APPENDIX

JASURBEK USMONALIEV
2255209
2023/2024

Data loading

````
# 2. Loading the data
```{r}
# Load and inspect the data
data <- read.csv("Obesity_CW.csv")
str(data)
```
````

```
'data.frame':   20758 obs. of  18 variables:
 $ id                            : int  0 1 2 3 4 5 6 7 8 9 ...
 $ Gender                        : chr  "Male" "Female" "Female" "Female" ...
 $ Age                           : num  24.4 18 18 21 31.6 ...
 $ Height                        : num  1.7 1.56 1.71 1.71 1.91 ...
 $ Weight                        : num  81.7 57 50.2 131.3 93.8 ...
 $ family_history_with_overweight: chr  "yes" "yes" "yes" "yes" ...
 $ FAVC                          : chr  "yes" "yes" "yes" "yes" ...
 $ FCVC                          : num  2 2 1.88 3 2.68 ...
 $ NCP                           : num  2.98 3 1.41 3 1.97 ...
 $ CAEC                          : chr  "Sometimes" "Frequently" "Sometimes" "Sometimes" ...
 $ SMOKE                         : chr  "no" "no" "no" "no" ...
 $ CH2O                          : num  2.76 2 1.91 1.67 1.98 ...
 $ SCC                           : chr  "no" "no" "no" "no" ...
 $ FAF                           : num  0 1 0.866 1.468 1.968 ...
 $ TUE                           : num  0.976 1 1.674 0.78 0.932 ...
 $ CALC                          : chr  "Sometimes" "no" "no" "Sometimes" ...
 $ MTRANS                        : chr  "Public_Transportation" "Automobile" "Public_Transportation" "Public_Transportation" ...
 $ NObeyesdad                    : chr  "Overweight_Level_II" "Normal_Weight" "Insufficient_Weight" "Obesity_Type_III" ...
```

Summary

````
```{r}
# Summary report of the variables
summary(data)
```
````

```
       id           Gender               Age            Height          Weight      
 Min.   :    0   Length:20758       Min.   :14.00   Min.   :1.450   Min.   : 39.00  
 1st Qu.: 5189   Class :character   1st Qu.:20.00   1st Qu.:1.632   1st Qu.: 66.00  
 Median :10378   Mode  :character   Median :22.82   Median :1.700   Median : 84.06  
 Mean   :10378                      Mean   :23.84   Mean   :1.700   Mean   : 87.89  
 3rd Qu.:15568                      3rd Qu.:26.00   3rd Qu.:1.763   3rd Qu.:111.60  
 Max.   :20757                      Max.   :61.00   Max.   :1.976   Max.   :165.06  
 family_history_with_overweight     FAVC                FCVC            NCP            CAEC          
 Length:20758                   Length:20758       Min.   :1.000   Min.   :1.000   Length:20758      
 Class :character               Class :character   1st Qu.:2.000   1st Qu.:3.000   Class :character  
 Mode  :character               Mode  :character   Median :2.394   Median :3.000   Mode  :character  
                                                   Mean   :2.446   Mean   :2.761                     
                                                   3rd Qu.:3.000   3rd Qu.:3.000                     
                                                   Max.   :3.000   Max.   :4.000                     
    SMOKE                CH2O           SCC                 FAF               TUE             CALC          
 Length:20758       Min.   :1.000   Length:20758       Min.   :0.000000   Min.   :0.0000   Length:20758      
 Class :character   1st Qu.:1.792   Class :character   1st Qu.:0.008013   1st Qu.:0.0000   Class :character  
 Mode  :character   Median :2.000   Mode  :character   Median :1.000000   Median :0.5739   Mode  :character  
                    Mean   :2.029                      Mean   :0.981747   Mean   :0.6168                     
                    3rd Qu.:2.550                      3rd Qu.:1.587406   3rd Qu.:1.0000                     
                    Max.   :3.000                      Max.   :3.000000   Max.   :2.0000                     
    MTRANS            NObeyesdad       
 Length:20758       Length:20758       
 Class :character   Class :character   
 Mode  :character   Mode  :character   
```

```
---------------------------------------------------------------------------------------------------------------------
CALC
       n  missing distinct
   20758        0        3

Value      Frequently         no  Sometimes
Frequency         529       5163      15066
Proportion      0.025      0.249      0.726
---------------------------------------------------------------------------------------------------------------------
MTRANS
       n  missing distinct
   20758        0        5

Value                 Automobile                 Bike           Motorbike Public_Transportation
Frequency                   3534                   32                  38                16687
Proportion                 0.170                0.002               0.002                0.804

Value                    Walking
Frequency                    467
Proportion                 0.022
---------------------------------------------------------------------------------------------------------------------
NObeyesdad
       n  missing distinct
   20758        0        7

Value      Insufficient_Weight       Normal_Weight      Obesity_Type_I     Obesity_Type_II    Obesity_Type_III
Frequency                 2523                3082                2910                3248                4046
Proportion               0.122               0.148               0.140               0.156               0.195

Value       Overweight_Level_I Overweight_Level_II
Frequency                 2427                2522
Proportion               0.117               0.121
---------------------------------------------------------------------------------------------------------------------
```

head()

Description: df [6 × 18]

	id <int>	Gender <chr>	Age <dbl>	Height <dbl>	Weight <dbl>	family_history_with_overweight <chr>	FCHighCal <chr>	FCvegetables <dbl>	NumMainMeals <dbl>
1	0	Male	24.44301	1.699998	81.66995	yes	yes	2.000000	2.983297
2	1	Female	18.00000	1.560000	57.00000	yes	yes	2.000000	3.000000
3	2	Female	18.00000	1.711460	50.16575	yes	yes	1.880534	1.411685
4	3	Female	20.95274	1.710730	131.27485	yes	yes	3.000000	3.000000
5	4	Male	31.64108	1.914186	93.79806	yes	yes	2.679664	1.971472
6	5	Male	18.12825	1.748524	51.55259	yes	yes	2.919751	3.000000

6 rows | 1–10 of 18 columns

Description: df [6 × 18]

ConsFoodBetwMeal <chr>	SMO... <chr>	DayWater <dbl>	MonitorCalory <chr>	PhysicalActFreq <dbl>	TechUsePerDay <dbl>	Alcohol <chr>	MTRANS <chr>	ObesityLevel <chr>
Sometimes	no	2.763573	no	0.000000	0.976473	Sometimes	Public_Transportation	Overweight_Level_II
Frequently	no	2.000000	no	1.000000	1.000000	no	Automobile	Normal_Weight
Sometimes	no	1.910378	no	0.866045	1.673584	no	Public_Transportation	Insufficient_Weight
Sometimes	no	1.674061	no	1.467863	0.780199	Sometimes	Public_Transportation	Obesity_Type_III
Sometimes	no	1.979848	no	1.967973	0.931721	Sometimes	Public_Transportation	Overweight_Level_II
Sometimes	no	2.137550	no	1.930033	1.000000	Sometimes	Public_Transportation	Insufficient_Weight

6 rows | 11–19 of 18 columns

```r
# Deleting id column as it's unneeded
data <- data %>%
  select(-id)

# Convert the Age column to integer
data$Age <- as.integer(floor(data$Age))

# Converting Height into cm
data$Height <- data$Height * 100
data$Height <- signif(data$Height, digits = 3)

# Converting Weight into integer
data$Weight <- data$Weight * 100
data$Weight <- signif(data$Weight, digits = 3)
data$Weight <- data$Weight %/% 100

# Remove leading and trailing white space & standardize levels in Gender
data$Gender <- trimws(tolower(data$Gender))
```

Data type converting

```r
# Convert brinary strings to integers
# In Gender, convert 'male' 'female' to 1 0 respectively
data <- data %>%
  mutate(Gender = ifelse(Gender == "male", 1, 0))

# In family_history_with_overweight, convert 'yes' 'no' to 1 0 respectively
data <- data %>%
  mutate(family_history_with_overweight = ifelse(family_history_with_overweight == "yes", 1, 0))

# In FCHCfood, convert 'yes' 'no' to 1 0 respectively
data <- data %>%
  mutate(FCHighCal = ifelse(FCHighCal== "yes", 1, 0))

# In SMOKE, convert 'yes' 'no' to 1 0 respectively
data <- data %>%
  mutate(SMOKE = ifelse(SMOKE== "yes", 1, 0))

# In MonitorCalory, convert 'yes' 'no' to 1 0 respectively
data <- data %>%
  mutate(MonitorCalory = ifelse(MonitorCalory== "yes", 1, 0))
```

```r
# Convert string factors to numerical levels
# Convert levels in ConsFoodBetwMeal to numerical values
data <- data %>%
  mutate(ConsFoodBetwMeal = recode_factor(ConsFoodBetwMeal,
                                          "Always" = 1,
                                          "Frequently" = 2,
                                          "no" = 3,
                                          "Sometimes" = 4))
data$ConsFoodBetwMeal <- as.numeric(data$ConsFoodBetwMeal)

# Convert levels in Alcohol to numerical values
data$Alcohol <- as.numeric(recode_factor(data$Alcohol,
                                         "Frequently" = 1,
                                         "no" = 2,
                                         "Sometimes" = 3))

# Convert levels in MTRANS to numerical values
data$MTRANS <- as.numeric(recode_factor(data$MTRANS,
                                        "Automobile" = 1,
                                        "Bike" = 2,
                                        "Motorbike" = 3,
                                        "Public_Transportation" = 4
                                        "Walking" = 5))
```

```r
# Convert string factors to numerical levels
# Convert levels in ConsFoodBetwMeal to numerical values
data <- data %>%
  mutate(ConsFoodBetwMeal = recode_factor(ConsFoodBetwMeal,
                                          "Always" = 1,
                                          "Frequently" = 2,
                                          "no" = 3,
                                          "Sometimes" = 4))
data$ConsFoodBetwMeal <- as.numeric(data$ConsFoodBetwMeal)

# Convert levels in Alcohol to numerical values
data$Alcohol <- as.numeric(recode_factor(data$Alcohol,
                                         "Frequently" = 1,
                                         "no" = 2,
                                         "Sometimes" = 3))

# Convert levels in MTRANS to numerical values
data$MTRANS <- as.numeric(recode_factor(data$MTRANS,
                                        "Automobile" = 1,
                                        "Bike" = 2,
                                        "Motorbike" = 3,
                                        "Public_Transportation" = 4,
                                        "Walking" = 5))
```

Data rounding

```r
# Round FCvegetables, NumMainMeals, DayWater, PhysicalActFreq, TUE to 2 decimals
data$FCvegetables <- round(data$FCvegetables, digits = 1)
data$NumMainMeals <- round(data$NumMainMeals, digits = 1)
data$DayWater <- round(data$DayWater, digits = 1)
data$PhysicalActFreq <- round(data$PhysicalActFreq, digits = 1)
data$TechUsePerDay <- round(data$TechUsePerDay, digits = 1)
```

```r
# Converting ObesityLevel levels to numeric values
data$ObesityLevel <- as.character(data$ObesityLevel)  # Converting to character before changing to numeric

data$ObesityLevel[data$ObesityLevel == "Insufficient_Weight"] <- "0"
data$ObesityLevel[data$ObesityLevel == "Normal_Weight"] <- "1"
data$ObesityLevel[data$ObesityLevel == "Overweight_Level_I"] <- "2"
data$ObesityLevel[data$ObesityLevel == "Overweight_Level_II"] <- "3"
data$ObesityLevel[data$ObesityLevel == "Obesity_Type_I"] <- "4"
data$ObesityLevel[data$ObesityLevel == "Obesity_Type_II"] <- "5"
data$ObesityLevel[data$ObesityLevel == "Obesity_Type_III"] <- "6"

data$ObesityLevel<- as.factor(data$ObesityLevel) # Convert to factor
```

```r
# Amend incorrectly loaded variables
data$Gender <- as.factor(data$Gender)
data$family_history_with_overweight <- as.factor(data$family_history_with_overweight)
data$FCHighCal <- as.factor(data$FCHighCal)
data$ConsFoodBetwMeal <- as.factor(data$ConsFoodBetwMeal)
data$SMOKE <- as.factor(data$SMOKE)
data$MonitorCalory <- as.factor(data$MonitorCalory)
data$Alcohol <- as.factor(data$Alcohol)
data$MTRANS <- as.factor(data$MTRANS)
data$ObesityLevel <- as.factor(data$ObesityLevel)


```

After cleaning

Description: df [6 × 17]

	Gender <fctr>	Age <int>	Height <dbl>	Weight <dbl>	family_history_with_overweight <fctr>	FCHighCal <fctr>	FCvegetables <dbl>	NumMainMeals <dbl>	ConsFoodBetwMeal <fctr>
1	1	24	170	81	1	1	2.000000	2.983297	4
2	0	18	156	57	1	1	2.000000	3.000000	2
3	0	18	171	50	1	1	1.880534	1.411685	4
4	0	20	171	131	1	1	3.000000	3.000000	4
5	1	31	191	93	1	1	2.679664	1.971472	4
6	1	18	175	51	1	1	2.919751	3.000000	4

6 rows | 1-10 of 17 columns

Description: df [6 × 17]

	NumMainMeals <dbl>	ConsFoodBetwMeal <fctr>	SMOKE <fctr>	DayWater <dbl>	MonitorCalory <fctr>	PhysicalActFreq <dbl>	TechUsePerDay <dbl>	Alcohol <fctr>	MTRANS <fctr>	ObesityLevel <fctr>
	3.0	4	0	2.8	0	0.0	1.0	3	4	3
	3.0	2	0	2.0	0	1.0	1.0	2	1	1
	1.4	4	0	1.9	0	0.9	1.7	2	4	0
	3.0	4	0	1.7	0	1.5	0.8	3	4	6
	2.0	4	0	2.0	0	2.0	0.9	3	4	3
	3.0	4	0	2.1	0	1.9	1.0	3	4	0

6 rows | 9-18 of 17 columns

```
# Exploratory Data Analysis

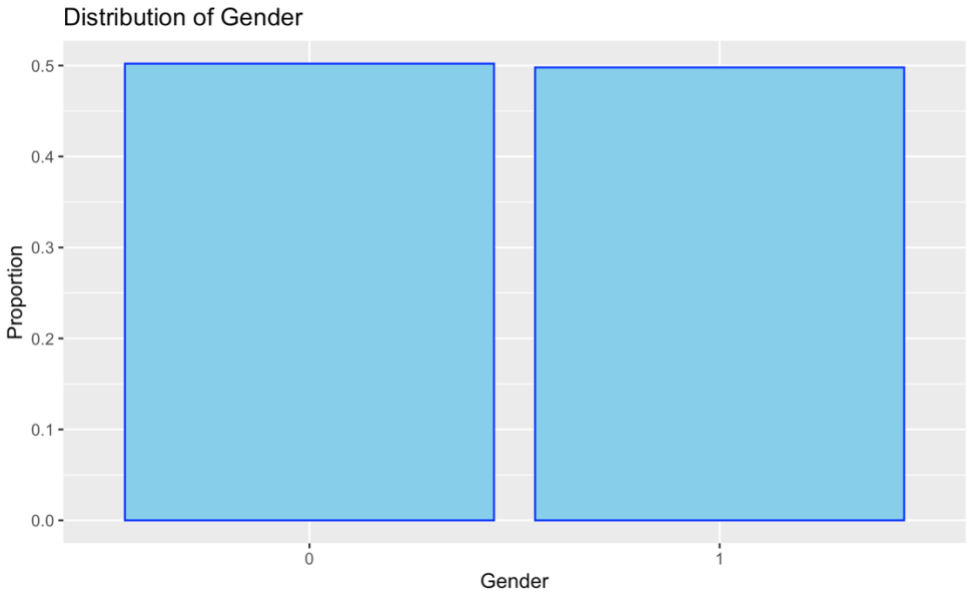
```{r}
# Analyze gender
gender_counts <- table(data$Gender)
gender_proportions <- prop.table(gender_counts)
gender_summary <- data.frame(Gender = names(gender_counts), Count = gender_counts, Proportion = gender_proportions)

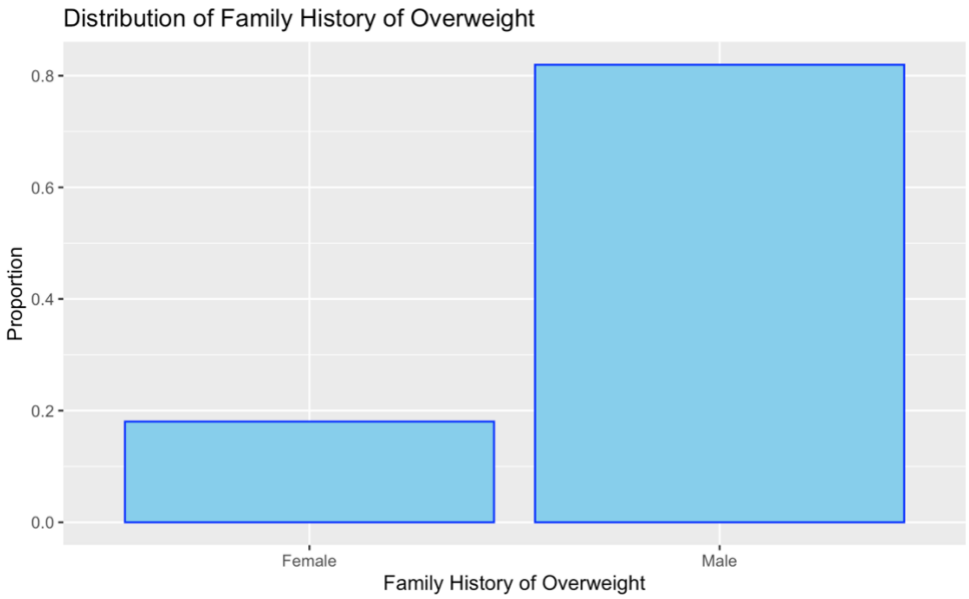
# Visualize gender distribution
ggplot(gender_summary, aes(x = Gender, y = Proportion.Freq)) +
  geom_bar(stat = "identity", fill = "skyblue", color = "blue") +
  labs(x = "Gender", y = "Proportion", title = "Distribution of Gender")

# Analyze family history of overweight
fam_history_counts <- table(data$family_history_with_overweight)
fam_history_proportions <- prop.table(fam_history_counts)
fam_history_summary <- data.frame(Family_History = names(fam_history_counts), Count = fam_history_counts, Proportion = fam_history_proportions)

# Visualize family history of overweight distribution
ggplot(fam_history_summary, aes(x = Family_History, y = Proportion.Freq)) +
  geom_bar(stat = "identity", fill = "skyblue", color = "blue") +
  labs(x = "Family History of Overweight", y = "Proportion", title = "Distribution of Family History of Overweight")+
  scale_x_discrete(labels = c("Female", "Male"))
```
```

Proportion of genders

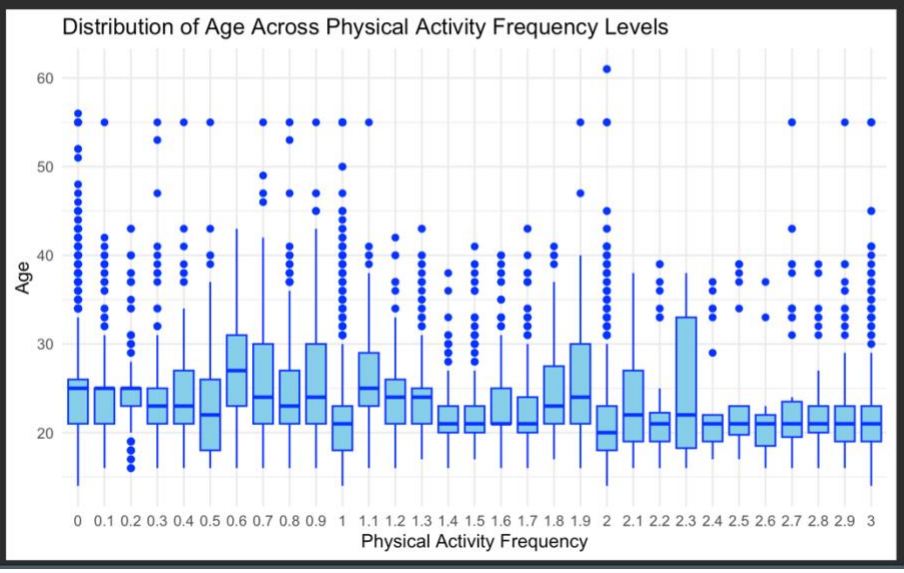




Boxplot Age and PhysicalAct Freq

```
# Calculate summary statistics for age within each level of physical activity frequency
age_summary <- aggregate(data$Age, by = list(data$PhysicalActFreq), FUN = summary)

# Visualize the distribution of age across different levels of physical activity frequency
ggplot(data, aes(x = factor(PhysicalActFreq), y = Age)) +
  geom_boxplot(fill = "skyblue", color = "blue") +
  labs(x = "Physical Activity Frequency", y = "Age", title = "Distribution of Age Across Physical Activity Frequency Levels") +
  theme_minimal()
```

