : the mean, median, max value, min value,1st quartile & 3rd quartile.

summary(diabaticposglucose)

summary(diabaticnegglucose)

```
> summary(diabatic$pos$glucose)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    0.0   119.0   140.0   141.3   167.0   199.0
> summary(diabatic$neg$glucose)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      0      93     107     110     125     197
```

Viewing the data in 'mydata'

view(mydata)

	pregnant	glucose	pressure	triceps	insulin	mass	pedigree	age	diabetes
1	6	148	72	35	0	33.6	0.627	50	pos
2	1	85	66	29	0	26.6	0.351	31	neg
3	8	183	64	0	0	23.3	0.672	32	pos
4	1	89	66	23	94	28.1	0.167	21	neg
5	0	137	40	35	168	43.1	2.288	33	pos
6	5	116	74	0	0	25.6	0.201	30	neg
7	3	78	50	32	88	31.0	0.248	26	pos
8	10	115	0	0	0	35.3	0.134	29	neg
9	2	197	70	45	543	30.5	0.158	53	pos

3. Compare the shape of the distribution of “glucose” in those with and without diabetes.

The density of glucose is significantly different for people who have diabetes(pos) and for people who do not have diabetes(neg). The values for people who have diabetes(pos) vary from 0.0 to 199.0 with a mean of 141.3, median of 140.0 and standard deviation of 31.093962. The values for people who do not have diabetes(neg) vary from 0 to 197 with a mean of 110, median of 107 and standard deviation of 26.1412. The same has been demonstrated on a ggplot and a histogram.

Exploratory Data Analysis

Attach

attach(mydata)

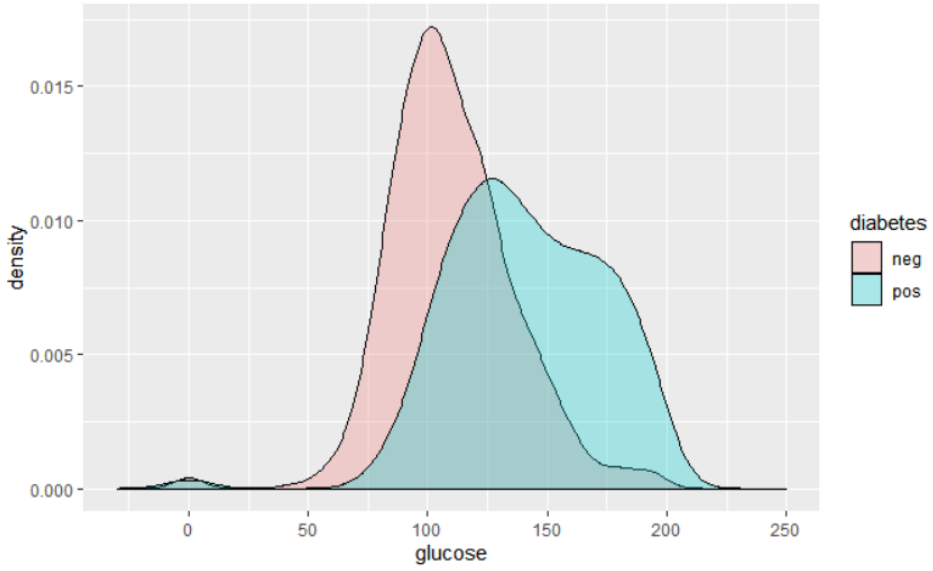
ggplot2 is a plotting package that makes it simple to create complex plots from data in a data frame providing a more programmatic interface for specifying what variables to plot, how they are displayed, and general visual properties.

```
install.packages(“ggplot2”)

library(ggplot2)
```

