Homework5

Kavana Manvi KrishnaMurthy

2024-06-03

Single Problem (100 pts – see rubric) For this assignment, you may pick data of your choice (given the rules below). You will perform all the majorsteps of data mining and report your results. The goal here is not an extensive report about the data, but practice applying the whole pipeline from start to finish. You have done all these steps already, but now you get to make choices yourself. That will take longer per step, but we are only going through the pipeline once on one set of data. This assignment is not meant to take longer than the previous ones. If you find it is substantially more work, consider switching to a simpler dataset or asking for guidance. The parts of this Problem correspond to the parts of the data mining pipeline, explaining the requirements of the assignment for each one. There is a rubric below that will be used to grade your submissions. The actual deliverable is ' just the report with each of the steps labeled and described (i.e., include the labels a-i in your report document.)

a. Data gathering and integration The first part is to get the data you will use. You may use anything that has not been used in an assignment or tutorial. It must have at least 100 data points and must include both numerical andcategorial (or ordinal) variables. I recommend keeping this relatively straightforward because data cleaning can take a lot of time if you choose a large, messy dataset. Kaggle (https://www.kaggle.com/datasets) and the University of California at Irvine (UCI) (https://archive.ics.uci.edu/ml/index.php) maintain collections of datasets, some even telling you if they are good examples for testing specific machine learning techniques. You may also choose to join together more than one dataset, for example to merge data on health outcomes by US state with a dataset on food statistics per state. Merging data is not required and will earn you a bonus point in this step.

Using the Obesity dataset from kaggle - https://www.kaggle.com/datasets/fatemehmehrparvar/obesity-levels

```
obesity <- read.csv("/Users/kavanamanvi/Desktop/FDS/HW5/ObesityDataSet.csv")
#no missing values
summary(obesity)
```

```
##     Gender               Age              Height          Weight
##  Length:2111        Min.   :14.00   Min.   :1.450   Min.   : 39.00
##  Class :character   1st Qu.:19.95   1st Qu.:1.630   1st Qu.: 65.47
##  Mode  :character   Median :22.78   Median :1.700   Median : 83.00
##                     Mean   :24.31   Mean   :1.702   Mean   : 86.59
##                     3rd Qu.:26.00   3rd Qu.:1.768   3rd Qu.:107.43
##                     Max.   :61.00   Max.   :1.980   Max.   :173.00
##  family_history_with_overweight     FAVC               FCVC
##  Length:2111                    Length:2111        Min.   :1.000
##  Class :character               Class :character   1st Qu.:2.000
##  Mode  :character               Mode  :character   Median :2.386
##                                                    Mean   :2.419
##                                                    3rd Qu.:3.000
```

```
##                                                  Max.   :3.000
##       NCP             CAEC              SMOKE                CH2O
##  Min.   :1.000   Length:2111        Length:2111        Min.   :1.000
##  1st Qu.:2.659   Class :character   Class :character   1st Qu.:1.585
##  Median :3.000   Mode  :character   Mode  :character   Median :2.000
##  Mean   :2.686                                         Mean   :2.008
##  3rd Qu.:3.000                                         3rd Qu.:2.477
##  Max.   :4.000                                         Max.   :3.000
##      SCC                 FAF              TUE             CALC
##  Length:2111        Min.   :0.0000   Min.   :0.0000   Length:2111
##  Class :character   1st Qu.:0.1245   1st Qu.:0.0000   Class :character
##  Mode  :character   Median :1.0000   Median :0.6253   Mode  :character
##                     Mean   :1.0103   Mean   :0.6579
##                     3rd Qu.:1.6667   3rd Qu.:1.0000
##                     Max.   :3.0000   Max.   :2.0000
##     MTRANS           NObeyesdad
##  Length:2111        Length:2111
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

```
#View the first 6 rows
head(obesity)
```

```
##   Gender Age Height Weight family_history_with_overweight FAVC FCVC NCP
## 1 Female  21   1.62   64.0                             yes   no    2   3
## 2 Female  21   1.52   56.0                             yes   no    3   3
## 3   Male  23   1.80   77.0                             yes   no    2   3
## 4   Male  27   1.80   87.0                              no   no    3   3
## 5   Male  22   1.78   89.8                              no   no    2   1
## 6   Male  29   1.62   53.0                              no  yes    2   3
##        CAEC SMOKE CH2O SCC FAF TUE       CALC                MTRANS
## 1 Sometimes    no    2  no   0   1         no Public_Transportation
## 2 Sometimes   yes    3 yes   3   0  Sometimes Public_Transportation
## 3 Sometimes    no    2  no   2   1 Frequently Public_Transportation
## 4 Sometimes    no    2  no   2   0 Frequently                Walking
## 5 Sometimes    no    2  no   0   0  Sometimes Public_Transportation
## 6 Sometimes    no    2  no   0   0  Sometimes            Automobile
##           NObeyesdad
## 1       Normal_Weight
## 2       Normal_Weight
## 3       Normal_Weight
## 4  Overweight_Level_I
## 5 Overweight_Level_II
## 6       Normal_Weight
```

