

# Project

Group 7

2023-05-01

## Importing the dataset from local

```
setwd("/Users/manaswinidasari/Downloads")
```

```
# Import CSV file
```

```
data <- read.csv("UA.csv")
```

```
head(data)
```

```
##   Gender  Age Height Weight      BMI  L1.4 L1.4T      FN  FNT      TL  TLT
ALT
## 1      2 61.9   164     47 17.47472 0.894  -2.4 0.6895 -2.95 0.7130 -2.90
17.0
## 2      2 55.0   162     54 20.57613 1.333   1.3 0.9130 -1.30 1.0675 -0.15
34.0
## 3      2 44.0   160     54 21.09375 1.157  -0.2 0.5190 -3.85 0.5770 -3.55
11.0
## 4      1 64.7   158     59 23.63403 0.948  -2.3 0.7920 -2.15 0.9050 -1.40
4.1
## 5      1 88.5   167     60 21.51386 1.114  -0.9 0.8250 -1.90 0.9385 -1.15
32.0
## 6      1 73.0   169     65 22.75831 0.956  -2.2 0.6965 -2.35 0.7550 -2.05
17.0
##   AST   BUN   CREA  URIC  FBG HDL.C LDL.C  Ca   P   Mg Calcium Calcitriol
## 1  21   8.10  84.8 265.4 4.55  1.16  2.06 2.15 1.04 0.76          1          1
## 2  25   7.64  85.0 466.0 3.89  1.22  2.35 2.15 1.11 0.96          1          0
## 3  21   5.14  76.2 479.1 4.66  1.54  1.65 2.18 1.10 0.89          1          1
## 4  28  12.99 104.0 473.1 6.84  0.93  1.39 2.35 0.87 0.78          0          0
## 5  19   4.10  72.4 429.8 5.07  1.25  1.85 1.98 0.93 0.86          1          1
## 6  23   5.09  81.6 366.1 6.65  1.16  3.13 2.43 1.09 0.78          1          1
##   Bisphosphonate Calcitonin HTN COPD DM Hyperlipidaemia Hyperuricemia AS
VT VD
## 1              0              0  1  1  0              0              1  1
0 1
## 2              0              0  1  0  1              0              1  1
0 1
## 3              0              1  1  1  1              0              1  1
0 1
## 4              0              1  1  1  1              0              1  1
0 1
## 5              0              0  1  0  1              0              1  1
```

```

0 1
## 6          1          1 1 1 1          0          1 1
0 1
##   OP CAD CKD Fracture Smoking Drinking
## 1 1  1  0      0      1      0
## 2 1  0  1      0      0      0
## 3 1  1  0      0      0      0
## 4 1  1  0      0      1      1
## 5 1  1  0      0      1      0
## 6 1  0  1      0      0      0

```

## Selecting columns of dataframe

```

# Select columns "A" and "C"
data <- data[, c("Age", "Gender", "BMI", "Ca", "P", "Mg", "OP", "Smoking",
"Drinking")]
head(data)

```

```

##   Age Gender      BMI   Ca    P   Mg OP Smoking Drinking
## 1 61.9      2 17.47472 2.15 1.04 0.76 1      1      0
## 2 55.0      2 20.57613 2.15 1.11 0.96 1      0      0
## 3 44.0      2 21.09375 2.18 1.10 0.89 1      0      0
## 4 64.7      1 23.63403 2.35 0.87 0.78 1      1      1
## 5 88.5      1 21.51386 1.98 0.93 0.86 1      1      0
## 6 73.0      1 22.75831 2.43 1.09 0.78 1      0      0

```

## Structure of dataframe

```

str(data)

## 'data.frame':    1537 obs. of  9 variables:
## $ Age      : num  61.9 55 44 64.7 88.5 73 61.5 52.9 53.1 86 ...
## $ Gender   : int   2 2 2 1 1 1 1 2 2 2 ...
## $ BMI      : num   17.5 20.6 21.1 23.6 21.5 ...
## $ Ca       : num   2.15 2.15 2.18 2.35 1.98 2.43 2.74 2.08 1.91 2.21 ...
## $ P        : num   1.04 1.11 1.1 0.87 0.93 1.09 0.76 0.94 0.93 0.94 ...
## $ Mg       : num   0.76 0.96 0.89 0.78 0.86 0.78 0.92 0.87 0.84 0.87 ...
## $ OP       : int    1 1 1 1 1 1 1 1 1 1 ...
## $ Smoking  : int    1 0 0 1 1 0 0 0 1 1 ...
## $ Drinking : int    0 0 0 1 0 0 0 0 1 1 ...

```

## Summary of the data

```

summary(data)

##      Age      Gender      BMI      Ca
## Min.   :28.60   Min.   :1.000   Min.   : 9.213   Min.   :1.780
## 1st Qu.:51.00   1st Qu.:1.000   1st Qu.:22.066   1st Qu.:2.160
## Median :57.00   Median :1.000   Median :24.219   Median :2.230

```

```
## Mean :59.85 Mean :1.385 Mean :24.313 Mean :2.238
## 3rd Qu.:67.30 3rd Qu.:2.000 3rd Qu.:26.356 3rd Qu.:2.310
## Max. :99.80 Max. :2.000 Max. :37.261 Max. :5.840
## NA's :36 NA's :34 NA's :2
## P Mg OP Smoking
## Min. :0.56 Min. :0.0970 Min. :0.0000 Min. :0.0000
## 1st Qu.:0.92 1st Qu.:0.8100 1st Qu.:0.0000 1st Qu.:0.0000
## Median :1.02 Median :0.8700 Median :0.0000 Median :0.0000
## Mean :1.04 Mean :0.8686 Mean :0.3696 Mean :0.2576
## 3rd Qu.:1.13 3rd Qu.:0.9300 3rd Qu.:1.0000 3rd Qu.:1.0000
## Max. :4.41 Max. :1.7300 Max. :1.0000 Max. :1.0000
## NA's :5 NA's :3
## Drinking
## Min. :0.0000
## 1st Qu.:0.0000
## Median :0.0000
## Mean :0.2277
## 3rd Qu.:0.0000
## Max. :1.0000
##
```

## Check for null values

```
# Check for null values
sum(is.na(data))

## [1] 80
```

## Replacing null values with mean values of the column

#It will not disturb overall structure of the data.

```
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

# Replace null values with mean value of each column in df: data
data <- select(data, -ncol(data)) %>%
  mutate_all(~ifelse(is.na(.), mean(., na.rm = TRUE), .)) %>%
  bind_cols(select(data, ncol(data)) %>% rename(Drinking = everything()))
```

```
sum(is.na(data))
```

