Capstone Project

Swati Jani Joshi

November 18 2016

Springboard Capstone Project

This is the Springboard Capstone Prject Code. The data set is from kaggle Prima Indians Diabetes Database found https://www.kaggle.com/uciml/pima-indians-diabetes-database

Read and Load Data

```r
library("ROCR")
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```r
library(ggplot2)
diabetes <- read.csv("~/Documents/Springboard Capstone Project/Data Files/diabetes.csv")
```

Examine the Structure of the Dataset

```r
str(diabetes)
```

```
## 'data.frame':    768 obs. of  9 variables:
##  $ Pregnancies             : int  6 1 8 1 0 5 3 10 2 8 ...
##  $ Glucose                 : int  148 85 183 89 137 116 78 115 197 125 ...
##  $ BloodPressure           : int  72 66 64 66 40 74 50 0 70 96 ...
##  $ SkinThickness           : int  35 29 0 23 35 0 32 0 45 0 ...
##  $ Insulin                 : int  0 0 0 94 168 0 88 0 543 0 ...
##  $ BMI                     : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
##  $ DiabetesPedigreeFunction: num  0.627 0.351 0.672 0.167 2.288 ...
##  $ Age                     : int  50 31 32 21 33 30 26 29 53 54 ...
##  $ Outcome                 : int  1 0 1 0 1 0 1 0 1 1 ...
```

Data Manipulation

Since it is not possible for a person to have Blood Pressure, Skin Thickness and Insulin of 0 those observations will be removed

```r
diabetes$Outcome <- factor(diabetes$Outcome)
for (i in 2:6) {
diabetes <- diabetes[-which(diabetes[,i] ==0), ]}
```

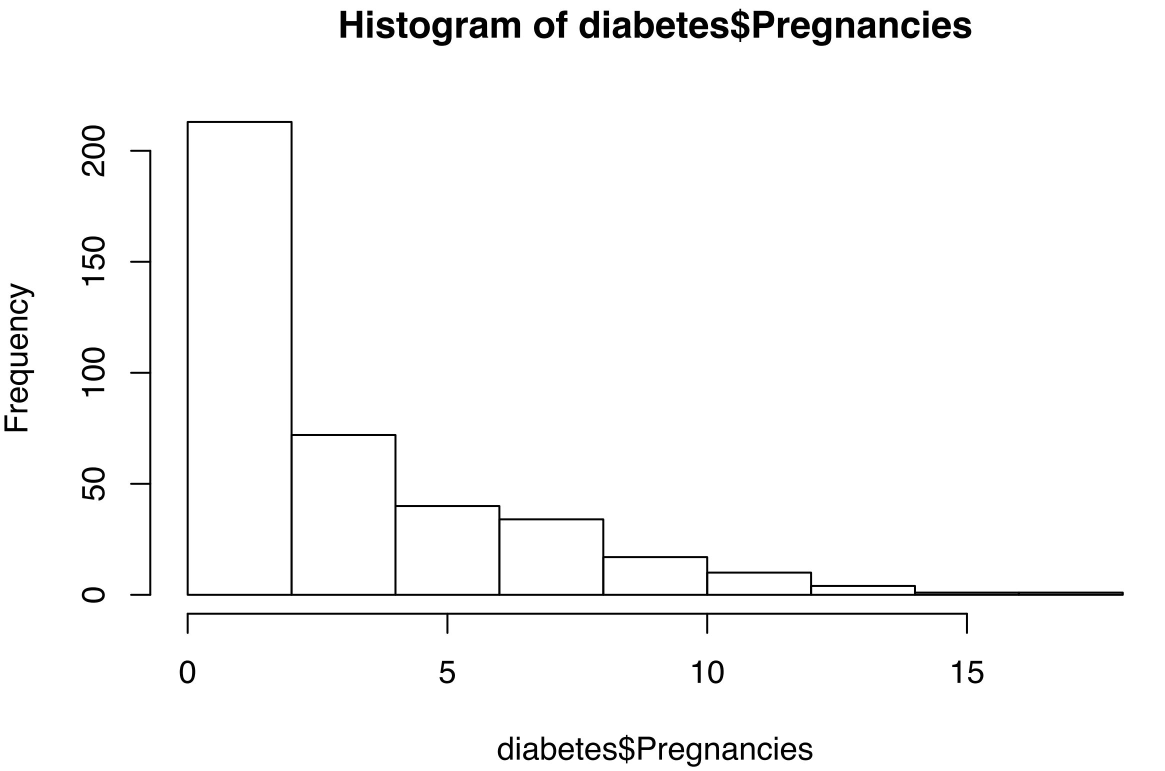
Examine the Structure of the Modified Dataset

```r
str(diabetes)
```

