Diabetes Project

Wenying Quan

3/7/2022

1. Introduction

Researchers have been interested in Pima Indians very long due to their high incidence rate of diabetes. From 1991 to 2003, the incidence rates of diabetes were 23.5 cases per 1,000 patient-years. Scientists have tried to determine why the rate of diabetes is so high. Researchers speculate that the transition from a traditional lifestyle to a modern lifestyle is associated with the high prevalence of diabetes among Pima Indians. The result was that, in 1970, the prevalence of Type2 diabetes among Pima older than thirty-five years was forty percent and fifty percent in 1990. Some researchers have employed machine learning techniques to discover the relationship. According to the study, body mass index, glucose levels, age, and pregnancy played a more significant role than skin thickness and blood pressure. Other researchers also used the neural networks approach. Using four crossover types, with eight inputs, the best performance was found with the 8-20-1 topology, with five inputs, plasma, diastolic blood pressure, body mass index, diabetes pedigree functions, and age identified by the DT algorithm28. An objective of this project is to identify the risk factors of type 2 diabetes in Pima Indians by using different machine learning approaches. We will investigate the relationship between eight covariates and diagnosis of diabetes.

2. Methods and Analysis

2.1 Data

The Pima Indians Diabetes Database consists of a population of community residents in Phoenix, Arizona, which was first studied in 1965 by the National Institute of Diabetes and Kidney Diseases. All participants are diagnosed with or without diabetes. Other eight risk factors are the number of times pregnant, glucose(mg/dl), diastolic blood pressure($mmHg$), triceps skinfold thickness(mm), 2-hour serum insulin ($\mu U/ml$), body mass index (BMI), diabetes pedigree function and age (years). The total number of observations in this dataset is 768. The bar plot shows the distribution of diagnosis of diabetes in this dataset, we can see the number of without diabetes are almost twice of the number of with diabetes.

```
diab <- read.csv("Diabetes.csv",header = T)
nrow(diab)
```

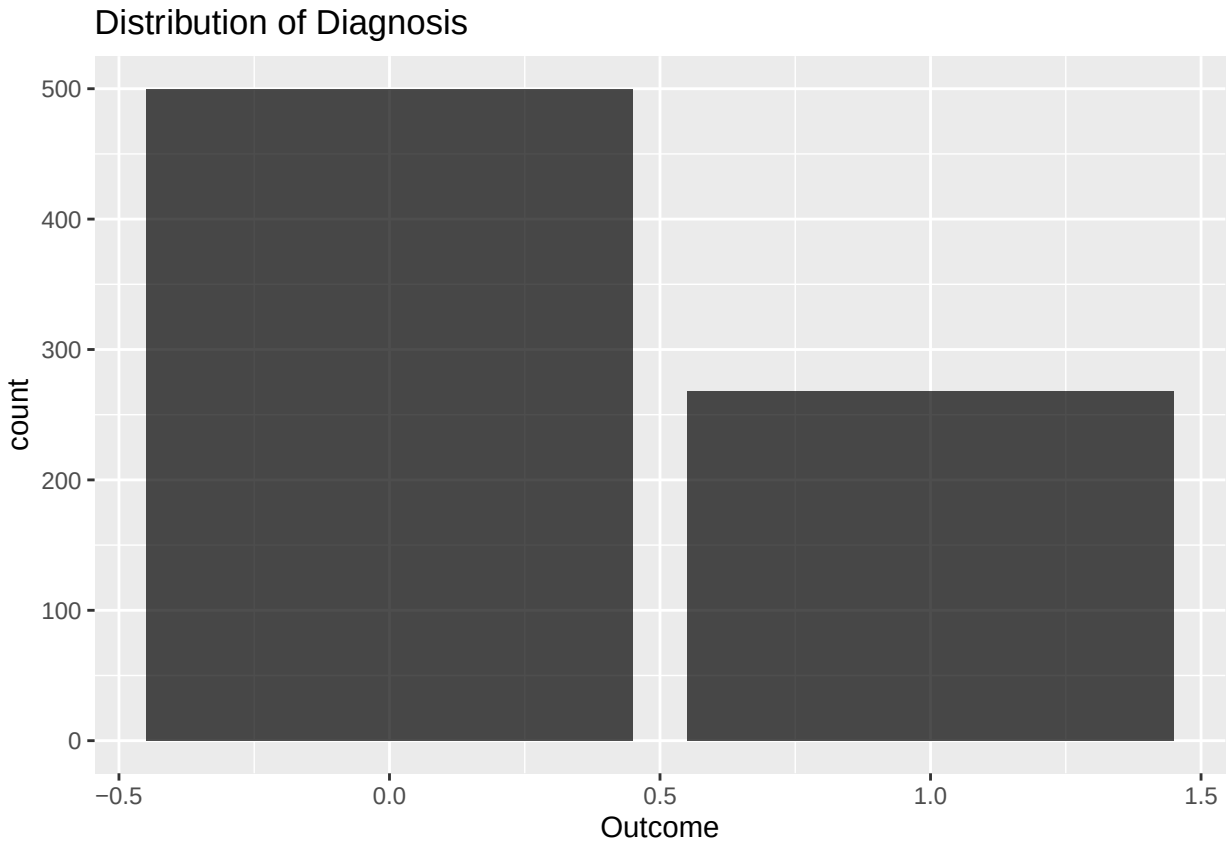
```
## [1] 768
```

```
head(diab)
```

```
##   Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
## 1           6     148            72            35       0 33.6
## 2           1      85            66            29       0 26.6
```

```
## 3           8     183            64             0       0 23.3
## 4           1      89            66            23      94 28.1
## 5           0     137            40            35     168 43.1
## 6           5     116            74             0       0 25.6
##   DiabetesPedigreeFunction Age Outcome
## 1                    0.627  50       1
## 2                    0.351  31       0
## 3                    0.672  32       1
## 4                    0.167  21       0
## 5                    2.288  33       1
## 6                    0.201  30       0
```

```
ggplot(diab, aes(x=Outcome)) + geom_bar(fill="black",alpha=0.7) +
      labs(title = "Distribution of Diagnosis")
```