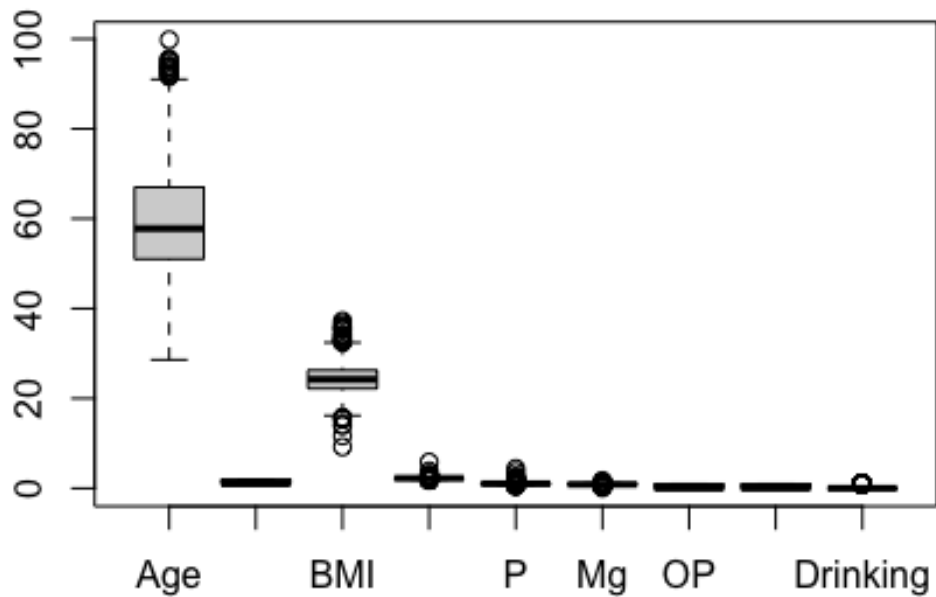
```
## [1] 0
```

```
head(data)
```

```
##   Age Gender    BMI   Ca    P   Mg OP Smoking Drinking
## 1 61.9     2 17.47472 2.15 1.04 0.76 1     1         0
## 2 55.0     2 20.57613 2.15 1.11 0.96 1     0         0
## 3 44.0     2 21.09375 2.18 1.10 0.89 1     0         0
## 4 64.7     1 23.63403 2.35 0.87 0.78 1     1         1
## 5 88.5     1 21.51386 1.98 0.93 0.86 1     1         0
## 6 73.0     1 22.75831 2.43 1.09 0.78 1     0         0
```

## Checking the outliers with Boxplot

```
boxplot(data, outline = TRUE)
```



## Print the outliers

```
# Load the required packages
```

```
library(dplyr)
```

```

# Calculate the number of outliers in df
outliers <- apply(data, 2, function(x) {
  qnt <- quantile(x, probs=c(.25, .75), na.rm = TRUE)
  H <- 1.5 * IQR(x, na.rm = TRUE)
  sum(x < (qnt[1] - H) | x > (qnt[2] + H), na.rm = TRUE)
})

# Print the number of outliers for each column
print(outliers)

##      Age  Gender    BMI    Ca    P    Mg    OP  Smoking
##      16      0     30    32   42    28     0         0
## Drinking
##      350

```

## Function to replace outliers with median values

```

# Define a function to replace outliers with the median value of a vector
replace_outliers <- function(x) {
  q1 <- quantile(x, 0.25, na.rm = TRUE)
  q3 <- quantile(x, 0.75, na.rm = TRUE)
  iqr <- q3 - q1
  upper <- q3 + 1.5 * iqr
  lower <- q1 - 1.5 * iqr
  replace(x, x > upper | x < lower, median(x, na.rm = TRUE))
}

# Replace outlier values with median value of each column in df
data <- select(data, -last_col()) %>%
  mutate_all(replace_outliers) %>%
  bind_cols(select(data, last_col()) %>% rename_with(~"Drinking", .cols =
everything()))

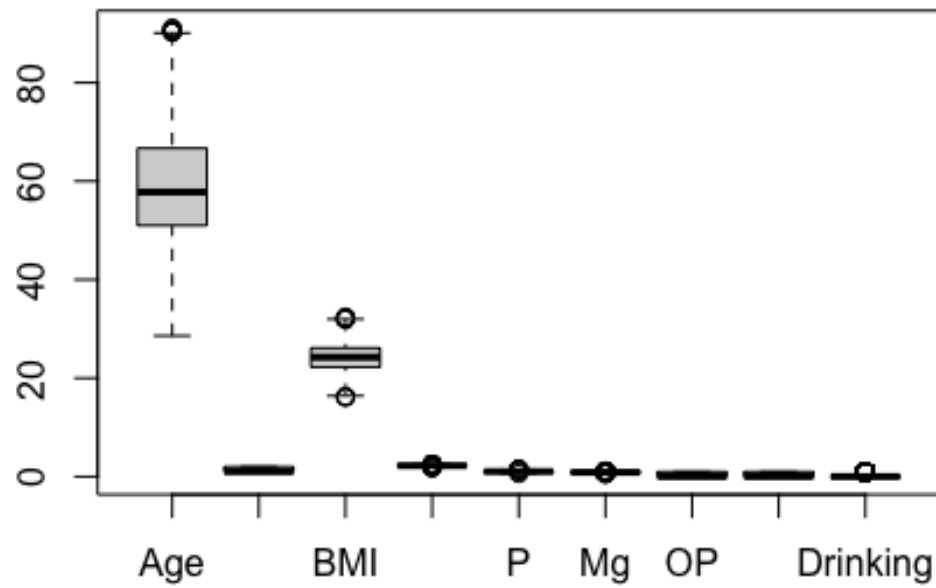
```

## Checking outliers after cleaning

```

boxplot(data, outline = TRUE)

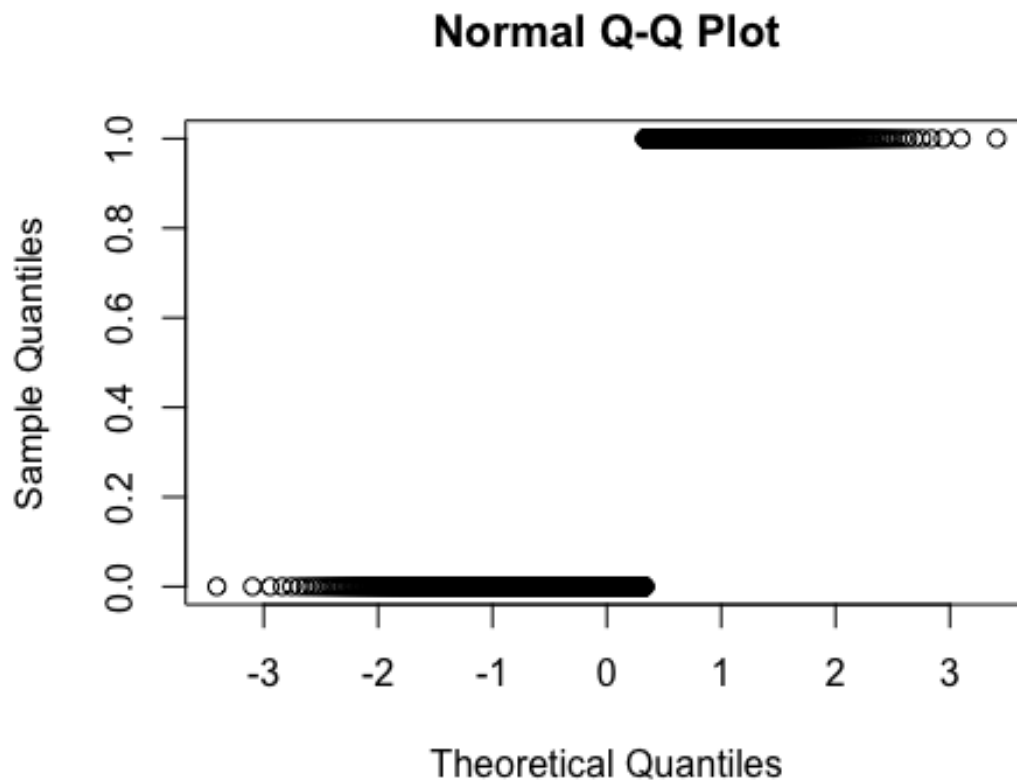
```



## Q-Q Plot

#to check normality

```
qqnorm(data$OP)
```



#the distribution of the data is approximately normal. The straight line means that the quantiles of the dataset are linearly related to the quantiles of the standard normal distribution.