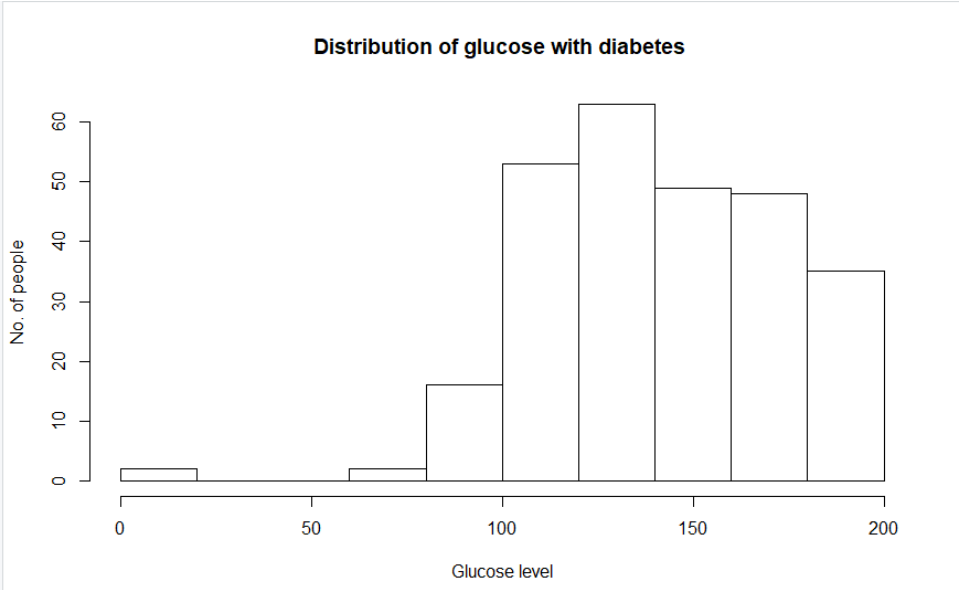
Graph to show distribution of glucose with diabetes represented by positive in blue and without diabetes represented by negative in pink.

```
ggplot(mydata, aes(x = glucose)) +
  geom_density(aes(fill = diabetes), alpha = 0.3) +
  scale_color_manual(values = c("#868686FF", "#EFC000FF")) +
  scale_fill_manual(values = c("lightcoral","darkturquoise")) + xlim(-30,250)
```

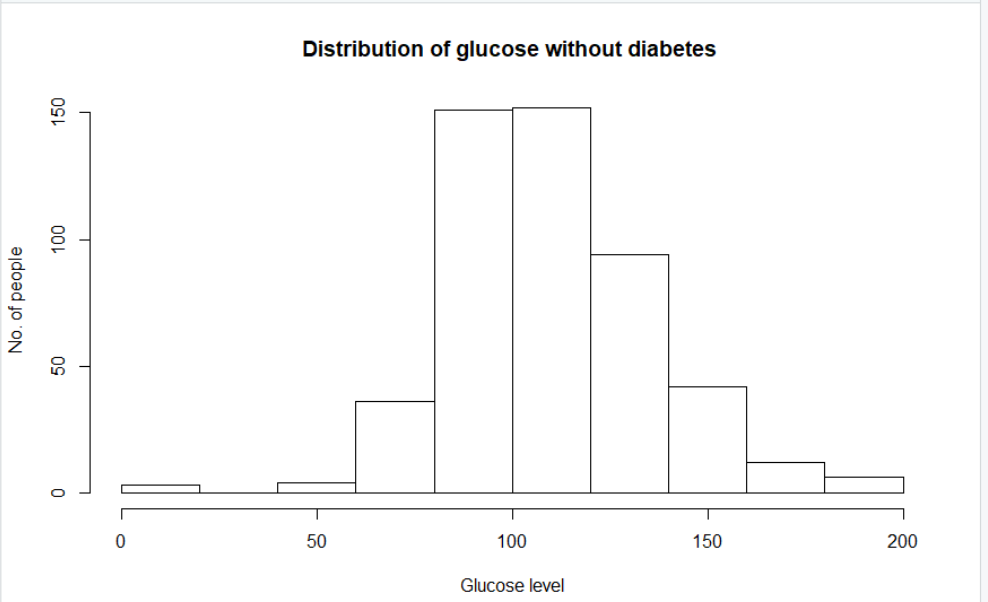


The hist() function is used to create a histogram. The histograms created in the code represent the Distribution of glucose with and without diabetes respectively.

```
hist(diabatic$pos$glucose, 10,main="Distribution of glucose with diabetes ",
    xlab="Glucose level",ylab="No. of people")
```




```
hist(diabatic$neg$glucose, 10,main="Distribution of glucose without diabetes ",
    xlab="Glucose level",ylab="No. of people")
```