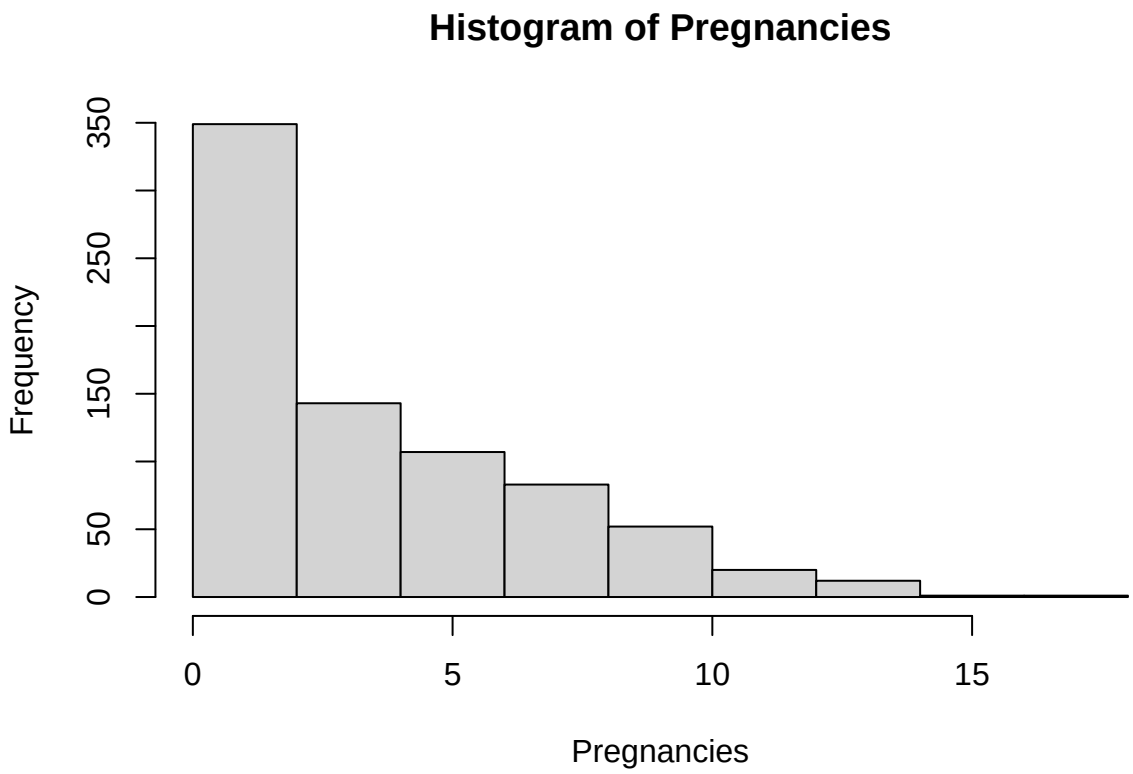
2.2 Univariate analysis

2.2.1 Number of pregnancies

```
summary(diab$Pregnancies)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.000   3.000   3.845   6.000  17.000
```

```
hist(diab$Pregnancies,main = "Histogram of Pregnancies", xlab = "Pregnancies")
```



The summary statistics and histogram of number of pregnancies show that, most of participants have history of pregnancies less than 5 times. There are some extreme cases that some of them have pregnancies more than 10 times.

2.2.2 Glucose

```
summary(diab$Glucose)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0    99.0   117.0   120.9   140.2   199.0
```

```
hist(diab$Glucose,main = "Histogram of Glucose", xlab = "Glucose")
```

Substitute missing values with mean of Glucose.

```
diab$Glucose <- na_if(diab$Glucose,0)
diab$Glucose[is.na(diab$Glucose)] <- mean(diab$Glucose, na.rm = T)
summary(diab$Glucose)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   44.00   99.75  117.00  121.69  140.25  199.00
```

```
hist(diab$Glucose,main = "Histogram of Glucose", xlab = "Glucose")
```

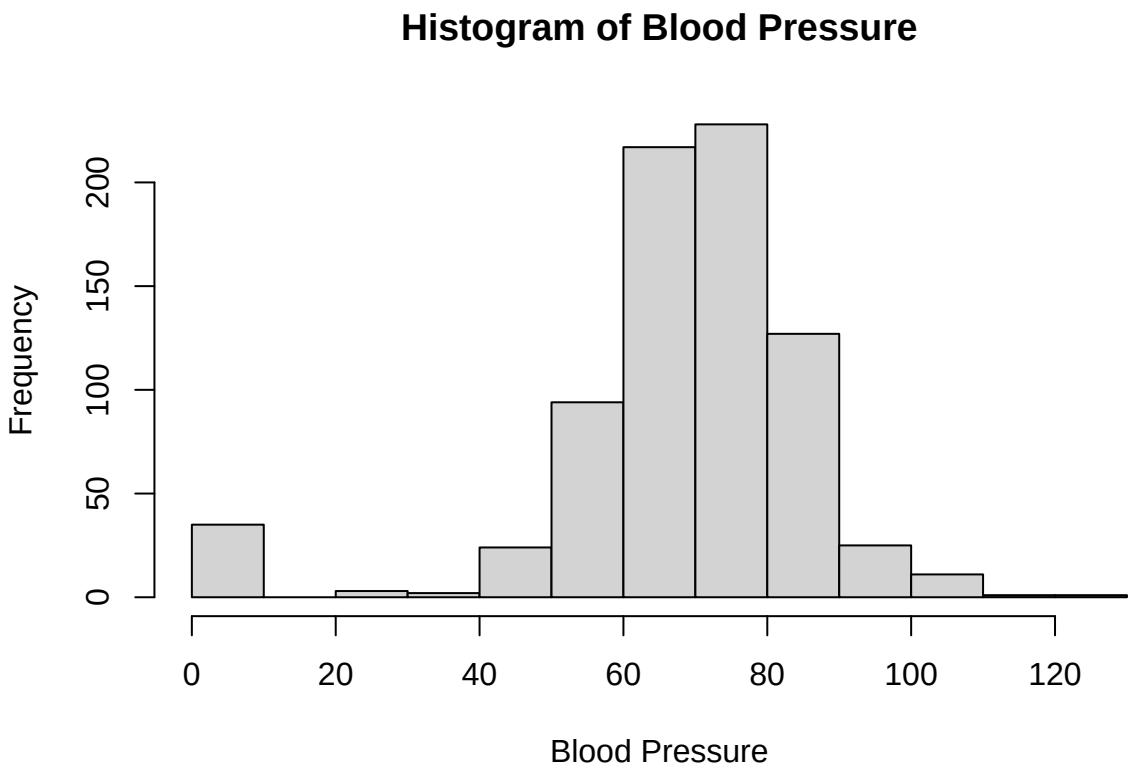
From the original summary statistics of glucose, we observe that there are some missing values, since the value of glucose cannot be zero for a person. Based on the distribution of glucose, we decided to substitute the missing values with mean of original data.