#shapiro-wilk test #to check normality for each column

```
library(dplyr)
data <- select_if(data, is.numeric)
# Perform Shapiro-Wilk test on each column
for (col in names(data)) {
  test_result <- shapiro.test(data[[col]])
  print(paste("Shapiro-Wilk test for column", col))
  print(test_result)
}

## [1] "Shapiro-Wilk test for column Age"
##
##  Shapiro-Wilk normality test
##
## data:  data[[col]]
## W = 0.97815, p-value = 1.545e-14
##
## [1] "Shapiro-Wilk test for column Gender"
##
##  Shapiro-Wilk normality test
```

```
##
## data: data[[col]]
## W = 0.61714, p-value < 2.2e-16
##
## [1] "Shapiro-Wilk test for column BMI"
##
## Shapiro-Wilk normality test
##
## data: data[[col]]
## W = 0.99699, p-value = 0.004661
##
## [1] "Shapiro-Wilk test for column Ca"
##
## Shapiro-Wilk normality test
##
## data: data[[col]]
## W = 0.99595, p-value = 0.0004033
##
## [1] "Shapiro-Wilk test for column P"
##
## Shapiro-Wilk normality test
##
## data: data[[col]]
## W = 0.99329, p-value = 1.916e-06
##
## [1] "Shapiro-Wilk test for column Mg"
##
## Shapiro-Wilk normality test
##
## data: data[[col]]
## W = 0.99627, p-value = 0.0008455
##
## [1] "Shapiro-Wilk test for column OP"
##
## Shapiro-Wilk normality test
##
## data: data[[col]]
## W = 0.61138, p-value < 2.2e-16
##
## [1] "Shapiro-Wilk test for column Smoking"
##
## Shapiro-Wilk normality test
##
## data: data[[col]]
## W = 0.54489, p-value < 2.2e-16
##
## [1] "Shapiro-Wilk test for column Drinking"
##
## Shapiro-Wilk normality test
##
```



```
## data: data[[col]]
## W = 0.5184, p-value < 2.2e-16
```

#none of the columns in our dataset have a normal distribution.

## Correlation

```
cor(data)
```

```
##           Age      Gender      BMI      Ca      P
## Age      1.000000000 -0.025185924  0.0367214161 -0.06893962  0.0037652633
## Gender   -0.025185924  1.000000000 -0.2433608838  0.02464990  0.0438763440
## BMI      0.036721416 -0.243360884  1.0000000000 -0.00443306 -0.0001982793
## Ca      -0.068939623  0.024649895 -0.0044330597  1.000000000  0.2172390220
## P       0.003765263  0.043876344 -0.0001982793  0.21723902  1.0000000000
## Mg      -0.104387459 -0.224082708  0.2099576475  0.06646473 -0.0543005275
## OP       0.055422445  0.003395221  0.0040782647 -0.03776777  0.0112742873
## Smoking  0.024180666 -0.007721263  0.0242814301 -0.04188546 -0.0849766112
## Drinking 0.008769412 -0.012140662 -0.0021391965 -0.01135212 -0.0453782737
##           Mg      OP      Smoking      Drinking
## Age      -0.104387459  0.055422445  0.024180666  0.008769412
## Gender   -0.224082708  0.003395221 -0.007721263 -0.012140662
## BMI      0.209957647  0.004078265  0.024281430 -0.002139196
## Ca      0.066464725 -0.037767766 -0.041885462 -0.011352121
## P      -0.054300528  0.011274287 -0.084976611 -0.045378274
## Mg      1.000000000  0.008308790  0.004678726  0.002864552
## OP      0.008308790  1.000000000 -0.010301088 -0.103957796
## Smoking  0.004678726 -0.010301088  1.000000000  0.524408834
## Drinking 0.002864552 -0.103957796  0.524408834  1.000000000
```

## Spearman Correlation

*# Calculate the Spearman correlation matrix*

```
cor(data, method = "spearman")
```

```
##           Age      Gender      BMI      Ca      P
## Age      1.000000000 -0.010631750  0.054427140 -0.050837223 -0.004496005
## Gender   -0.010631750  1.000000000 -0.257175375  0.028577861  0.042703248
## BMI      0.054427140 -0.257175375  1.000000000  0.008806547  0.007583249
## Ca      -0.050837223  0.028577861  0.008806547  1.000000000  0.192135098
## P      -0.004496005  0.042703248  0.007583249  0.192135098  1.000000000
## Mg      -0.129500641 -0.211954050  0.220823297  0.064020829 -0.054805644
## OP       0.051667692  0.003395221 -0.003232426 -0.044172827  0.010144822
## Smoking  0.011338926 -0.007721263  0.015982038 -0.029030032 -0.089295242
## Drinking 0.004138680 -0.012140662 -0.006923549 -0.003804442 -0.049830261
##           Mg      OP      Smoking      Drinking
## Age      -0.129500641  0.051667692  0.011338926  0.004138680
## Gender   -0.211954050  0.003395221 -0.007721263 -0.012140662
## BMI      0.220823297 -0.003232426  0.015982038 -0.006923549
## Ca      0.064020829 -0.044172827 -0.029030032 -0.003804442
```

```
## P      -0.054805644  0.010144822 -0.089295242 -0.049830261
## Mg      1.000000000  0.003133075  0.007011499  0.008185209
## OP      0.003133075  1.000000000 -0.010301088 -0.103957796
## Smoking 0.007011499 -0.010301088  1.000000000  0.524408834
## Drinking 0.008185209 -0.103957796  0.524408834  1.000000000
```

#there is a slight correlation between Age and OP .

