Data606FinalProj

Mathew Katz

2022-12-06

Abstract:

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. In 2019, 37.3 million Americans, or 11.3% of the population, had diabetes. 1.4 million Americans are diagnosed with diabetes every year. It is important to be able to diagnose such a worldwide problem. 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.

Import Libraries we will need:

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(readr)
library(ggplot2)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
library(caret)
```

```
## Loading required package: lattice
```

```r
library(party)
```

```
## Loading required package: grid
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: modeltools
```

```
## Loading required package: stats4
```

```
## Loading required package: strucchange
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

```r
library(e1071)
```

Read in our Diabetes Dataset:

```r
df <- read_csv("diabetes.csv", show_col_types = FALSE)
```

Look at every column in the data frame:

```r
df %>% str()
```

```
## spc_tbl_ [768 x 9] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Pregnancies             : num [1:768] 6 1 8 1 0 5 3 10 2 8 ...
##  $ Glucose                 : num [1:768] 148 85 183 89 137 116 78 115 197 125 ...
##  $ BloodPressure           : num [1:768] 72 66 64 66 40 74 50 0 70 96 ...
##  $ SkinThickness           : num [1:768] 35 29 0 23 35 0 32 0 45 0 ...
##  $ Insulin                 : num [1:768] 0 0 0 94 168 0 88 0 543 0 ...
##  $ BMI                     : num [1:768] 33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
##  $ DiabetesPedigreeFunction: num [1:768] 0.627 0.351 0.672 0.167 2.288 ...
##  $ Age                     : num [1:768] 50 31 32 21 33 30 26 29 53 54 ...
##  $ Outcome                 : num [1:768] 1 0 1 0 1 0 1 0 1 1 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Pregnancies = col_double(),
##   ..   Glucose = col_double(),
##   ..   BloodPressure = col_double(),
##   ..   SkinThickness = col_double(),
##   ..   Insulin = col_double(),
##   ..   BMI = col_double(),
##   ..   DiabetesPedigreeFunction = col_double(),
##   ..   Age = col_double(),
##   ..   Outcome = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Based off of pregnancies, glucose level, blood pressure, skin thickness, insulin level, BMI ,Diabetes Pedigree Function, and age let's predict whether someone has diabetes or not but first lets take a quick look at the data:

```
df%>% head()
```

```
## # A tibble: 6 x 9
##   Pregnancies Glucose BloodPressure SkinTh~1 Insulin   BMI Diabe~2   Age Outcome
##         <dbl>   <dbl>         <dbl>    <dbl>   <dbl> <dbl>   <dbl> <dbl>   <dbl>
## 1           6     148            72       35       0  33.6   0.627    50       1
## 2           1      85            66       29       0  26.6   0.351    31       0
## 3           8     183            64        0       0  23.3   0.672    32       1
## 4           1      89            66       23      94  28.1   0.167    21       0
## 5           0     137            40       35     168  43.1   2.29     33       1
## 6           5     116            74        0       0  25.6   0.201    30       0
## # ... with abbreviated variable names 1: SkinThickness,
## #   2: DiabetesPedigreeFunction
```

The outcome column needs to be changed from a number to a factor (also called categorical or enumerative):

```
df$Outcome <- as.factor(df$Outcome)

levels(df$Outcome) <- c("No","Yes")
```

Summarize variables:

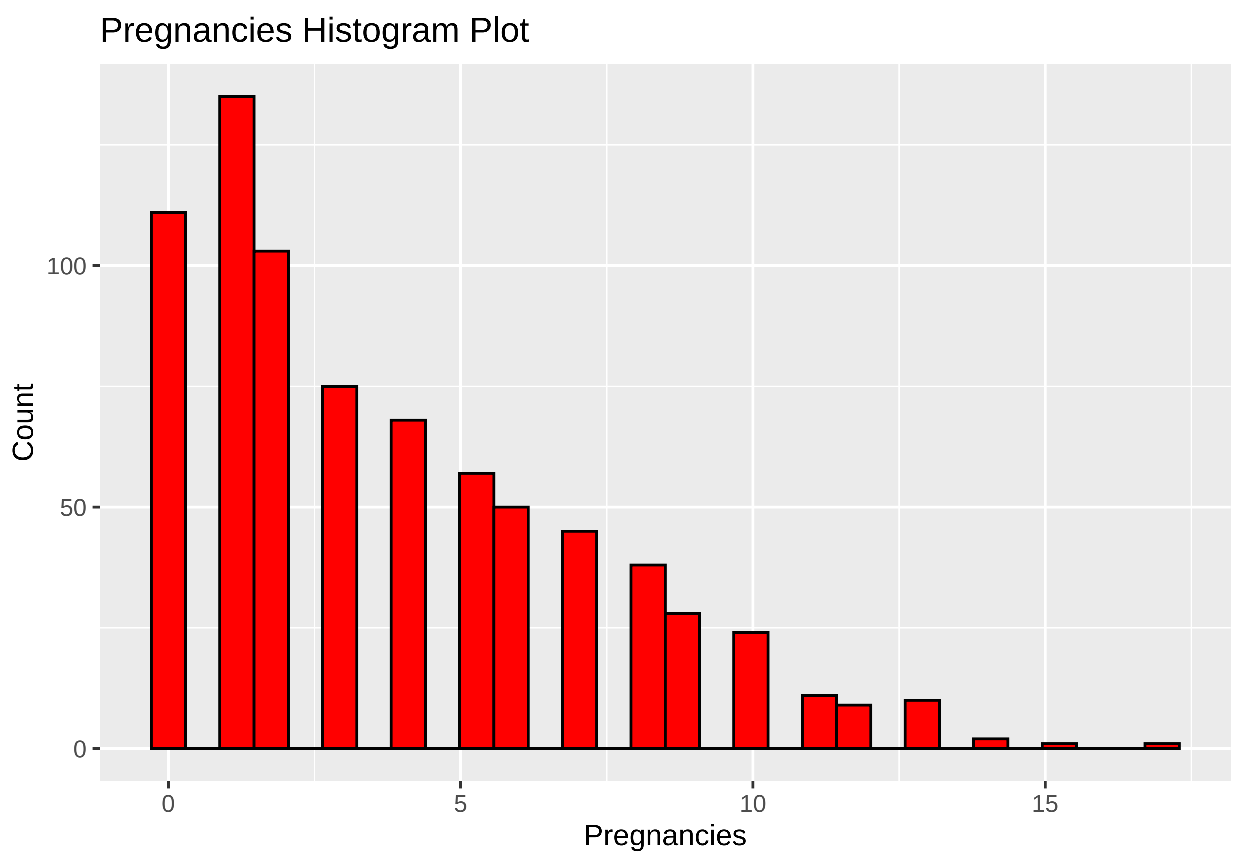
```
df %>% summary()
```