2.2.3 Blood Pressure

```
summary(diab$BloodPressure)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   62.00   72.00   69.11   80.00  122.00
```

```
hist(diab$BloodPressure,main = "Histogram of Blood Pressure", xlab = "Blood Pressure")
```

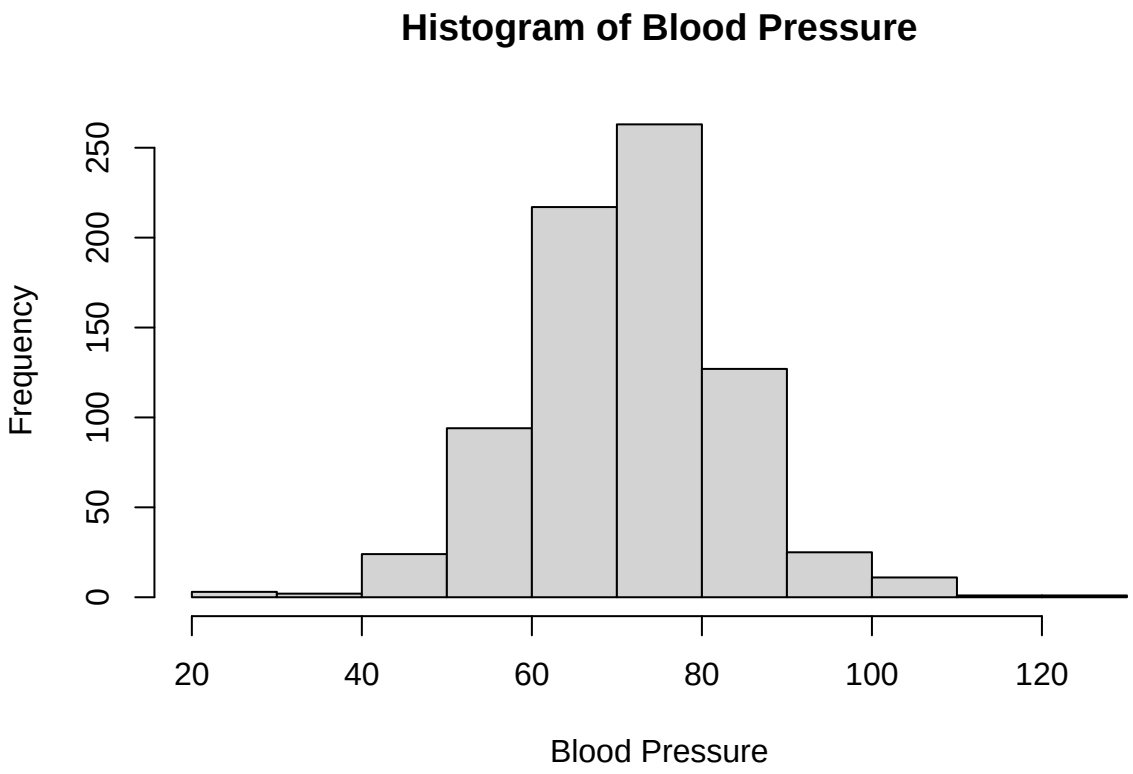


Substitute missing values with mean of Blood Pressure.

```
diab$BloodPressure <- na_if(diab$BloodPressure,0)
diab$BloodPressure[is.na(diab$BloodPressure)] <- mean(diab$BloodPressure, na.rm = T)
summary(diab$BloodPressure)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   24.00   64.00   72.20   72.41   80.00  122.00
```

```
hist(diab$BloodPressure,main = "Histogram of Blood Pressure", xlab = "Blood Pressure")
```



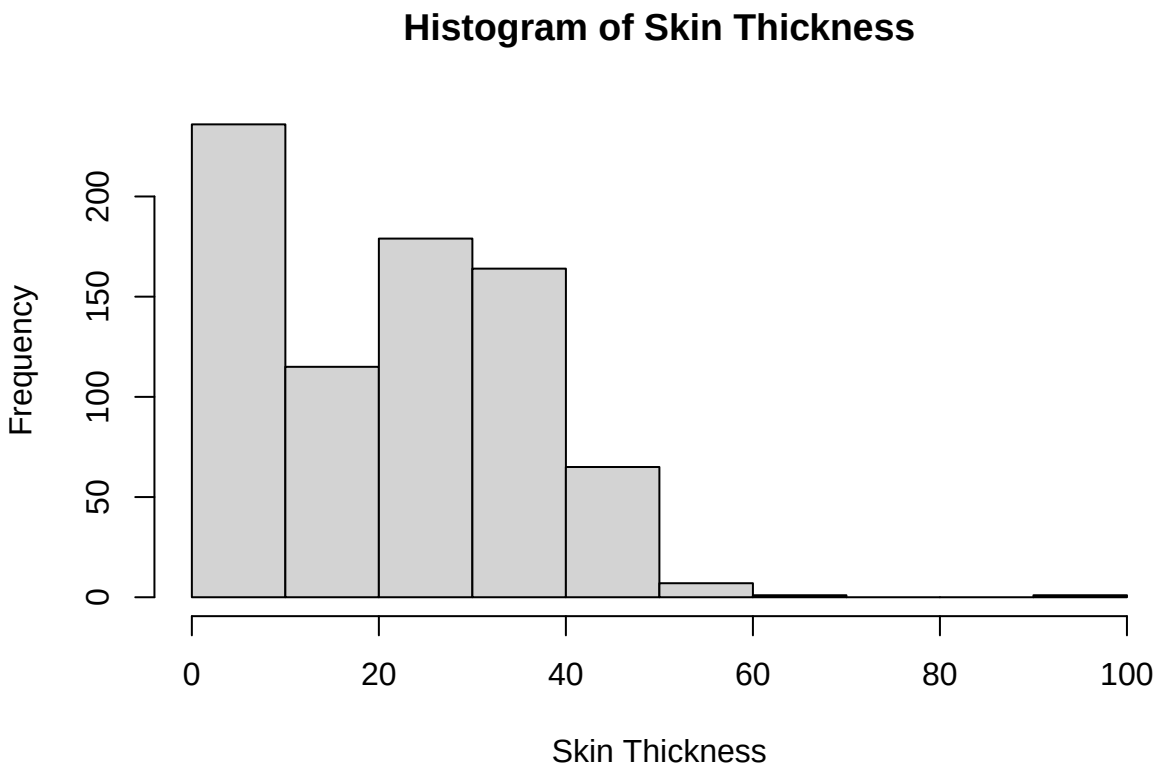
From the original summary statistics of blood pressure, we observe that there are some missing values, since the value of blood pressure cannot be zero for a person. Based on the distribution of blood pressure, we decided to substitute the missing values with mean of original data.