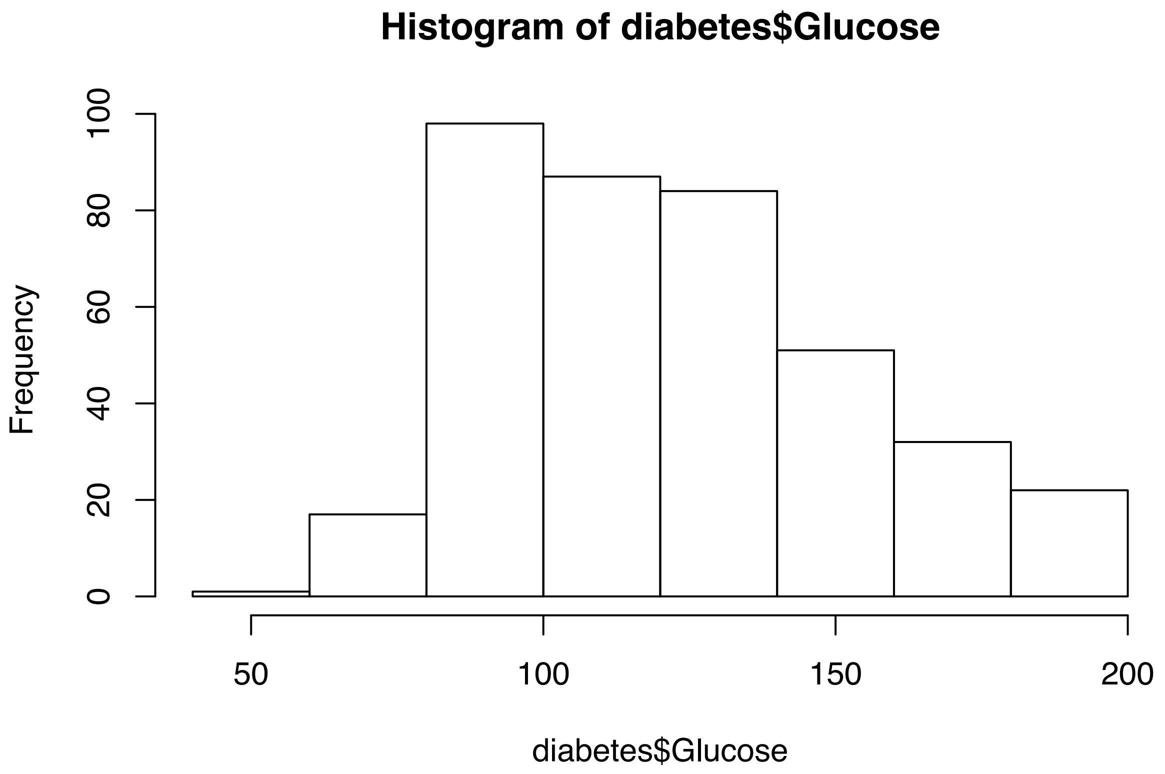
```
## 'data.frame':    392 obs. of  9 variables:
##  $ Pregnancies             : int  1 0 3 2 1 5 0 1 1 3 ...
##  $ Glucose                 : int  89 137 78 197 189 166 118 103 115 126 ...
##  $ BloodPressure           : int  66 40 50 70 60 72 84 30 70 88 ...
##  $ SkinThickness           : int  23 35 32 45 23 19 47 38 30 41 ...
##  $ Insulin                 : int  94 168 88 543 846 175 230 83 96 235 ...
##  $ BMI                     : num  28.1 43.1 31 30.5 30.1 25.8 45.8 43.3 34.6 39.3 ...
##  $ DiabetesPedigreeFunction: num  0.167 2.288 0.248 0.158 0.398 ...
##  $ Age                     : int  21 33 26 53 59 51 31 33 32 27 ...
##  $ Outcome                 : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 1 2 1 ...
```

