test

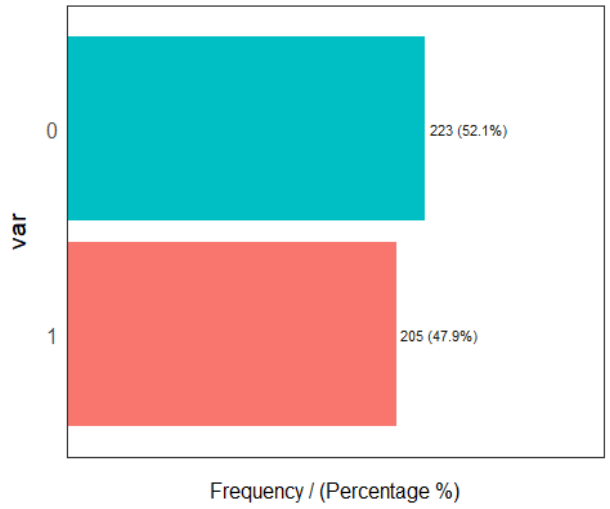
```
basic_eda <- function(data)
{
  glimpse(data)
  print(status(data))
  freq(data)
  print(profiling_num(data))
  plot_num(data)
  describe(data)
}

status(train_data)
```

```
##                    variable q_zeros     p_zeros q_na p_na q_inf p_inf
    type
## 1               Pregnancies      64 0.149532710    0    0     0     0
 integer
## 2                   Glucose       3 0.007009346    0    0     0     0
 integer
## 3             BloodPressure      19 0.044392523    0    0     0     0
 integer
## 4             SkinThickness     139 0.324766355    0    0     0     0
 integer
## 5                   Insulin     218 0.509345794    0    0     0     0
 integer
## 6                       BMI       6 0.014018692    0    0     0     0
 numeric
## 7 DiabetesPedigreeFunction       0 0.000000000    0    0     0     0
 numeric
## 8                       Age       0 0.000000000    0    0     0     0
 integer
## 9                   Outcome     223 0.521028037    0    0     0     0
 integer
##   unique
## 1     17
## 2    124
## 3     41
## 4     48
## 5    130
## 6    200
## 7    335
## 8     49
## 9      2
```

```
#From the table shown, we can conclude the percertanges of zeros, N/A,
inf. And the table shows that there are 223 0s, which correspond
#52.1% of total outcome. The zeros represents the cases of not having d
```

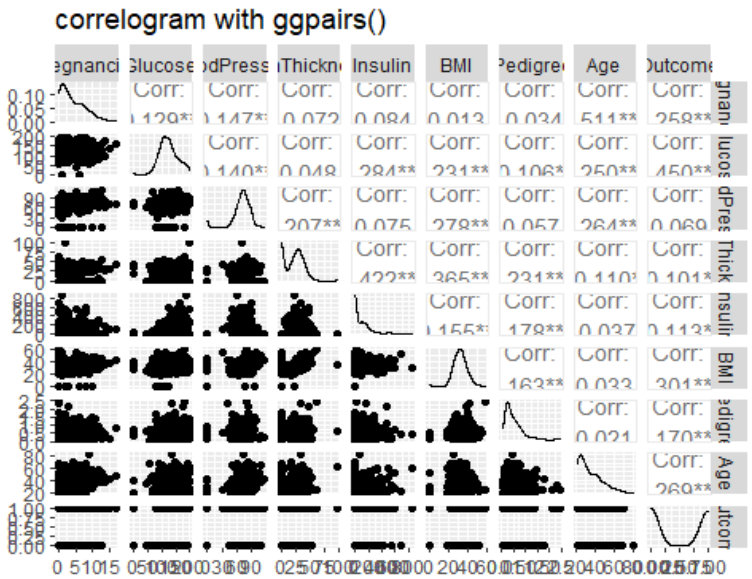
```
iabetes
freq(train_data$Outcome)#To view the percertage of the outcomes
```




```
##   var frequency percentage cumulative_perc
## 1   0       223       52.1            52.1
## 2   1       205       47.9           100.0

train_data_prof=profiling_num(train_data)
#Check the mean, std_dev of each variable
#Check the relationship between each pair of two variables.
df = train_data[, c("Pregnancies", "Glucose",
                    "BloodPressure", "SkinThickness", "Insulin", "BMI",

                    "DiabetesPedigreeFunction", "Age", "Outcome")]
ggpairs(df, title="correlogram with ggpairs()")
```



```r
#Check the correlations between each pair of two variables.

x_train = train_data[,1:8]
y_train = train_data$Outcome
x_test = test_data[,1:8]
y_test = test_data$Outcome

scaled_x_train = scale(x_train)
scaled_x_test = scale(x_test)

k = 1:20
test_error <- rep(x = 0, times = length(k))
train_error <- rep(x = 0, times = length(k))

for (i in seq_along(k)) {
  pred <- knn(train = scaled_x_train, test  = scaled_x_test, cl = y_tra
in, k = k[i])
  test_error[i] <- mean(y_test != pred)
}

for (j in seq_along(k)){
  pred <- knn(train=scaled_x_train, test = scaled_x_train, cl = y_train,
 k = k[j])
  train_error[j] <- mean(y_train != pred)
}

matplot(cbind(test_error,train_error),type="b",col=c("red","green"),
        lend = par("lend"), ylab = "Error", xlab = "Model Complexity (1
/K)",
        main = "Training and Testing Error")
```