```
##   Pregnancies        Glucose      BloodPressure    SkinThickness
##  Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00
##  1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00
##  Median : 3.000   Median :117.0   Median : 72.00   Median :23.00
##  Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54
##  3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00
##  Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
##     Insulin           BMI        DiabetesPedigreeFunction      Age
##  Min.   :  0.0   Min.   : 0.00   Min.   :0.0780           Min.   :21.00
##  1st Qu.:  0.0   1st Qu.:27.30   1st Qu.:0.2437           1st Qu.:24.00
##  Median : 30.5   Median :32.00   Median :0.3725           Median :29.00
##  Mean   : 79.8   Mean   :31.99   Mean   :0.4719           Mean   :33.24
##  3rd Qu.:127.2   3rd Qu.:36.60   3rd Qu.:0.6262           3rd Qu.:41.00
##  Max.   :846.0   Max.   :67.10   Max.   :2.4200           Max.   :81.00
##  Outcome
##  No :500
##  Yes:268
##
##
##
##
```

Plots:

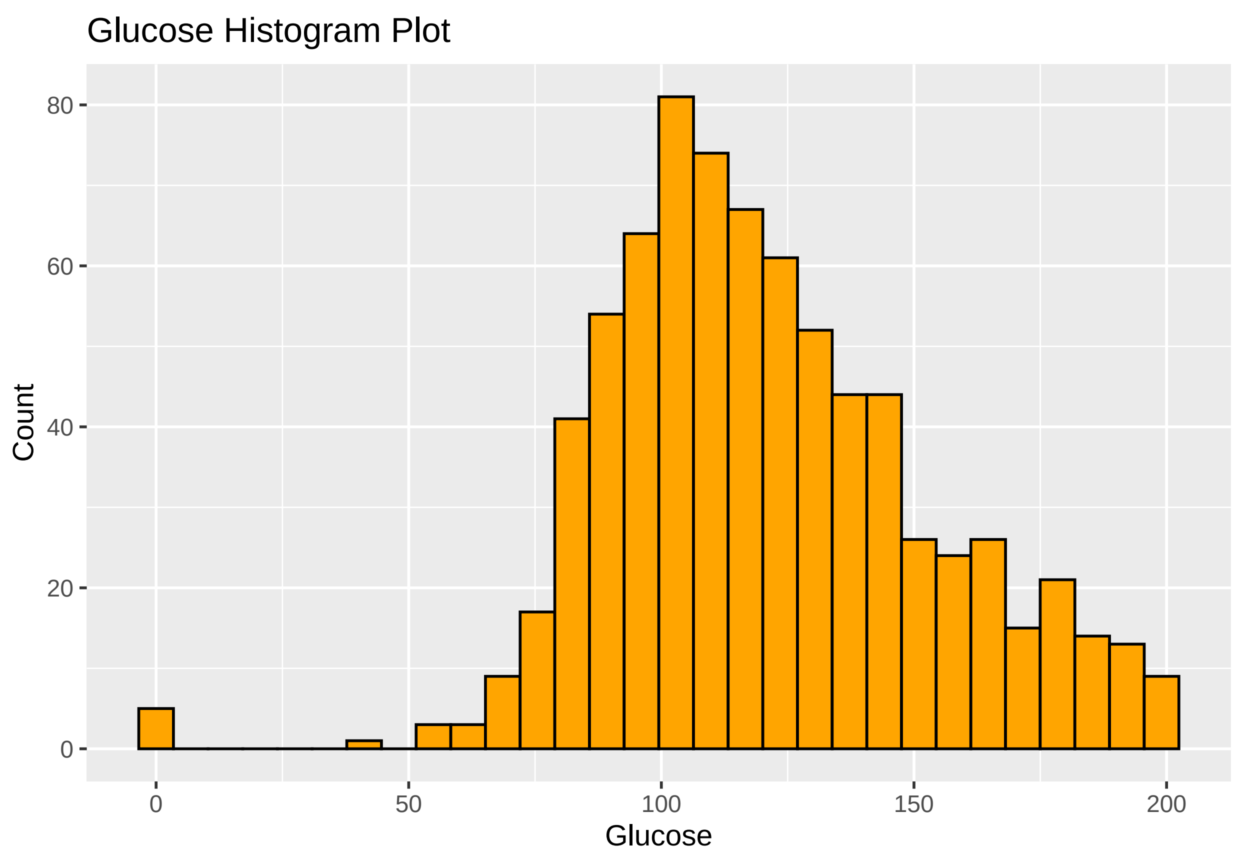
```
plotpreg <- ggplot(data = df, aes(x = Pregnancies)) +
  geom_histogram(color = "black", fill = "red") +
  labs(title = "Pregnancies Histogram Plot", x = "Pregnancies", y = "Count")
plotpreg
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