Plotting Histograms to better understand the type of distribution

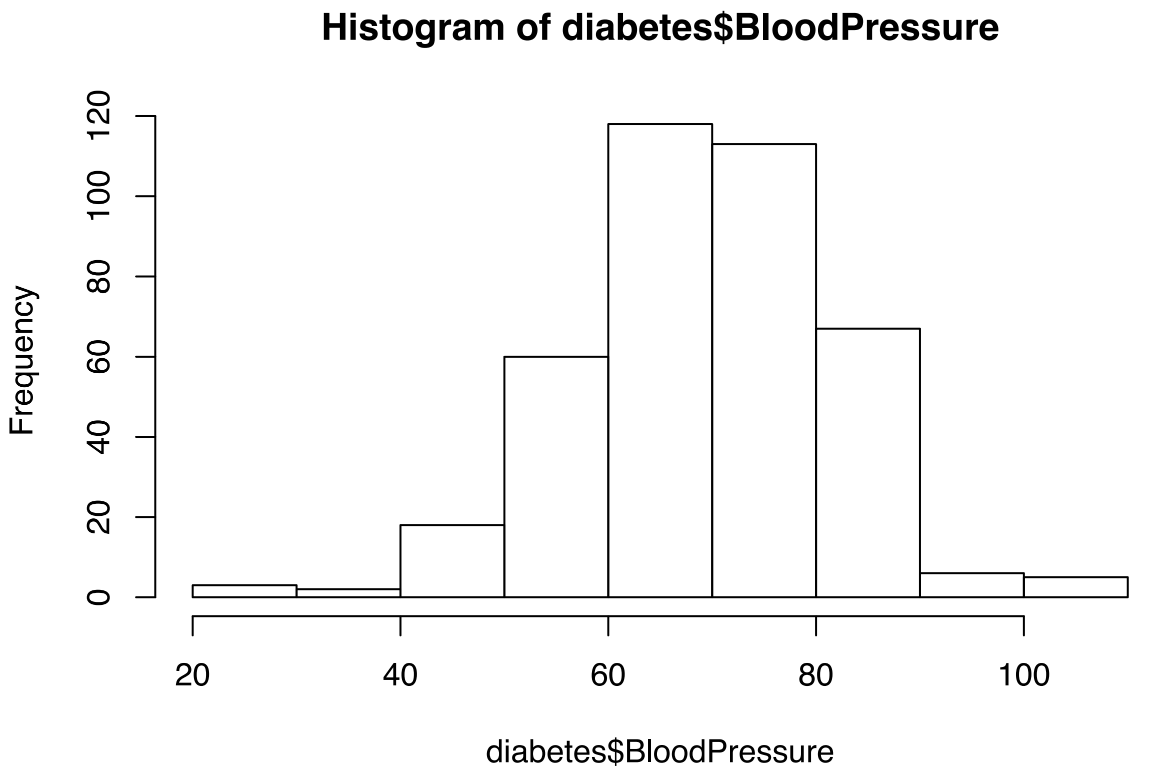
```r
hist(diabetes$Pregnancies)
```



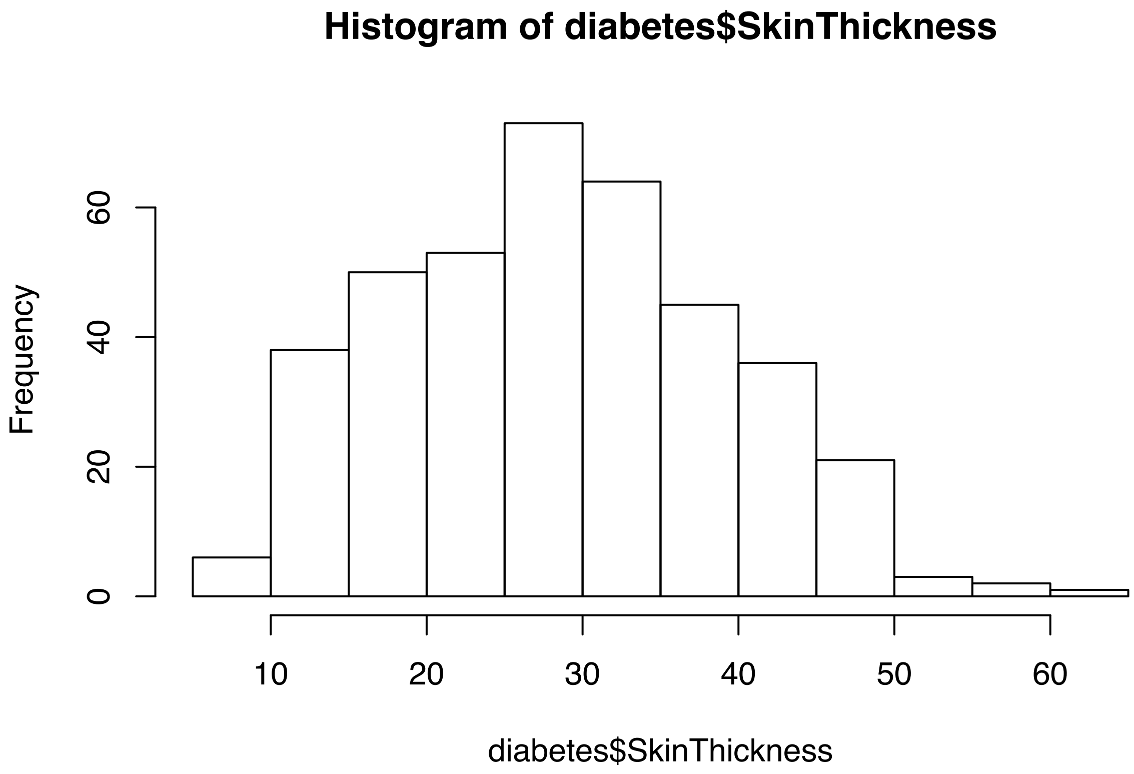
```r
hist(diabetes$Glucose)
```



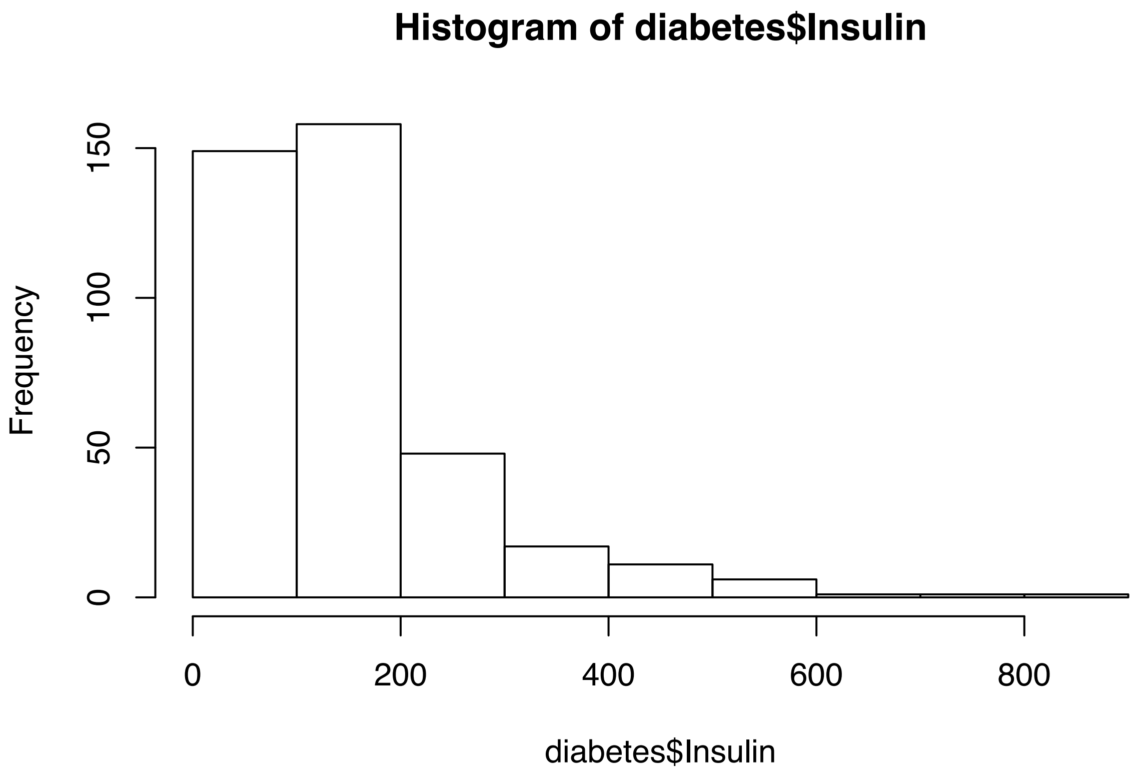
```r
hist(diabetes$BloodPressure)
```



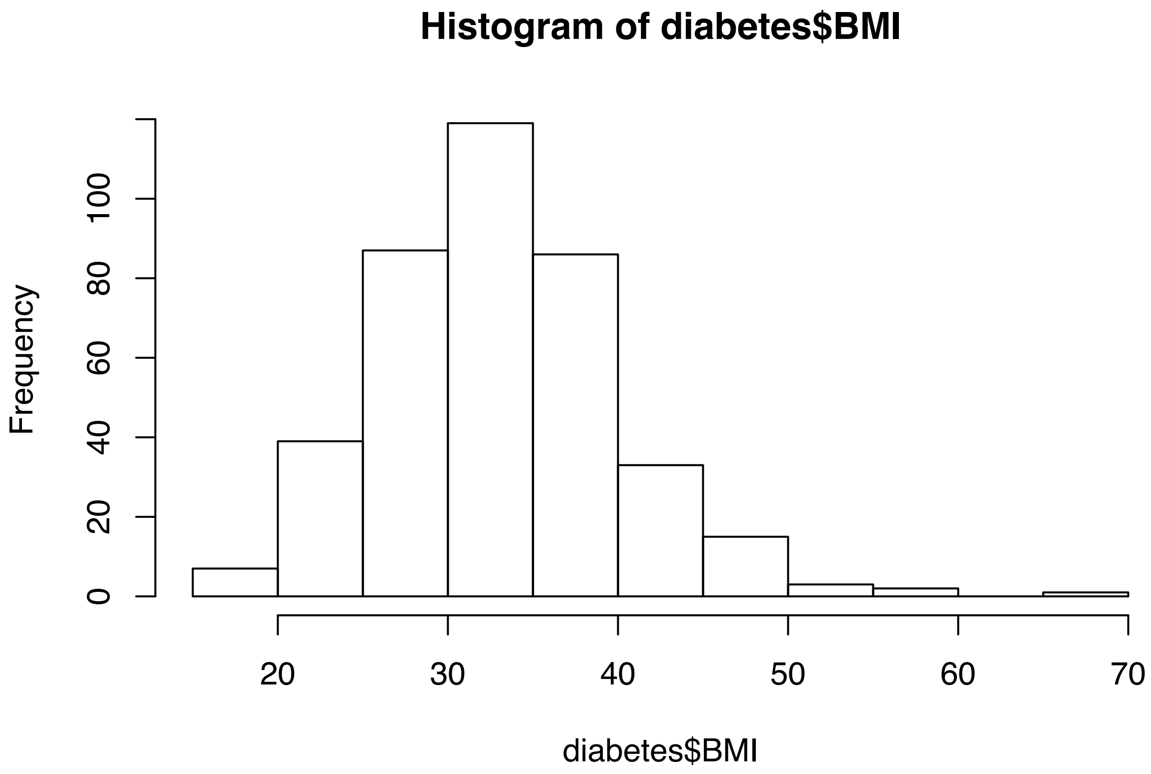
```r
hist(diabetes$SkinThickness)
```



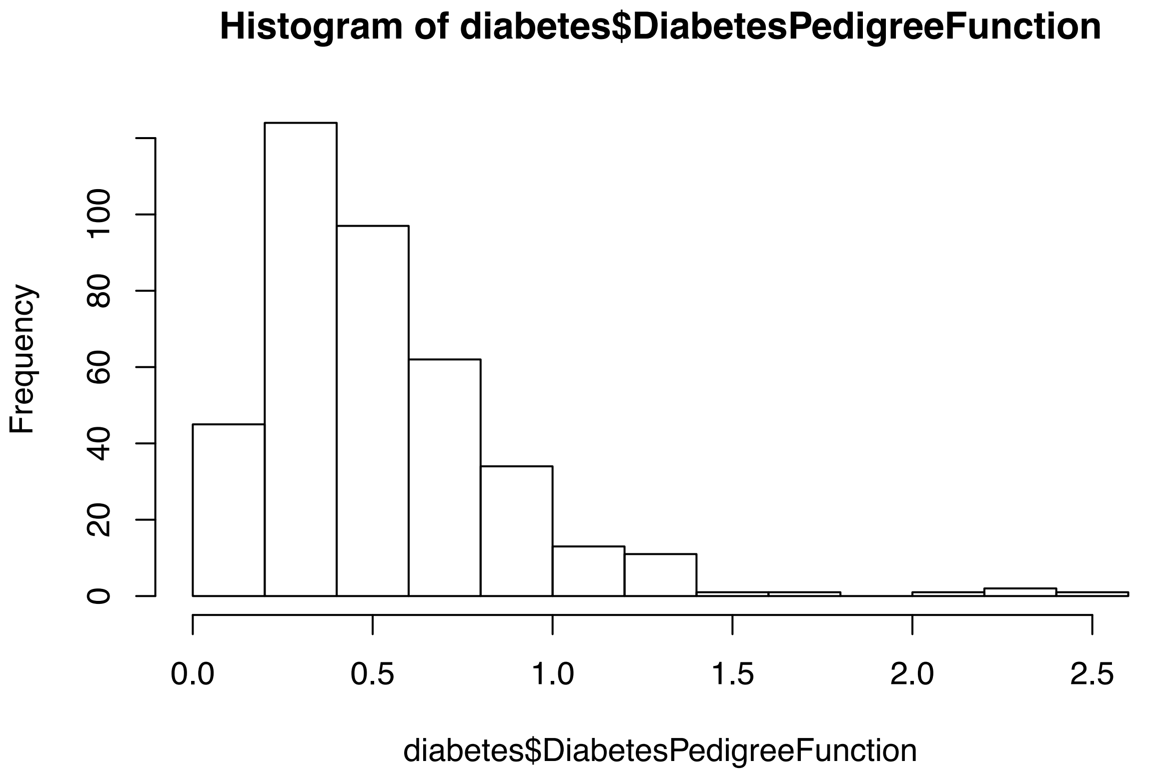
```r