## Pearson Correlation

*# Calculate the Pearson correlation matrix*

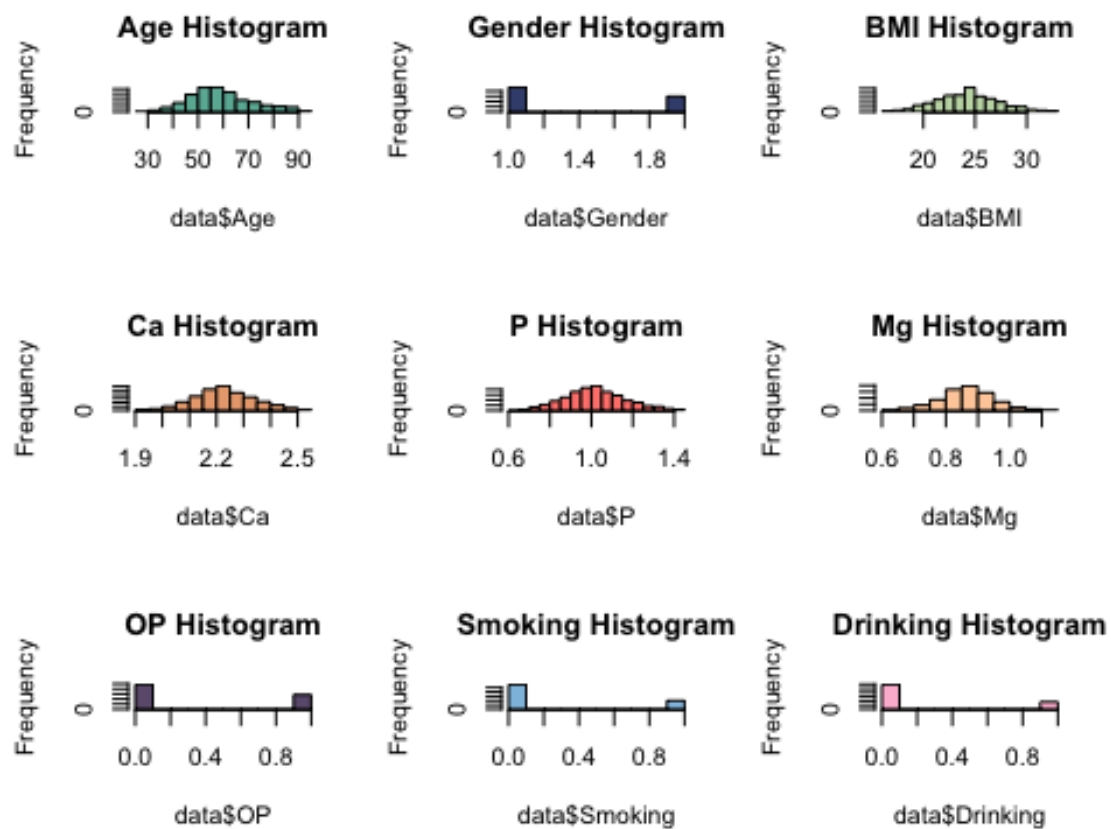
```
cor(data, method = "pearson")
```

```
##           Age      Gender      BMI      Ca      P
## Age      1.000000000 -0.025185924  0.0367214161 -0.06893962  0.0037652633
## Gender   -0.025185924  1.000000000 -0.2433608838  0.02464990  0.0438763440
## BMI      0.036721416 -0.243360884  1.0000000000 -0.00443306 -0.0001982793
## Ca      -0.068939623  0.024649895 -0.0044330597  1.00000000  0.2172390220
## P        0.003765263  0.043876344 -0.0001982793  0.21723902  1.0000000000
## Mg      -0.104387459 -0.224082708  0.2099576475  0.06646473 -0.0543005275
## OP       0.055422445  0.003395221  0.0040782647 -0.03776777  0.0112742873
## Smoking  0.024180666 -0.007721263  0.0242814301 -0.04188546 -0.0849766112
## Drinking 0.008769412 -0.012140662 -0.0021391965 -0.01135212 -0.0453782737
##           Mg      OP      Smoking      Drinking
## Age      -0.104387459  0.055422445  0.024180666  0.008769412
## Gender   -0.224082708  0.003395221 -0.007721263 -0.012140662
## BMI      0.209957647  0.004078265  0.024281430 -0.002139196
## Ca      0.066464725 -0.037767766 -0.041885462 -0.011352121
## P      -0.054300528  0.011274287 -0.084976611 -0.045378274
## Mg      1.000000000  0.008308790  0.004678726  0.002864552
## OP      0.008308790  1.000000000 -0.010301088 -0.103957796
## Smoking  0.004678726 -0.010301088  1.000000000  0.524408834
## Drinking 0.002864552 -0.103957796  0.524408834  1.000000000
```

#there is a slight correlation between Age and OP .

## Histogram

```
par(mfrow=c(3,3))
hist(data$Age, col = "#69b3a2", main="Age Histogram")
hist(data$Gender,col = "#404e7c", main="Gender Histogram")
hist(data$BMI,col = "#c2d9b1", main="BMI Histogram")
hist(data$Ca,col = "#e8a87c", main="Ca Histogram")
hist(data$P,col = "#ff847c", main="P Histogram")
hist(data$Mg,col = "#fecea8", main="Mg Histogram")
hist(data$OP,col = "#6c5b7b", main="OP Histogram")
hist(data$Smoking,col = "#8fbfe0", main="Smoking Histogram")
hist(data$Drinking,col = "#fcbad3", main="Drinking Histogram")
```



#we

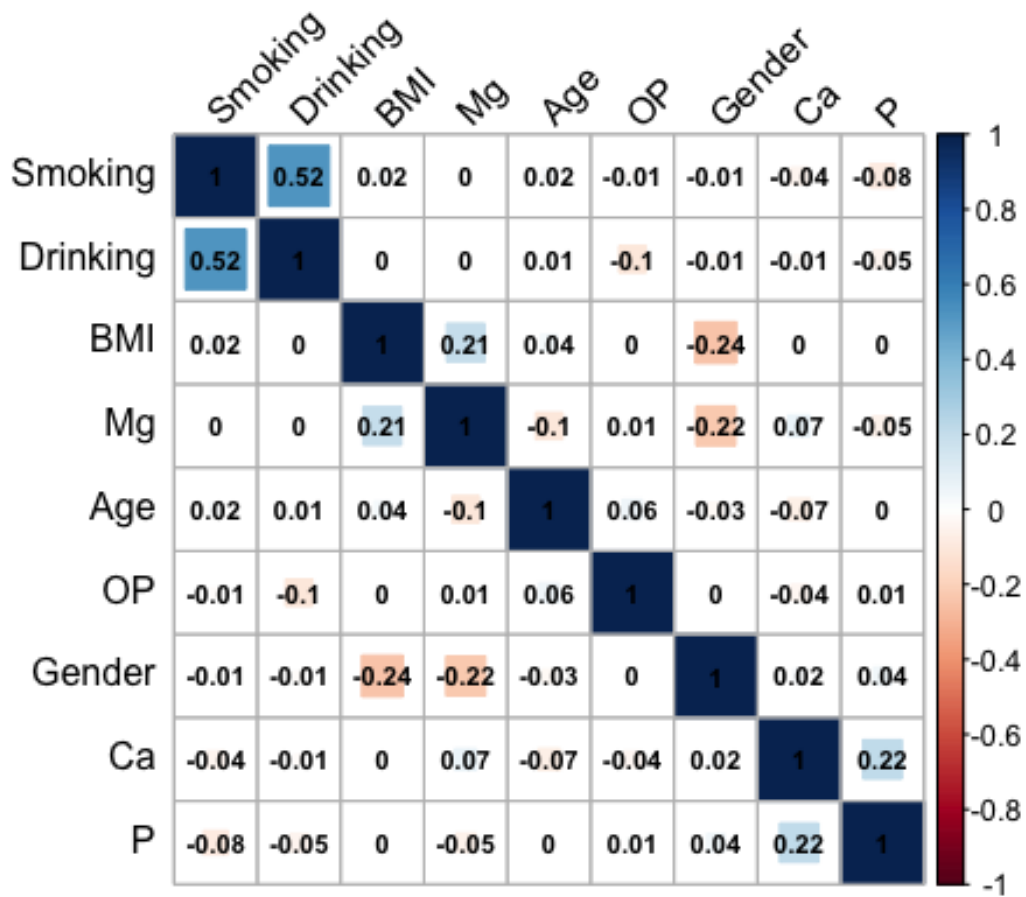
found Gender, OP, Smoking and Drinking has highly skewed data.