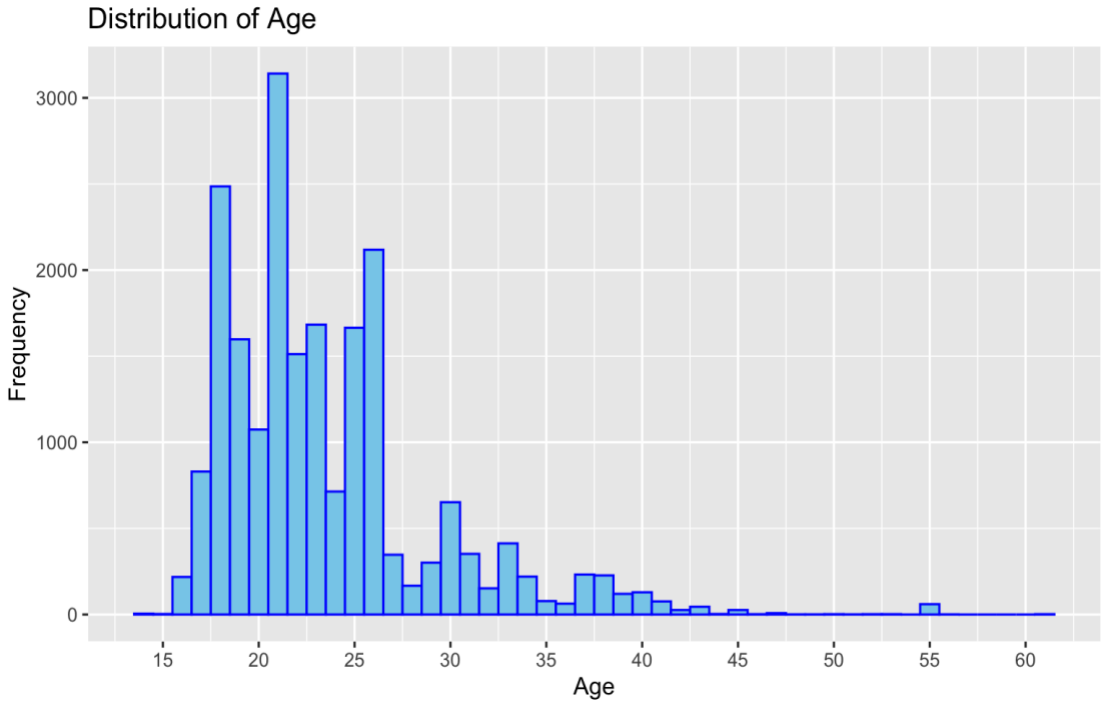


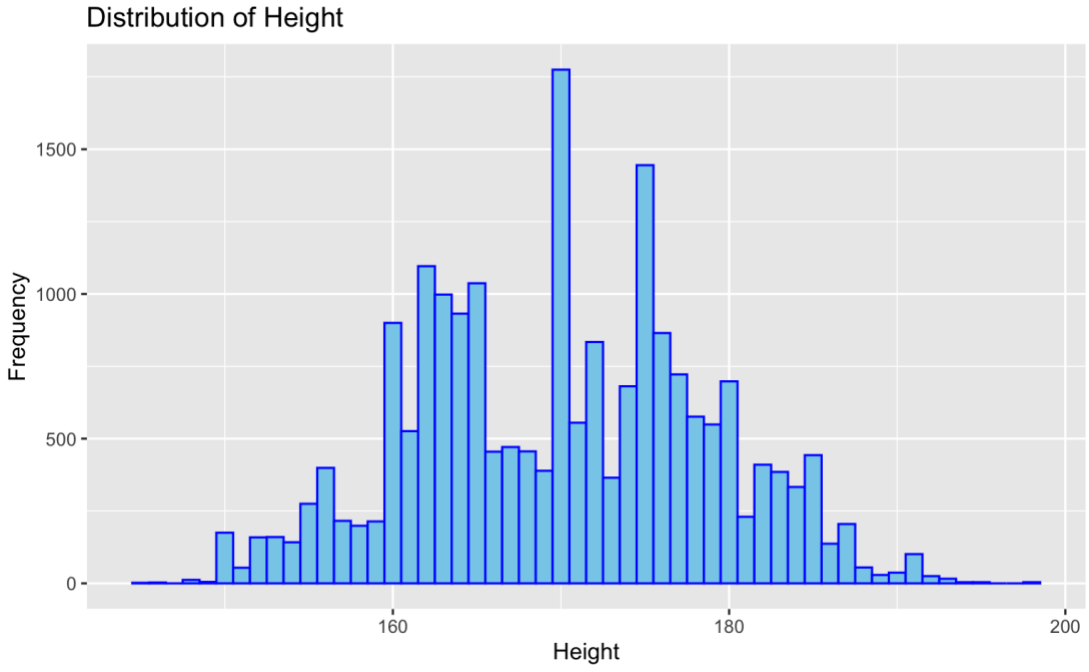
Histogram codes

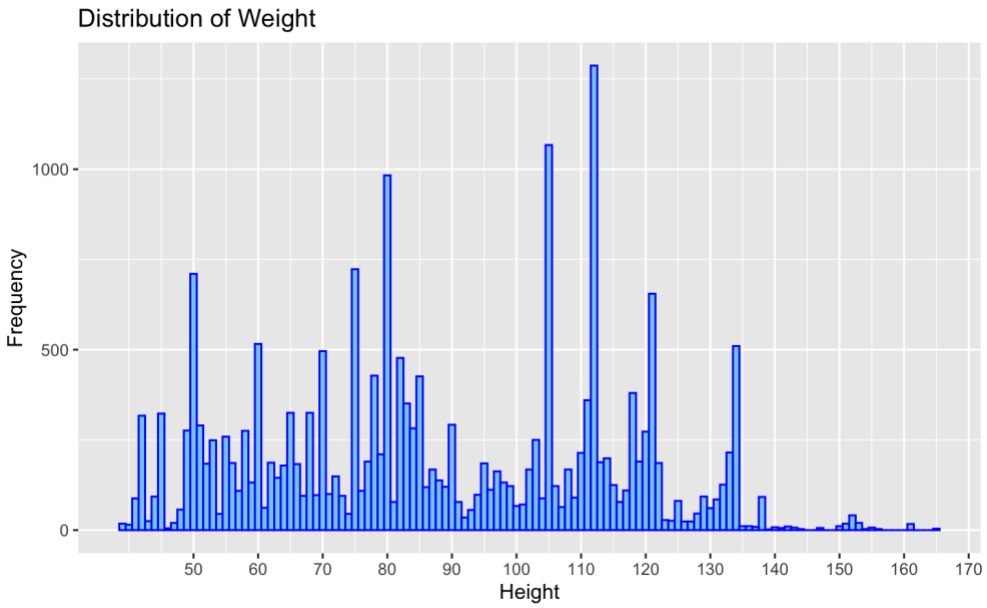
```r
# FCvegetables
ggplot(data, aes(x = FCvegetables)) +
  geom_histogram(color = "blue", fill = "skyblue", alpha = 1, binwidth = 0.1, bins = 30) +
  labs(title = "Distribution of FCvegetables",
       x = "FCvegetables",
       y = "Frequency")

# NumMainMeals
ggplot(data, aes(x = NumMainMeals)) +
  geom_histogram( color = "blue", fill = "skyblue", alpha = 1, binwidth = 0.1, bins = 30) +
  labs(title = "Distribution of NumMainMeals",
       x = "NumMainMeals",
       y = "Frequency")

# DayWater
ggplot(data, aes(x = DayWater)) +
  geom_histogram(color = "blue", fill = "skyblue", alpha = 1, binwidth = 0.1, bins = 30) +
  labs(title = "Distribution of CH2O",
       x = "DayWater",
       y = "Frequency")

# PhysicalActFreq
ggplot(data, aes(x = PhysicalActFreq)) +
  geom_histogram( color = "blue", fill = "skyblue", alpha = 1, binwidth = 0.1, bins = 30) +
  labs(title = "Distribution of PhysicalActFreq",
       x = "PhysicalActFreq",
       y = "Frequency")

# TechUsePerDay
ggplot(data, aes(x = TechUsePerDay)) +
  geom_histogram(color = "blue", fill = "skyblue", alpha = 1, binwidth = 0.1, bins = 30) +
  labs(title = "Distribution of TechUsePerDay",
       x = "TechUsePerDay",
       y = "Frequency")
```

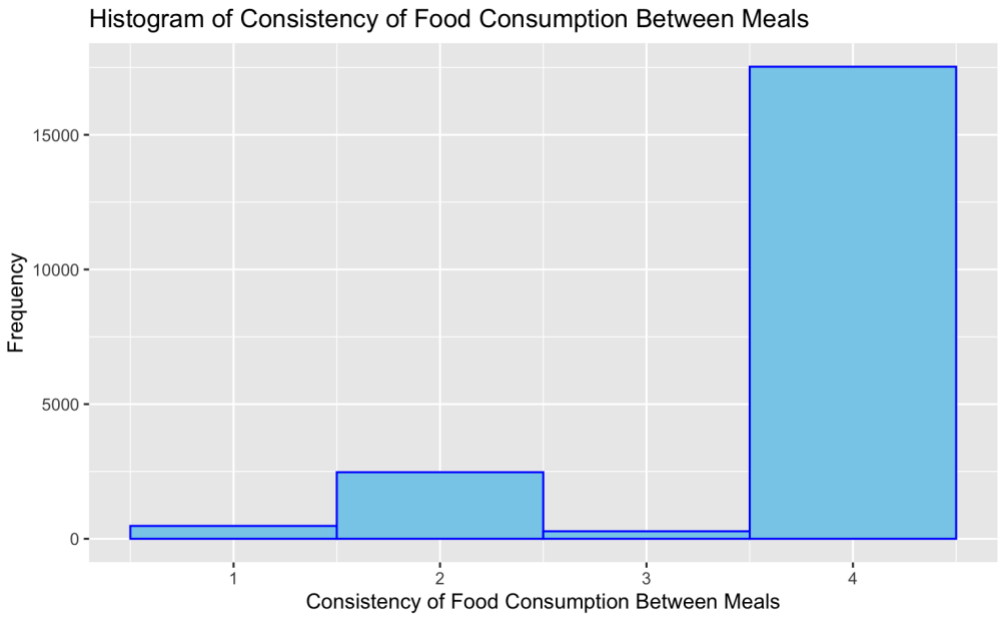
```r
# Histograms
# age
ggplot(data, aes(x = Age)) +
  geom_histogram(binwidth = 1, color = "blue", fill = "skyblue", alpha = 1) +
  labs(title = "Distribution of Age",
       x = "Age",
       y = "Frequency") +
  scale_x_continuous(breaks = seq(5, 65, by = 5))

# height
ggplot(data, aes(x = Height)) +
  geom_histogram(binwidth = 1, color = "blue", fill = "skyblue", alpha = 1) +
  labs(title = "Distribution of Height",
       x = "Height",
       y = "Frequency")

# weight
ggplot(data, aes(x = Weight)) +
  geom_histogram(binwidth = 1, color = "blue", fill = "skyblue", alpha = 1) +
  labs(title = "Distribution of Weight",
       x = "Height",
       y = "Frequency") +
  scale_x_continuous(breaks = seq(50, 180, by = 10))

# ConsFoodBetwMeal
# Convert 'ConsFoodBetwMeal' to a numeric variable
data$ConsFoodBetwMeal <- as.numeric(as.character(data$ConsFoodBetwMeal))

# Create histogram using ggplot
ggplot(data = data, aes(x = ConsFoodBetwMeal)) +
  geom_histogram(binwidth = 1, fill = "skyblue", color = "blue") +
  labs(x = "Consistency of Food Consumption Between Meals", y = "Frequency", title = "Histogram of Consistency of Food Consumption Between Meals")
```

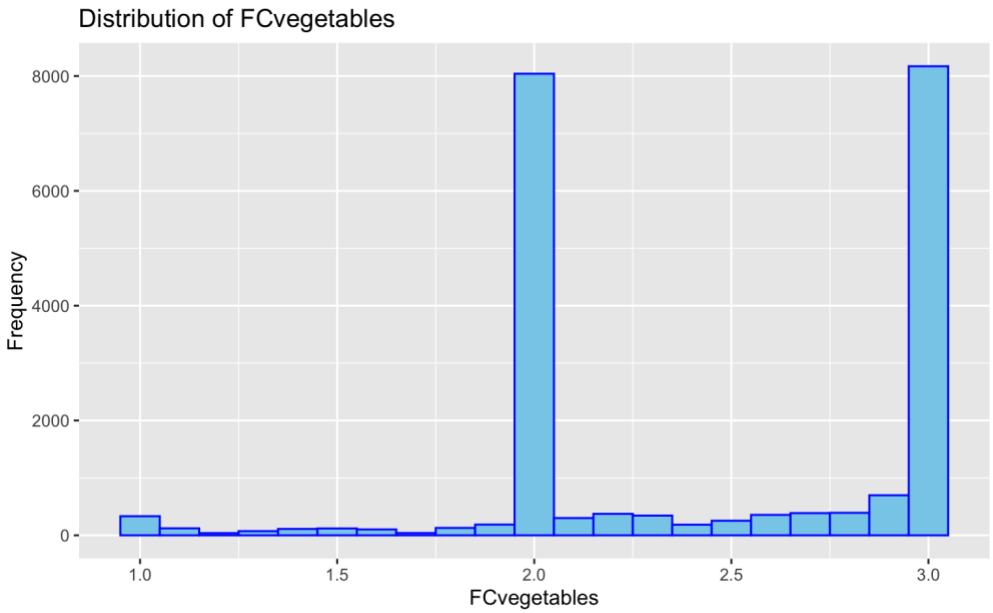
HIstograms

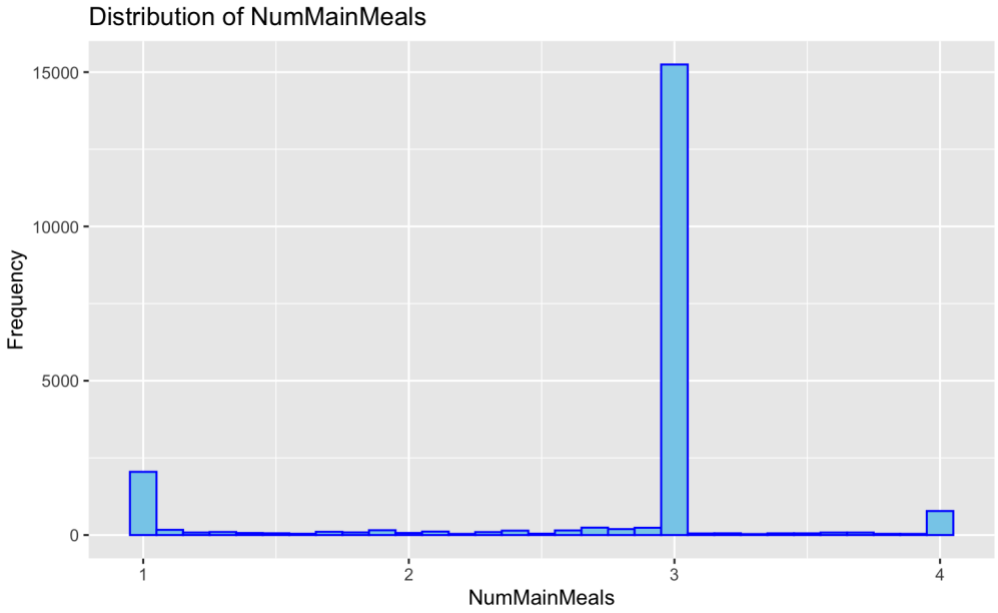

Distribution of Age
Frequency
3000
2000
1000
0
15
20
25
30
35
40
45
50
55
60
Age

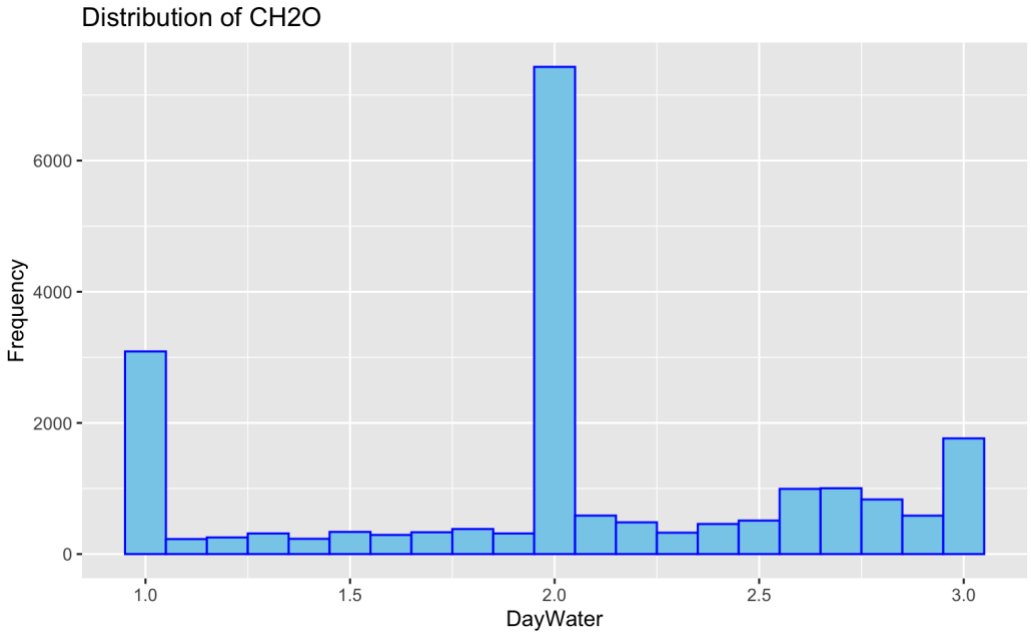

Distribution of Height
Frequency
1500
1000
500
0
160
180
200
Height

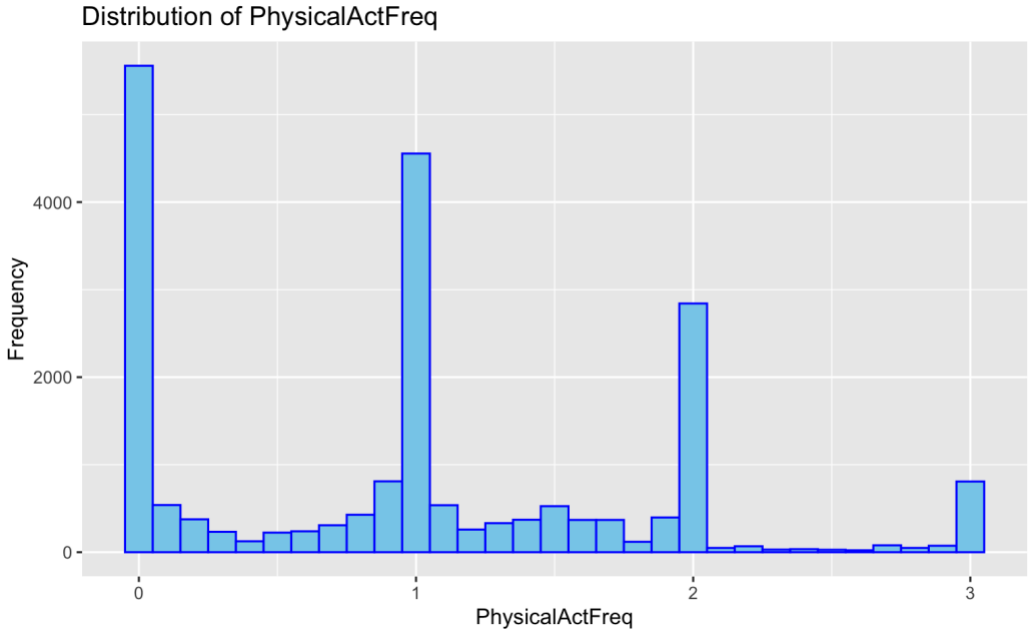

Distribution of Weight
Frequency
1000
500
0
50
60
70
80
90
100
110
120
130
140
150
160
170
Height

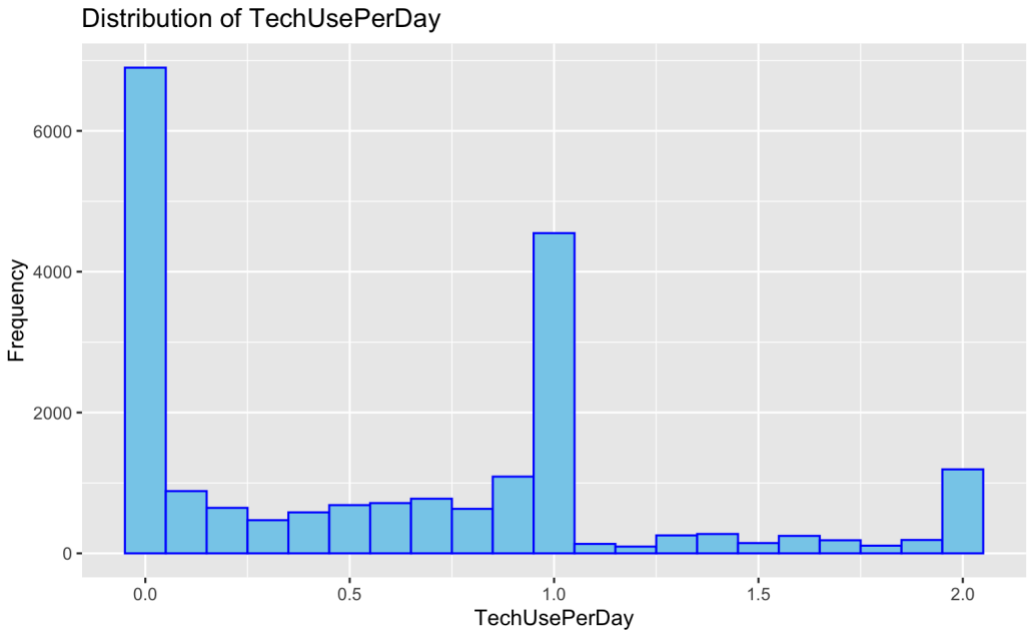

Histogram of Consistency of Food Consumption Between Meals
Frequency
15000
10000
5000
0
1
2
3
4
Consistency of Food Consumption Between Meals


Distribution of FCvegetables
Frequency
8000
6000
4000
2000
0
1.0
1.5
2.0
2.5
3.0
FCvegetables