2.2.4 Skin Thickness

```
summary(diab$SkinThickness)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    0.00   23.00   20.54   32.00   99.00
```

```
hist(diab$SkinThickness,main = "Histogram of Skin Thickness", xlab = "Skin Thickness")
```

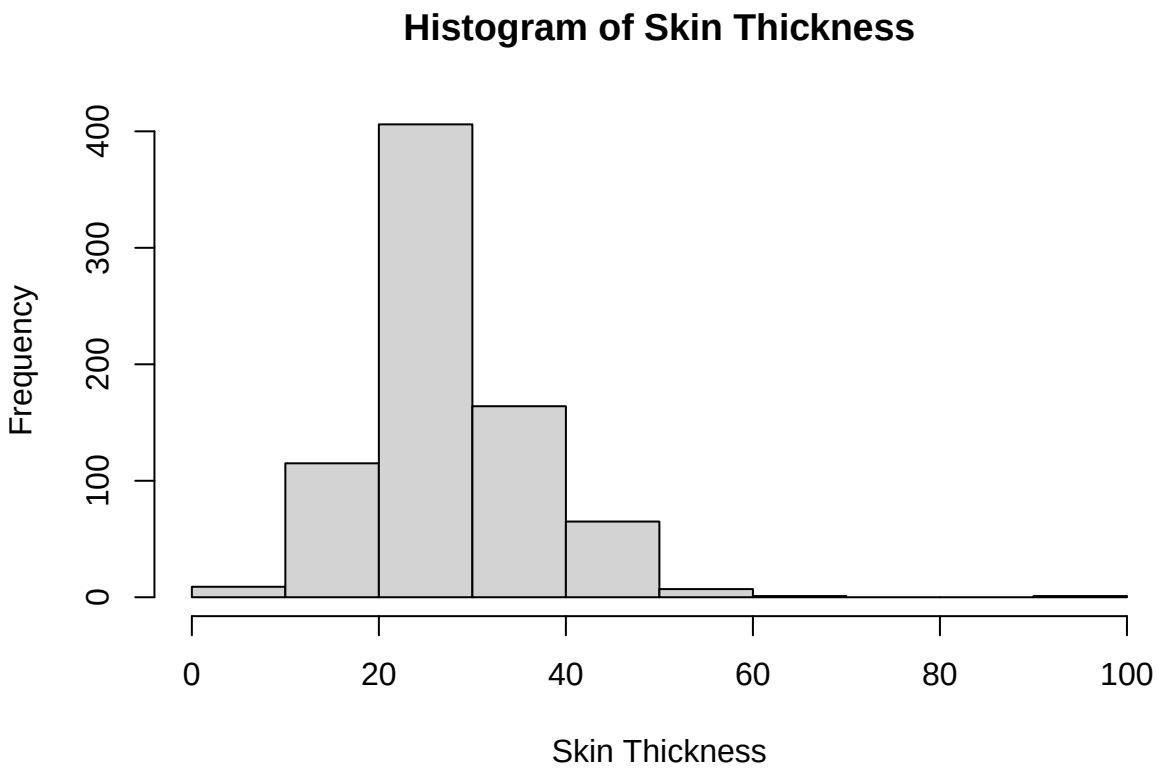


Substitute missing values with median of Skin Thickness.

```
diab$SkinThickness <- na_if(diab$SkinThickness,0)
diab$SkinThickness[is.na(diab$SkinThickness)] <- median(diab$SkinThickness, na.rm = T)
summary(diab$SkinThickness)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    7.00   25.00   29.00   29.11   32.00   99.00
```

```
hist(diab$SkinThickness,main = "Histogram of Skin Thickness", xlab = "Skin Thickness")
```



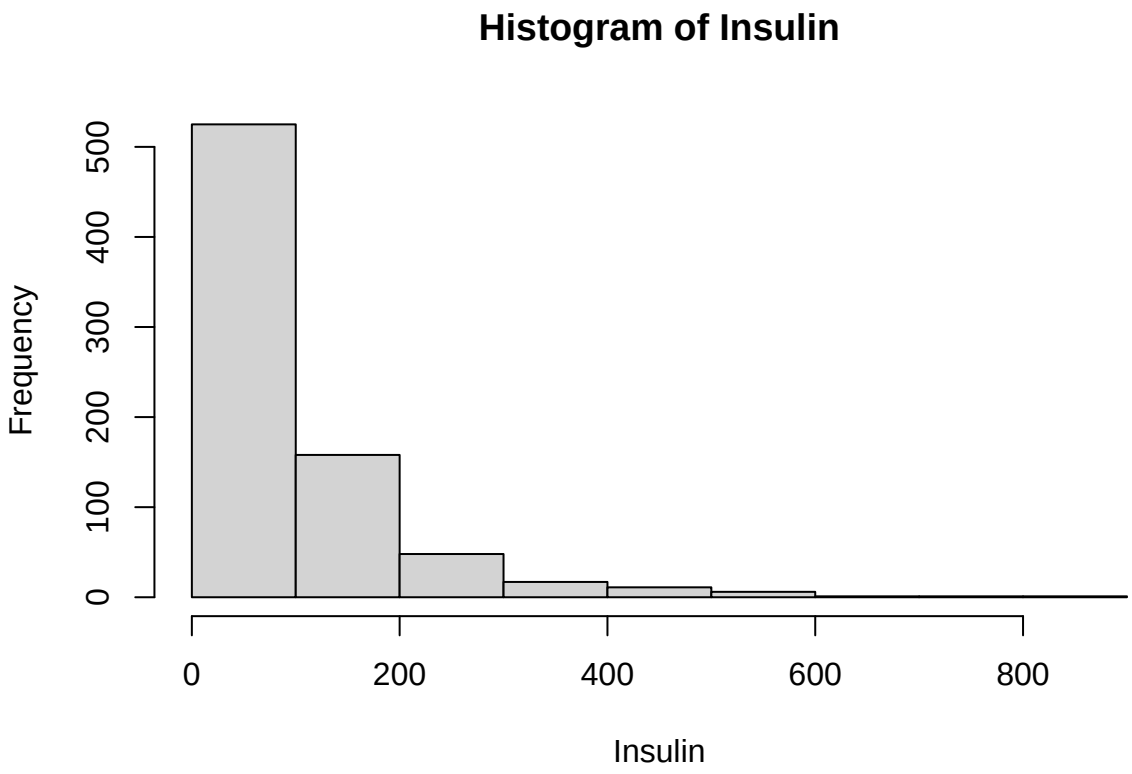
From the original summary statistics of triceps skinfold thickness, we observe that there are some missing values, since the value of triceps skinfold thickness cannot be zero for a person. Based on the distribution of triceps skinfold thickness, we decided to substitute the missing values with median of original data.