## Correlation plot

```
library(corrplot)

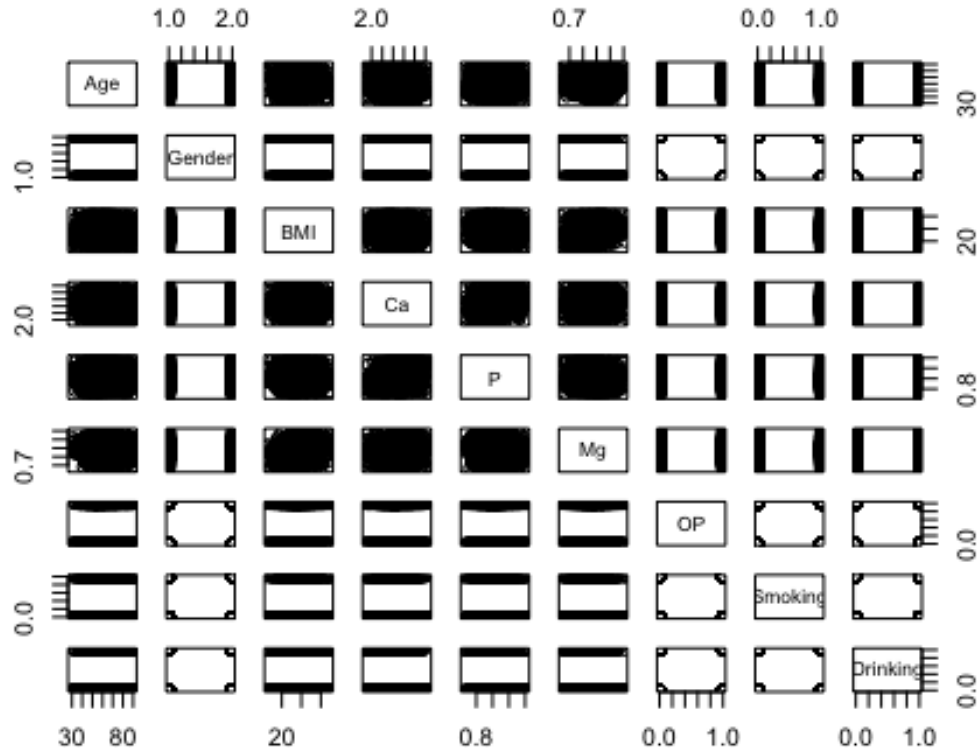
## corrplot 0.92 loaded

corrplot(cor(data), method = "square", order = "hclust",
          tl.col = "black", tl.srt = 45,
          addCoef.col = "black", number.cex = 0.7)
```



there is

correlation between drinking and smoking, BMI and Mg, Ca and P. # Pairplot  
pairs(data)



#One-sample t-test

```
# Loop through each column and perform t-test
for (col in colnames(data)) {
  t.test_result <- t.test(data[,col], mu = 0)
  if (t.test_result$p.value < 0.05) {
    print(paste("Null hypothesis for", col, "rejected with p-value =",
t.test_result$p.value))
  } else {
    print(paste("Null hypothesis for", col, "not rejected with p-value =",
t.test_result$p.value))
  }
}

## [1] "Null hypothesis for Age rejected with p-value = 0"
## [1] "Null hypothesis for Gender rejected with p-value = 0"
## [1] "Null hypothesis for BMI rejected with p-value = 0"
## [1] "Null hypothesis for Ca rejected with p-value = 0"
## [1] "Null hypothesis for P rejected with p-value = 0"
## [1] "Null hypothesis for Mg rejected with p-value = 0"
## [1] "Null hypothesis for OP rejected with p-value = 4.52362573006446e-156"
## [1] "Null hypothesis for Smoking rejected with p-value =
1.70686014534895e-101"
```

```
## [1] "Null hypothesis for Drinking rejected with p-value =  
2.7646881617891e-88"
```

## t-test with OP as target variable

```
# Loop through each column (except OP) and perform t-test with OP as target  
variable  
for (col in colnames(data)) {  
  t.test_result <- t.test(data[,col], data$OP)  
  if (t.test_result$p.value < 0.05) {  
    print(paste("Null hypothesis for", col, "rejected with p-value =",  
t.test_result$p.value))  
  } else {  
    print(paste("Null hypothesis for", col, "not rejected with p-value =",  
t.test_result$p.value))  
  }  
}  
  
## [1] "Null hypothesis for Age rejected with p-value = 0"  
## [1] "Null hypothesis for Gender rejected with p-value = 0"  
## [1] "Null hypothesis for BMI rejected with p-value = 0"  
## [1] "Null hypothesis for Ca rejected with p-value = 0"  
## [1] "Null hypothesis for P rejected with p-value = 0"  
## [1] "Null hypothesis for Mg rejected with p-value = 3.00304774759658e-244"  
## [1] "Null hypothesis for OP not rejected with p-value = 1"  
## [1] "Null hypothesis for Smoking rejected with p-value =  
1.97224248413668e-11"  
## [1] "Null hypothesis for Drinking rejected with p-value =  
5.66641836262055e-18"
```

## Wilcoxon rank-sum test

#to test significance of subgroups with OP variable

```
# Subset data by gender  
group1 <- subset(data, Gender == 1)  
group2 <- subset(data, Gender == 2)  
  
# Perform Wilcoxon rank-sum test on Score column  
wilcox.test(group1$OP, group2$OP)  
  
##  
## Wilcoxon rank sum test with continuity correction  
##  
## data: group1$OP and group2$OP  
## W = 278778, p-value = 0.8942  
## alternative hypothesis: true location shift is not equal to 0
```

This result indicates that there is no significant difference between the median of “OP” for group 1 and group 2. The p-value is greater than 0.05, which suggests that we cannot reject the null hypothesis that the true location shift is equal to 0.

## Kruskal-Wallis test

```
# subset the data into two groups based on Gender
group1 <- subset(data, Gender == 1)
group2 <- subset(data, Gender == 2)

# create a data frame with the combined groups
Kruskal_test <- data.frame(value = c(group1$OP, group2$OP),
                           group = factor(rep(c("Group 1", "Group 2"),
                                              c(nrow(group1),
                                                nrow(group2))))))

# perform the Kruskal-Wallis test
kruskal.test(value ~ group, data = Kruskal_test)

##
##  Kruskal-Wallis rank sum test
##
## data:  value by group
## Kruskal-Wallis chi-squared = 0.017706, df = 1, p-value = 0.8941
```

The Kruskal-Wallis test was performed successfully, with a chi-squared value of 0.017706 and a p-value of 0.8941. This suggests that there is no significant difference in the median values of the “value” variable between the two groups.

## Kruskal-Wallis test by age groups

```
# subset the data into 4 groups based on Age
age1 <- subset(data, Age >= 28.6 & Age < 51)
age2 <- subset(data, Age >= 51 & Age < 57)
age3 <- subset(data, Age >= 57 & Age < 67.3)
age4 <- subset(data, Age >= 67.3 & Age < 100)

# create a data frame with the combined groups
Kruskal_test <- data.frame(value = c(age1$OP, age2$OP, age3$OP, age4$OP),
                           group = factor(rep(c("age1", "age2", "age3",
                                              "age4"),
                                              c(nrow(age1), nrow(age2),
                                                nrow(age3), nrow(age4))))))

# perform the Kruskal-Wallis test
kruskal.test(value ~ group, data = Kruskal_test)

##
##  Kruskal-Wallis rank sum test
```

```
##  
## data: value by group  
## Kruskal-Wallis chi-squared = 7.5839, df = 3, p-value = 0.05544
```

The Kruskal-Wallis test suggests that there might be a significant difference between the groups based on Age. However, the p-value is just above the usual significance level of 0.05. Therefore, further investigation with post-hoc tests such as Dunn's test or pairwise Wilcoxon rank-sum tests should be performed to identify which groups are significantly different from each other.