Distribution of NumMainMeals
Frequency
15000
10000
5000
0
1
2
3
4
NumMainMeals


Distribution of CH2O
Frequency
6000
4000
2000
0
1.0
1.5
2.0
2.5
3.0
DayWater

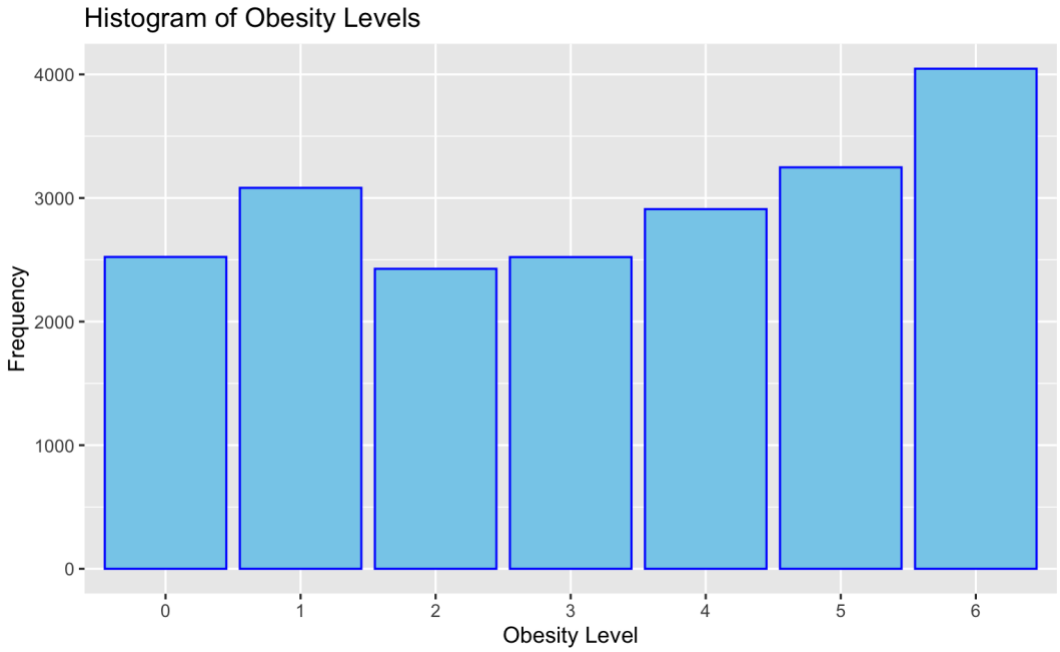

Distribution of PhysicalActFreq
Frequency
4000
2000
0
0
1
2
3
PhysicalActFreq

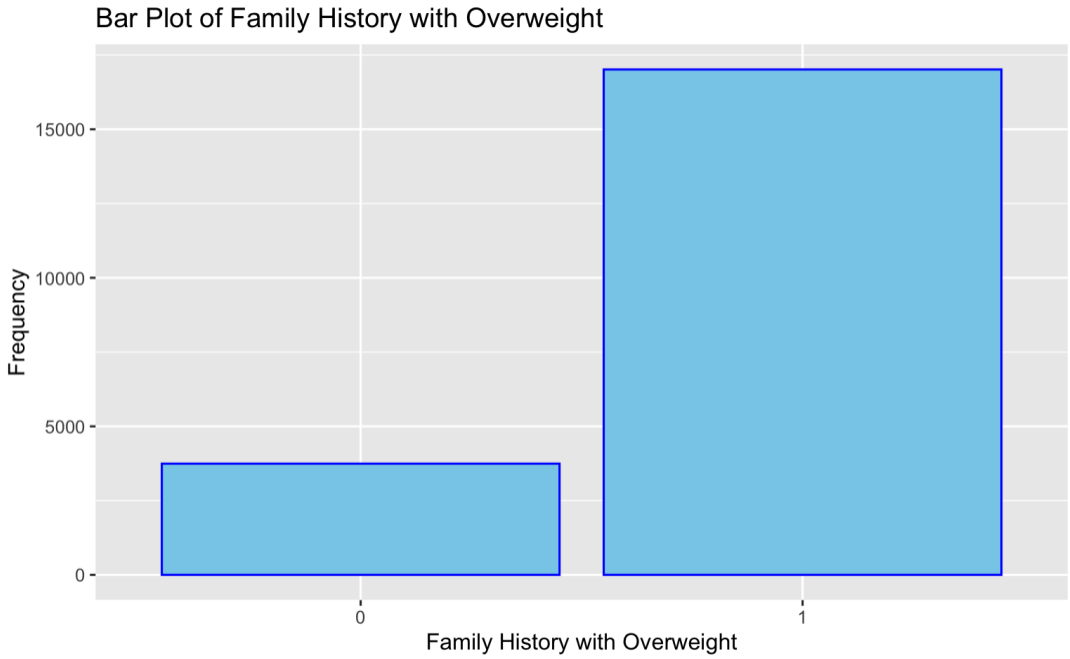

Distribution of TechUsePerDay
Frequency
6000
4000
2000
0
0.0
0.5
1.0
1.5
2.0
TechUsePerDay

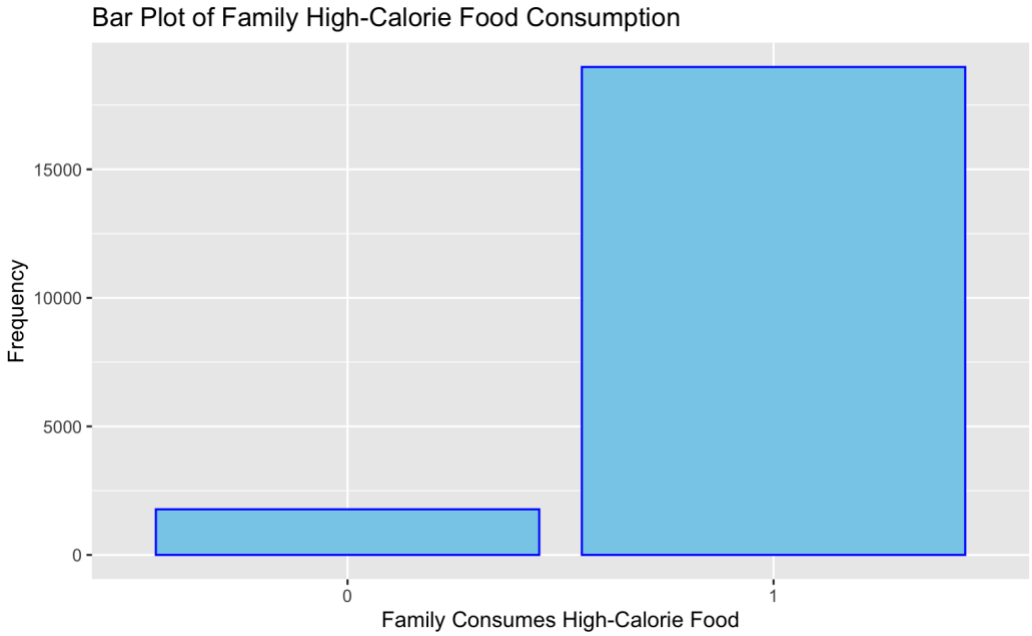
bar plots & code

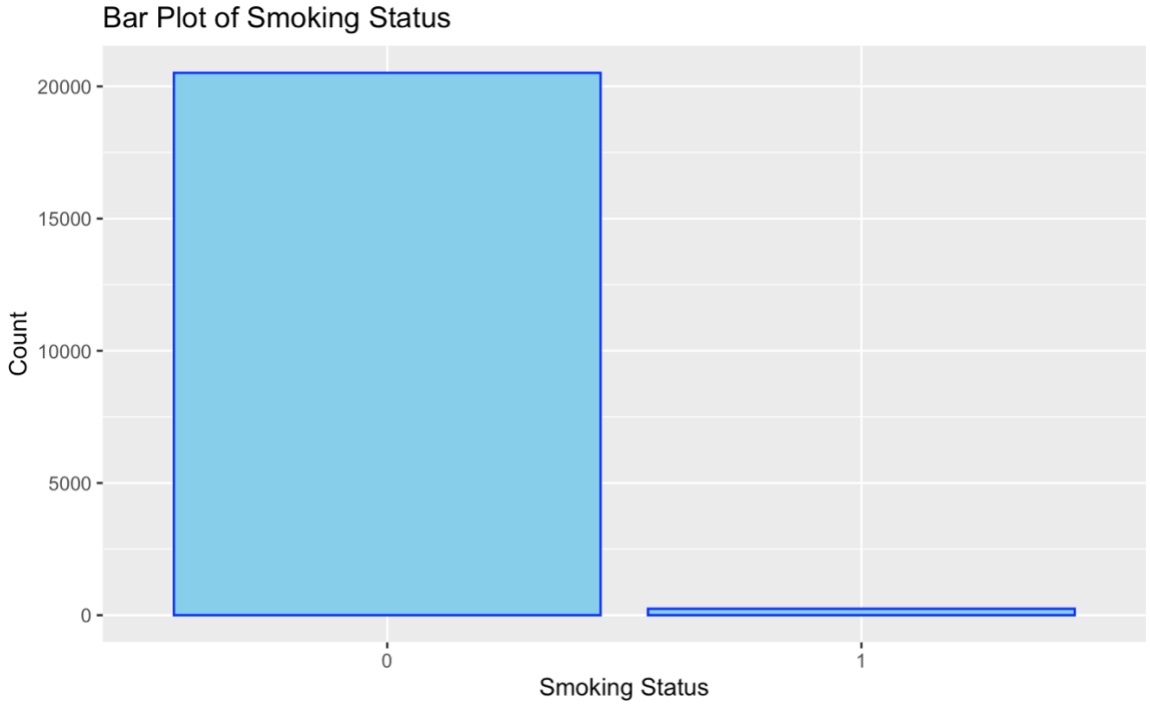
```r
# Bar plots of factor/binary variables
# gender
ggplot(data = data, aes(x = factor(Gender))) +
  geom_bar(fill = "skyblue", color = "blue") +
  labs(x = "Gender", y = "Count", title = "Bar Plot of Gender")

ggplot(data = data, aes(x = factor(ObesityLevel))) +
  geom_bar(fill = "skyblue", color = "blue") +
  labs(x = "Obesity Level", y = "Frequency", title = "Histogram of Obesity Levels")

ggplot(data = data, aes(x = factor(family_history_with_overweight))) +
  geom_bar(fill = "skyblue", color = "blue") +
  labs(x = "Family History with Overweight", y = "Frequency", title = "Bar Plot of Family History with Overweight")

ggplot(data = data, aes(x = factor(FCHighCal))) +
  geom_bar(fill = "skyblue", color = "blue") +
  labs(x = "Family Consumes High-Calorie Food", y = "Frequency", title = "Bar Plot of Family High-Calorie Food Consumption")

ggplot(data = data, aes(x = factor(SMOKE))) +
  geom_bar(fill = "skyblue", color = "blue") +
  labs(x = "Smoking Status", y = "Count", title = "Bar Plot of Smoking Status")

ggplot(data = data, aes(x = factor(MonitorCalory))) +
  geom_bar(fill = "skyblue", color = "blue") +
  labs(x = "Monitor Calorie Intake", y = "Count", title = "Bar Plot of Monitor Calorie Intake")

ggplot(data = data, aes(x = factor(Alcohol))) +
  geom_bar(fill = "skyblue", color = "blue") +
  labs(x = "Alcohol Consumption", y = "Count", title = "Bar Plot of Alcohol Consumption")

ggplot(data = data, aes(x = factor(MTRANS))) +
  geom_bar(fill = "skyblue", color = "blue") +
  labs(x = "Transportation Mode", y = "Count", title = "Bar Plot of Transportation Mode")
```

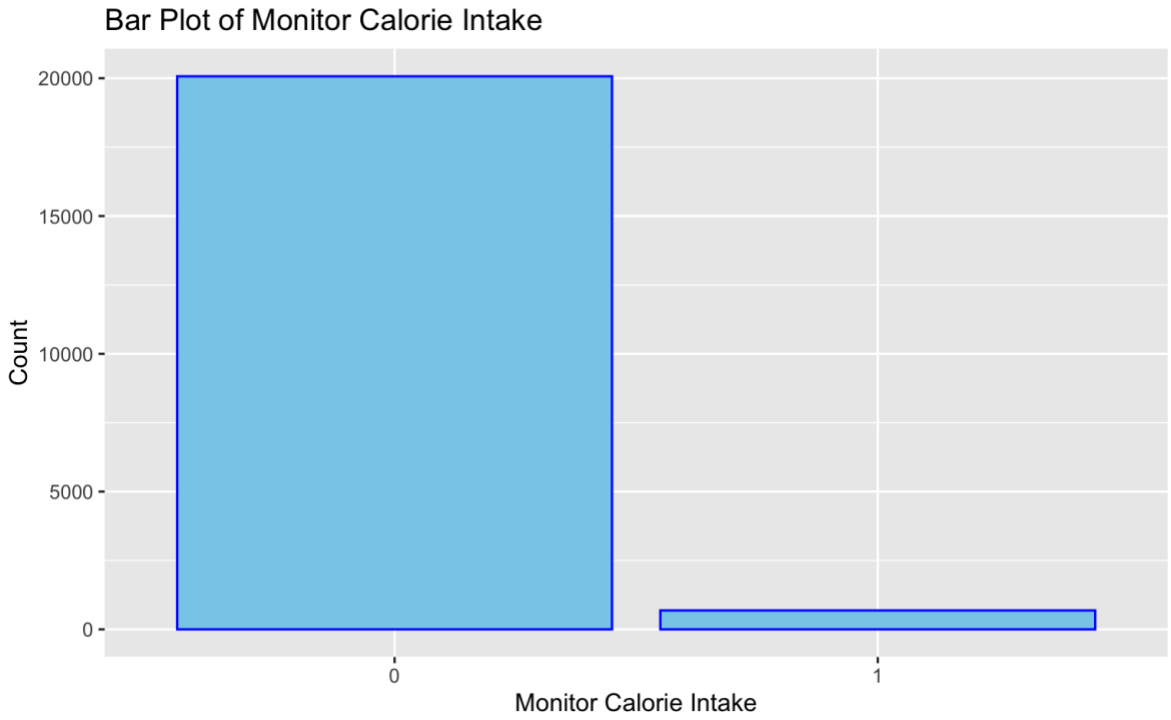
Barplot of factor variables

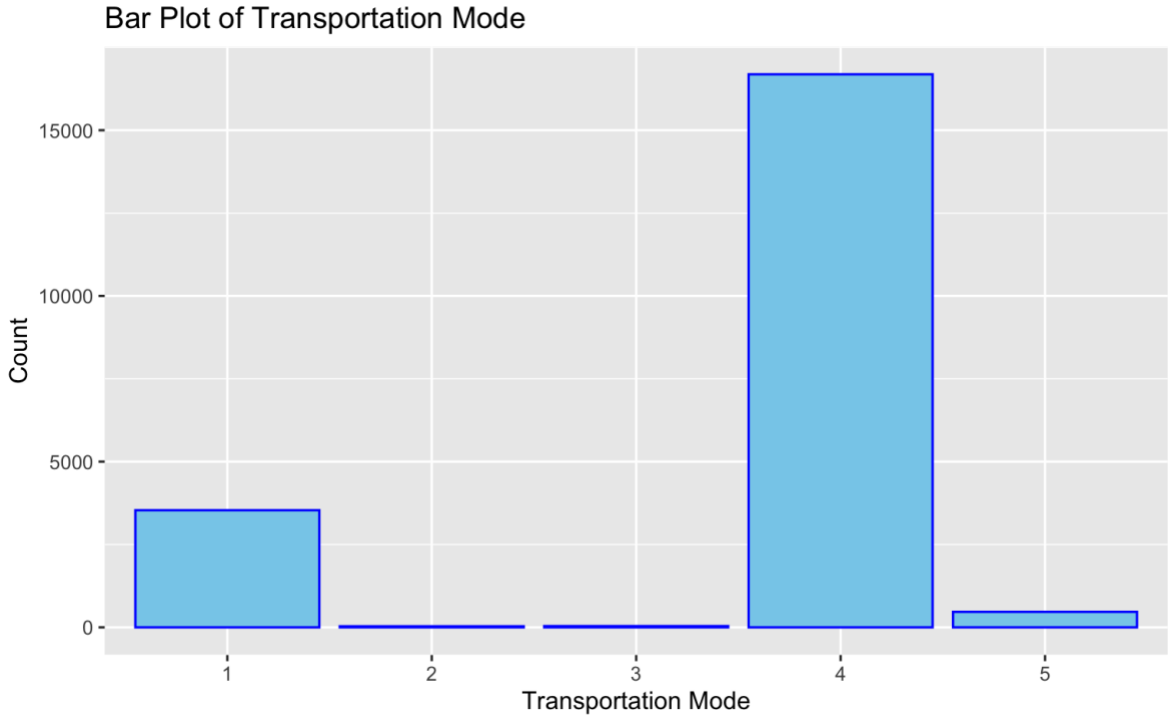





Bar Plot of Family High-Calorie Food Consumption
Frequency
15000
10000
5000
0
0
1
Family Consumes High-Calorie Food


Bar Plot of Smoking Status
Count
20000
15000
10000
5000
0
0
1
Smoking Status


Bar Plot of Monitor Calorie Intake
20000
15000
10000
5000
0
Count
0
1
Monitor Calorie Intake

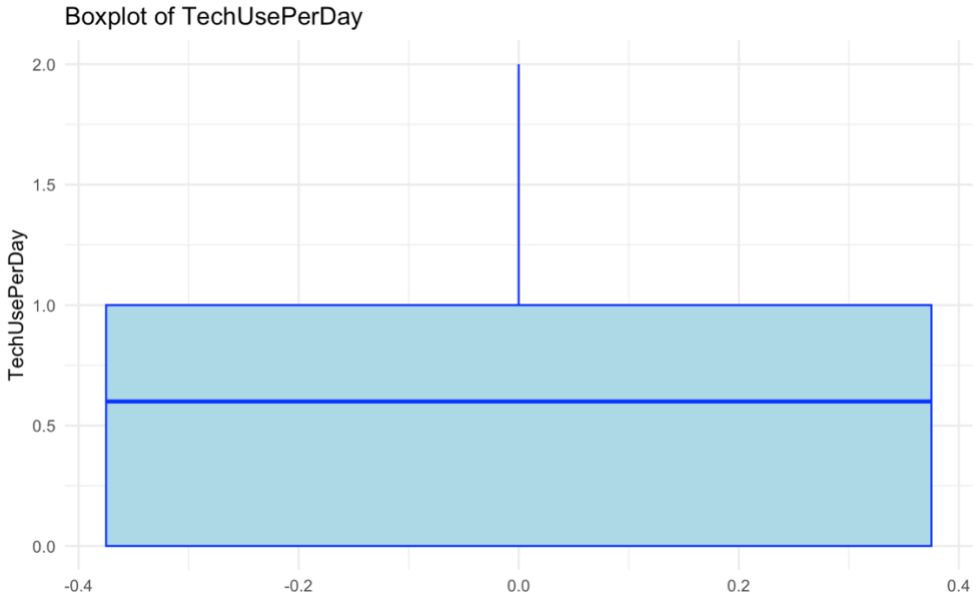

Bar Plot of Transportation Mode
15000
10000
5000
0
Count
1
2
3
4
5
Transportation Mode

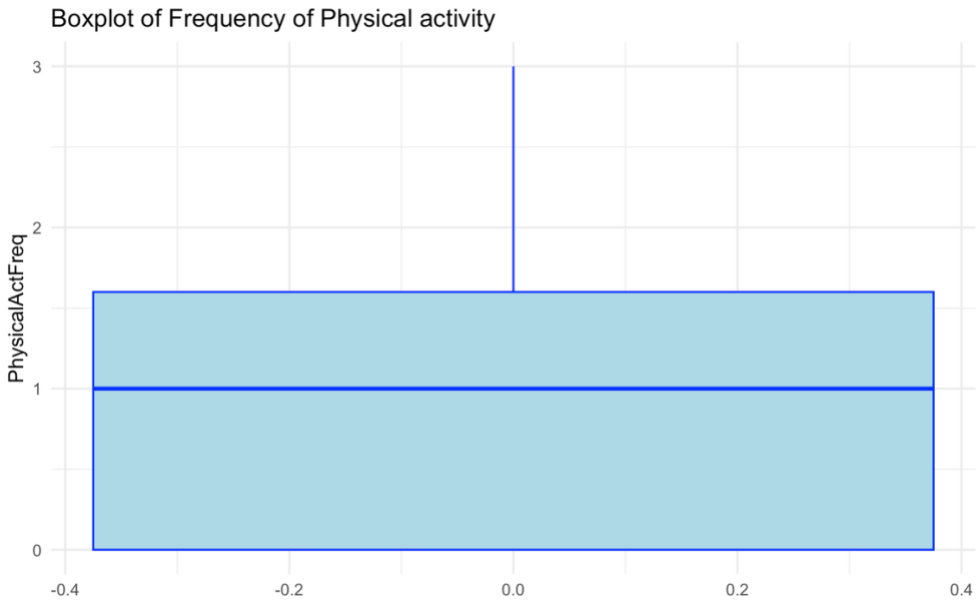
BOXPLOT code

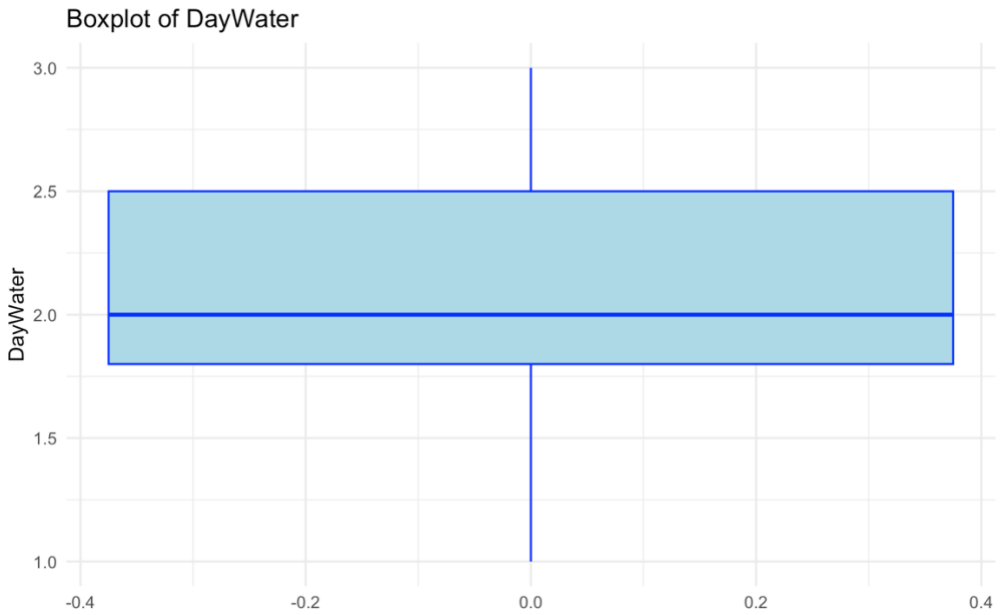
```
# FCvegetables
ggplot(data, aes( y = FCvegetables)) +
  geom_boxplot(fill = "lightblue", color = "blue", outlier.colour = 'red') +
  labs(title = "Boxplot of Frequent Consumption of vegetables",
  y = "FCvegetables") +
  theme_minimal()


# DayWater
ggplot(data, aes(y = DayWater)) +
  geom_boxplot(fill = "lightblue", color = "blue", outlier.colour = 'red') +
  labs(title = "Boxplot of DayWater",
       y = "DayWater") +
  theme_minimal()


# PhysicalActFreq
ggplot(data, aes(y = PhysicalActFreq)) +
  geom_boxplot(fill = "lightblue", color = "blue", outlier.colour = 'red') +
  labs(title = "Boxplot of Frequency of Physical activity",

       y = "PhysicalActFreq") +
  theme_minimal()


# TechUsePerDay
ggplot(data, aes(y = TechUsePerDay)) +
  geom_boxplot(fill = "lightblue", color = "blue", outlier.colour = 'red') +
  labs(title = "Boxplot of TechUsePerDay",

       y = "TechUsePerDay") +
  theme_minimal()