2.2.5 Insulin

```
summary(diab$Insulin)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0     0.0    30.5    79.8   127.2   846.0
```

```
hist(diab$Insulin,main = "Histogram of Insulin", xlab = "Insulin")
```

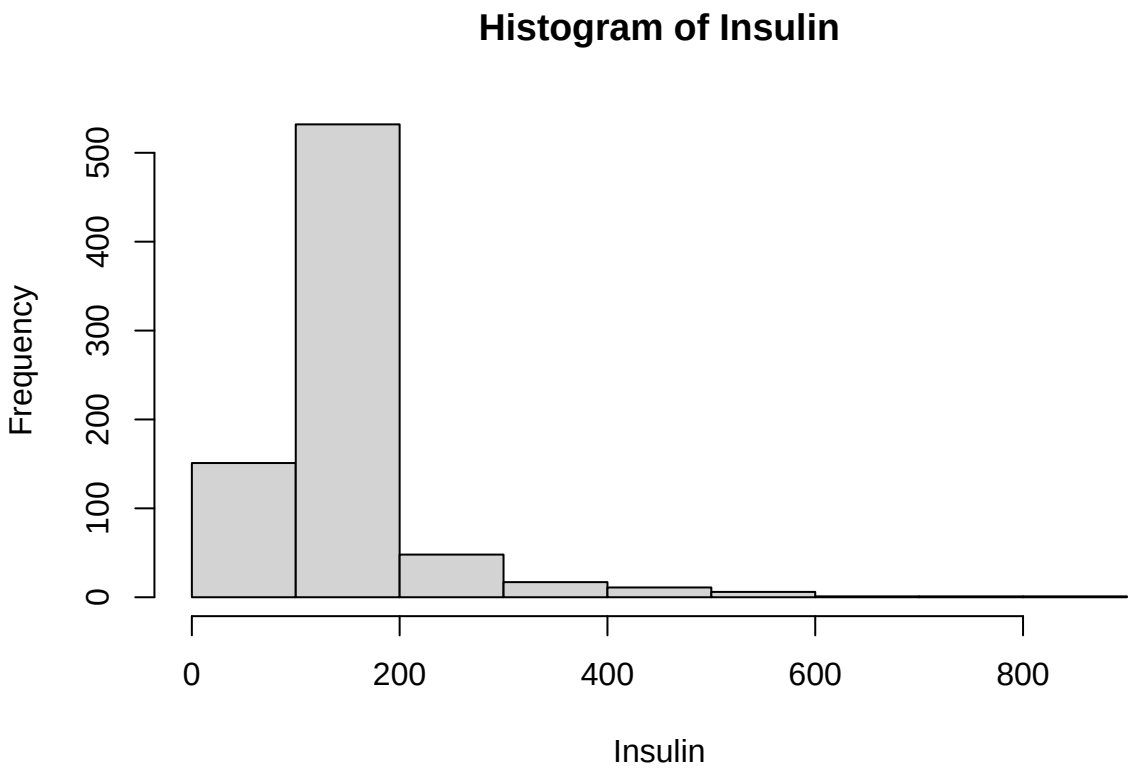


Substitute missing values with median of Insulin.

```
diab$Insulin <- na_if(diab$Insulin,0)
diab$Insulin[is.na(diab$Insulin)] <- median(diab$Insulin, na.rm = T)
summary(diab$Insulin)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    14.0   121.5   125.0   140.7   127.2   846.0
```

```
hist(diab$Insulin,main = "Histogram of Insulin", xlab = "Insulin")
```



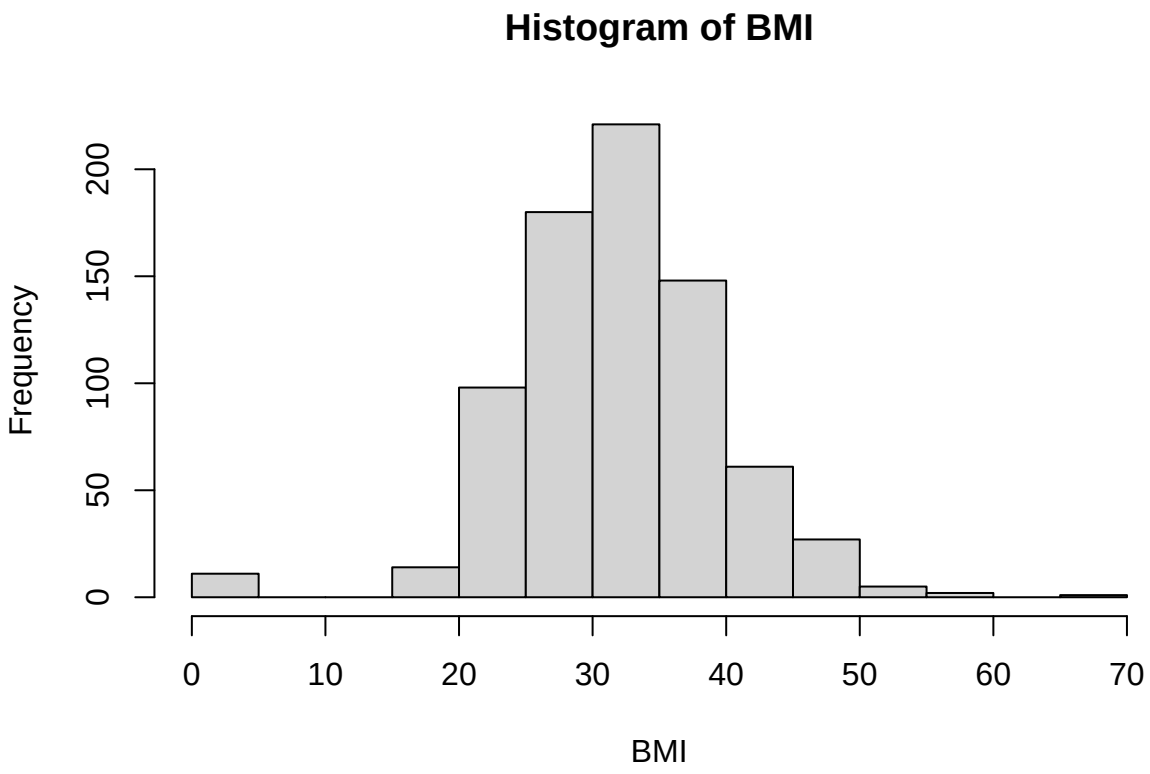
From the original summary statistics of insulin , we observe that there are some missing values, since the value of insulin cannot be zero for a person. Based on the distribution of insulin, we decided to substitute the missing values with median of original data.

2.2.6 BMI

```
summary(diab$BMI)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   27.30   32.00   31.99   36.60   67.10
```

```
hist(diab$BMI,main = "Histogram of BMI", xlab = "BMI")
```