## Mann-Whitney U test

```
# subset the data into 4 groups based on Age  
age1 <- subset(data, Age >= 28.6 & Age < 51)  
age2 <- subset(data, Age >= 51 & Age < 57)  
age3 <- subset(data, Age >= 57 & Age < 67.3)  
age4 <- subset(data, Age >= 67.3 & Age < 100)  
  
# perform pairwise Wilcoxon rank-sum tests  
mann_test <- pairwise.wilcox.test(x = data$OP, g = data$Age_group,  
p.adjust.method = "bonferroni")  
  
mann_test  
  
##  
## Pairwise comparisons using  
##  
## data: data$OP and data$Age_group  
##  
## <0 x 0 matrix>  
##  
## P value adjustment method: bonferroni
```

The output of the pairwise Wilcoxon rank-sum test suggests that there are no significant differences between the "OP" values for the different age groups. The p-values are all greater than 0.05 after Bonferroni correction for multiple comparisons. Therefore, we cannot reject the null hypothesis that there are no differences in "OP" values between the age groups.

Based on the output of the Kruskal-Wallis test and the pairwise comparison results, there is no significant difference between the groups in terms of the variable "x". The p-value for the Kruskal-Wallis test is 0.4, which is greater than the significance level of 0.05, indicating that there is no significant difference between the groups. Furthermore, the pairwise comparison results also show that all the p-values are greater than the adjusted significance level ( $0.05/15 = 0.0033$  using the Bonferroni correction), suggesting that there are no significant differences between any pairs of groups. Therefore, we can conclude that there is no significant difference between the groups in terms of the variable "x".



## Multiple regression model on BMI, Smoking, Drinking

*# Fit the multiple regression model*

```
reg_model <- lm(OP ~ BMI + Smoking + Drinking, data = data)
```

```
summary(reg_model)
```

```
##
## Call:
## lm(formula = OP ~ BMI + Smoking + Drinking, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.4580 -0.3879 -0.2992  0.6116  0.7703
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.3787067  0.1014464   3.733 0.000196 ***
## BMI          0.0003774  0.0041491   0.091 0.927529
## Smoking      0.0672212  0.0328963   2.043 0.041181 *
## Drinking     -0.1564128  0.0342966  -4.561 5.51e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.48 on 1533 degrees of freedom
## Multiple R-squared:  0.01351,    Adjusted R-squared:  0.01158
## F-statistic: 6.998 on 3 and 1533 DF,  p-value: 0.0001127
```

Smoking and Drinking are significant predictors of OP, but BMI is not. The model overall is not a strong predictor of OP, as indicated by the low R-squared value.

## Multiple Regression on OP vs Ca, Mg, P

*# Fit the multiple regression model*

```
reg_model <- lm(OP ~ Ca + Mg + P, data = data)
```

```
summary(reg_model)
```

```
##
## Call:
## lm(formula = OP ~ Ca + Mg + P, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.4316 -0.3749 -0.3512  0.6180  0.7035
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.66023    0.27984   2.359  0.0184 *
## Ca          -0.19114    0.11597  -1.648  0.0995 .
```

```
## Mg          0.07265    0.15093    0.481    0.6303
## P           0.07103    0.08724    0.814    0.4157
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4828 on 1533 degrees of freedom
## Multiple R-squared:  0.001975,    Adjusted R-squared:  2.239e-05
## F-statistic: 1.011 on 3 and 1533 DF,  p-value: 0.3867
```

only the intercept has a statistically significant relationship with the response variable, while the three predictor variables do not have significant relationships. The R-squared value indicates that the predictor variables explain very little of the variance in the response variable, and the F-statistic and p-value suggest that the overall model is not significant.

## Multiple regression model on Age, Gender

```
# Fit the multiple regression model
reg_model <- lm(OP ~ Age + Gender, data = data)

summary(reg_model)

##
## Call:
## lm(formula = OP ~ Age + Gender, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.4390 -0.3719 -0.3471  0.6181  0.6863
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.233286   0.070913   3.290  0.00103 **
## Age          0.002180   0.001001   2.178  0.02955 *
## Gender       0.004755   0.025294   0.188  0.85090
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4824 on 1534 degrees of freedom
## Multiple R-squared:  0.003095,    Adjusted R-squared:  0.001795
## F-statistic: 2.381 on 2 and 1534 DF,  p-value: 0.09281
```

## Logistic regression model on BMI, Gender

```
# Fit a logistic regression model
model <- glm(OP ~ BMI + Gender, data = data, family = binomial)

# Print the model summary
summary(model)
```

```
##
## Call:
## glm(formula = OP ~ BMI + Gender, family = binomial, data = data)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -0.9769  -0.9616  -0.9570   1.4087   1.4270
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.650242   0.510309  -1.274   0.203
## BMI          0.003657   0.018449   0.198   0.843
## Gender       0.019846   0.111909   0.177   0.859
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2024.9  on 1536  degrees of freedom
## Residual deviance: 2024.8  on 1534  degrees of freedom
## AIC: 2030.8
##
## Number of Fisher Scoring iterations: 4
```

The results indicate that neither “BMI” nor “Gender” has a statistically significant effect on “OP” (both p-values > 0.05).

## Logistic Regression on OP vs BMI, Smoking, Drinking

```
# Fit a Logistic regression model
model <- glm(OP ~ BMI + Smoking + Drinking, data = data, family = binomial)

# Print the summary of the model
summary(model)

##
## Call:
## glm(formula = OP ~ BMI + Smoking + Drinking, family = binomial,
##      data = data)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.1153  -0.9898  -0.8382   1.3764   1.6989
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.503516   0.441361  -1.141   0.2539
## BMI          0.001827   0.018050   0.101   0.9194
## Smoking      0.297149   0.144074   2.062   0.0392 *
## Drinking     -0.705845   0.156887  -4.499 6.83e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2024.9  on 1536  degrees of freedom
## Residual deviance: 2003.5  on 1533  degrees of freedom
## AIC: 2011.5
##
## Number of Fisher Scoring iterations: 4
```

the model suggests that Smoking and Drinking are significant predictors of OP, while BMI is not.