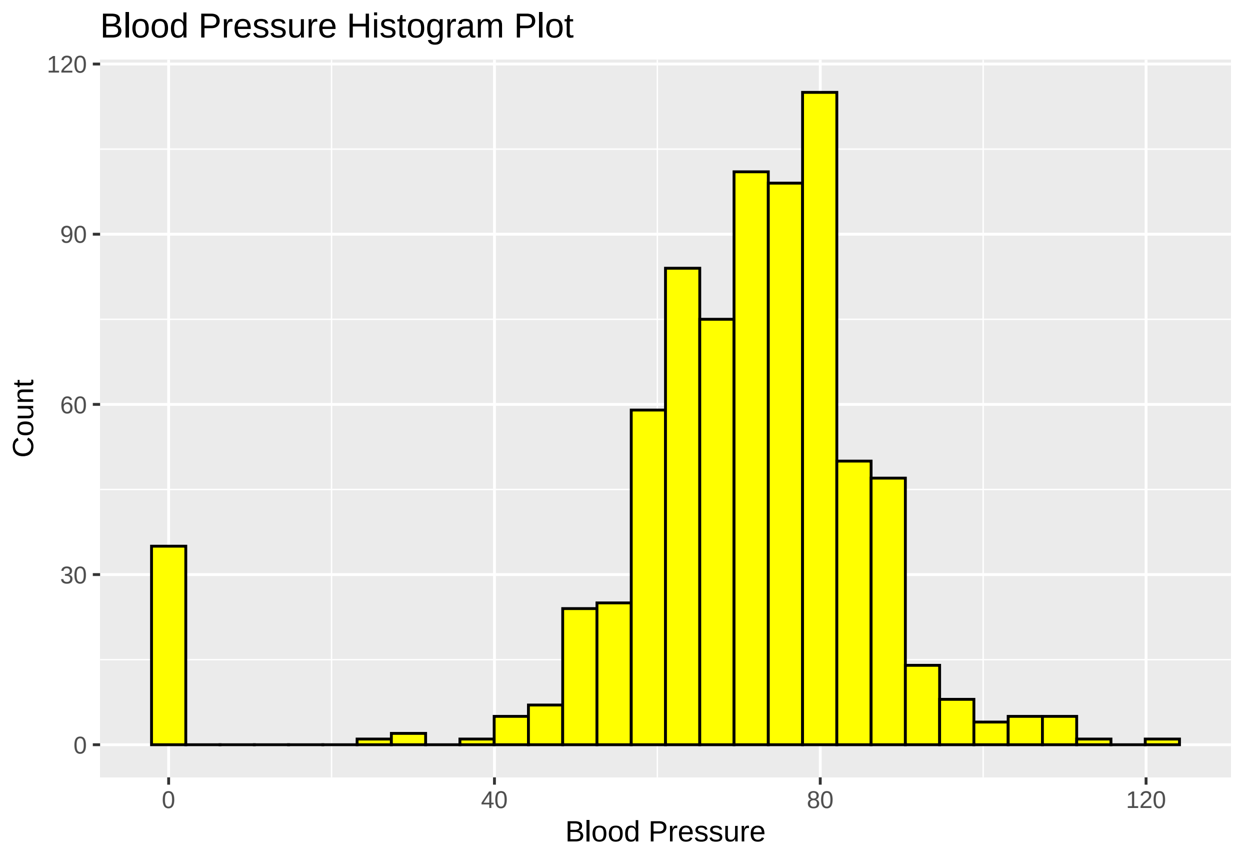
```r
plotgluc <- ggplot(data = df, aes(x = Glucose)) +
  geom_histogram(color = "black", fill = "orange") +
  labs(title = "Glucose Histogram Plot", x = "Glucose", y = "Count")
plotgluc
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



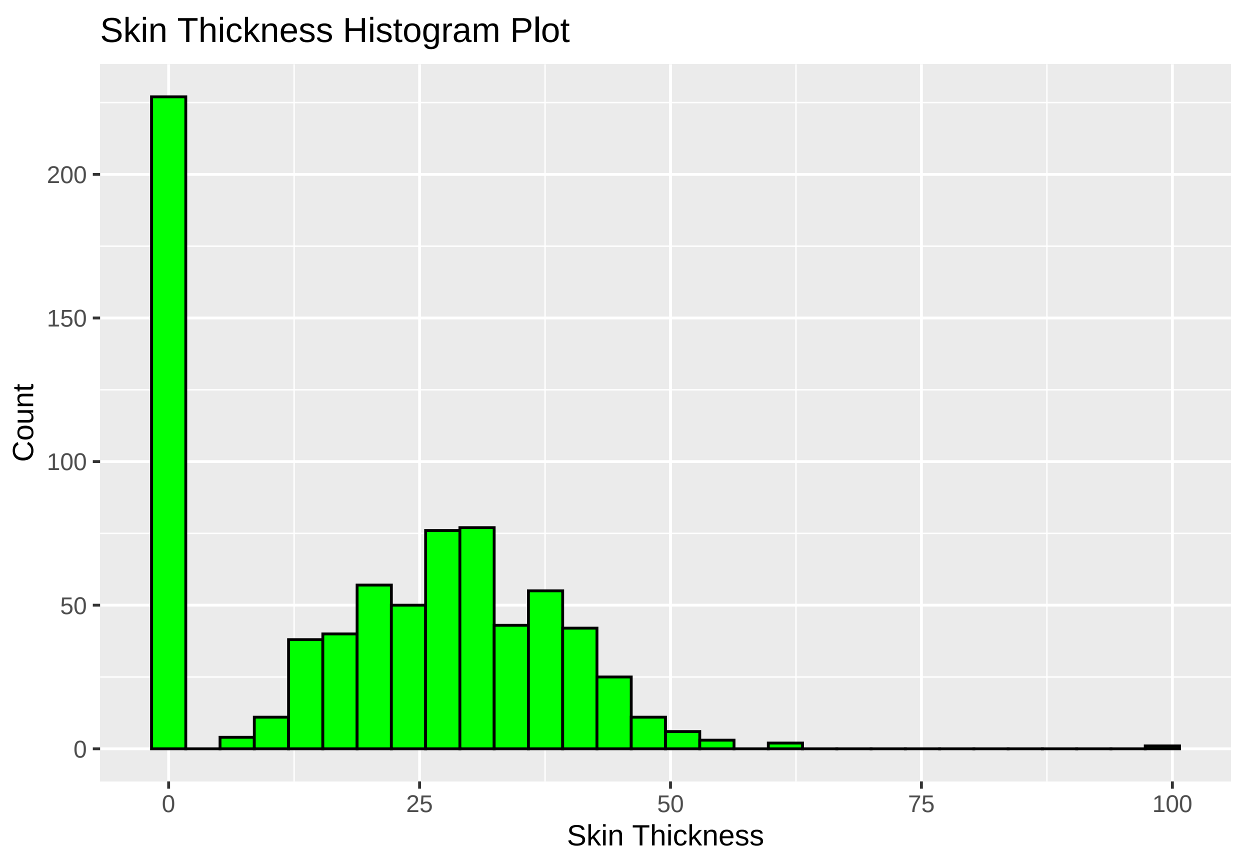
```
plotbp <- ggplot(data = df, aes(x = BloodPressure)) +
  geom_histogram(color = "black", fill = "yellow") +
  labs(title = "Blood Pressure Histogram Plot", x = "Blood Pressure", y = "Count")
plotbp
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