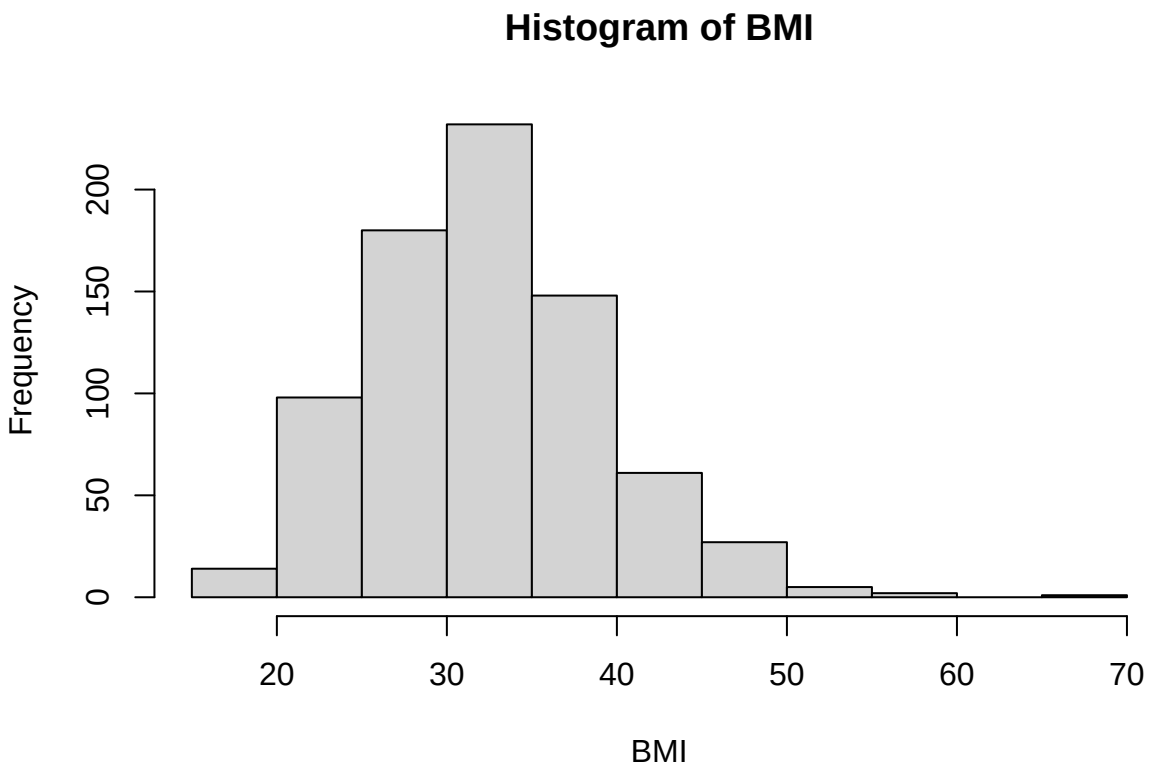


Substitute missing values with median of BMI.

```
diab$BMI <- na_if(diab$BMI,0)
diab$BMI[is.na(diab$BMI)] <- median(diab$BMI, na.rm = T)
summary(diab$BMI)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.20   27.50   32.30   32.46   36.60   67.10
```

```
hist(diab$BMI,main = "Histogram of BMI", xlab = "BMI")
```



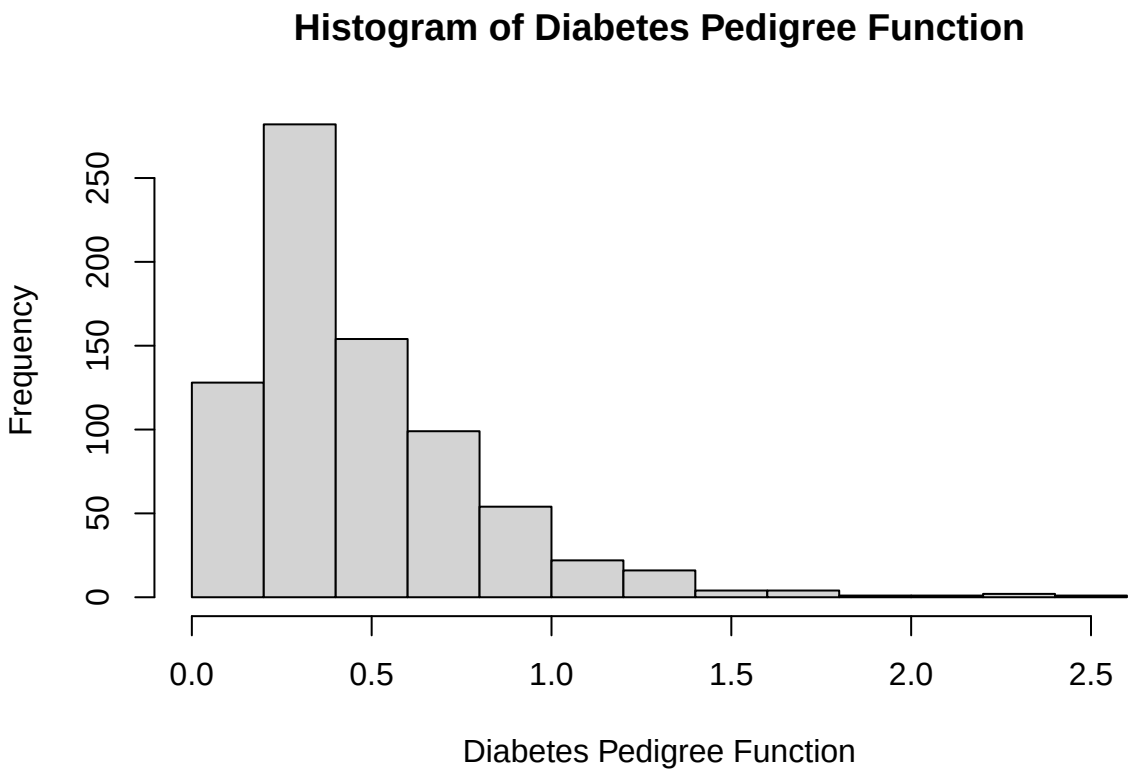
From the original summary statistics of BMI , we observe that there are some missing values, since the value of BMI cannot be zero for a person. Based on the distribution of BMI, we decided to substitute the missing values with median of original data.

2.2.7 Diabetes Pedigree Function

```
summary(diab$DiabetesPedigreeFunction)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0780  0.2437  0.3725  0.4719  0.6262  2.4200
```

```
hist(diab$DiabetesPedigreeFunction,main = "Histogram of Diabetes Pedigree Function",
     xlab = "Diabetes Pedigree Function")
```