## Logistic Regression on OP vs Ca, Mg, P

```
# Fit a logistic regression model
model2 <- glm(OP ~ Ca + Mg + P, data = data, family = binomial)

# Print the summary of the model
summary(model2)

##
## Call:
## glm(formula = OP ~ Ca + Mg + P, family = binomial, data = data)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.0658  -0.9690  -0.9301   1.3876   1.5528
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.7166     1.2027   0.596   0.5513
## Ca           -0.8228     0.4994  -1.648   0.0994 .
## Mg             0.3121     0.6482   0.481   0.6302
## P             0.3058     0.3750   0.816   0.4148
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2024.9  on 1536  degrees of freedom
## Residual deviance: 2021.9  on 1533  degrees of freedom
## AIC: 2029.9
##
## Number of Fisher Scoring iterations: 4
```

Ca may have a weak negative relationship with the outcome variable, while Mg and P may not be significant predictors

## Logistic Regression on OP vs BMI, Gender

```
# Fit a logistic regression model
model <- glm(OP ~ BMI + Gender, data = data, family = binomial)

# Print the summary of the model
summary(model)

##
## Call:
## glm(formula = OP ~ BMI + Gender, family = binomial, data = data)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -0.9769  -0.9616  -0.9570   1.4087   1.4270
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.650242   0.510309  -1.274   0.203
## BMI          0.003657   0.018449   0.198   0.843
## Gender       0.019846   0.111909   0.177   0.859
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2024.9  on 1536  degrees of freedom
## Residual deviance: 2024.8  on 1534  degrees of freedom
## AIC: 2030.8
##
## Number of Fisher Scoring iterations: 4
```

None of the predictors are statistically significant at the 0.05 level, as all their p-values are greater than 0.05.

## Random Forest Regression

```
# Load required library
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:dplyr':
##
##      combine

library(caret)
```

```

## Loading required package: ggplot2

##
## Attaching package: 'ggplot2'

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

## Loading required package: lattice

data$OP <- as.factor(data$OP)

# Split data into training and testing sets
set.seed(123)
train_idx <- sample(1:nrow(data), size = 0.7*nrow(data))
train_data <- data[train_idx,]
test_data <- data[-train_idx,]

rf_model <- randomForest(OP ~ ., data = train_data, importance = TRUE, ntree
= 500, mtry = 3)

# Predict using the fitted model and test data
rf_pred <- predict(rf_model, newdata = test_data)

# Create contingency table
tab <- table(rf_pred, test_data$OP)

# Convert to confusion matrix
cm <- confusionMatrix(tab)

# Print confusion matrix
print(cm)

## Confusion Matrix and Statistics
##
##
## rf_pred    0    1
##           0 248 137
##           1  47  30
##
##               Accuracy : 0.6017
##               95% CI   : (0.5555, 0.6467)
##      No Information Rate : 0.6385
##      P-Value [Acc > NIR] : 0.9542
##
##               Kappa   : 0.023
##
##  Mcnemar's Test P-Value : 5.339e-11

```

```
##
##          Sensitivity : 0.8407
##          Specificity : 0.1796
##          Pos Pred Value : 0.6442
##          Neg Pred Value : 0.3896
##          Prevalence : 0.6385
##          Detection Rate : 0.5368
##          Detection Prevalence : 0.8333
##          Balanced Accuracy : 0.5102
##
##          'Positive' Class : 0
##
```

Based on this confusion matrix, we can evaluate the performance of the random forest model. We can see that the model predicted 295 observations as category 0, out of which 248 were correctly classified (true negatives) and 47 were incorrectly classified (false positives). The model predicted 167 observations as category 1, out of which only 30 were correctly classified (true positives) and 137 were incorrectly classified (false negatives), which gives an overall accuracy of approximately 60.17%.

## Tuning the hyperparameters using grid search to improve the accuracy of the model

```
# Load required library
library(randomForest)

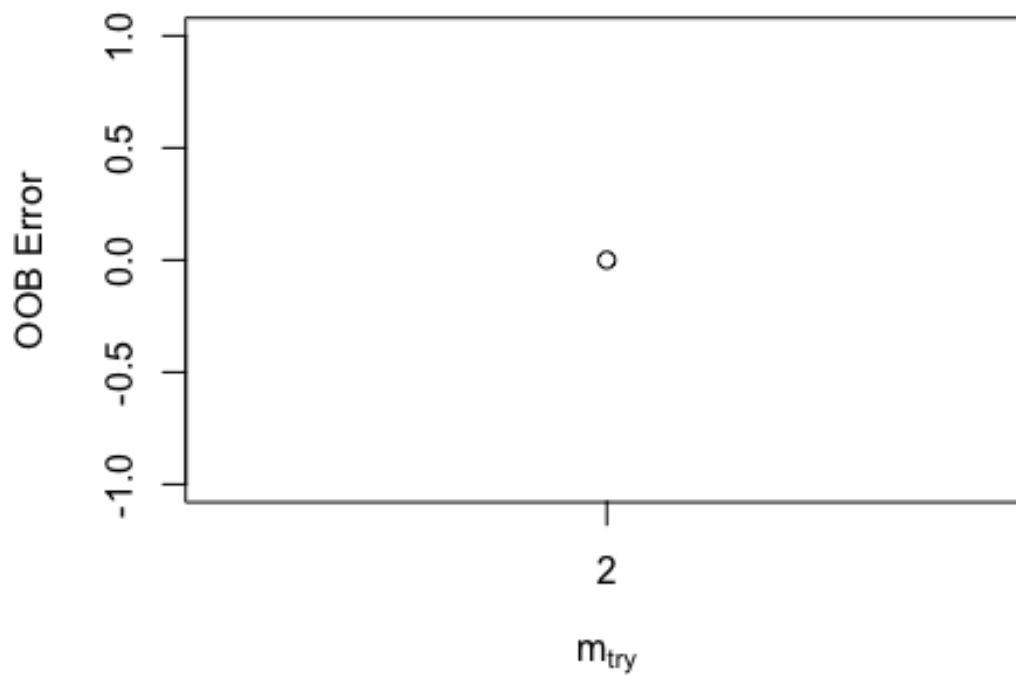
# Split data into training and testing sets
set.seed(123)
train_idx <- sample(1:nrow(data), size = 0.7*nrow(data))
train_data <- data[train_idx,]
test_data <- data[-train_idx,]

# Define grid for tuning
rf_grid <- expand.grid(ntree = c(500, 1000),
                      mtry = c(2, 4))

# Define the grid of ntree and mtry values
ntree_vals <- c(500, 1000)
mtry_vals <- c(2, 4)

# Tune random forest model
rf_tune <- tuneRF(x = train_data[, -1], y = train_data$OP,
                 ntreeTry = ntree_vals[1],
                 mtryStart = mtry_vals[1],
                 stepFactor = 1.2,
                 improve = 0.01)
```

```
## mtry = 2  OOB error = 0%
## Searching left ...
## Searching right ...
```



```
# Train random forest model with optimal hyperparameters
rf_model <- randomForest(OP ~ ., data = train_data[-1], ntree = 1000, mtry =
rf_tune[1, "mtry"])

# Predict on test data
rf_pred <- predict(rf_model, test_data)

# Evaluate model performance
confusionMatrix(table(rf_pred, test_data$OP))

## Confusion Matrix and Statistics
##
##
## rf_pred    0    1
##           0 272 149
##           1  23  18
##
##               Accuracy : 0.6277
##               95% CI : (0.5818, 0.6719)
```



```

##      No Information Rate : 0.6385
##      P-Value [Acc > NIR] : 0.7039
##
##      Kappa : 0.0357
##
##      McNemar's Test P-Value : <2e-16
##
##      Sensitivity : 0.9220
##      Specificity : 0.1078
##      Pos Pred Value : 0.6461
##      Neg Pred Value : 0.4390
##      Prevalence : 0.6385
##      Detection Rate : 0.5887
##      Detection Prevalence : 0.9113
##      Balanced Accuracy : 0.5149
##
##      'Positive' Class : 0
##

# Load required libraries
library(caret)
library(randomForest)

# Split data into training and testing sets
set.seed(123)
train_idx <- sample(1:nrow(data), size = 0.7*nrow(data))
train_data <- data[train_idx,]
test_data <- data[-train_idx,]

# Set up a grid of hyperparameters to tune
rf_grid <- expand.grid(mtry = c(2, 3, 4, 5))

# Tune random forest model
rf_tune <- train(OP ~ ., data = train_data, method = "rf",
                 trControl = trainControl(method = "cv", number = 10),
                 tuneGrid = rf_grid, importance = TRUE)

# Train random forest model with optimal hyperparameters
rf_model <- randomForest(train_data$OP ~ ., data = train_data, mtry =
rf_tune$bestTune$mtry)

# Predict on test data
rf_pred <- predict(rf_model, test_data)

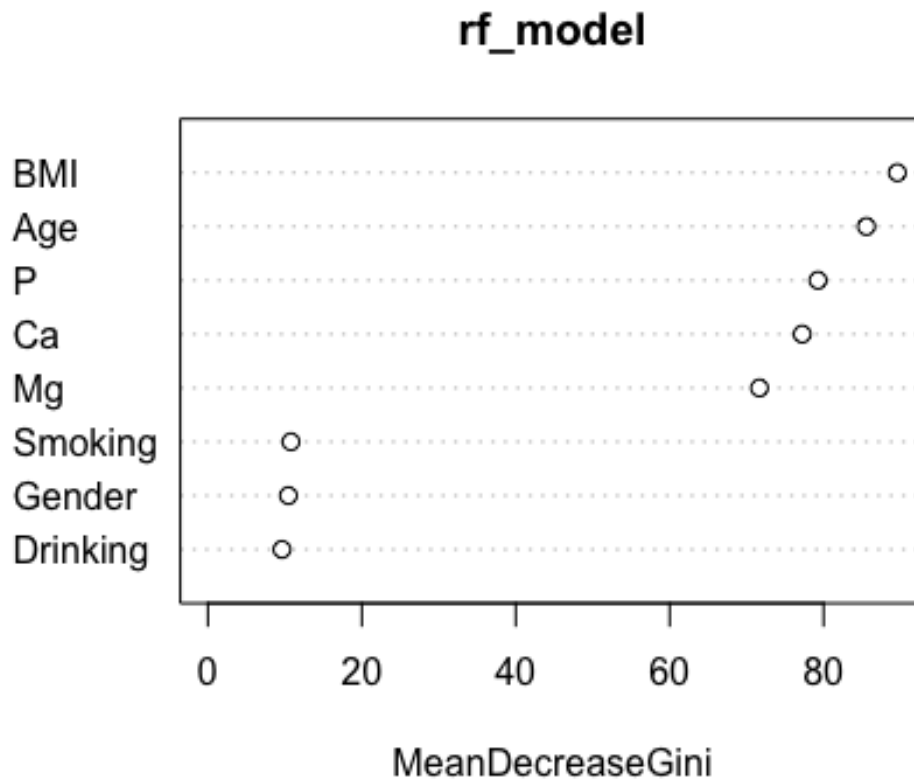
# Evaluate model performance
table(rf_pred, test_data$OP)

##
## rf_pred    0    1