```

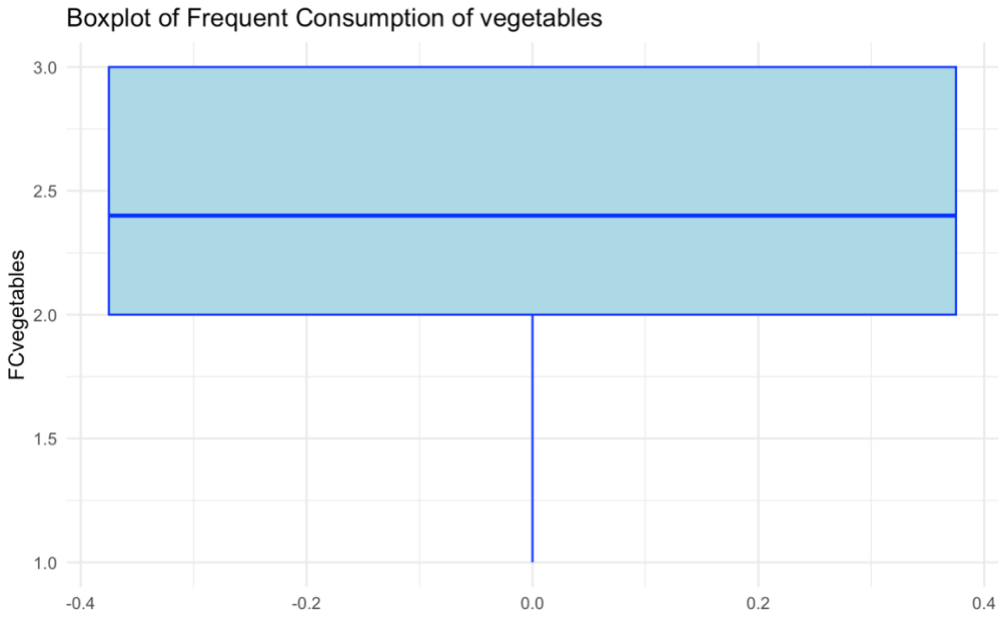
```{r}
# Boxplots
# Age
ggplot(data, aes(Age)) +
  geom_boxplot(fill = "lightblue", color = "blue", outlier.colour = 'red') +
  labs(title = "Box plot of Age",
       y = "Age") +
  theme_minimal()


# Height
ggplot(data = data, aes(Height)) +
  geom_boxplot(fill = "skyblue", color = "blue", outlier.colour = 'red') +
  labs( y = "Height", title = "Box Plot of Height")


# Weight
ggplot(data, aes(y = Weight)) +
  geom_boxplot(fill = "lightblue", color = "blue", outlier.colour = 'red') +
  labs(title = "Boxplot of Weight",

       y = "Weight") +
  theme_minimal()


# FCvegetables
ggplot(data, aes( y = FCvegetables)) +
  geom_boxplot(fill = "lightblue", color = "blue", outlier.colour = 'red') +
  labs(title = "Boxplot of Frequent Consumption of vegetables",
  y = "FCvegetables") +
  theme_minimal()
```

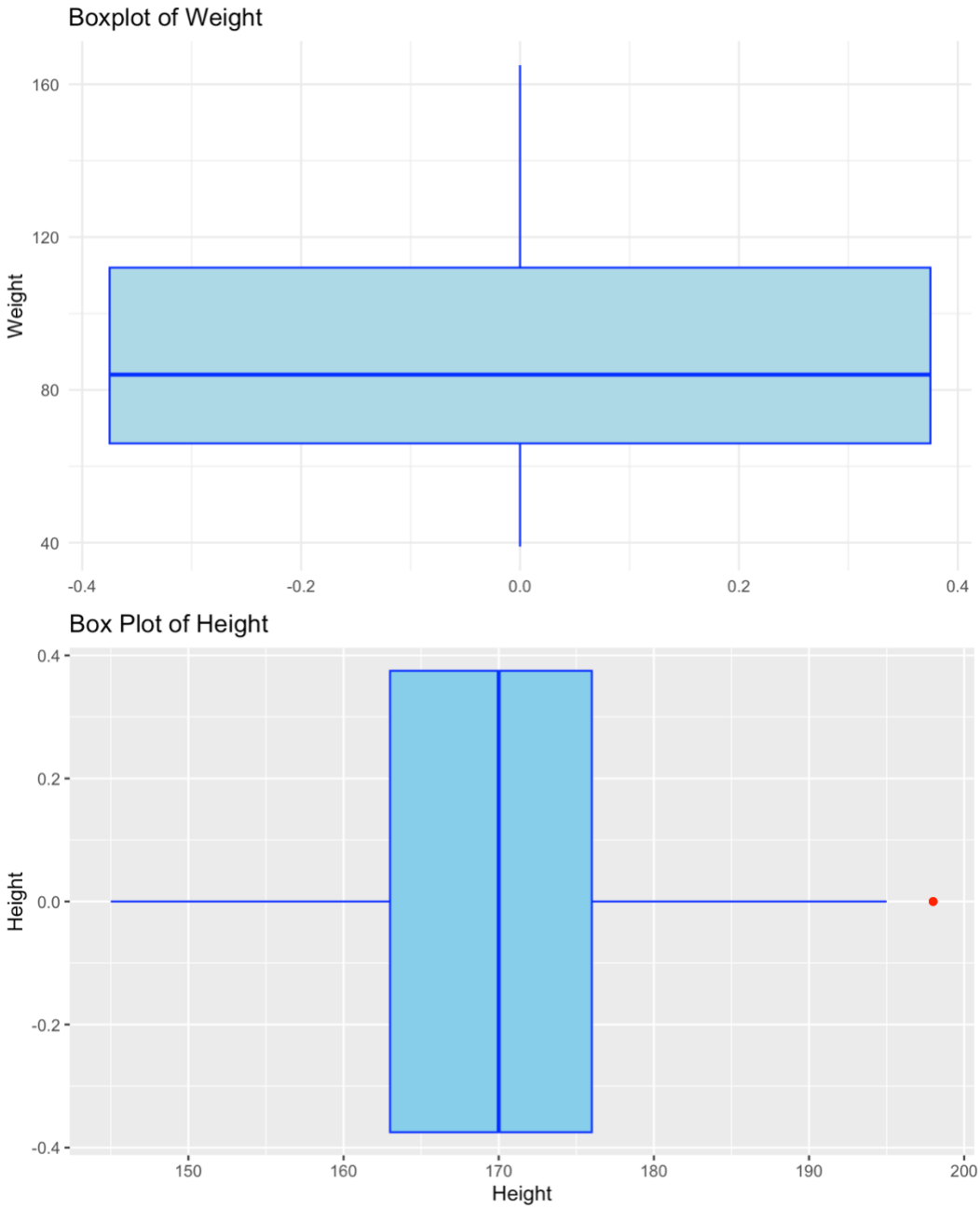
BOXPLOTS

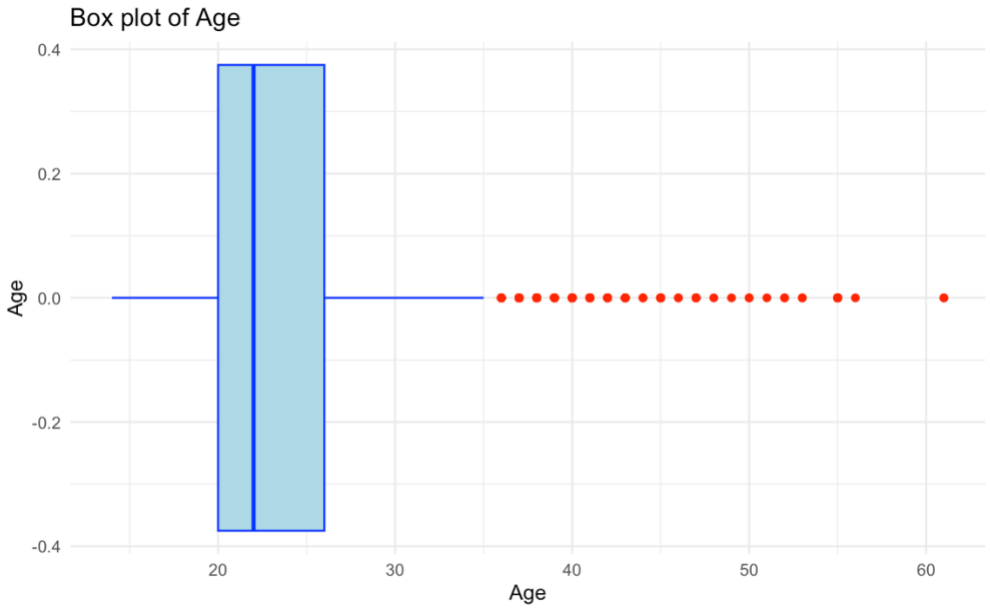

Boxplot of TechUsePerDay
TechUsePerDay
2.0
1.5
1.0
0.5
0.0
-0.4
-0.2
0.0
0.2
0.4


Boxplot of Frequency of Physical activity
PhysicalActFreq
3
2
1
0
-0.4
-0.2
0.0
0.2
0.4


Boxplot of DayWater
DayWater
3.0
2.5
2.0
1.5
1.0
-0.4
-0.2
0.0
0.2
0.4


Boxplot of Frequent Consumption of vegetables
FCvegetables
3.0
2.5
2.0
1.5
1.0
-0.4
-0.2
0.0
0.2
0.4

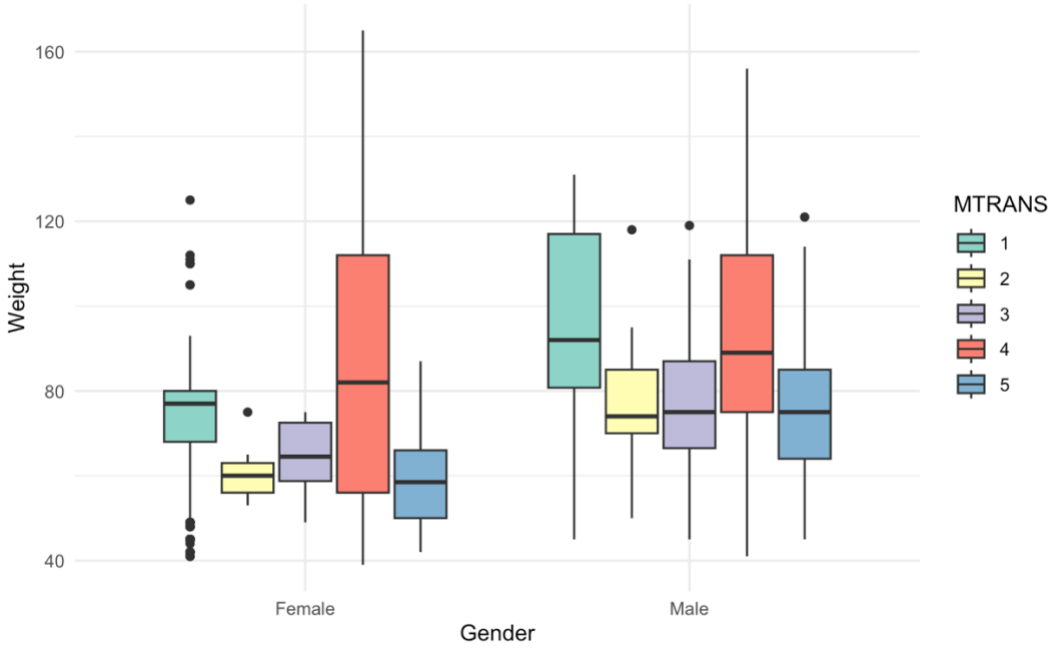

Boxplot of Weight
Weight
160
120
80
40
-0.4
-0.2
0.0
0.2
0.4
Box Plot of Height
Height
0.4
0.2
0.0
-0.2
-0.4
150
160
170
180
190
200
Height

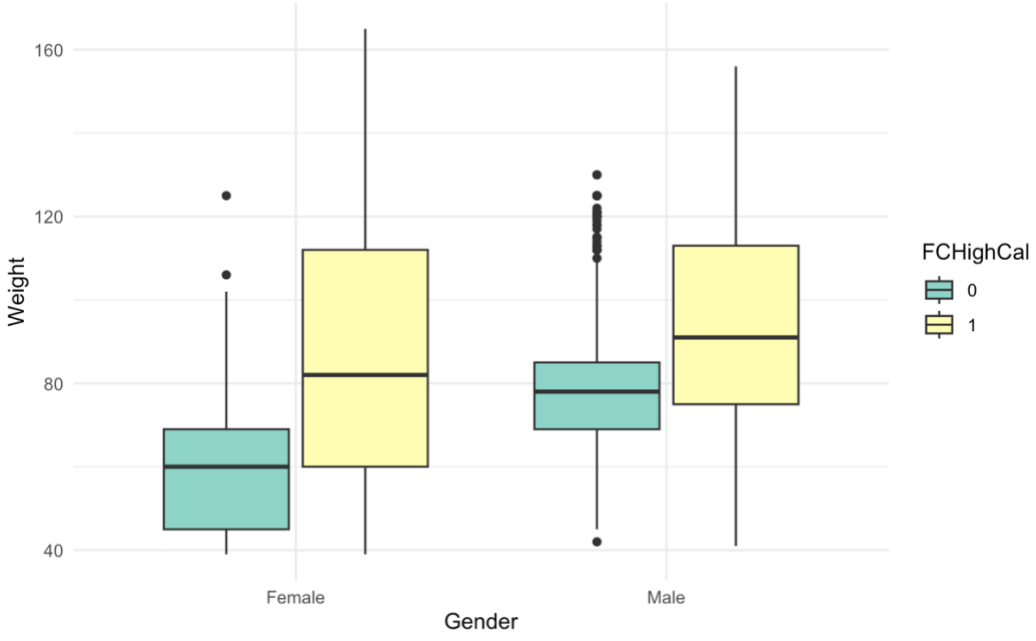

Box plot of Age
0.4
0.2
0.0
-0.2
-0.4
Age
20
30
40
50
60
Age

Sede by side box plot code

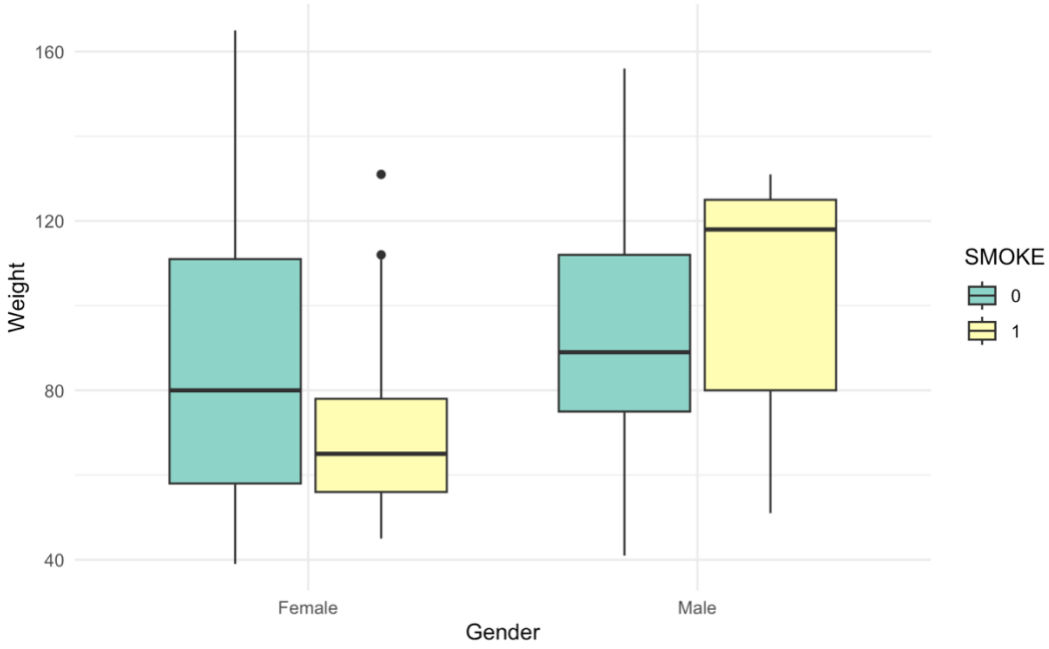
```
{r}
# Plot side-by-sides of both continuous and factor
# Weight & Obesity Level
ggplot(data = data, aes(x = factor(Gender), y = Weight, fill = ObesityLevel)) +
  geom_boxplot() +
  labs(x = "Gender", y = "Weight", fill = "Obesity Level") +
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()+
  scale_x_discrete(labels = c("Female", "Male"))

# Weight & MTRANS
ggplot(data = data, aes(x = factor(Gender), y = Weight, fill = MTRANS)) +
  geom_boxplot() +
  labs(x = "Gender", y = "Weight", fill = "MTRANS") +
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()+
  scale_x_discrete(labels = c("Female", "Male"))

# Weight & FCHighCal
ggplot(data = data, aes(x = factor(Gender), y = Weight, fill = FCHighCal)) +
  geom_boxplot() +
  labs(x = "Gender", y = "Weight", fill = "FCHighCal") +
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()+
  scale_x_discrete(labels = c("Female", "Male"))

# Weight & family_history_with_overweight
ggplot(data = data, aes(x = factor(Gender), y = Weight, fill = family_history_with_overweight)) +
  geom_boxplot() +
  labs(x = "Gender", y = "Weight", fill = "family_history_with_overweight") +
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()+
  scale_x_discrete(labels = c("Female", "Male"))
```

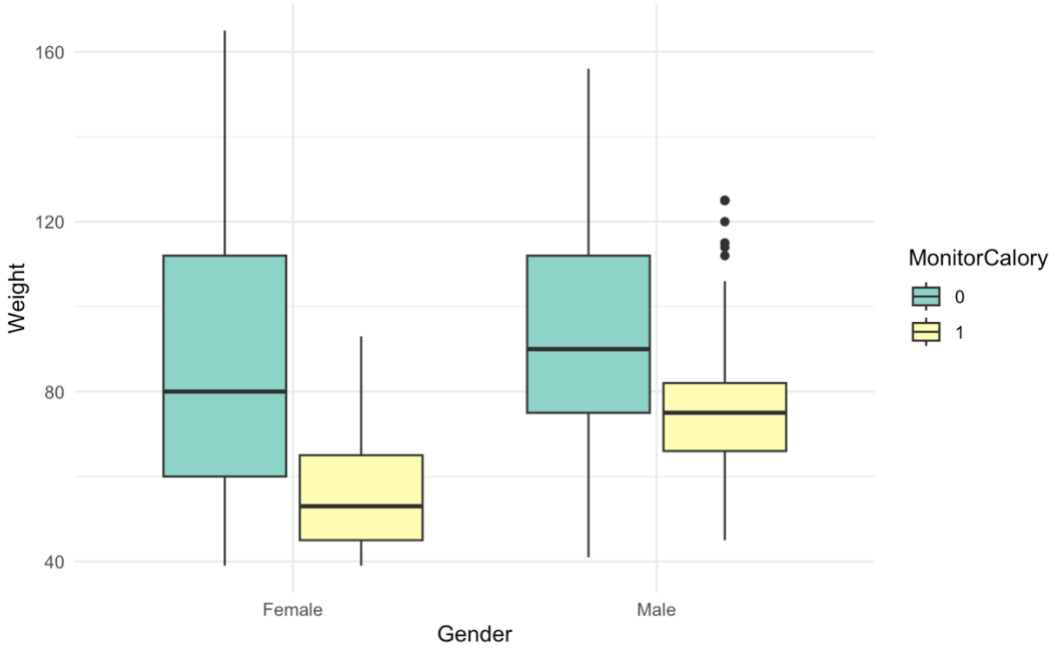
```
# Weight & SMOKE
ggplot(data = data, aes(x = factor(Gender), y = Weight, fill = SMOKE)) +
  geom_boxplot() +
  labs(x = "Gender", y = "Weight", fill = "SMOKE") +
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()+
  scale_x_discrete(labels = c("Female", "Male"))

# Weight & MonitorCalory
ggplot(data = data, aes(x = factor(Gender), y = Weight, fill = MonitorCalory)) +
  geom_boxplot() +
  labs(x = "Gender", y = "Weight", fill = "MonitorCalory") +
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()+
  scale_x_discrete(labels = c("Female", "Male"))

# Weight & Alcohol
ggplot(data = data, aes(x = factor(Gender), y = Weight, fill = Alcohol)) +
  geom_boxplot() +
  labs(x = "Gender", y = "Weight", fill = "Alcohol") +
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()+
  scale_x_discrete(labels = c("Female", "Male"))
```