hist(diabetes$Insulin)
```



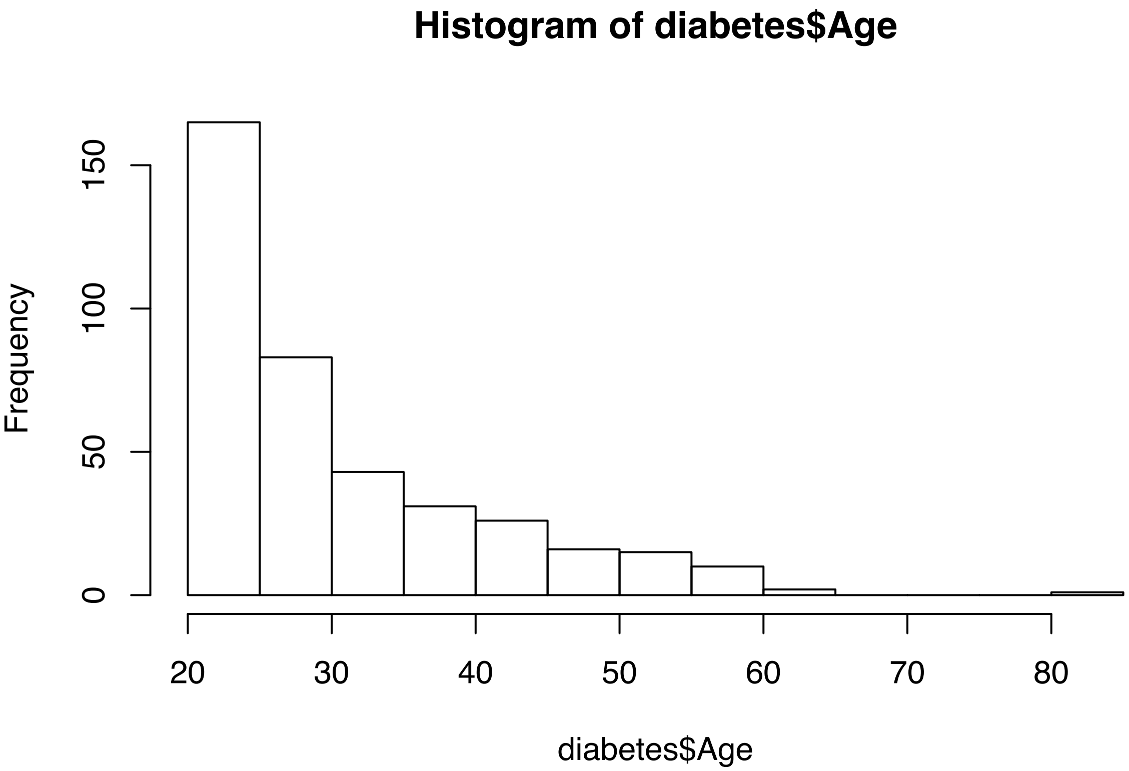
```r
hist(diabetes$BMI)
```



```r
hist(diabetes$DiabetesPedigreeFunction)
```



```r
hist(diabetes$Age)
```



Splitting the Dataset

```r
set.seed(140)
training_set <- sort(sample(nrow(diabetes), nrow(diabetes)*.7))
diabetes_train <- diabetes[training_set,]
diabetes_test <- diabetes[-training_set,]
```

Logistic Regression Model

```r