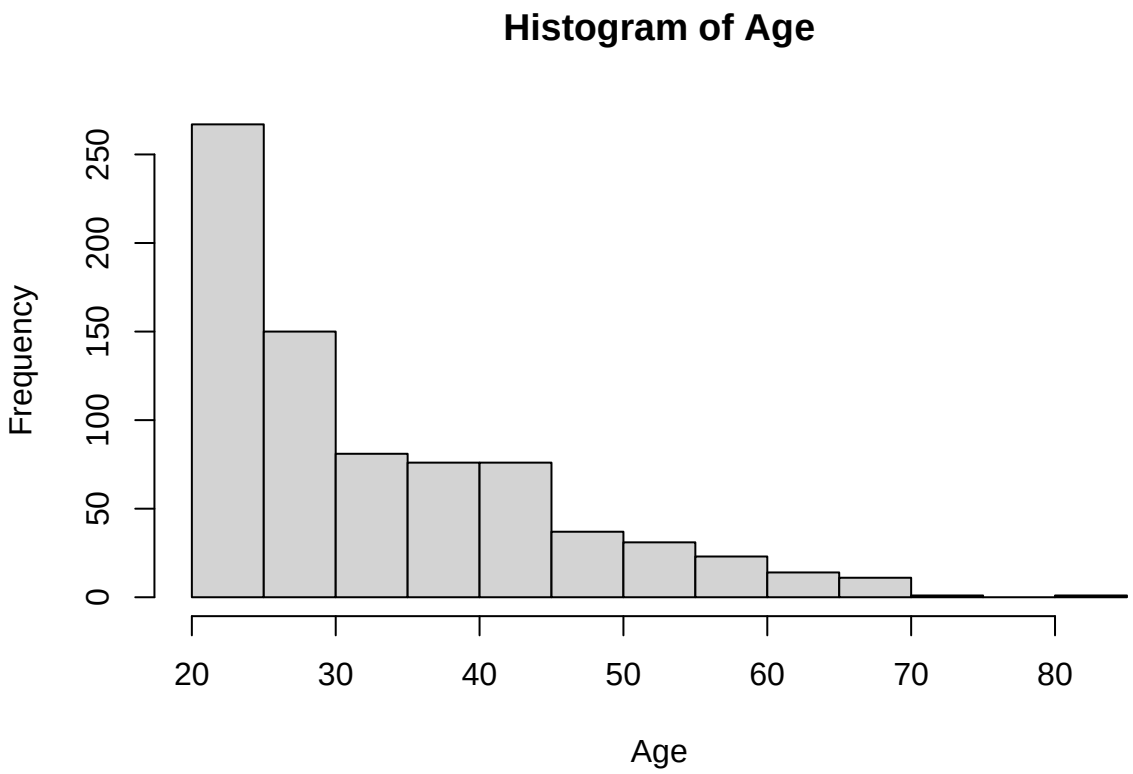
The original data of diabetes pedigree function has no missing values, and we can observe a right-skewed distribution from the histogram.

2.2.8 Age

```
summary(diab$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.00   24.00   29.00   33.24   41.00   81.00
```

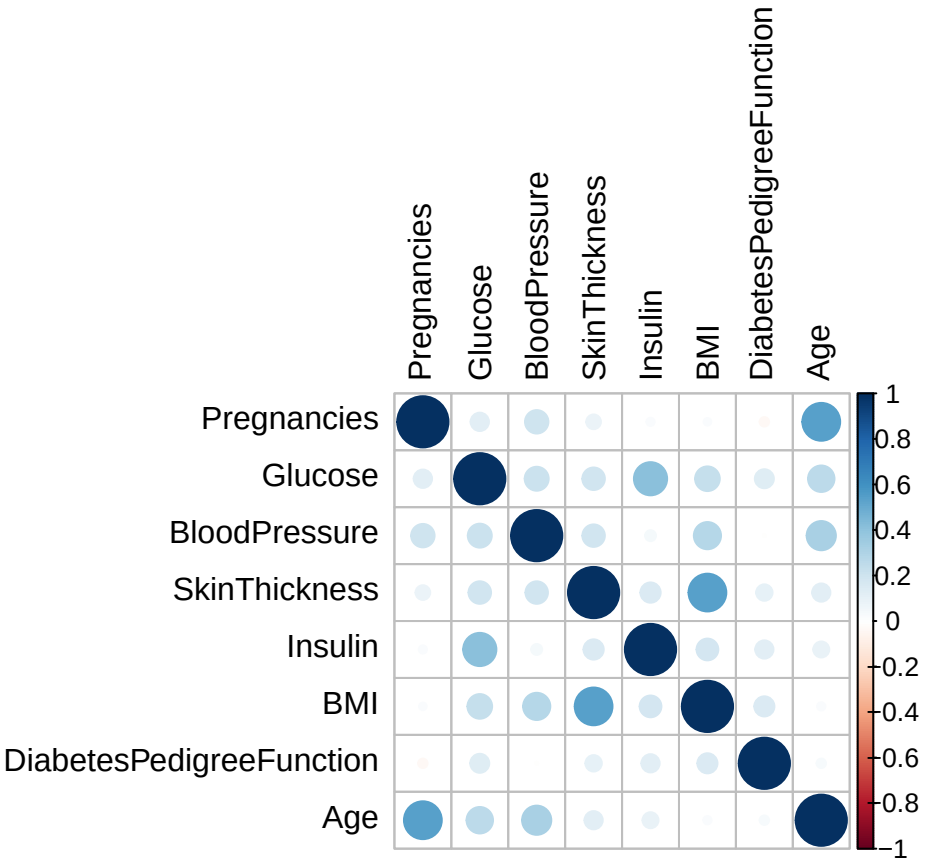
```
hist(diab$Age,main = "Histogram of Age", xlab = "Age")
```



The original data of age has no missing values either, and from the histogram we can see most of the participants are about 20-30 years old. The maximun age in this dataset is 81 years old.

2.3 Check correlation between 8 risk factors

```
corrplot(cor(diab[1:8]),tl.col = "black")
```




```
correlation <- cor(diab[1:8])
findCorrelation(correlation, cutoff = 0.7)
```

```
## integer(0)
```

Since multicollinearity may have severe impact to the results, we need to check correlations between continuous variables. From the figure, we can observe a significant correlation between age and pregnancies, BMI and skin thickness. By a rule of thumb, we consider a high correlation if the correlation value r is equal or bigger than 0.7. After calculation, there is no significant multicollinearity observed.

2.4 Model Fitting

Split the data into traing and test set.

```
set.seed(1,sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
test_index <-createDataPartition(y = diab$Outcome, times = 1, p = 0.25, list = FALSE)
train <-diab[-test_index,]
test <- diab[test_index,]
```

In this step, split our dataset into training and test sets. We use only training set to develop our machine learning models and then use test set to create confusion matrix so that we can have a thorough idea about overall accuracy, precision, and sensitivity, etc.

2.4.1 Logistic Regression Model

Logistic regression is an extension of linear regression that assures that the estimate of conditional probability $P_r(Y = 1|X = x)$ is between 0 and 1. This approach makes use of the logistic transformation: $g(p) = log(p/1 - p)$. In R, we can fit the logistic regression model with the function glm() (generalized linear models).

```
fit_glm <- glm(Outcome ~., data=train, family="binomial")
p_hat_glm <- predict(fit_glm, test)
y_hat_glm <- factor(ifelse(p_hat_glm > 0.5,1,0))
glm_confusion <- confusionMatrix(data = y_hat_glm, reference = factor(test$Outcome))
glm_confusion
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 116  28
##          1  10  38
##
##                Accuracy : 0.8021
##                  95% CI : (0.7386, 0.856)
##     No Information Rate : 0.6562
##     P-Value [Acc > NIR] : 6.487e-06
##
##                   Kappa : 0.5309
##
##  Mcnemar's Test P-Value : 0.00582
##
##             Sensitivity : 0.9206
##             Specificity : 0.5758
##          Pos Pred Value : 0.8056
##          Neg Pred Value : 0.7917
##              Prevalence : 0.6562
##          Detection Rate : 0.6042
##    Detection Prevalence : 0.7500
##       Balanced Accuracy : 0.7482
##
##        'Positive' Class : 0
##
```

```
varImp(fit_glm)
```

```
##                             Overall
## Pregnancies              3.86037769
## Glucose                  8.03177888
## BloodPressure            0.54559479
## SkinThickness            0.08543942
```

```
## Insulin                   0.88519971
## BMI                       4.72013769
## DiabetesPedigreeFunction 2.83603659
## Age                       0.46451948
```

Based on the output, for logistic regression model, we get an overall accuracy is equal to 0.8021. And the first three most important variables that contribute the most to the model are glucose, BMI, and pregnancies.