4. Test whether there is a statistically significant difference in the mean glucose of those with and without diabetes.

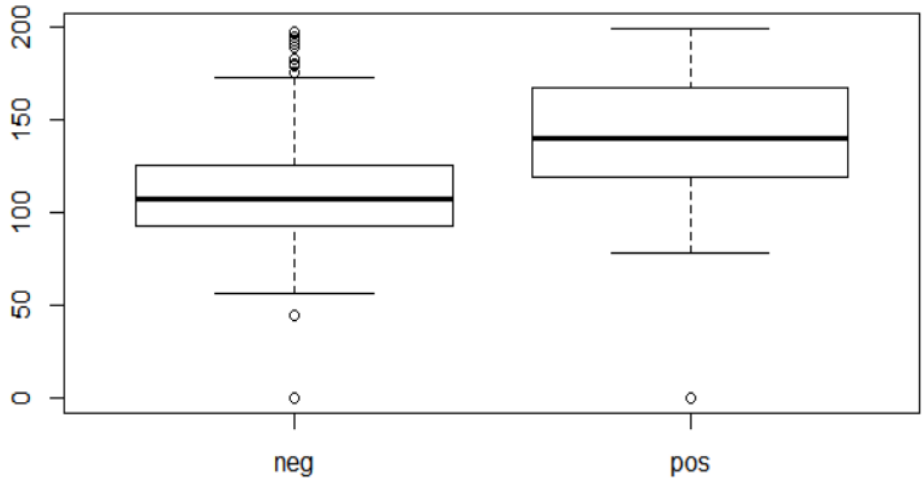
A statistically significant t-test result is one in which a difference between two groups (in this case positive mean which represents glucose with diabetes and negative mean which represents glucose without diabetes) is unlikely to have occurred because the sample happened to be atypical. Statistical significance suggests that the two larger populations from which we sample are "actually" different where in this case the positive mean is 141.2575 and the negative mean is 109.9800 which is significantly different.

```
> posmean<-diabatic$pos$glucose
> negmean<-diabatic$neg$glucose
> t.test(posmean, negmean)

        Welch Two Sample t-test

data:  posmean and negmean
t = 13.752, df = 461.33, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 26.80786 35.74707
sample estimates:
mean of x mean of y
 141.2575  109.9800
```

boxplot(glucose~diabetes)



From the box plot it is clear that in both cases, it is positively skewed however in case of glucose with diabetes the distribution is more skewed compared to the distribution in case of glucose without diabetes. The solid lines inside the boxes represent the means for both the cases and it is clearly visible that the positive mean is greater than the negative mean. Furthermore, the points lying outside the quartile ranges are the outliers.

5.We want to fit a logistic regression model to the data set, with "diabetes" as the binary dependent variable and "age", "mass", "glucose" and "pregnant" as predictors. Write a code to fit this model and clearly explain the output.

Model building:

5.1) Data Slicing:

For classification problem, it is important to ensure that the train and test sets have approximately the same percentage of samples of each target class. Hence, we will use stratified sampling.

```
> # Binary variable needs to be converted into factor variable
> mydata$diabetes = as.factor(mydata$diabetes)
> library(caret)
> set.seed(1234)
> trainIndex = createDataPartition(diabetes, p=0.7, list = FALSE, times = 1)
> train.data = mydata[trainIndex, ]
> test.data  = mydata[-trainIndex,]
> dim(train.data)
[1] 538   9
> dim(test.data)
[1] 230   9
```

prop.table(table(mydata$diabetes))

```
> prop.table(table(mydata$diabetes))

        1         2
0.6510417 0.3489583
```

prop.table(table(train.data$diabetes))

```
> prop.table(table(train.data$diabetes))

        1         2
0.6505576 0.3494424
```

prop.table(table(test.data$diabetes))

```
> prop.table(table(test.data$diabetes))

        1         2
0.6521739 0.3478261
```

5.2) Logistic Regression

```
> ### Model refining - Logistic Regression
> logit_model1  =  glm(diabetes ~ age+mass+glucose+pregnant,
+                     data = train.data,
+                     family = binomial(link="logit"))
>
> summary(logit_model1)

Call:
glm(formula = diabetes ~ age + mass + glucose + pregnant, family = binomial(link = "logit"),
    data = train.data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1772  -0.7472  -0.4485   0.7756   2.7961

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -8.30072    0.80768 -10.277  < 2e-16 ***
age          0.01290    0.01057   1.220   0.2223
mass         0.08513    0.01670   5.099 3.42e-07 ***
glucose      0.03255    0.00400   8.137 4.05e-16 ***
pregnant     0.09386    0.03723   2.521   0.0117 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 696.28  on 537  degrees of freedom
Residual deviance: 529.44  on 533  degrees of freedom
AIC: 539.44

Number of Fisher Scoring iterations: 5
```

The three significant variables highlighted in yellow are:

- Mass
- Glucose
- Pregnant

Also, the AIC[#] Score is 539.44. This should be ideally observed in subsequent stages when the model is refined. The model having least AIC Score would be the most preferred and optimized one.