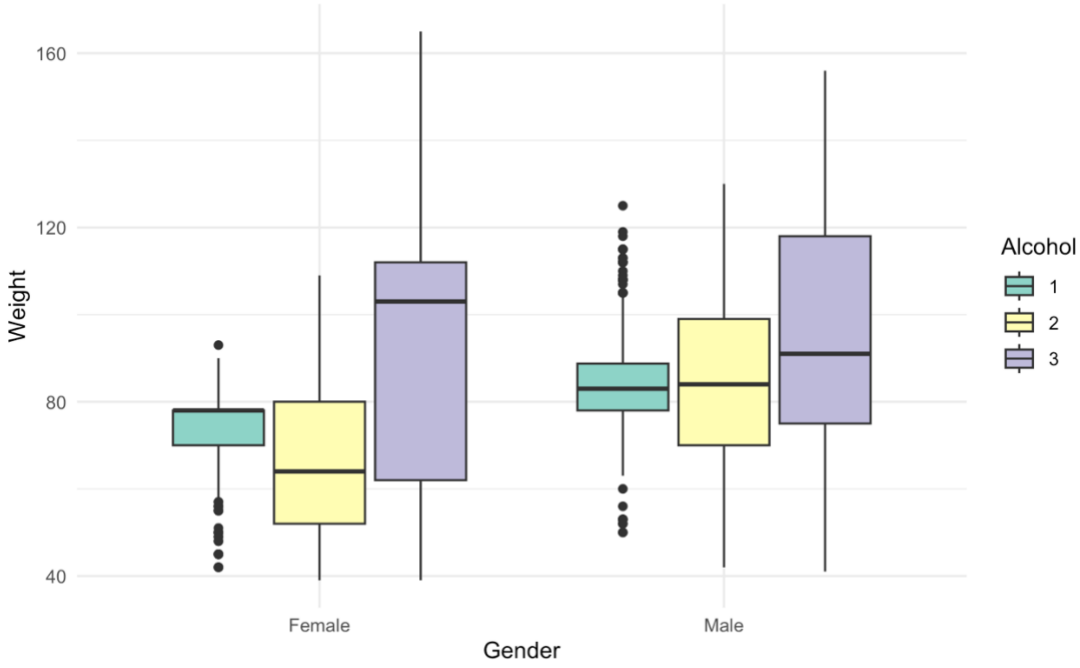

160
120
80
40
Weight
Female
Male
Gender
MTRANS
1
2
3
4
5


160
120
80
40
Weight
Female
Male
Gender
FCHighCal
0
1


160
120
80
40
Weight
Female
Male
Gender
family_history_with_overweight
0
1


160
120
80
40
Weight
Female
Male
Gender
SMOKE
0
1


160
120
80
40
Weight
Female
Male
Gender
MonitorCalory
0
1


160
120
80
40
Weight
Female
Male
Gender
Alcohol
1
2
3

```r
# Convert ObesityLevel from factor to numeric
data$ObesityLevel <- as.numeric(as.character(data$ObesityLevel))

# Calculate Pearson correlation coefficient between weight and obesity level
correlation_weight_obesity <- cor(data$Weight, data$ObesityLevel)

# Print the correlation coefficient
print(paste("Pearson correlation coefficient between weight and obesity level:", correlation_weight_obesity))
```

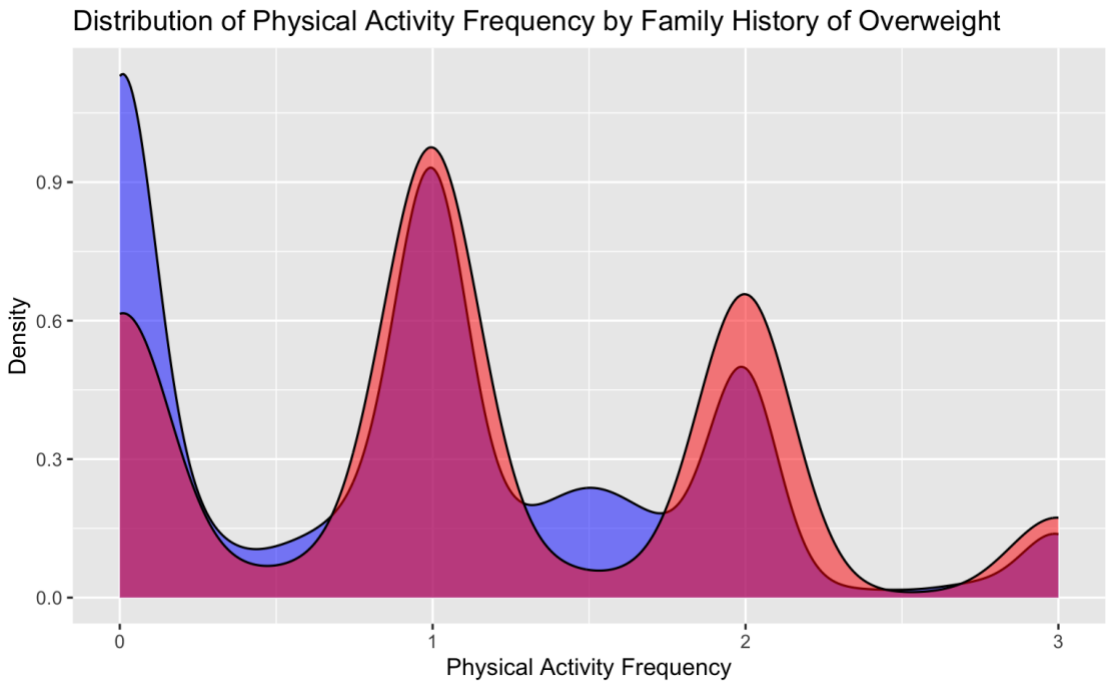
```
[1] "Pearson correlation coefficient between weight and obesity level: 0.921590274722955"
```

```r
# Subset the data into two groups: with and without family history of overweight
with_history <- subset(data, family_history_with_overweight == 1)
without_history <- subset(data, family_history_with_overweight == 0)

# Create histograms or density plots for physical activity frequency for each group
ggplot() +
  geom_density(data = with_history, aes(x = PhysicalActFreq), fill = "blue", alpha = 0.5) +
  geom_density(data = without_history, aes(x = PhysicalActFreq), fill = "red", alpha = 0.5) +
  labs(x = "Physical Activity Frequency", y = "Density", title = "Distribution of Physical Activity Frequency by Family History of Overweight") +
  scale_fill_manual(values = c("blue", "red"), labels = c("With History", "Without History"))

# Calculate summary statistics for physical activity frequency for each group
summary_physical_activity <- data.frame(
  Group = c("With History", "Without History"),
  Mean = c(mean(with_history$PhysicalActFreq), mean(without_history$PhysicalActFreq)),
  SD = c(sd(with_history$PhysicalActFreq), sd(without_history$PhysicalActFreq))
)

print(summary_physical_activity)
```

Group <chr>	Mean <dbl>	SD <dbl>
With History	0.944722	0.8284450
Without History	1.150214	0.8650021

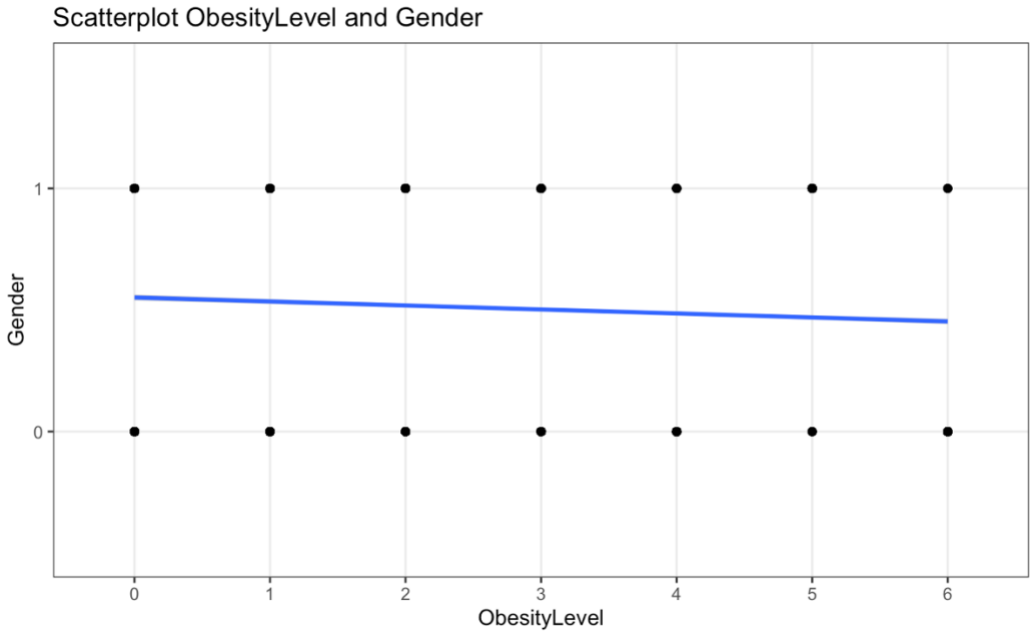
2 rows

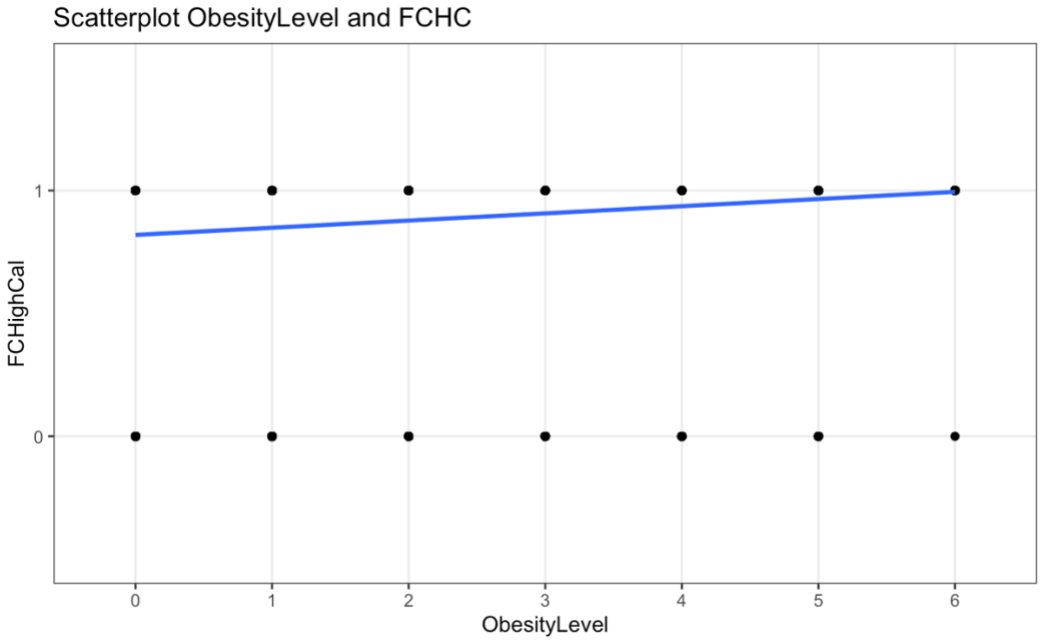
This density plot shows that objects without a familial history of obesity excercise more than those that do.

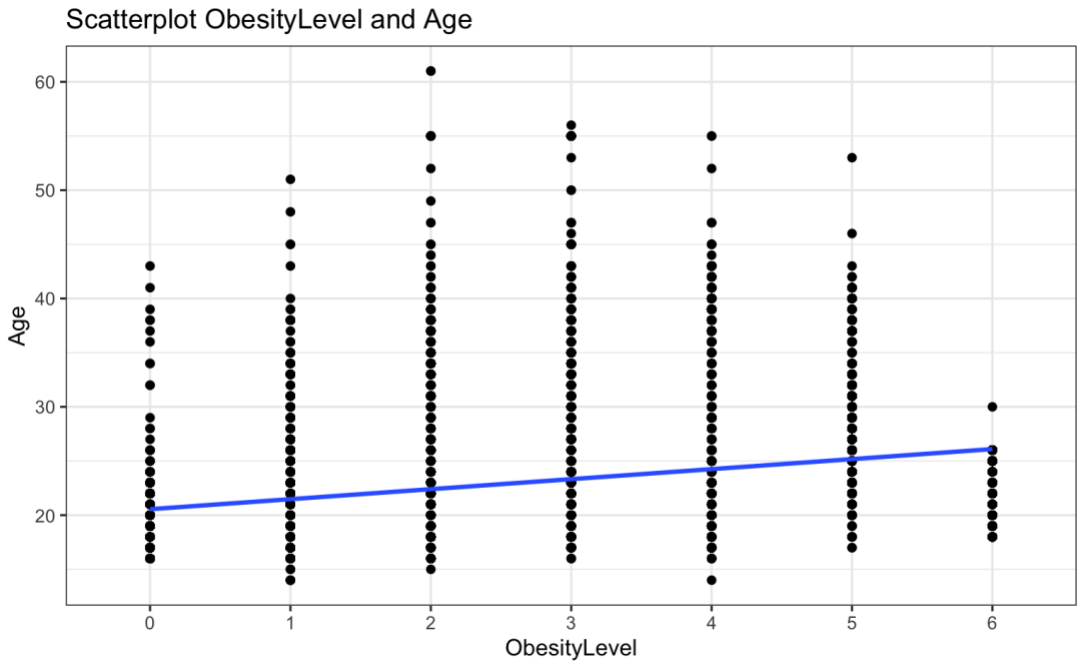


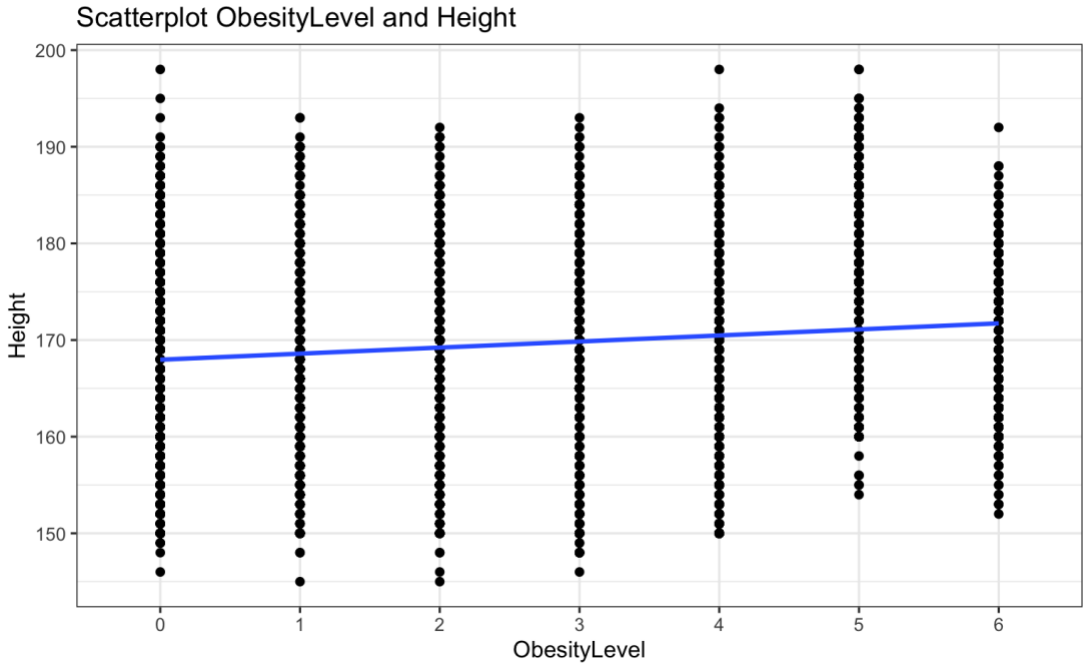

```r
# Scatterplots
# Scatterplots of ObesityLevel(target var) with all variables
ggplot(data, aes(x=ObesityLevel, y=Gender, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and Gender")

ggplot(data, aes(x=ObesityLevel, y=Age, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and Age")

ggplot(data, aes(x=ObesityLevel, y=Height, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and Height")

ggplot(data, aes(x=ObesityLevel, y=Weight, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and Weight")

ggplot(data, aes(x=ObesityLevel, y=family_history_with_overweight, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot
ObesityLevel and family_history_with_overweight")

ggplot(data, aes(x=ObesityLevel, y=FCHighCal, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and FCHC")

ggplot(data, aes(x=ObesityLevel, y=FCvegetables, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and
FCvegetables")

ggplot(data, aes(x=ObesityLevel, y=NumMainMeals, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and
NumMainMeals")

ggplot(data, aes(x=ObesityLevel, y=ConsFoodBetwMeal, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and
ConsFoodBetwMeal")

ggplot(data, aes(x=ObesityLevel, y=SMOKE, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and SMOKE")

ggplot(data, aes(x=ObesityLevel, y=MonitorCalory, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and
MonitorCalory")

ggplot(data, aes(x=ObesityLevel, y=DayWater, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and WayWater")

ggplot(data, aes(x=ObesityLevel, y=PhysicalActFreq, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and
PyysicalActFreq")

ggplot(data, aes(x=ObesityLevel, y=TechUsePerDay, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and
TechUsePerDay")

ggplot(data, aes(x=ObesityLevel, y=MTRANS, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and MTRANS")

ggplot(data, aes(x=ObesityLevel, y=Alcohol, group=1)) +geom_point() +geom_smooth(method = 'lm') +theme_bw() +ggtitle("Scatterplot ObesityLevel and Alcohol")
```