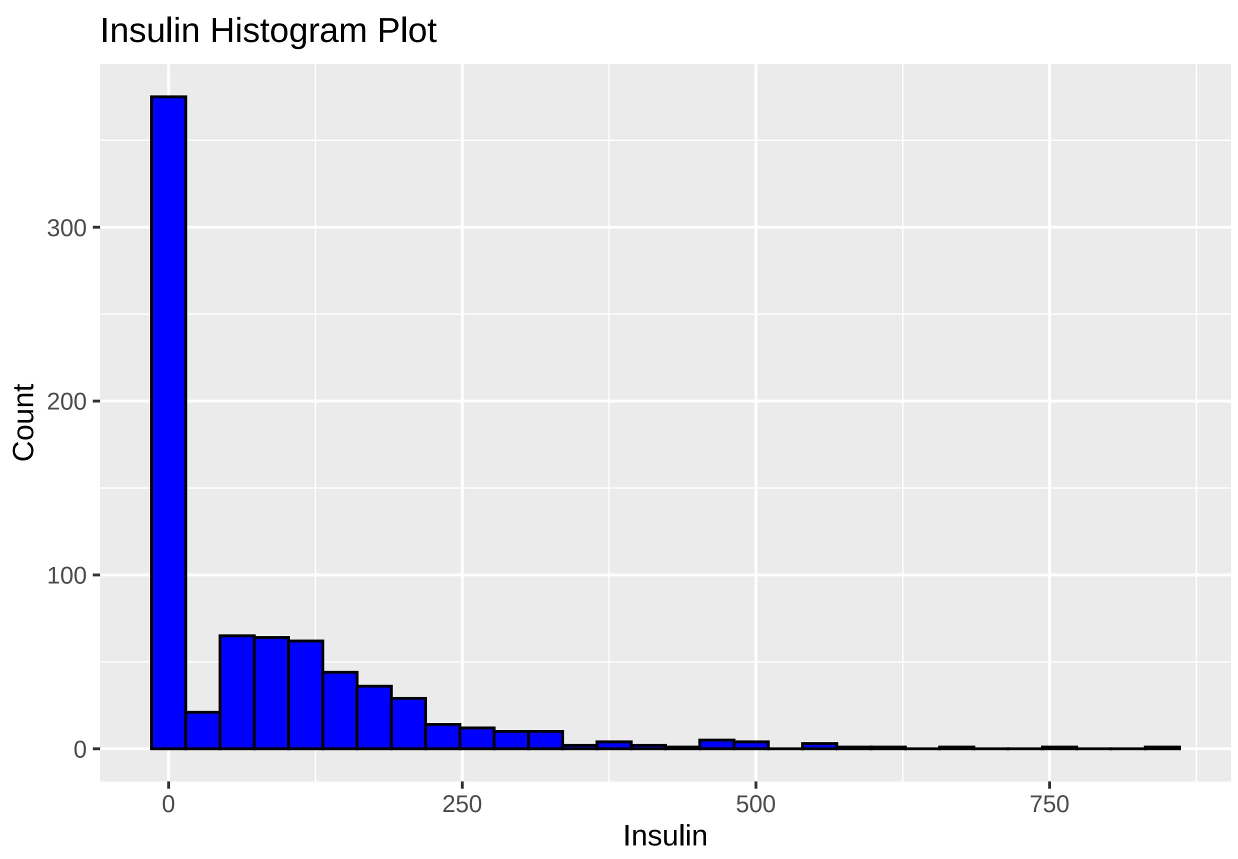
```r
plotst <- ggplot(data = df, aes(x = SkinThickness)) +
  geom_histogram(color = "black", fill = "green") +
  labs(title = "Skin Thickness Histogram Plot", x = "Skin Thickness", y = "Count")
plotst
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



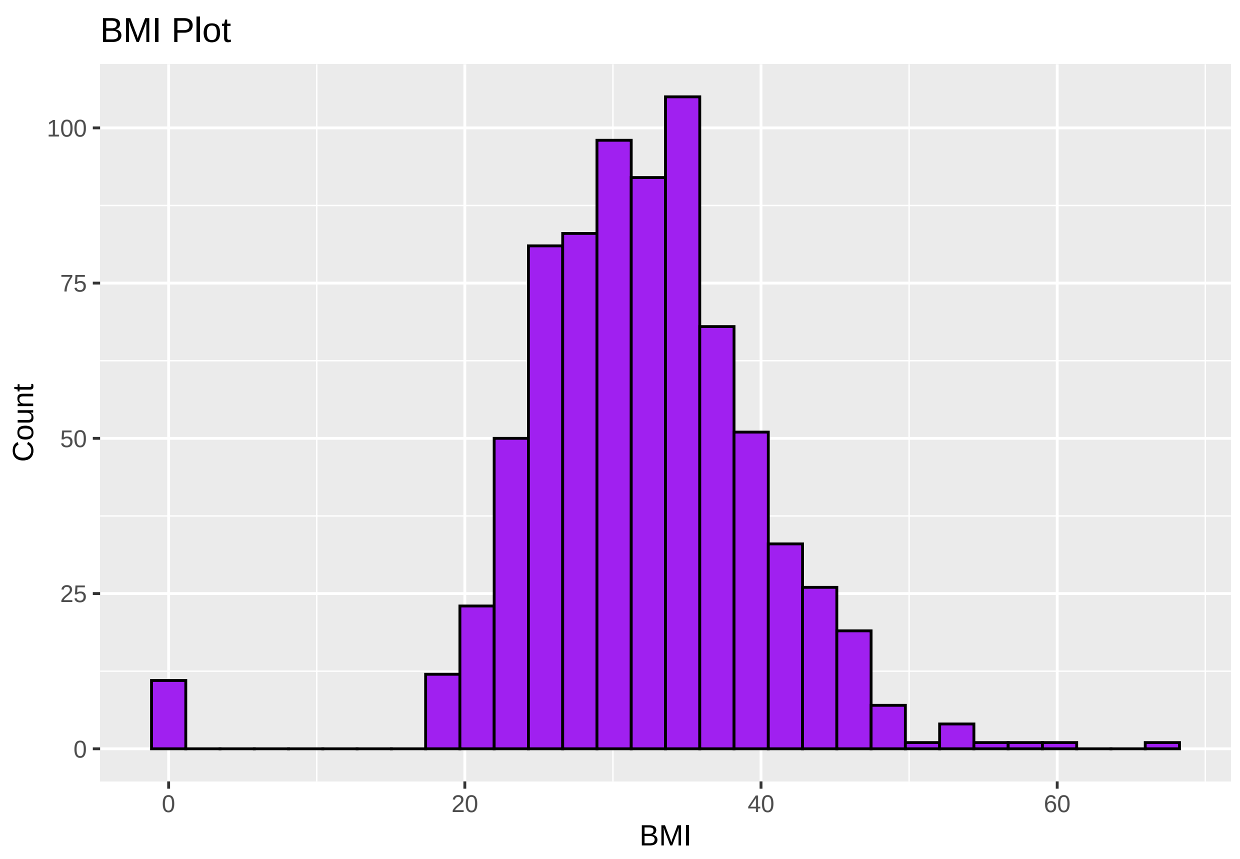
```
plotins <- ggplot(data = df, aes(x = Insulin)) +
  geom_histogram(color = "black", fill = "blue") +
  labs(title = "Insulin Histogram Plot", x = "Insulin", y = "Count")
plotins
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