model <- glm(Outcome~.,data=diabetes_train,family = binomial(link='logit'))
summary(model)
```

```
##
## Call:
## glm(formula = Outcome ~ ., family = binomial(link = "logit"),
##     data = diabetes_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0625  -0.6012  -0.3276   0.5499   2.3017
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -11.125464   1.634157  -6.808 9.89e-12 ***
## Pregnancies               0.157412   0.069340   2.270   0.0232 *
## Glucose                   0.035440   0.006782   5.226 1.73e-07 ***
## BloodPressure             0.013118   0.015103   0.869   0.3851
## SkinThickness             0.015583   0.021990   0.709   0.4785
## Insulin                   0.001034   0.001676   0.617   0.5375
## BMI                       0.080148   0.037468   2.139   0.0324 *
## DiabetesPedigreeFunction  1.057814   0.540177   1.958   0.0502 .
## Age                       0.015937   0.022099   0.721   0.4708
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 344.01  on 273  degrees of freedom
## Residual deviance: 226.50  on 265  degrees of freedom
## AIC: 244.5
##
## Number of Fisher Scoring iterations: 5
```

Using the model on the Test Set

```
predictions <- predict(model,newdata=diabetes_test,type="response")
predictions <- round(predictions)
mean(predictions==diabetes_test$Outcome)
```

```
## [1] 0.7372881
```