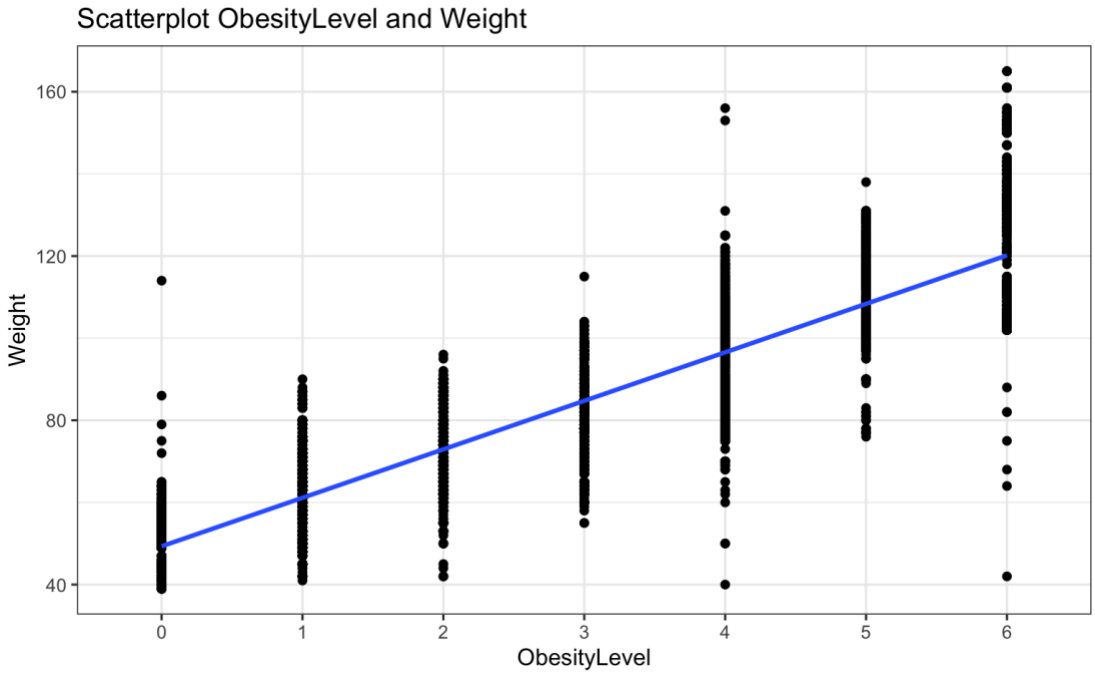
SCATTERPLOTS

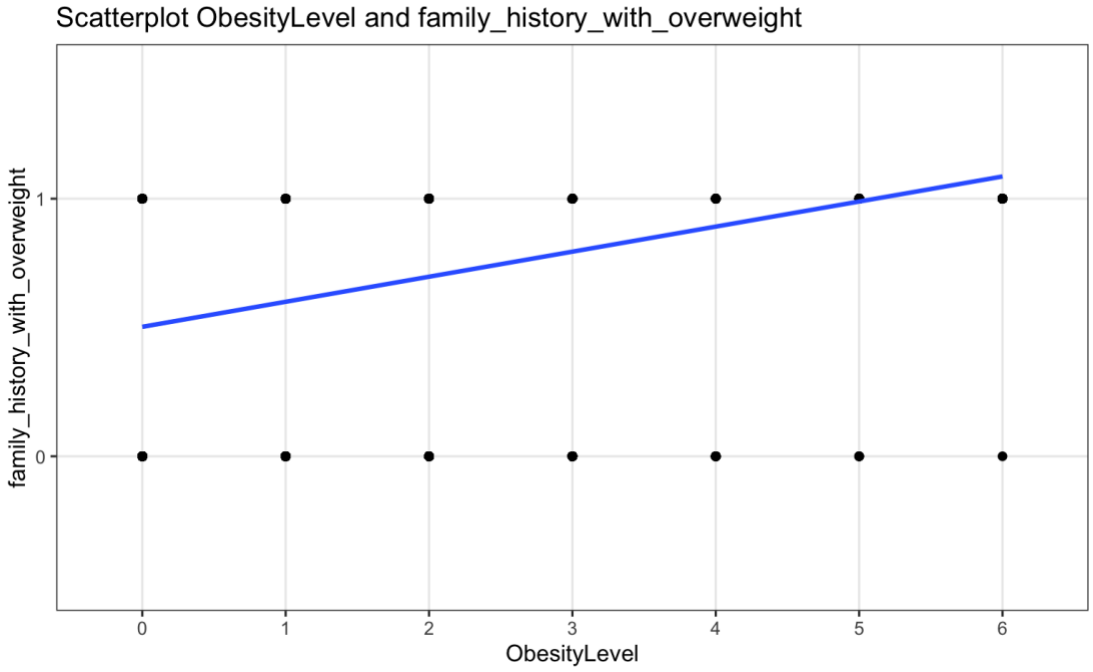


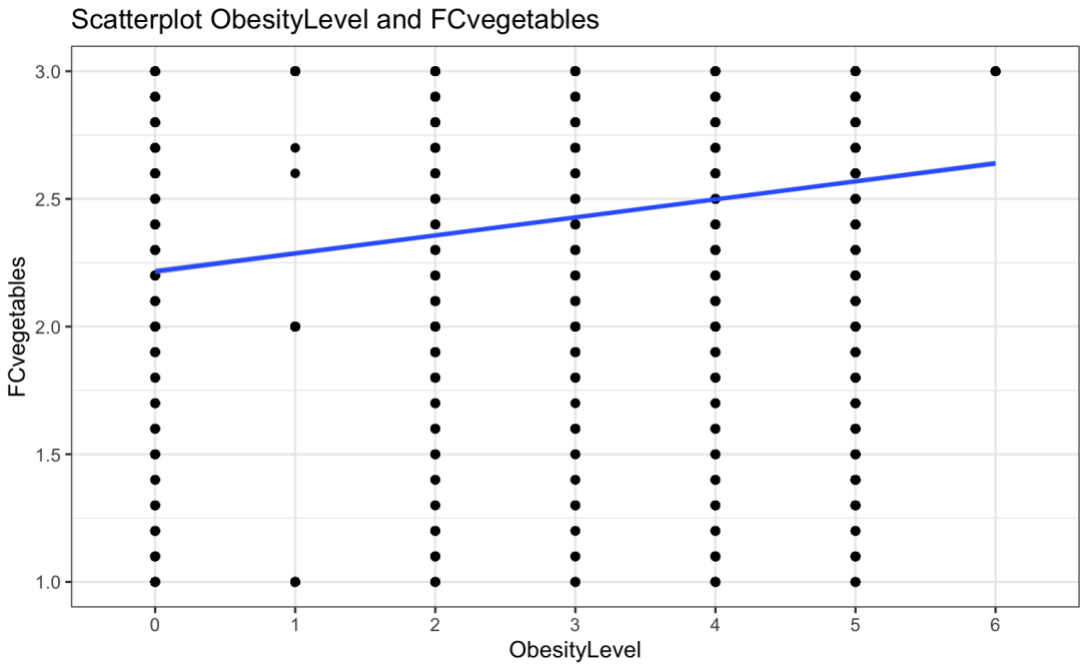


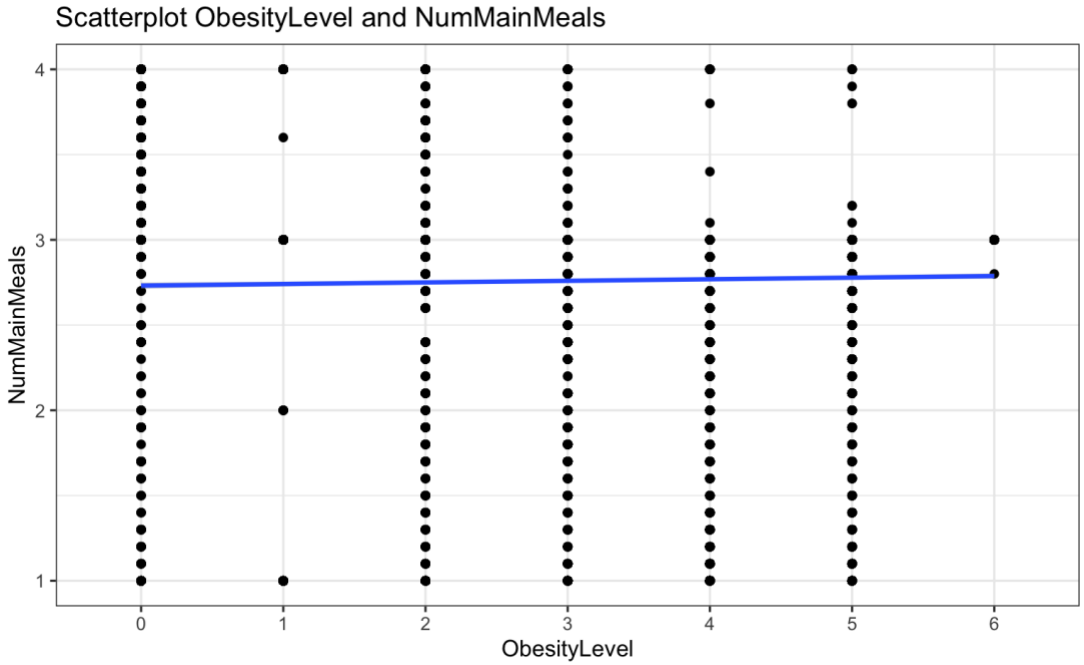

Scatterplot ObesityLevel and Age
Age
60
50
40
30
20
0
1
2
3
4
5
6
ObesityLevel

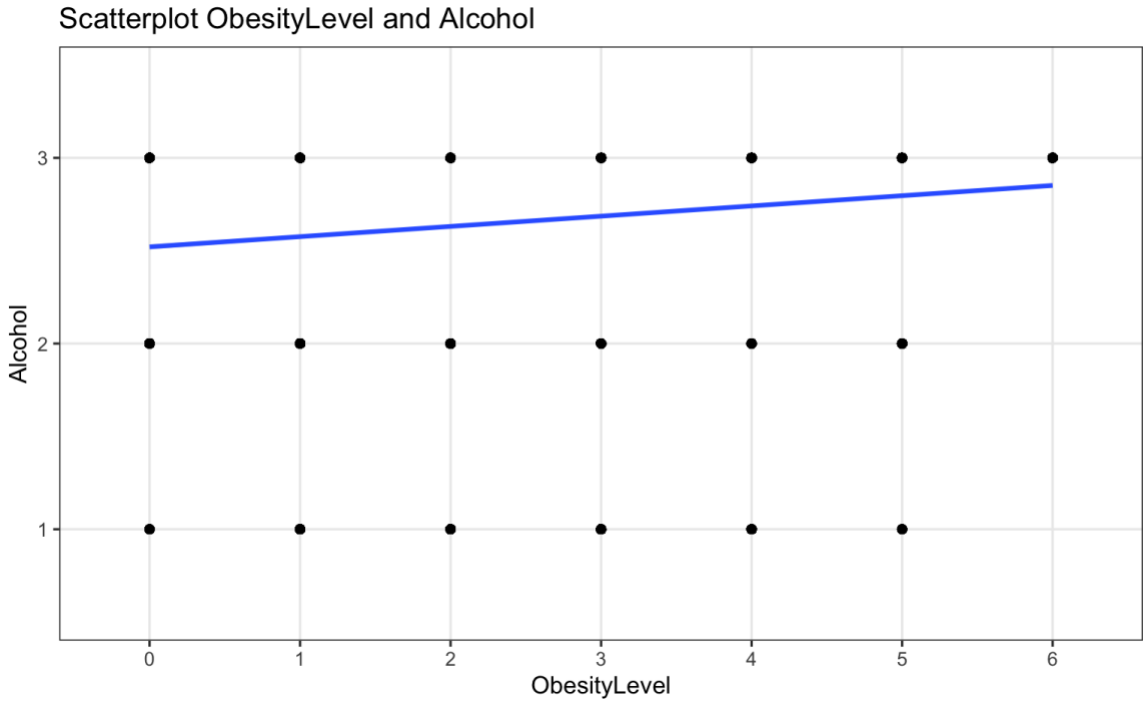

Scatterplot ObesityLevel and Height
Height
200
190
180
170
160
150
0
1
2
3
4
5
6
ObesityLevel

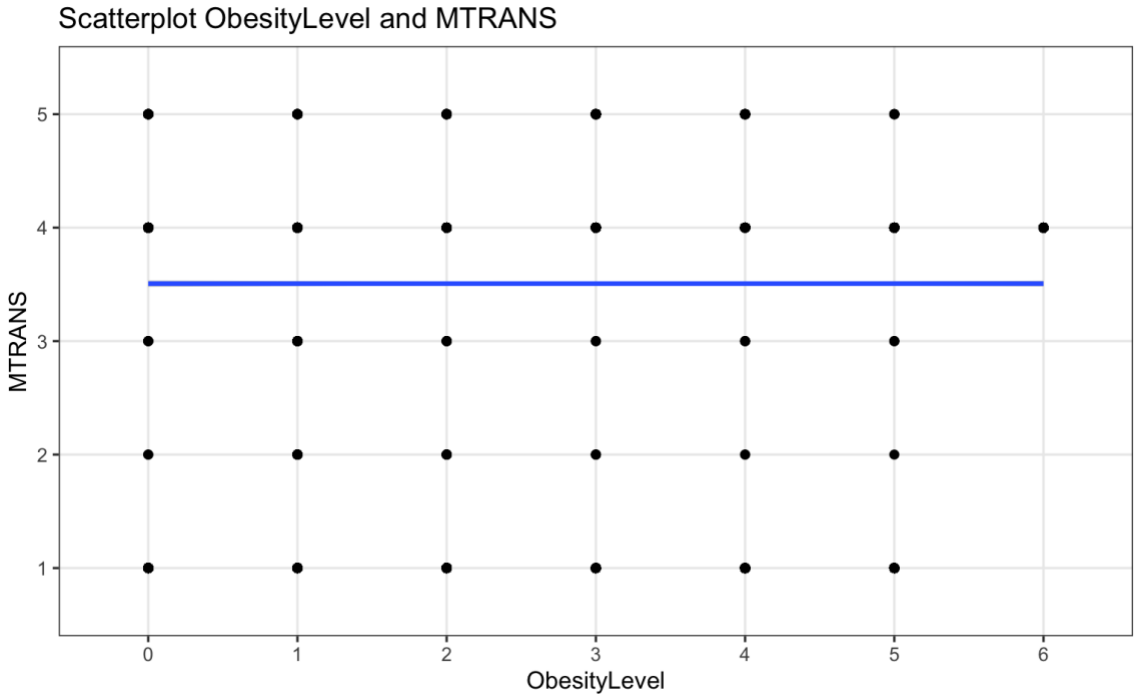

Scatterplot ObesityLevel and Weight
Weight
160
120
80
40
0
1
2
3
4
5
6
ObesityLevel

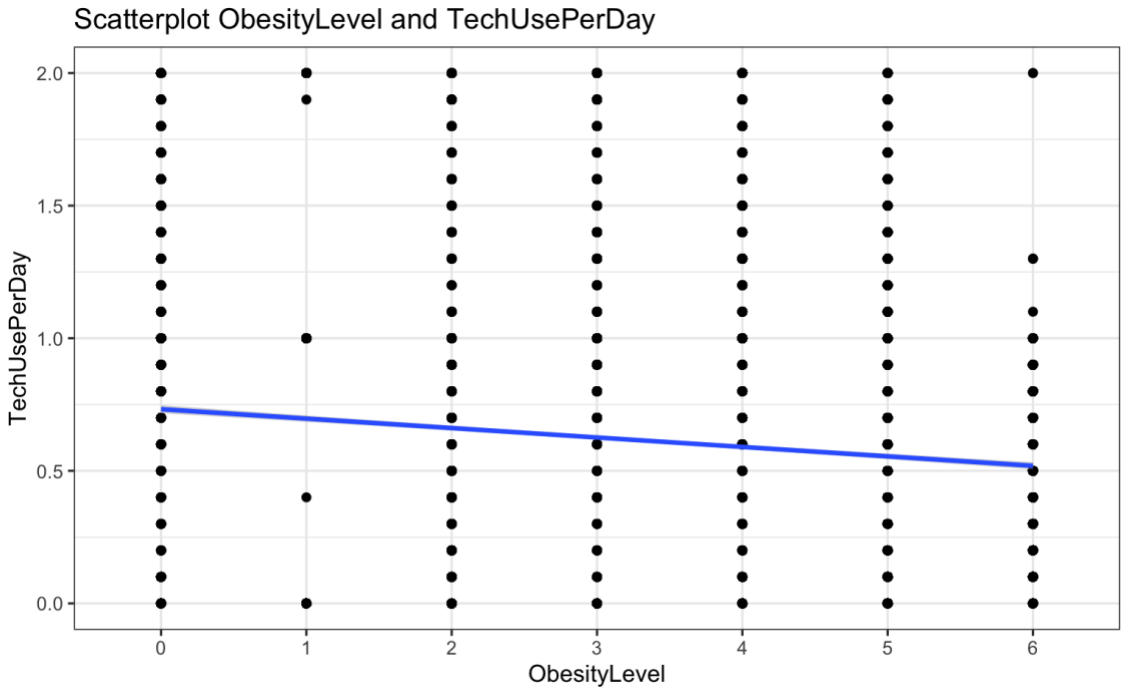

Scatterplot ObesityLevel and family_history_with_overweight
family_history_with_overweight
1
0
0
1
2
3
4
5
6
ObesityLevel

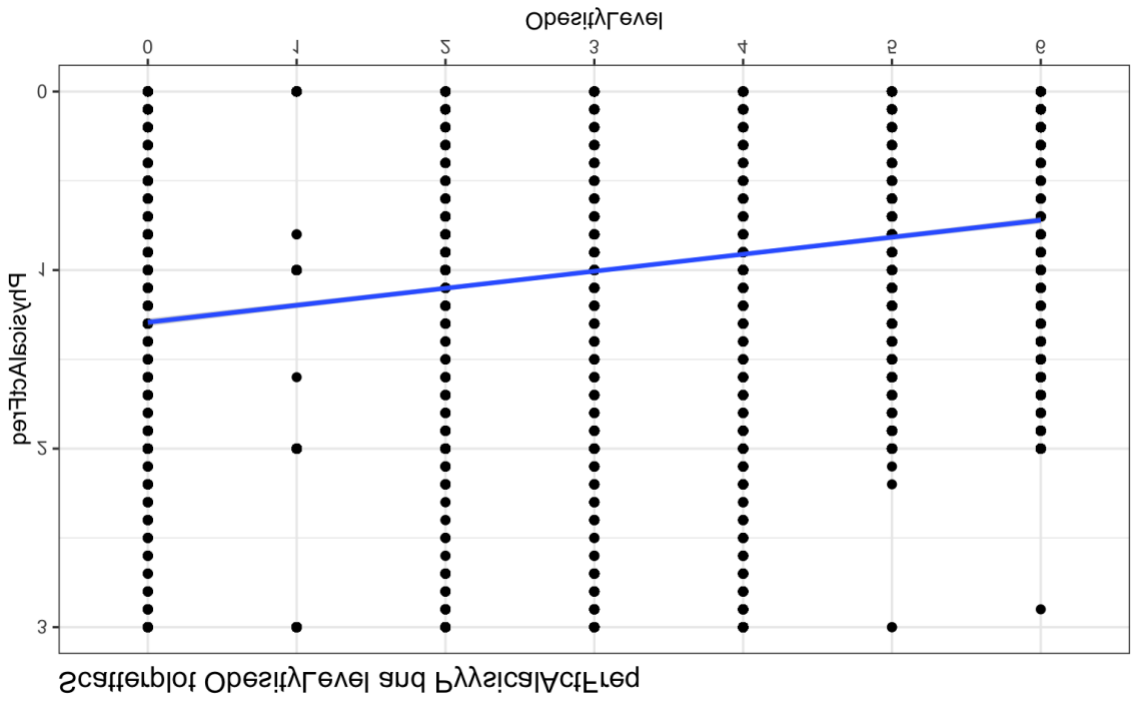

Scatterplot ObesityLevel and FCvegetables
FCvegetables
3.0
2.5
2.0
1.5
1.0
0
1
2
3
4
5
6
ObesityLevel

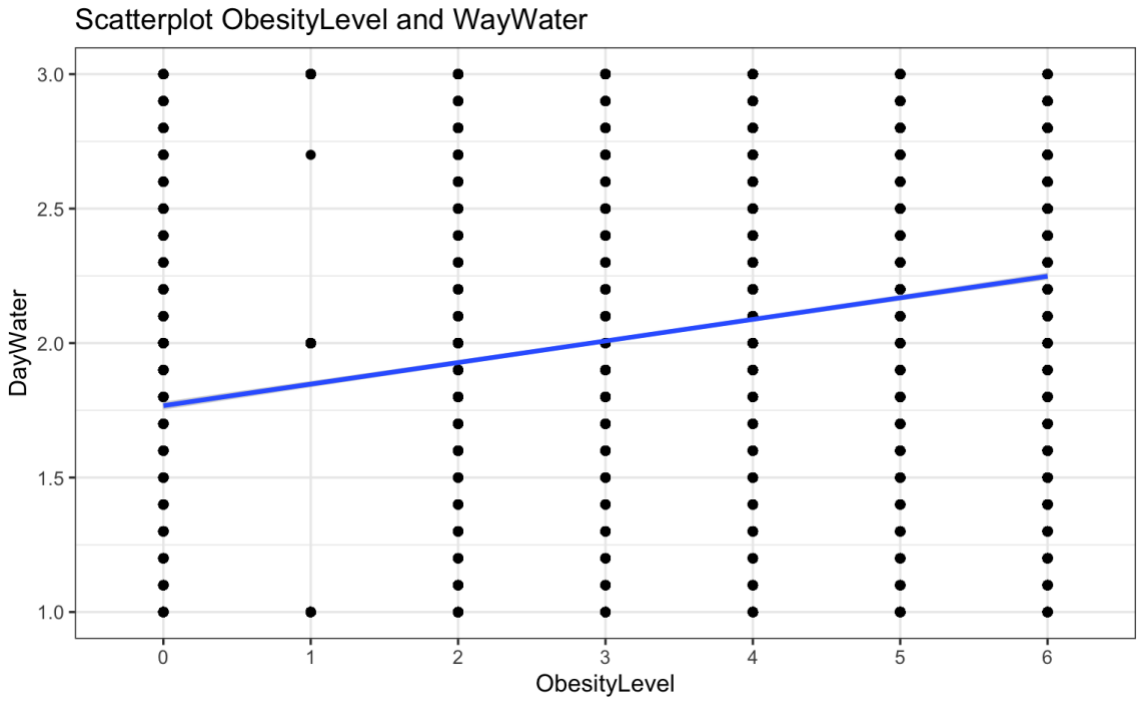

Scatterplot ObesityLevel and NumMainMeals
NumMainMeals
4
3
2
1
0
1
2
3
4
5
6
ObesityLevel