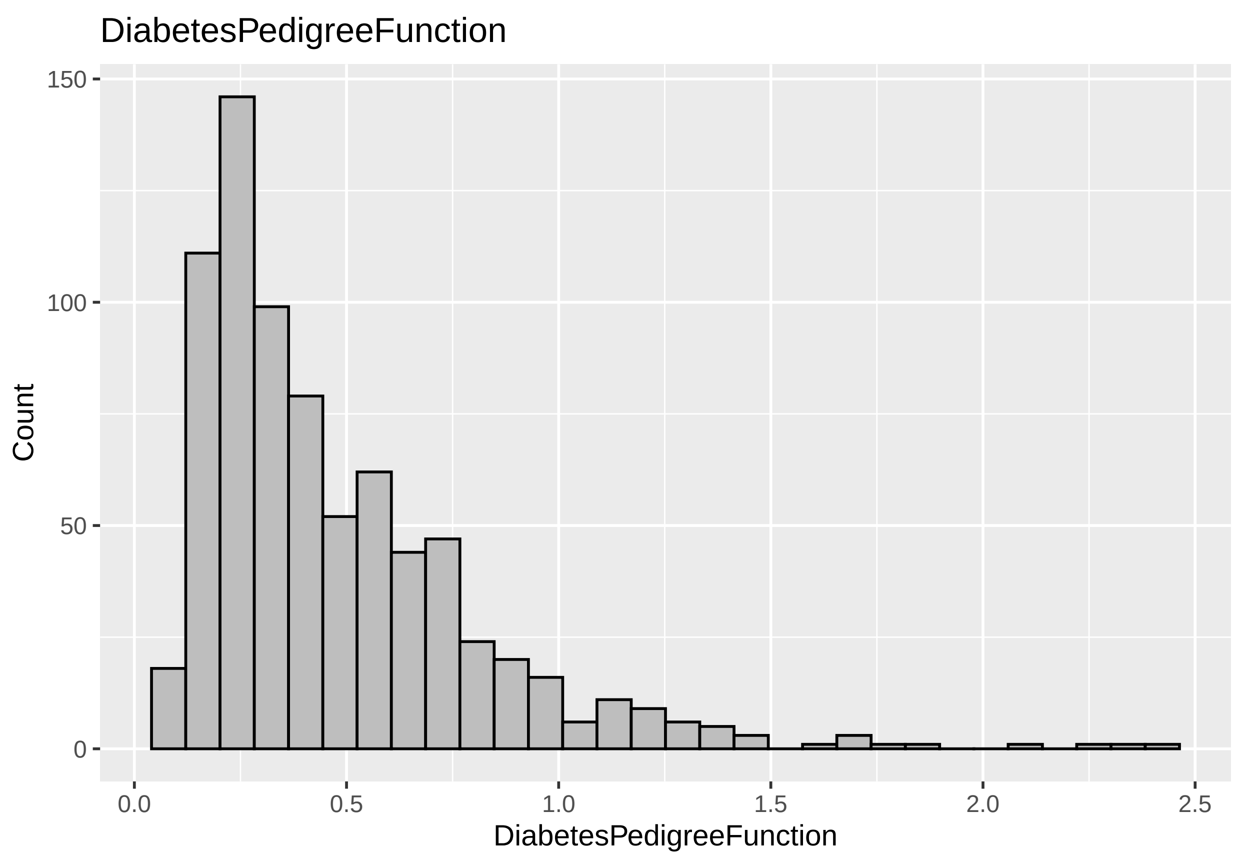
```r
plotbmi <- ggplot(data = df, aes(x = BMI)) +
  geom_histogram(color = "black", fill = "purple") +
  labs(title = "BMI Plot", x = "BMI", y = "Count")
plotbmi
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



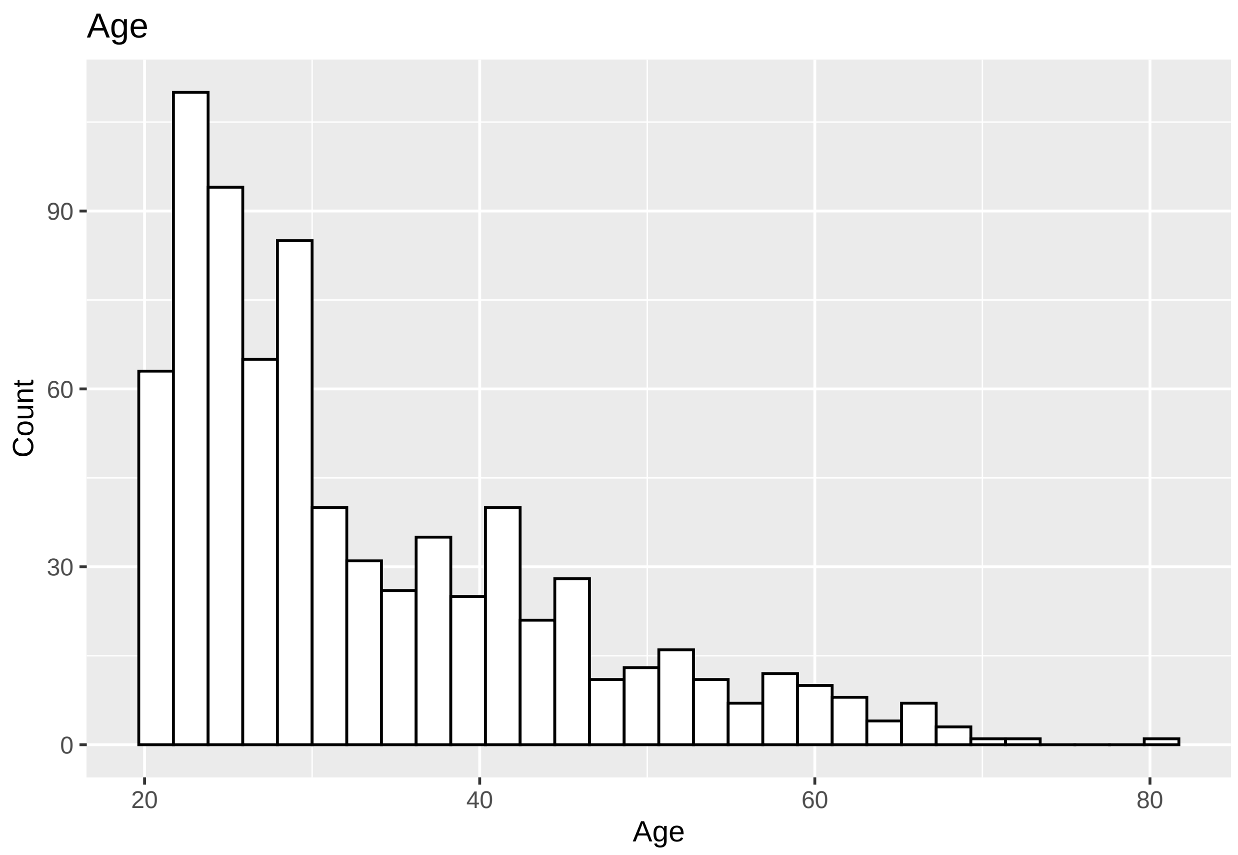
```
plotdpf <- ggplot(data = df, aes(x = DiabetesPedigreeFunction)) +
  geom_histogram(color = "black", fill = "grey") +
  labs(title = "DiabetesPedigreeFunction", x = "DiabetesPedigreeFunction", y = "Count")
plotdpf
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```