5.3) Multi-collinearity(VIF)

```
> library(car)
> vif(logit_model1)
     age     mass  glucose pregnant
1.447698 1.016803 1.029065 1.409220
```

Solution of regression problem becomes unstable in presence of 2 or more correlated predictors. Multicollinearity can be measured by computing variance inflation factor (VIF) which gauges – how much the variance of regression coefficient is inflated due to multicollinearity.
Here the multicollinearity of age, mass, glucose and pregnant is nearly 1 for each, therefore we have considered them while building the logistics model. And these parameters can be further considered while refining the model.
As a thumb rule, VIF of more than 5 or 10 is considered significant. And such variables can be removed to improve stability of the regression model.

5.4) ROC plot:

The Receiver Operating Characteristic (ROC) plot measures the True Positive Rate against the False Positive Rate in a diagnostic test.

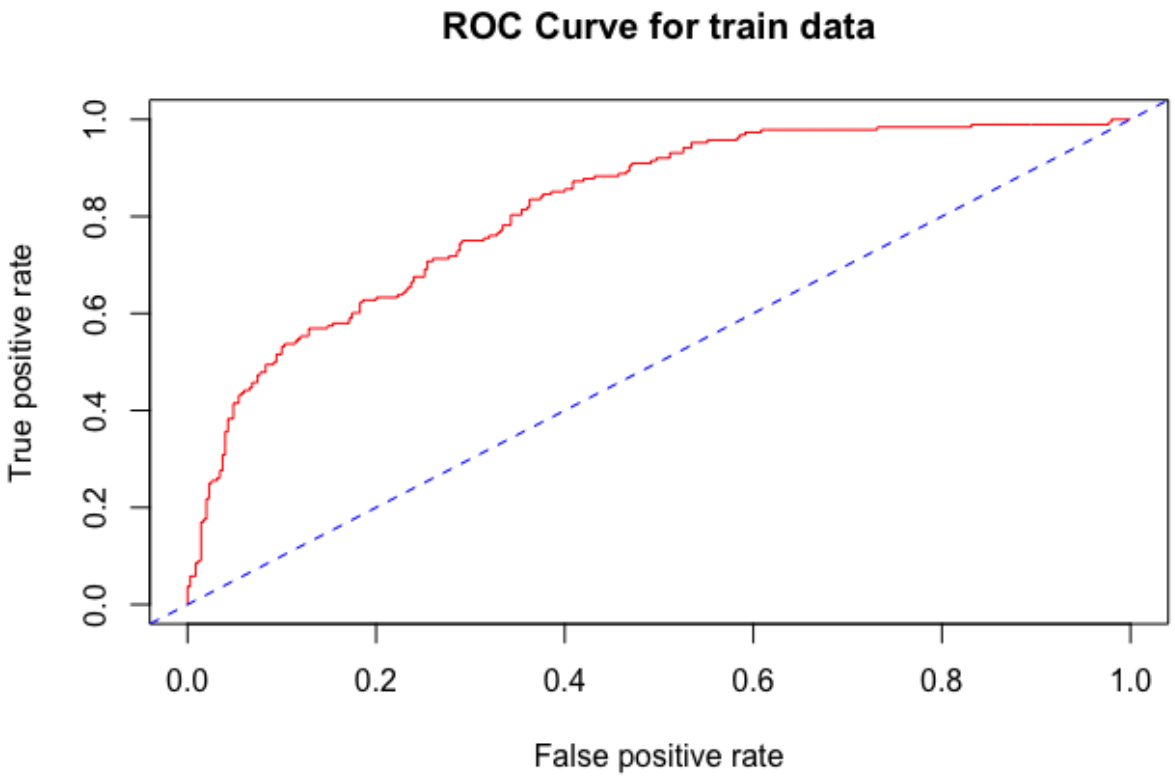
Accuracy of the model is measured by the area under the ROC curve. An area of 1 represents 100% accuracy; an area of 0.5 represents a test which should be not be accepted in general. A rough guide for classifying the accuracy of a model

verification test is the traditional academic point system, as follows:

0.90-1 = excellent (A) 0.80-0.90 = good (B) 0.70-0.80 = fair (C) 0.60-0.70 = poor (D) 0.50-0.60 = fail (F)

AUC At 0.8182827, the ROC Curve of our model demonstrates fairly good results.

AUC At 0.8182827, the ROC Curve of our model demonstrates fairly good results.



5.5) Area under the curve

AUC of a classifier is equal to the probability that the classifier will rank a randomly chosen positive data point higher than a randomly chosen negative data point. Higher the probability better is the classifier.

```
> # AUC
> train.auc = performance(train.roc, "auc")
> train.area = as.numeric(slot(train.auc, "y.values"))
> train.area
[1] 0.8182827
```