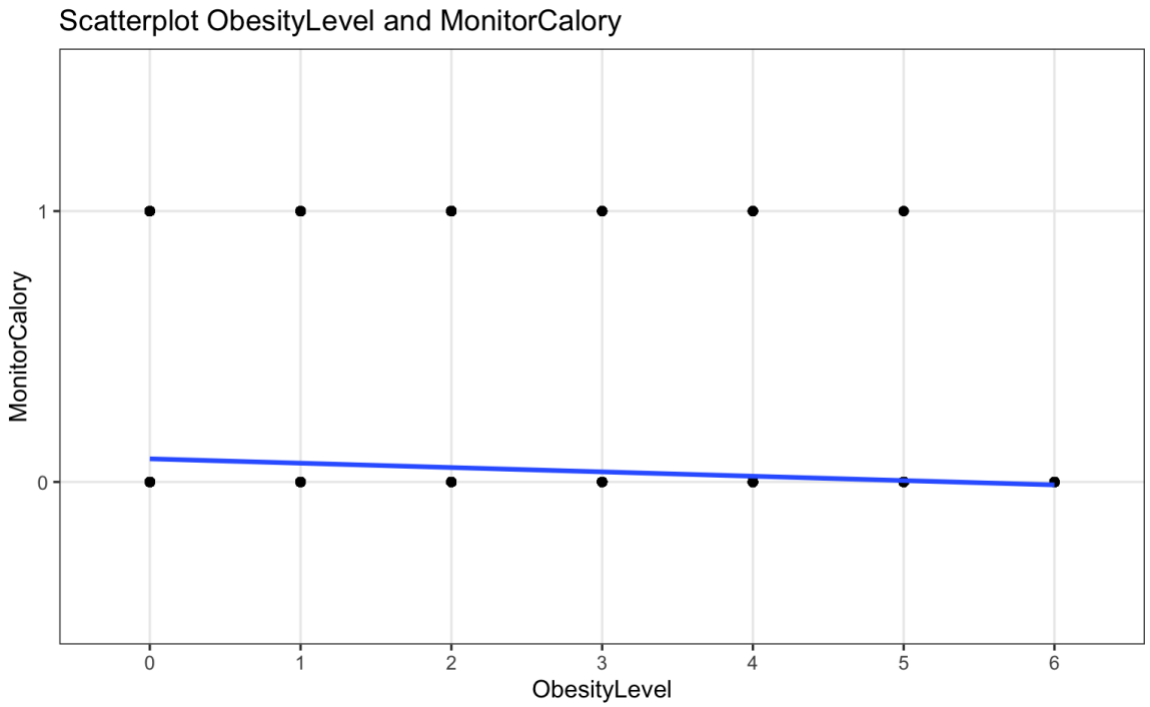

Scatterplot ObesityLevel and Alcohol
Alcohol
ObesityLevel
1
2
3
0
1
2
3
4
5
6

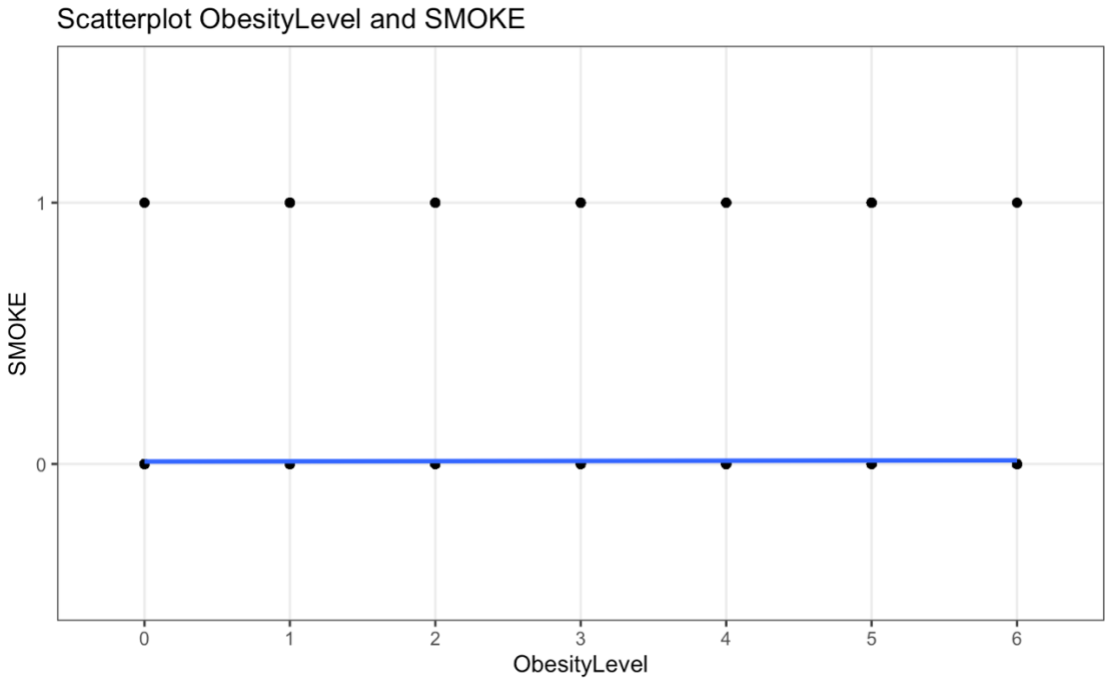

Scatterplot ObesityLevel and MTRANS
MTRANS
ObesityLevel
1
2
3
4
5
0
1
2
3
4
5
6

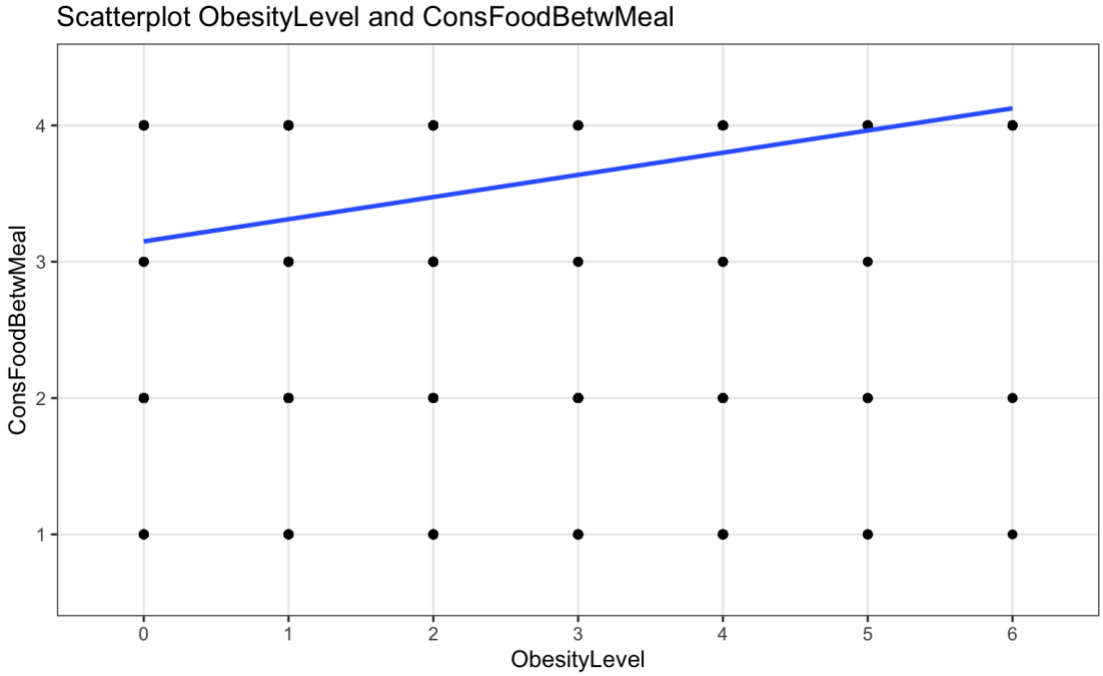

Scatterplot ObesityLevel and TechUsePerDay
TechUsePerDay
2.0
1.5
1.0
0.5
0.0
0
1
2
3
4
5
6
ObesityLevel


ObesityLevel
0
1
2
3
4
5
6
PhysicalActFreq
0
1
2
3


Scatterplot ObesityLevel and WayWater
DayWater
3.0
2.5
2.0
1.5
1.0
0
1
2
3
4
5
6
ObesityLevel

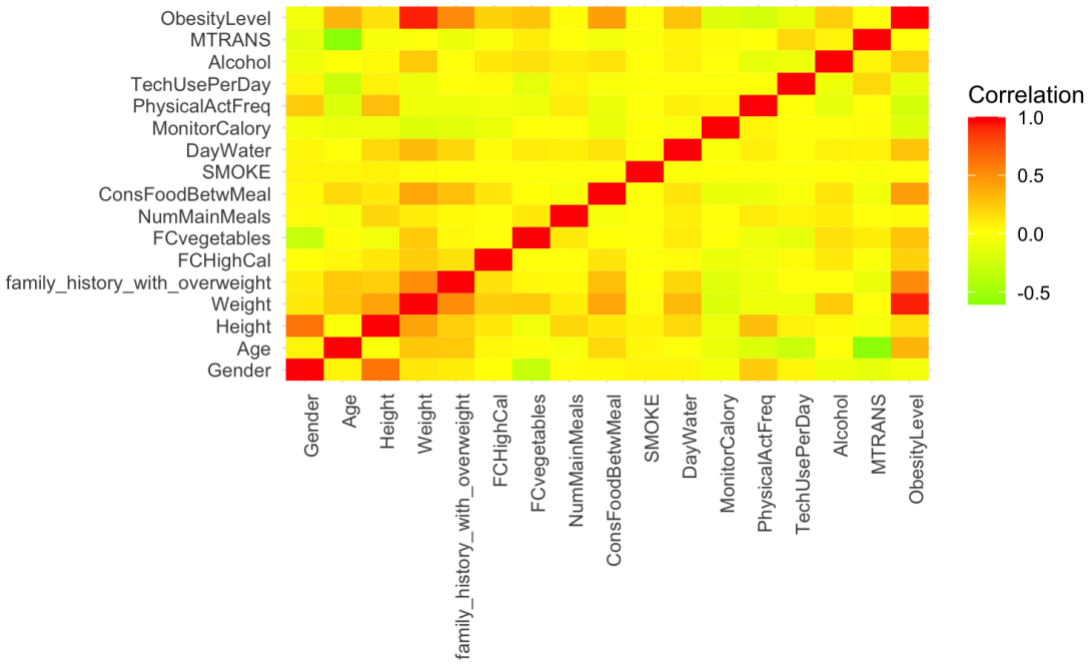

Scatterplot ObesityLevel and MonitorCalory
MonitorCalory
1
0
0
1
2
3
4
5
6
ObesityLevel


Scatterplot ObesityLevel and SMOKE
SMOKE
1
0
0
1
2
3
4
5
6
ObesityLevel


Scatterplot ObesityLevel and ConsFoodBetwMeal
ConsFoodBetwMeal
4
3
2
1
0
1
2
3
4
5
6
ObesityLevel

HEATMAP

```r
# Heatmap
num_data <- data[, sapply(data, is.numeric)]

# Calculate the correlation matrix on just the numeric data
cor_matrix <- cor(num_data, use = "complete.obs")

# Melt the correlation matrix into long format
melted_matrix <- melt(cor_matrix)

# Create the heatmap using ggplot2
heatmap_plot <- ggplot(melted_matrix, aes(Var1, Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "green", high = "red", mid = "yellow", midpoint = 0) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        axis.text.y = element_text(angle = 0, hjust = 1)) +
  labs(x = '', y = '', fill = 'Correlation')

# Print the heatmap plot
print(heatmap_plot)
```

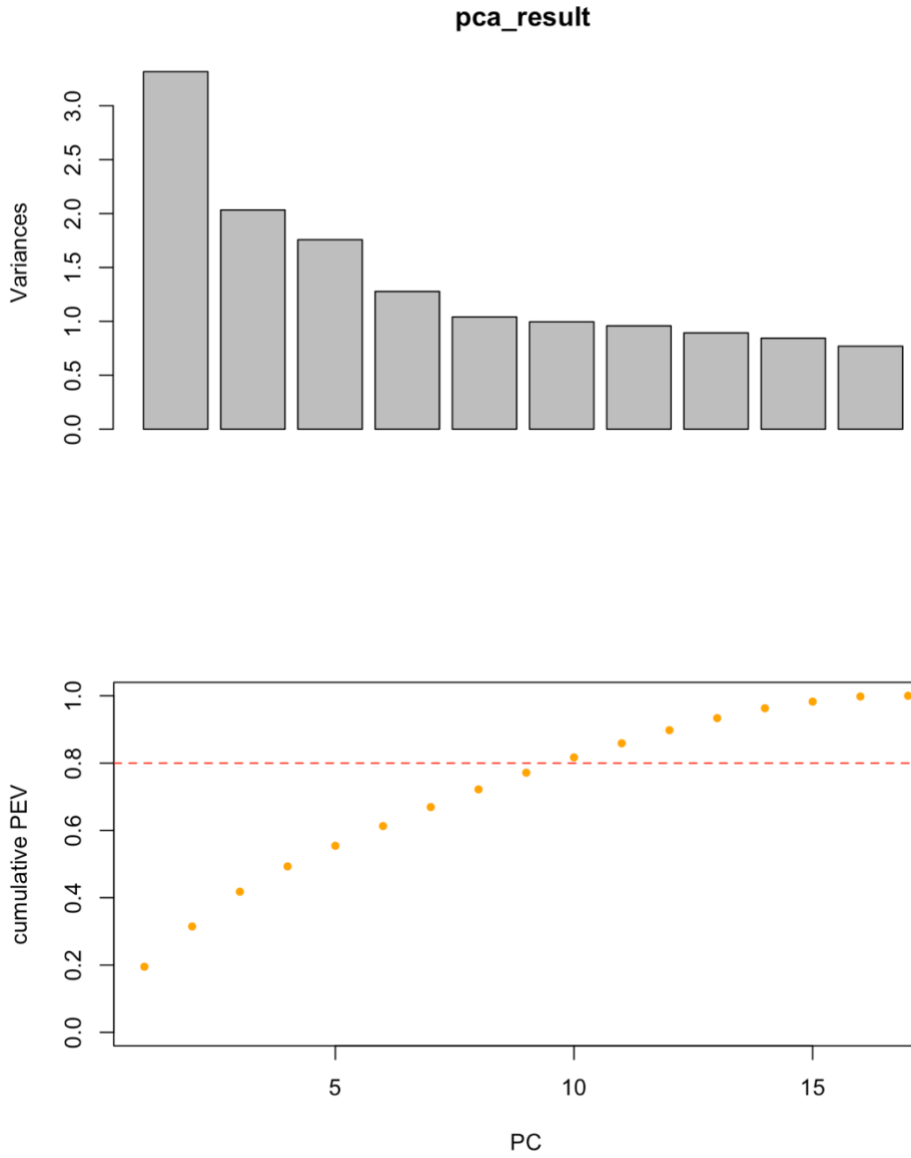


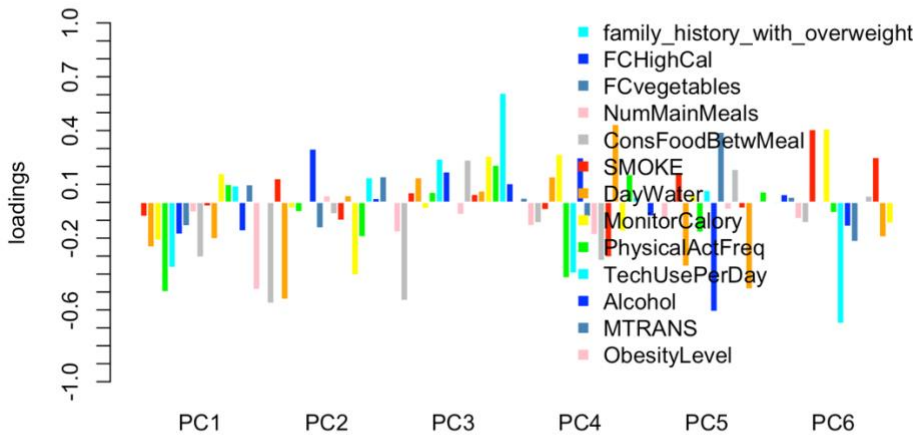
PCA

```r
# Principal Component Analysis (PCA)
# Split the data into training and test sets
set.seed(123)
splitIndex <- createDataPartition(data$ObesityLevel, p = 0.7, list = FALSE)
train <- data[splitIndex, ]
test <- data[-splitIndex, ]

# Select only numeric columns (excluding the target variable)
numeric_columns <- sapply(train[, names(train) != "ObesityLevel"], is.numeric)
train_numeric <- train[, numeric_columns]

# Scale the numeric training data
train_scaled <- scale(train_numeric)

# Apply PCA
pca_result <- prcomp(train_scaled, center = TRUE, scale. = TRUE)
attributes(pca_result)

# Calculate the proportion of explained variance (PEV) from the std values
pca_var <- pca_result$sdev^2
pca_PEV <- pca_var/sum(pca_var)
plot(pca_result)

# plot the cumulative PEV
opar <- par(no.readonly = TRUE)
plot(
  cumsum(pca_PEV),
  ylim = c(0,1),
  xlab = 'PC',
  ylab = 'cumulative PEV',
  pch = 20,
  col = 'orange'
)
abline(h = 0.8, col = 'red', lty = 'dashed')
abline(h = 0.8, col = 'red', lty = 'dashed')
par(opar)

# Get and inspect the loadings
pca_loadings <- pca_result$rotation
pca_loadings
```

```
0.5832291761
Alcohol                          0.41041327 -0.044930383 -0.381995641  0.22184406  0.114204844
0.1847588597
MTRANS                           0.11296924  0.147129798 -0.061474868 -0.17942516  0.044546786
-0.3105681905
ObesityLevel                    -0.02613767 -0.031774603  0.010211510 -0.13149719  0.034704024
-0.0365646997
                                       PC13        PC14         PC15         PC16         PC17
Gender                          0.273758731 -0.13153897  0.461209755 -0.465138076 -0.018458880
Age                            -0.037900928  0.20417049  0.519428009  0.493923008  0.045982831
Height                          0.150534518 -0.08408743 -0.417338404  0.563555137 -0.174954686
Weight                         -0.021646741  0.38020905 -0.152706288 -0.145150532  0.706769097
family_history_with_overweight -0.525504602 -0.58100902  0.055837174  0.005786286  0.020392808
FCHighCal                       0.117915326 -0.04517570  0.068863014  0.009154263 -0.001370212
FCvegetables                    0.260100442 -0.36927703  0.146332543 -0.065246128 -0.011393509
NumMainMeals                    0.121999994  0.07774859  0.078332770 -0.031997562 -0.006359799
ConsFoodBetwMeal                0.299289873 -0.22921706  0.042481269  0.070970545  0.015841364
SMOKE                           0.027009072 -0.03384560 -0.014957570 -0.025368280 -0.007447823
DayWater                        0.165277035 -0.09722295  0.027389615  0.020347544 -0.005922542
MonitorCalory                   0.002936986  0.02161463  0.009387043  0.020117288 -0.001427489
PhysicalActFreq                -0.486388791  0.17062115  0.157807618 -0.046912257 -0.036372031
TechUsePerDay                  -0.119128930  0.12412737  0.105975633  0.026506690  0.001140370
Alcohol                        -0.394835289 -0.12630098  0.053014119 -0.032105036 -0.024447002
MTRANS                          0.038963511  0.09506543  0.492395319  0.366026634  0.030188499
ObesityLevel                   -0.045686369  0.41742731 -0.017776122 -0.226311047 -0.680917314
```

PCA result


pca_result
Variances
0.0
0.5
1.0
1.5
2.0
2.5
3.0
cumulative PEV
0.0
0.2
0.4
0.6
0.8
1.0
5
10
15
PC



CLSUTERING

```r
```{r}
# DBSCAN clustering
# Select the first 6 principal components
pcs <- pca_result$x[, 1:6]

# Perform DBSCAN clustering with different parameter values
dbscan_result <- dbscan(pcs, eps = 0.5, minPts = 5)

# Print the clusters
print(dbscan_result$cluster)

# Visualize clusters if possible
# For example, if you have 2D or 3D data, you can plot the clusters
if (ncol(pcs) == 2) {
  plot(pcs, col = dbscan_result$cluster + 1, pch = 20, main = "DBSCAN Clustering")
} else if (ncol(pcs) == 3) {
  scatter3D(pcs[,1], pcs[,2], pcs[,3], colvar = dbscan_result$cluster + 1, pch = 20, main = "DBSCAN Clustering")
}