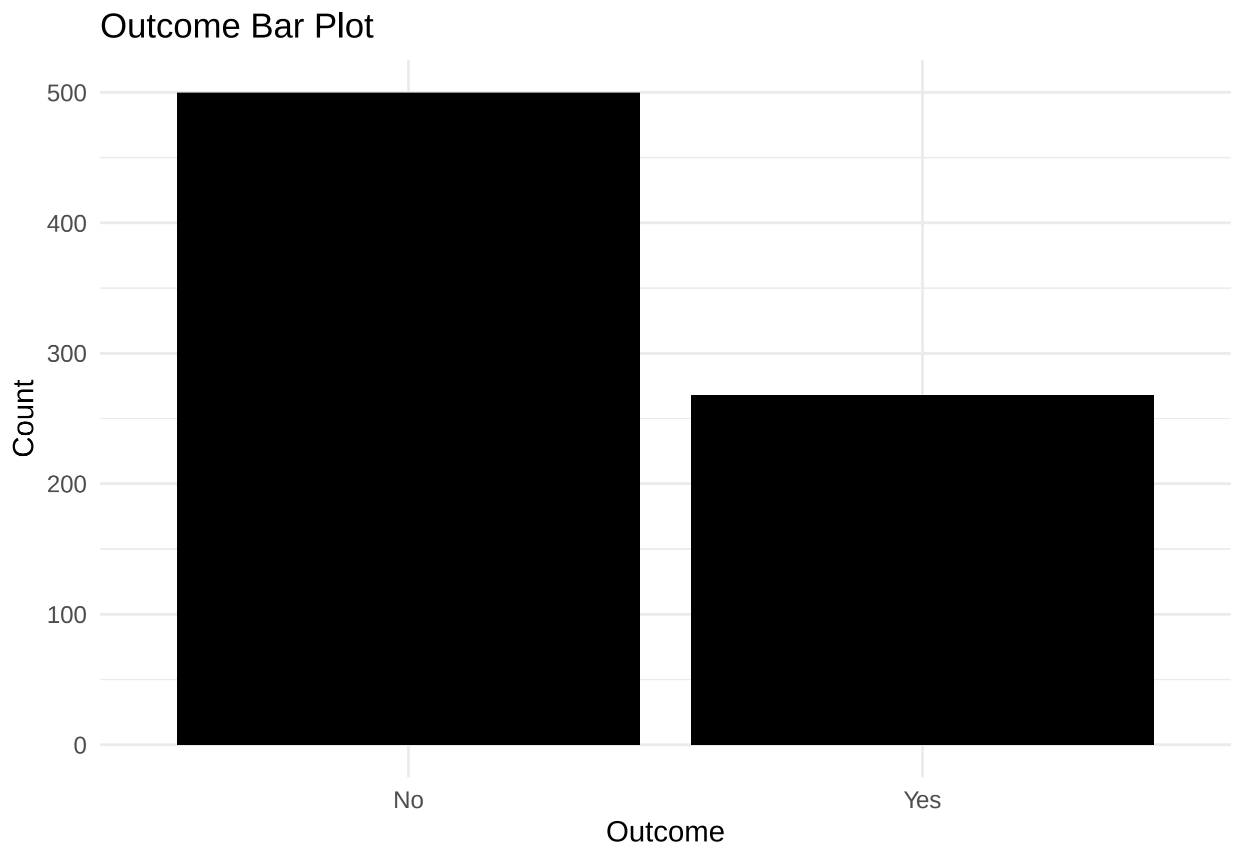


```
plotage <- ggplot(data = df, aes(x = Age)) +
  geom_histogram(color = "black", fill = "white") +
  labs(title = "Age", x = "Age", y = "Count")
plotage
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```

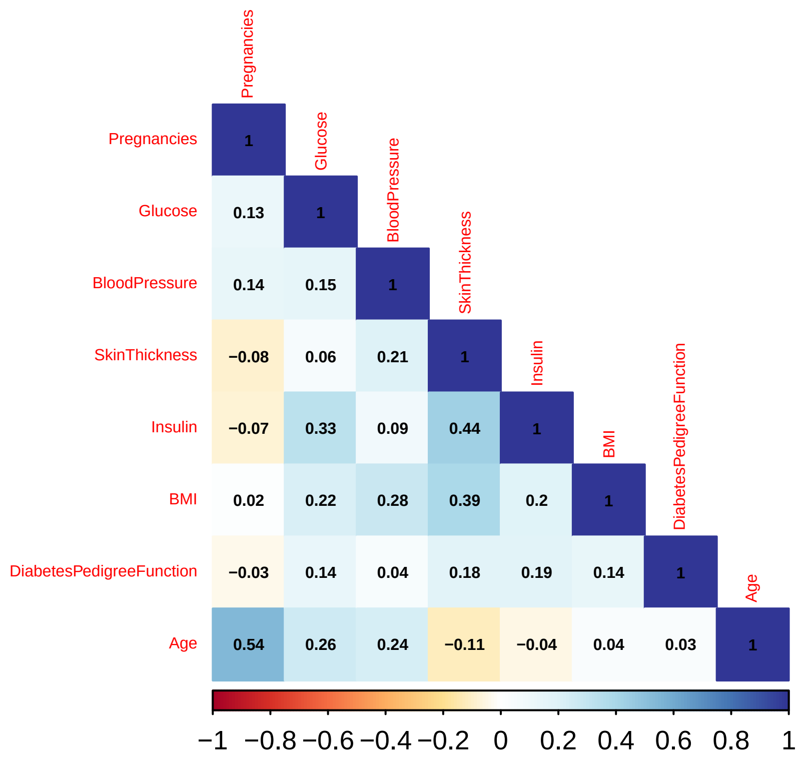