Better Logistic Regression Model

```
model2 <- glm(Outcome~Glucose + BMI + DiabetesPedigreeFunction,data=diabetes_train,family = binomial(li
summary(model2)
```

```
##
## Call:
## glm(formula = Outcome ~ Glucose + BMI + DiabetesPedigreeFunction,
##     family = binomial(link = "logit"), data = diabetes_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9432  -0.6826  -0.4079   0.6030   2.1613
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -9.298309   1.235601  -7.525 5.26e-14 ***
## Glucose                   0.040891   0.005835   7.008 2.42e-12 ***
## BMI                       0.083913   0.026804   3.131  0.00174 **
## DiabetesPedigreeFunction  1.065370   0.519535   2.051  0.04030 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 344.01  on 273  degrees of freedom
## Residual deviance: 246.35  on 270  degrees of freedom
## AIC: 254.35
##
## Number of Fisher Scoring iterations: 5
```

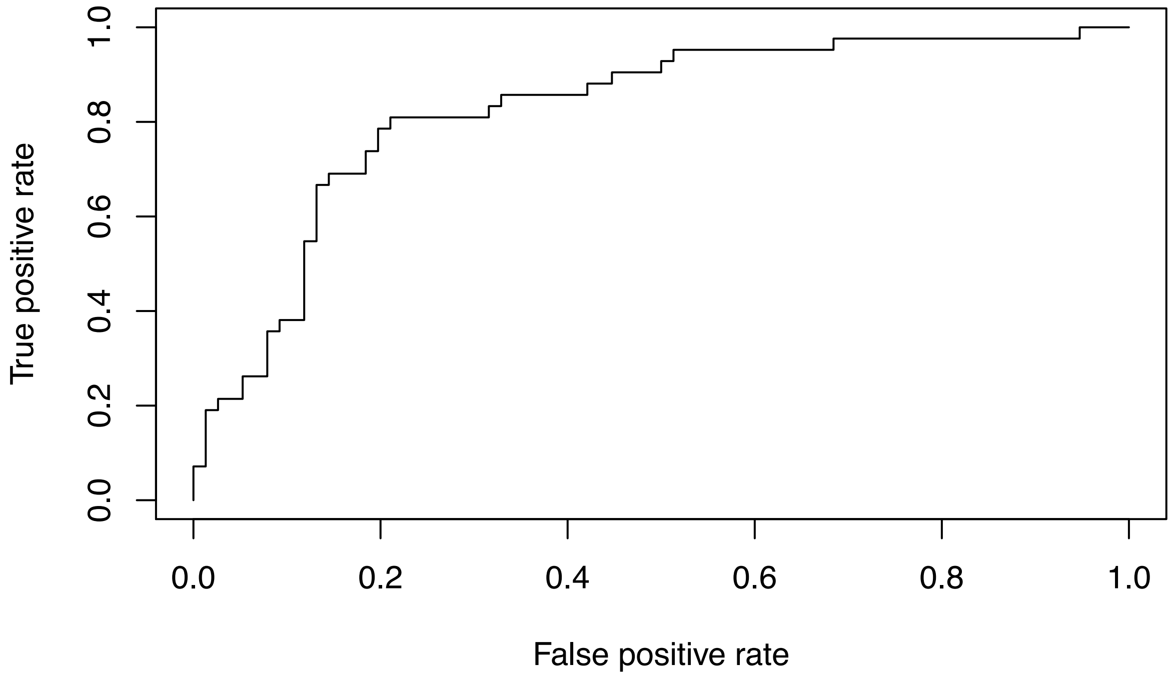
Using the model on the Test Set

```
predictions2 <- predict(model2,newdata=diabetes_test,type="response")
predictions2 <- round(predictions2)
mean(predictions2==diabetes_test$Outcome)
```

```
## [1] 0.7627119
```

ROC Curve

```
p = predict(model2, diabetes_test, type="response")
pr = prediction(p, diabetes_test$Outcome)
prf = performance(pr, measure = "tpr", x.measure = "fpr")
plot(prf)
```



AUC

```
auc = performance(pr, measure = "auc")
auc = auc@y.values[[1]]
print(paste("Model Accuracy", auc))
```

```
## [1] "Model Accuracy 0.824561403508772"
```