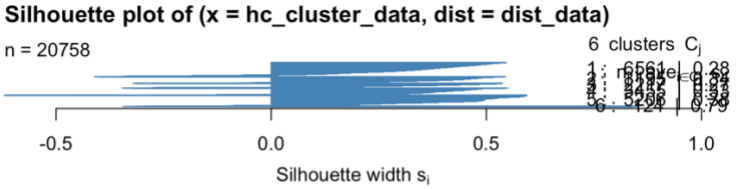
# Evaluate clustering if ground truth labels are available
# For example, using silhouette score
silhouette_score <- silhouette(dbscan_result$cluster, dist(pcs))
```
```

```
  [1]   3   0   0   3   1   1   0   2   0   3   0   3   0 147  60   2   0   0   2   0   0   0   2   0   1   0   3
 [28]   0  78   0   3   2   0   0   0   2   4   2   0   5   0   0   0   0   6   0 143   0   3   0   7   3   3   0
 [55]   3   2   0   0   0   0   3   0   0   0   3   2   0   2   0   0   3  51   2   3   0   3   0   2   2   2   2
 [82]   0   8   0   9   0   0   0   2   2   0   0   2   0   3   0   0   0   3   0   3   2  10   0   2   0   3   0
[109]   0   2   0   0   0 134   0   0   1  48   0  44   0   3   0   3   0   0   0   3   0   0   2   0   0   2   0
[136]  70   2   0   0   0   0   3   3   3   3   2   0   1   3   1   0   3   2   0   0  11   2   3   0   2   0   2
[163]   0   2   2   0   0   2   0  32   3   3   3   0  12   0   3   0   0   2   2   0   0   2   0   6   2   0   0
[190]   3   3   0   3   0   0   3   2   0   0   0   1   3   0  13   0   0   2   0   0  83   2   0   3   0   0  59
[217]   0   1   3   1   0   0   0   4   0   0   0   1   0   3  14   0   3   2   0   0   0  99   2   0   2   2   2
[244]   0   1   2   0   3   0   0   0   0   2  15   2   0   2   3   3   0   2   0   0   3   0   3   2   0   2   0
[271]  16   0   3   2   0   2   0   2   2   1   3  17   0   0   2   3   0   3   3   0   2   0   0   0   0   2   2
[298]   0   0   0   2   3   0   3   3   0   1   3   0   2  18   1   0   3  15   3   0   3   0   2  39   2   0   0
[325]   2   2   0   0   2   0   3   0   2  42   4   2   3   0  19   0   2   3   3   3   2   3   0   0   0   3   3
[352]   3   2   2   3   2   0  20   0   0   7   2  54   0  21   2   0   3 133   2   3   0   0   0   2  20   0   0
[379]   1   0   0   0   0   1   0   0   0   3   1   2   0   2   0 135   0   2   0   0   2   0   3   0   2   0   2
[406]   3 131   0   0   2   2   2   2   0   0   2   2   0   0   0   0   2   0   0   2   0   2   0   1   3   2   0
[433]   0   6   3   0  12   0   3   0  78   0   2   3   2   1   0   0   0   0   2  54   3   3   3   3   0   3   2   0
[460]   0   4  28   0 123   3   2   0   0   2   3   0   0   3   3   0   3   0   3   2   0   3   2  22   0   0   0
[487]  23   2   2   0  24   4   2  25   2   0   0   4   3   0   2   3   2   0   0   0   2   0   3   0   2   3   0
[514]   0   0   3   3   3   0   1   2   0   1   3   0   0   2   2   0 133   2   2   0   2   3   2  22   0   0   0
[541]   0   0   2 103  26   2   0   3   3   2   2   0   1   0   0   3   2   0   2   2   0   3  27   0   0   0  14
[568]   0   0   0   3   3   0   3   0   0   2   0   0   0   3   3   0   2   3   0   0   2   0   0   3   0   0   0
[595]   1   0  28  29   0   0   3  83   2   0  30   0   2   0  54   0   0   0  76   2   0   0   0   0   0   0  43
[622]   0   3   3   2   0   2   0   2  90   0   2   0   0   3  96   2   0   3   0   3   3   0   3   0   0   3  20
[649]  16   2   0   0   0   0   0   0   0   2   0   3   3   3   0   2   2   3   2   2   0   0   4   2   0   0   0
[676]   0  31   0   3   2   2   0   3   0   1   3   0   0   3   0   0   2   0  32   0   2   2   3   0 110 132   0
[703]   0   3   0   3   0   0   1   3   2   0   0   0   0  33  24   3   0  34   3   2   3   0   1   0   0   0   0
[730]  17   3   3  43  54   0   0   3   0  35   1   0  12   2   2   3   3  33   0   0   0   1   0   0   2   0   2
[757]   0   0   2   3   0   0   2   3  36   1  37   2   0   0   0   0   3   2   0   0   0   2   0   0   0   3   3
[784]   2   0   2   0   0   0   0   0   0   0   0  24   0   0   2   3   2   3  33   2   0   0   2   2   3  76   3
[811]   2   2   0  38   2   0   3   2   0  39   9   0   3   0   3  43   0   4   3  40   2   0   0 124   2   1  54
[838]   0   0   2   2   0   2   1   0   3   2   2   3   0   2   2   3   0   3  41   0   3   2   3   3   2   0   0
[865]   3   2   2   0   0   3   0   3   0   0   0   0  42   0   3   3   3   2   0   0   0 148   2   1   0   2   0
[892]   3   3   2   0   3   0   0   0   0   2   1   0  33   0  42   3   0   0   0   0   0   2   2   0  43   0   0
[919]   2   0   0   0  91   0   3   2   3   0   2   3   0   2   0   0   2   4   0   4   3   3   3   3   3   3   3
```

```r
# Hierarchichal clustering
# First generate the distance matrix with euclidian distance
dist_data <- dist(data[,-17], method = 'euclidian')
# Apply complete linkage
hc_data <- hclust(dist_data, method = 'complete')
hc_data

# plot the associated dendrogram
plot(hc_data, hang = -0.1, labels = data$ObesityLevel)

# 'cut' the dendrogram to select one partition with 6 groups
hc_cluster_data <- cutree(hc_data, k = 6)
```

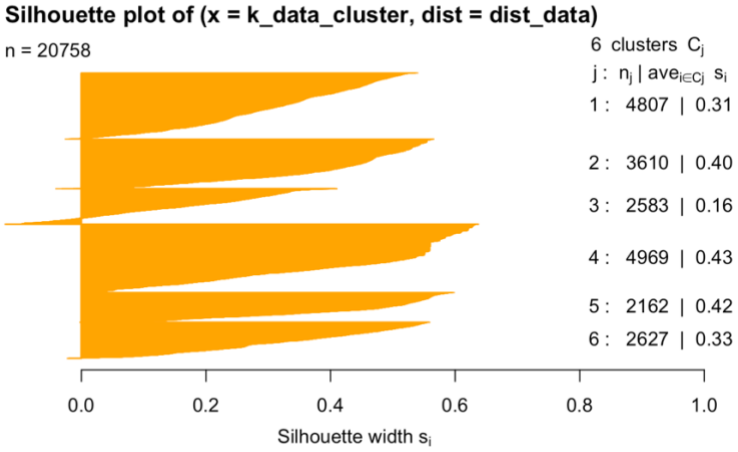
```r
# K-means
k_data = kmeans(data[,-17], 6)
k_data

# Get the cluster id from the kmeans object
k_data_cluster <- k_data$cluster

# Calculate the silhoutte score for the two cluster solutions
sil_hc_data <- cluster::silhouette(hc_cluster_data, dist_data)
sil_k_data <- cluster::silhouette(k_data_cluster, dist_data)

# Plot the results of the silhoutte analysis for the two cluster solutions
opar <- par(no.readonly = TRUE)
par(mfrow = c(2,1))
plot(sil_hc_data, border = 'steelblue')
plot(sil_k_data, border = 'orange')
par(opar)

# Plot the k-means clus
plot(sil_k_data, border = 'orange')
```

Silhouette



Average silhouette width : 0.32

Average silhouette width : 0.35



Average silhouette width : 0.35

ML: SUPPORT VECTOR MACHINES

```
Confusion Matrix and Statistics

          Reference
Prediction   3   1   0   6   5   2   4
         3 365   5   2   0   5  70  41
         1   7 515  23   0   0  52   0
         0   0  51 478   0   0   3   1
         6   0   0   0 808   1   0   2
         5   3   0   0   0 631   0  24
         2  79  44   1   1   0 349  16
         4  50   1   0   0  12  11 498

Overall Statistics

               Accuracy : 0.8783
                 95% CI : (0.8679, 0.8881)
    No Information Rate : 0.195
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8573

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 3 Class: 1 Class: 0 Class: 6 Class: 5 Class: 2 Class: 4
Sensitivity           0.72421   0.8360   0.9484   0.9988   0.9723  0.71959   0.8557
Specificity           0.96626   0.9768   0.9849   0.9991   0.9923  0.96152   0.9793
Pos Pred Value        0.74795   0.8626   0.8968   0.9963   0.9590  0.71224   0.8706
Neg Pred Value        0.96203   0.9716   0.9928   0.9997   0.9948  0.96283   0.9765
Prevalence            0.12148   0.1485   0.1215   0.1950   0.1564  0.11690   0.1403
Detection Rate        0.08797   0.1241   0.1152   0.1947   0.1521  0.08412   0.1200
Detection Prevalence  0.11762   0.1439   0.1285   0.1955   0.1586  0.11810   0.1379
Balanced Accuracy     0.84523   0.9064   0.9667   0.9989   0.9823  0.84055   0.9175
```

```r
```{r}
cm <- confusionMatrix(predictions, as.factor(test$ObesityLevel))
cm_table <- as.table(cm$table)
melted_cm <- melt(cm_table)
names(melted_cm) <- c("Reference", "Prediction", "Count")

#heatmap
ggplot(data = melted_cm, aes(x = Reference, y = Prediction, fill = Count)) +
  geom_tile() +
  scale_fill_gradient(low = "white", high = "blue") +
  theme_minimal() +
  labs(x = "Reference", y = "Prediction", fill = "Count") +
  ggtitle("Confusion Matrix Heatmap") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```
```

HPCI : SPARK , SVM

QUESTIONS

```
### Q1 Is there a difference in obesity prevalence between genders?


from pyspark.sql.functions import col


total_gender = df.groupBy('Gender').count().withColumnRenamed('count', 'TotalCount')
obese_gender = df.filter(col('ObesityLevel') > 0).groupBy('Gender').count().withColumnRenamed('count', 'ObeseCount')
prevalence = total_gender.join(obese_gender, 'Gender', 'left_outer').fillna(0)
obesity_prev = prevalence.withColumn('ObesityPrevalence',
        prevalence['ObeseCount'] / prevalence['TotalCount'])
obesity_prev.show()
```

```
+------+----------+----------+------------------+
|Gender|TotalCount|ObeseCount| ObesityPrevalence|
+------+----------+----------+------------------+
|   1.0|     10336|      9434|0.9127321981424149|
|   0.0|     10422|      8801| 0.844463634619075|
+------+----------+----------+------------------+
```

```
### Q2 Are individuals with higher physical activity levels less likely to be obese?

activ = df.groupBy('PhysicalActFreq').count().withColumnRenamed('count', 'TotalCount')
obese_activ = df.filter(col('ObesityLevel') > 0).groupBy('PhysicalActFreq').count().withColumnRenamed('count', 'ObeseCount')

prev_activ = activ.join(obese_activ, 'PhysicalActFreq', 'left_outer').fillna(0)
prev_activ = prev_activ.withColumn('ObesityPrevalence',
        prev_activ['ObeseCount'] / prev_activ['TotalCount'])

prev_activ.show()
```

```
+--------------+----------+----------+-------------------+
|PhysicalActFreq|TotalCount|ObeseCount| ObesityPrevalence|
+--------------+----------+----------+-------------------+
|           2.5|        28|        21|               0.75|
|           2.2|        68|        49|0.7205882352941176|
|           2.0|      2842|      2018|0.7100633356790992|
|           1.8|       119|       112|0.9411764705882353|
|           0.1|       539|       455|0.8441558441558441|
|           3.0|       808|       753| 0.931930693069307|
|           2.9|        74|        63|0.8513513513513513|
|           0.6|       239|       219|0.9163179916317992|
|           0.9|       810|       735|0.9074074074074074|
|           1.5|       526|       505|0.9600760456273765|
|           2.6|        22|        16|0.7272727272727273|
|           0.5|       224|       161|            0.71875|
|           1.1|       537|       513|0.9553072625698324|
|           2.8|        49|        41|0.8367346938775511|
|           1.0|      4555|      4141|0.9091108671789243|
|           1.4|       371|       343|0.9245283018867925|
|           0.3|       233|       206|0.8841201716738197|
|           0.4|       125|       112|              0.896|
|           1.6|       369|       343|0.9295392953929539|
|           2.4|        35|        24|0.6857142857142857|
+--------------+----------+----------+-------------------+
only showing top 20 rows
```

```
###  Q3 What is the maximum weight for individuals for each obesity level

#importing max from spark sql functions
from pyspark.sql.functions import max
mweight_level = df.groupBy('ObesityLevel').agg(max(col('Weight')).alias('MaxWeight'))

mweight_level.show()
```

```
+------------+---------+
|ObesityLevel|MaxWeight|
+------------+---------+
|         5.0|    138.0|
|         2.0|     96.0|
|         3.0|    115.0|
|         1.0|     90.0|
|         6.0|    165.0|
|         4.0|    156.0|
|         0.0|    114.0|
+------------+---------+
```

```
#Finding error row
error_row = df.filter((df['ObesityLevel'] == 0.0) & (df['Weight'] == 114.0))
error_row.show()
```

```
+------+------+----+------+------+------------------------------+---------+-------------+-------------+-------------+-----+--------+-----------
|    id|Gender| Age|Height|Weight|family_history_with_overweight|FCHighCal|FCvegetables|NumMainMeals|ConsFoodBetwMeal|SMOKE|DayWater|MonitorCalor
+------+------+----+------+------+------------------------------+---------+-------------+-------------+-------------+-----+--------+-----------
|8743.0|   1.0|28.0| 176.0| 114.0|                           1.0|      1.0|          1.1|          4.0|          4.0|  0.0|     2.0|          0.
+------+------+----+------+------+------------------------------+---------+-------------+-------------+-------------+-----+--------+-----------
```

```
#importing SparkSession
from pyspark.sql import SparkSession

#deleting it
data = df.filter(df['id'] != 8743.0)
data.show()
```

```
+----+------+----+------+------+------------------------------+---------+-------------+-------------+-------------+-----+--------+-------------+
|  id|Gender| Age|Height|Weight|family_history_with_overweight|FCHighCal|FCvegetables|NumMainMeals|ConsFoodBetwMeal|SMOKE|DayWater|MonitorCalory|
+----+------+----+------+------+------------------------------+---------+-------------+-------------+-------------+-----+--------+-------------+
| 1.0|   1.0|24.0| 170.0|  81.0|                           1.0|      1.0|          2.0|          3.0|          4.0|  0.0|     2.8|          0.0|
| 2.0|   0.0|18.0| 156.0|  57.0|                           1.0|      1.0|          2.0|          3.0|          2.0|  0.0|     2.0|          0.0|
| 3.0|   0.0|18.0| 171.0|  50.0|                           1.0|      1.0|          1.9|          1.4|          4.0|  0.0|     1.9|          0.0|
| 4.0|   0.0|20.0| 171.0| 131.0|                           1.0|      1.0|          3.0|          3.0|          4.0|  0.0|     1.7|          0.0|
| 5.0|   1.0|31.0| 191.0|  93.0|                           1.0|      1.0|          2.7|          2.0|          4.0|  0.0|     2.0|          0.0|
| 6.0|   1.0|18.0| 175.0|  51.0|                           1.0|      1.0|          2.9|          3.0|          4.0|  0.0|     2.1|          0.0|
| 7.0|   1.0|29.0| 175.0| 113.0|                           1.0|      1.0|          2.0|          3.0|          4.0|  0.0|     2.0|          0.0|
| 8.0|   1.0|29.0| 175.0| 118.0|                           1.0|      1.0|          1.4|          3.0|          4.0|  0.0|     2.0|          0.0|
| 9.0|   1.0|17.0| 170.0|  70.0|                           0.0|      1.0|          2.0|          3.0|          4.0|  0.0|     3.0|          1.0|
|10.0|   0.0|26.0| 164.0| 111.0|                           1.0|      1.0|          3.0|          3.0|          4.0|  0.0|     2.6|          0.0|
|11.0|   0.0|20.0| 165.0|  65.0|                           1.0|      1.0|          3.0|          3.0|          4.0|  0.0|     3.0|          0.0|
|12.0|   1.0|22.0| 170.0|  70.0|                           1.0|      0.0|          2.0|          3.0|          3.0|  0.0|     2.0|          0.0|
|13.0|   1.0|18.0| 181.0| 108.0|                           1.0|      1.0|          2.0|          2.2|          4.0|  0.0|     2.5|          0.0|
|14.0|   0.0|21.0| 173.0| 132.0|                           1.0|      1.0|          3.0|          3.0|          4.0|  0.0|     2.0|          0.0|
```

```
#Recheck

mweight_level = data.groupBy('ObesityLevel').agg(max(col('Weight')).alias('MaxWeight'))
mweight_level.show()

##-----------------ANSWER-----------------
#Now, answer is sarisfactory
# Level 0: 86kg
# Level 1: 90kg
# Level 2: 96kg
# Level 3: 115kg
# Level 4: 156kg
# Level 5: 138kg
# LEvel 6: 165kg
```

```
+-------------+---------+
|ObesityLevel|MaxWeight|
+-------------+---------+
|          5.0|    138.0|
|          2.0|     96.0|
|          3.0|    115.0|
|          1.0|     90.0|
|          6.0|    165.0|
|          4.0|    156.0|
|          0.0|     86.0|
+-------------+---------+
```

```
### Q4 Which transportation type is most poular among all ObesityLevels

from pyspark.sql.functions import count, col
from pyspark.sql.window import Window
from pyspark.sql.functions import rank


trans_ol = df.groupBy('MTRANS', 'ObesityLevel').agg(count('*').alias('Count'))

w_spec = Window.partitionBy('ObesityLevel').orderBy(col('Count').desc())

mc_trans = trans_ol.withColumn('rank', rank().over(w_spec)) \
    .filter(col('rank') == 1) \
    .drop('rank') \
    .orderBy('ObesityLevel')

# Display the result
mc_trans.show()
```

```
+------+-------------+-----+
|MTRANS|ObesityLevel|Count|
+------+-------------+-----+
|   4.0|          0.0| 2166|
|   4.0|          1.0| 2565|
|   4.0|          2.0| 1835|
|   4.0|          3.0| 1649|
|   4.0|          4.0| 2132|
|   4.0|          5.0| 2294|
|   4.0|          6.0| 4046|
```

ChiSqSelector for modelling

```
from pyspark.ml.feature import ChiSqSelector

# Initialize and configure the ChiSqSelector
selector = ChiSqSelector(numTopFeatures=10, featuresCol="features", outputCol="selectedFeatures", labelCol="indexedLabel")

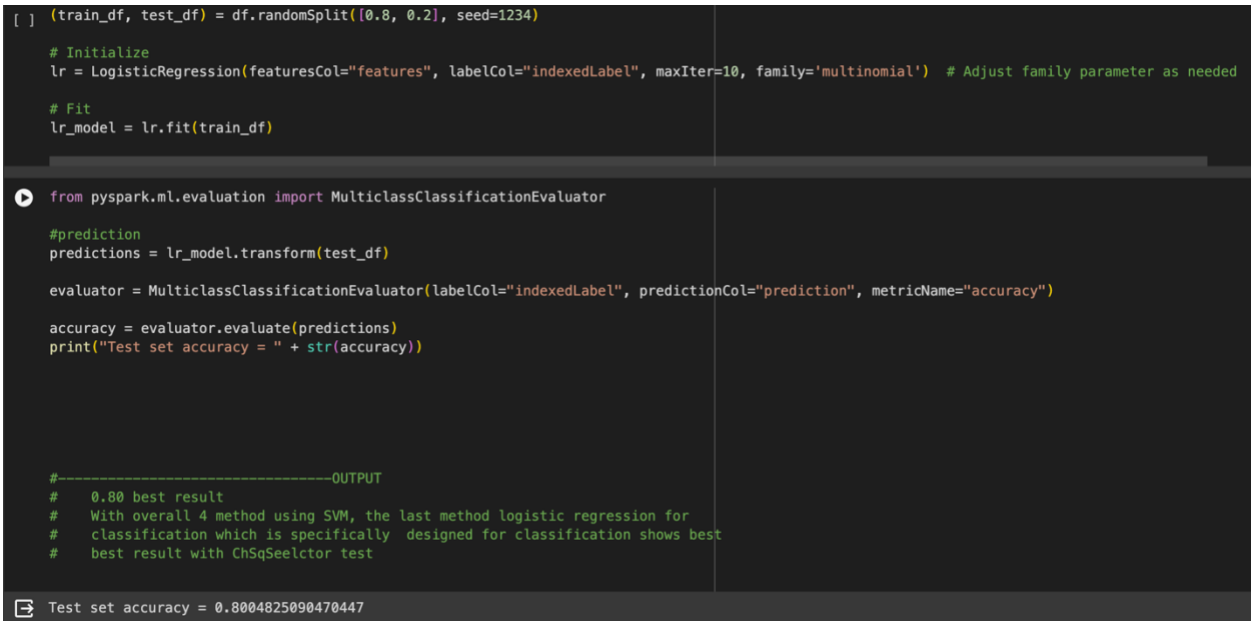
# Fit and transform to select features
model = selector.fit(df)
df_selected = model.transform(df)

# Display the result with selected features
df_selected.show()


#--------------------------------OUTPUT
#want to select the top 10 features that have the most significant
#relationship with the target variable based on the Chi-Squared test
#featuresCol="features"
#outputCol="selectedFeatures"
#labelCol="indexedLabel"
```

```
+----+------+----+------+------+-----------------------------+---------+-----------+-------------+----------------+-----+--
|  id|Gender| Age|Height|Weight|family_history_with_overweight|FCHighCal|FCvegetables|NumMainMeals|ConsFoodBetwMeal|SMOKE|Da
+----+------+----+------+------+-----------------------------+---------+-----------+-------------+----------------+-----+--
| 1.0|   1.0|24.0| 170.0|  81.0|                          1.0|      1.0|        2.0|          3.0|             4.0|  0.0|
| 2.0|   0.0|18.0| 156.0|  57.0|                          1.0|      1.0|        2.0|          3.0|             2.0|  0.0|
| 3.0|   0.0|18.0| 171.0|  50.0|                          1.0|      1.0|        1.9|          1.4|             4.0|  0.0|
| 4.0|   0.0|20.0| 171.0| 131.0|                          1.0|      1.0|        3.0|          3.0|             4.0|  0.0|
| 5.0|   1.0|31.0| 191.0|  93.0|                          1.0|      1.0|        2.7|          2.0|             4.0|  0.0|
| 6.0|   1.0|18.0| 175.0|  51.0|                          1.0|      1.0|        2.9|          3.0|             4.0|  0.0|
| 7.0|   1.0|29.0| 175.0| 113.0|                          1.0|      1.0|        2.0|          3.0|             4.0|  0.0|
| 8.0|   1.0|29.0| 175.0| 118.0|                          1.0|      1.0|        1.4|          3.0|             4.0|  0.0|
| 9.0|   1.0|17.0| 170.0|  70.0|                          0.0|      1.0|        2.0|          3.0|             4.0|  0.0|
|10.0|   0.0|26.0| 164.0| 111.0|                          1.0|      1.0|        3.0|          3.0|             4.0|  0.0|
```

HPCI result



```
[ ] (train_df, test_df) = df.randomSplit([0.8, 0.2], seed=1234)

    # Initialize
    lr = LogisticRegression(featuresCol="features", labelCol="indexedLabel", maxIter=10, family='multinomial')  # Adjust family parameter as needed

    # Fit
    lr_model = lr.fit(train_df)
```

```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator

#prediction
predictions = lr_model.transform(test_df)

evaluator = MulticlassClassificationEvaluator(labelCol="indexedLabel", predictionCol="prediction", metricName="accuracy")

accuracy = evaluator.evaluate(predictions)
print("Test set accuracy = " + str(accuracy))




#---------------------------------OUTPUT
#    0.80 best result
#    With overall 4 method using SVM, the last method logistic regression for
#    classification which is specifically  designed for classification shows best
#    best result with ChSqSeelctor test
```

```
Test set accuracy = 0.8004825090470447
```