```

```
##      0 269 150
##      1  26  17

# Variable importance measures
varImpPlot(rf_model)
```



```
# Evaluate model performance
confusionMatrix(rf_pred, test_data$OP)

## Confusion Matrix and Statistics
##
##              Reference
## Prediction    0    1
##              0 269 150
##              1  26  17
##
##              Accuracy : 0.619
##              95% CI : (0.573, 0.6635)
##              No Information Rate : 0.6385
##              P-Value [Acc > NIR] : 0.8214
##
##              Kappa : 0.0163
##
## Mcnemar's Test P-Value : <2e-16
```



138 ## 0	104	107	109	113	114	116	118	123	125	128	131	132	133	135	136
188 ## 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
246 ## 0	140	146	147	149	150	154	157	173	175	176	181	182	183	184	187
299 ## 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
336 ## 0	192	198	203	213	214	216	219	225	226	228	231	232	233	244	245
369 ## 1	0	0	0	1	0	0	0	0	0	0	1	0	1	0	0
427 ## 0	249	252	254	257	260	267	268	272	273	276	283	284	288	295	296
491 ## 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
546 ## 0	300	301	302	305	306	307	308	313	314	317	322	324	325	333	334
590 ## 0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
664 ## 0	338	340	342	345	346	351	353	354	356	359	360	361	362	364	368
736 ## 0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0
798 ## 0	371	377	388	389	390	395	399	404	405	408	410	412	414	417	418
	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0
	428	429	437	438	439	440	442	443	453	454	465	472	476	483	486
	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
	492	495	497	501	506	509	510	511	513	514	518	521	527	532	543
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	550	552	553	558	563	568	569	570	571	573	574	580	582	583	585
	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
	607	614	624	629	630	632	634	636	638	644	645	652	659	661	663
	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0
	681	689	693	695	697	704	707	716	717	718	721	723	725	727	735
	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
	739	745	754	761	762	764	770	771	773	777	780	781	782	790	795

```
## 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0
## 799 800 801 805 809 816 823 824 830 833 837 849 852 854 862
863
## 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0
## 864 867 868 871 874 875 880 881 882 883 884 885 887 895 897
909
## 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0
0
## 911 912 913 914 918 925 934 936 942 943 948 954 958 961 962
963
## 0 0 0 0 0 1 0 1 0 1 0 1 0 0 0
0
## 964 967 968 969 972 980 984 998 1000 1001 1003 1004 1008 1010 1012
1018
## 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0
## 1019 1024 1025 1026 1027 1037 1039 1041 1042 1045 1049 1050 1051 1052 1053
1058
## 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0
## 1059 1061 1067 1074 1084 1085 1086 1089 1092 1094 1095 1104 1106 1111 1114
1116
## 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0
## 1119 1138 1144 1145 1147 1148 1150 1151 1152 1156 1157 1160 1162 1170 1173
1179
## 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0
## 1186 1189 1190 1191 1192 1194 1197 1198 1200 1206 1208 1209 1210 1211 1213
1214
## 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
0
## 1215 1217 1218 1223 1225 1226 1227 1228 1229 1232 1237 1243 1252 1254 1264
1265
## 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0
## 1266 1267 1271 1272 1287 1292 1295 1297 1303 1305 1311 1312 1325 1330 1335
1339
## 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0
## 1341 1342 1345 1346 1347 1350 1354 1358 1360 1362 1367 1375 1383 1385 1388
1394
## 0 0 1 1 0 0 0 0 0 1 0 0 0 1 0
0
## 1395 1400 1402 1404 1405 1409 1413 1414 1419 1424 1425 1432 1434 1437 1443
1447
## 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0
0
```

```
## 1449 1458 1460 1461 1465 1470 1471 1473 1480 1481 1482 1485 1487 1490 1495
1498
##      0      0      0      0      0      0      0      1      0      0      1      0      0      0      0
0
## 1500 1502 1504 1506 1509 1513 1514 1518 1522 1526 1527 1529 1534 1535
##      0      0      0      0      0      0      0      1      0      0      0      0      0      0
## Levels: 0 1
```