```
plotout <- ggplot(data = df, aes(x = Outcome)) +
  stat_count(fill = "black") +
  labs(title = "Outcome Bar Plot", x = "Outcome", y = "Count") +
  theme_minimal()
plotout
```



Correlation of Variables:

```
df_corr <- df[-9]
df_corr <- cor(df_corr)
corrplot(df_corr, method = "color", type = "lower",
         addCoef.col = "black", col = COL2("RdYlBu"), number.cex = .5, tl.cex = .5)
```



There are moderately positive correlations between the Age and Pregnancy, and the Insulin and Skin Thickness attributes. This indicates that as the age of the patients increased so did the number of pregnancies, also as the quantity of insulin administered to the patients increased; the skin thickness increased likewise.

Weak positive correlations can also be observed in the following attributes of the dataset; Insulin & Glucose, BMI & Skin Thickness, Blood Pressure & BMI, Age & Blood Pressure…

Time to Train-Test-Split:

```
nrows <- NROW(df)

set.seed(42)

index <- sample(1:nrows, 0.7 * nrows)

train <- df[index,]

test <- df[-index,]
```

Random Forest Classifier:

```
learn_rf <- randomForest(Outcome~., data=train, ntree=500, proximity=T, importance=T)

pre_rf   <- predict(learn_rf, test[,-9])

cm_rf    <- confusionMatrix(pre_rf, test$Outcome)
```

```r
cm_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  134  28
##        Yes  22  47
##
##                Accuracy : 0.7835
##                  95% CI : (0.7248, 0.8349)
##     No Information Rate : 0.6753
##     P-Value [Acc > NIR] : 0.0001885
##
##                   Kappa : 0.4959
##
##  Mcnemar's Test P-Value : 0.4795001
##
##             Sensitivity : 0.8590
##             Specificity : 0.6267
##          Pos Pred Value : 0.8272
##          Neg Pred Value : 0.6812
##              Prevalence : 0.6753
##          Detection Rate : 0.5801
##    Detection Prevalence : 0.7013
##       Balanced Accuracy : 0.7428
##
##        'Positive' Class : No
##
```

CTree Classifier:

```r
learn_ct <- ctree(Outcome~., data=train, controls=ctree_control(maxdepth=2))

pre_ct   <- predict(learn_ct, test[,-9])

cm_ct    <- confusionMatrix(pre_ct, test$Outcome)

cm_ct
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  132  36
##        Yes  24  39
##
##                Accuracy : 0.7403
##                  95% CI : (0.6787, 0.7956)
##     No Information Rate : 0.6753
##     P-Value [Acc > NIR] : 0.01935
##
```

```
##                   Kappa : 0.382
##
##  Mcnemar's Test P-Value : 0.15558
##
##             Sensitivity : 0.8462
##             Specificity : 0.5200
##          Pos Pred Value : 0.7857
##          Neg Pred Value : 0.6190
##              Prevalence : 0.6753
##          Detection Rate : 0.5714
##    Detection Prevalence : 0.7273
##       Balanced Accuracy : 0.6831
##
##        'Positive' Class : No
##
```

Naive Bayes Classifier:

```
learn_nb <- naiveBayes(train[,-9], train$Outcome)

pre_nb <- predict(learn_nb, test[,-9])

cm_nb <- confusionMatrix(pre_nb, test$Outcome)

cm_nb
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  125  27
##        Yes  31  48
##
##                Accuracy : 0.7489
##                  95% CI : (0.6878, 0.8035)
##     No Information Rate : 0.6753
##     P-Value [Acc > NIR] : 0.009121
##
##                   Kappa : 0.4353
##
##  Mcnemar's Test P-Value : 0.693641
##
##             Sensitivity : 0.8013
##             Specificity : 0.6400
##          Pos Pred Value : 0.8224
##          Neg Pred Value : 0.6076
##              Prevalence : 0.6753
##          Detection Rate : 0.5411
##    Detection Prevalence : 0.6580
##       Balanced Accuracy : 0.7206
##
##        'Positive' Class : No
##
```

The Random Forest Classifier performed the best with a 78% accuracy, 85.9% Sensitivity, and 63% Specificity.

Let's see how our model does on two test subjects. A yes and a no diabetes:

```
Y <- test[1,]
N <- test[2,]

print(Y)
```

```
## # A tibble: 1 x 9
##   Pregnancies Glucose BloodPressure SkinTh~1 Insulin   BMI Diabe~2   Age Outcome
##         <dbl>   <dbl>         <dbl>    <dbl>   <dbl> <dbl>   <dbl> <dbl> <fct>
## 1           6     148            72       35       0  33.6   0.627    50 Yes
## # ... with abbreviated variable names 1: SkinThickness,
## #   2: DiabetesPedigreeFunction
```

```
print(N)
```

```
## # A tibble: 1 x 9
##   Pregnancies Glucose BloodPressure SkinTh~1 Insulin   BMI Diabe~2   Age Outcome
##         <dbl>   <dbl>         <dbl>    <dbl>   <dbl> <dbl>   <dbl> <dbl> <fct>
## 1           1      89            66       23      94  28.1   0.167    21 No
## # ... with abbreviated variable names 1: SkinThickness,
## #   2: DiabetesPedigreeFunction
```

Remove the outcome variable so it isn't there to mess with the model prediciton:

```
Y$Outcome <- NULL

N$Outcome <- NULL
```

Function that takes in a patient's variables and predicts if they have diabetes or not:

```
patient_diabetes_predict <- function(new, method=learn_rf) {

    new_pre <- predict(method, new)

    new_res <- as.character(new_pre)

    return(paste("Result: ", new_res, sep=""))

}
```

Yes patient's prediction:

```
patient_diabetes_predict(Y)
```

```
## [1] "Result: Yes"
```

No patient's prediction:

```
patient_diabetes_predict(N)
```

```
## [1] "Result: No"
```