2.4.2 K Nearest Neighbor(KNN) Model

K-nearest neighbors (kNN) estimates the conditional probabilities in a similar way to bin smoothing. However, kNN is easier to adapt to multiple dimensions. Using kNN, for any point (x_1, x_2) for which we want an estimate of $p(x_1, x_2)$, we look for the k nearest points to (x_1, x_2) and take an average of the 0s and 1s associated with these points. We refer to the set of points used to compute the average as the neighborhood.

```
train$Outcome <- factor(train$Outcome)
test$Outcome <- factor(test$Outcome)
train_knn<- train(Outcome~., method = "knn", data = train)
y_hat_knn <- predict(train_knn,test)
knn_confusion <- confusionMatrix(data = y_hat_knn, reference=test$Outcome)
knn_confusion
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 94 19
##          1 32 47
##
##                Accuracy : 0.7344
##                  95% CI : (0.666, 0.7954)
##     No Information Rate : 0.6562
##     P-Value [Acc > NIR] : 0.01256
##
##                   Kappa : 0.4376
##
##  Mcnemar's Test P-Value : 0.09289
##
##             Sensitivity : 0.7460
##             Specificity : 0.7121
##          Pos Pred Value : 0.8319
##          Neg Pred Value : 0.5949
##              Prevalence : 0.6562
##          Detection Rate : 0.4896
##    Detection Prevalence : 0.5885
##       Balanced Accuracy : 0.7291
##
##        'Positive' Class : 0
##
```

```
varImp(train_knn)
```

```
## ROC curve variable importance
##
##                           Importance
## Glucose                      100.0000
## BMI                           47.9407
## Age                           39.0532
## Insulin                       25.2342
## SkinThickness                  5.4231
## DiabetesPedigreeFunction       4.9237
## Pregnancies                    0.9909
## BloodPressure                  0.0000
```

For kNN model, we get an overall accuracy is equal to 0.7344. And the first three most important variables that contribute the most to the model are glucose, BMI, and age.

2.4.3 Random Forest Model

Random forests are a very popular machine learning approach that addresses the shortcomings of decision trees. The goal is to improve prediction performance and reduce instability by averaging multiple decision trees (a forest of trees constructed with randomness). The general idea of random forests is to generate many predictors, each using regression or classification trees, and then forming a final prediction based on the average prediction of all these trees.

```
train_rf <- train(Outcome~., method="rf", data=train)
rf_confusion <- confusionMatrix(predict(train_rf,test),test$Outcome)
rf_confusion
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 105  21
##          1  21  45
##
##                Accuracy : 0.7812
##                  95% CI : (0.716, 0.8376)
##     No Information Rate : 0.6562
##     P-Value [Acc > NIR] : 0.0001103
##
##                   Kappa : 0.5152
##
##  Mcnemar's Test P-Value : 1.0000000
##
##             Sensitivity : 0.8333
##             Specificity : 0.6818
##          Pos Pred Value : 0.8333
##          Neg Pred Value : 0.6818
##              Prevalence : 0.6562
##          Detection Rate : 0.5469
##    Detection Prevalence : 0.6562
##       Balanced Accuracy : 0.7576
##
##        'Positive' Class : 0
```

```
##
```

```
varImp(train_rf)
```

```
## rf variable importance
##
##                           Overall
## Glucose                   100.000
## BMI                        55.415
## DiabetesPedigreeFunction   35.298
## Age                        33.440
## Pregnancies                12.146
## Insulin                    11.039
## BloodPressure               7.151
## SkinThickness               0.000
```

For random forest model, we get an overall accuracy is equal to 0.7812. And the first three most important variables that contribute the most to the model are glucose, BMI, and diabetes pedigree function.

3. Results

Now, we create a comprehensive table to compare all the results we have obtained before.

```
confusionmatrix <- list(logistic_regression = glm_confusion,
                        knn = knn_confusion, random_forest = rf_confusion)
final_result <- sapply(confusionmatrix, function(x) x$byClass)
final_result %>% knitr::kable()
```

	logistic_regression	knn	random_forest
Sensitivity	0.9206349	0.7460317	0.8333333
Specificity	0.5757576	0.7121212	0.6818182
Pos Pred Value	0.8055556	0.8318584	0.8333333
Neg Pred Value	0.7916667	0.5949367	0.6818182
Precision	0.8055556	0.8318584	0.8333333
Recall	0.9206349	0.7460317	0.8333333
F1	0.8592593	0.7866109	0.8333333
Prevalence	0.6562500	0.6562500	0.6562500
Detection Rate	0.6041667	0.4895833	0.5468750
Detection Prevalence	0.7500000	0.5885417	0.6562500
Balanced Accuracy	0.7481962	0.7290765	0.7575758

For logistic regression model, we have a precision equals to 0.8056, which means 80.56% of the time this model will classify the patients in a high risk category when they actually had a high risk of getting diabetes. The recall/sensitivity is 0.9206, indicates 92.06% of the time, patients actually having high risk are classified by this model. For kNN model, the precision is 0.8319, and the recall/sensitivity is 0.7460. It has slight higher precision compare to logistic regression, but has a much lower recall/sensitivity score. For random forest model, the precision is 0.8333, and the recall/sensitivity is 0.8333. It has slight higher precision compare to logistic regression, but has a lower recall/sensitivity score. Since the $F1 - score$ is the harmonic average of precision and recall, we can consider $F1 - score$ as a decision rule as well. Comparing these three models,

logistic regression has the highest value, which is equal to 0.8593.

4. Conclusion

In this project, we investigate the connection between eight different risk factors and developing diabetes in Pima Indians. We tried three different machine learning models to figure out which is the best model to predict a risk of developing diabetes. Based on the previous result section, our conclusion is that, the logistic regression model performs better than other two models given this dataset. Since it has the highest recall/sensitivity score and $F1 - score$. Although, the precision is not the best, but it has no much big difference compare to others. And the most important factors in determining diabetes are glucose, BMI, age, diabetes pedigree function, and pregnancies. There are several limitations of our study. There are only 768 observations in this dataset, a larger dataset would give a more reliable result with greater precision and power. To verify our conclusion, a further study such as including a larger sample size or using other model-fitted methods should be conducted in our future investigation.