```
#number of rows is 2111
nrow(obesity)
```

```
## [1] 2111
```

```r
#number of columns
ncol(obesity)
```

```
## [1] 17
```

```r
#column names
names(obesity)
```

```
##  [1] "Gender"                          "Age"
##  [3] "Height"                          "Weight"
##  [5] "family_history_with_overweight" "FAVC"
##  [7] "FCVC"                            "NCP"
##  [9] "CAEC"                            "SMOKE"
## [11] "CH2O"                            "SCC"
## [13] "FAF"                             "TUE"
## [15] "CALC"                            "MTRANS"
## [17] "NObeyesdad"
```

b. Data Exploration Using data exploration to understand what is happening is important throughout the pipeline, and is not limited to this step. However, it is important to use some exploration early on to make sure you understand your data. You must at least consider the distributions of each variable and at least some of the relationships between pairs of variables.

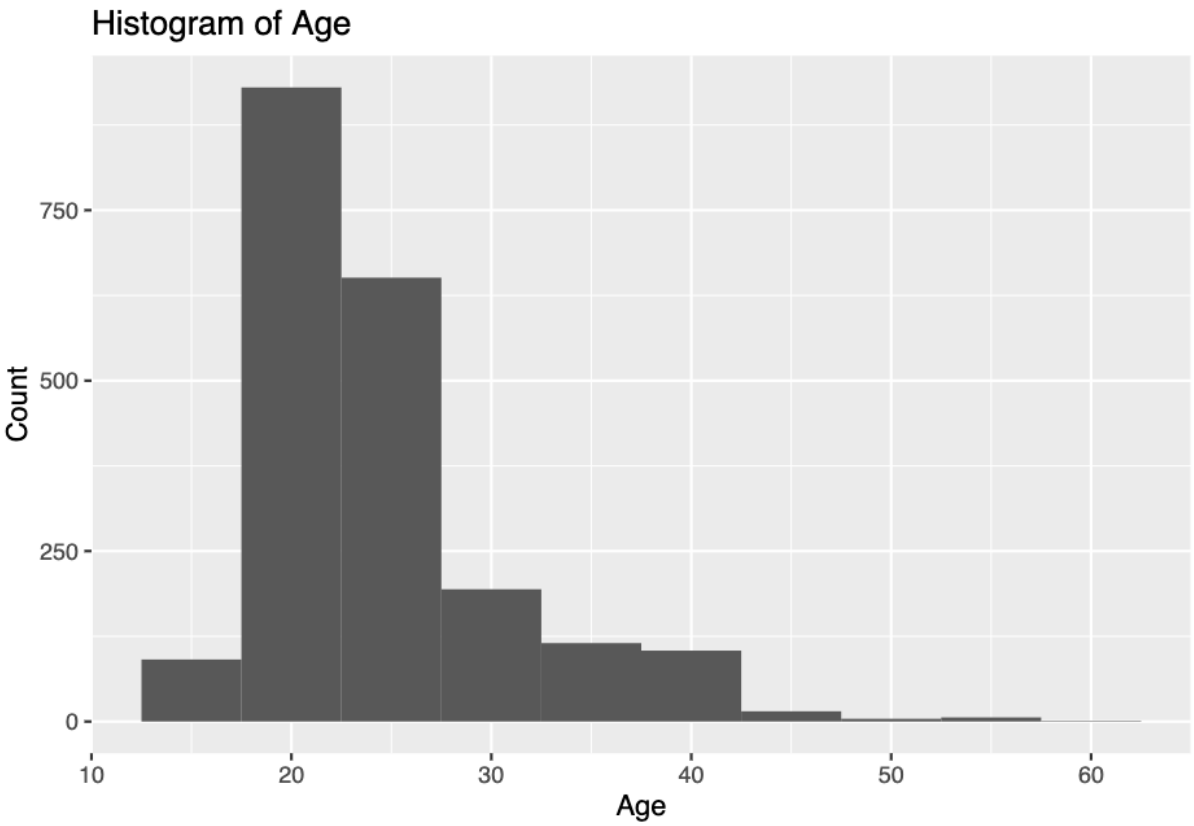
##Numerical Variables:

#Age

```r
summary(obesity$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   14.00   19.95   22.78   24.31   26.00   61.00
```

```r
library("ggplot2")
ggplot(obesity, aes(x = Age)) +
     geom_histogram(binwidth = 5) +
     labs(title = "Histogram of Age", x = "Age", y = "Count")
```



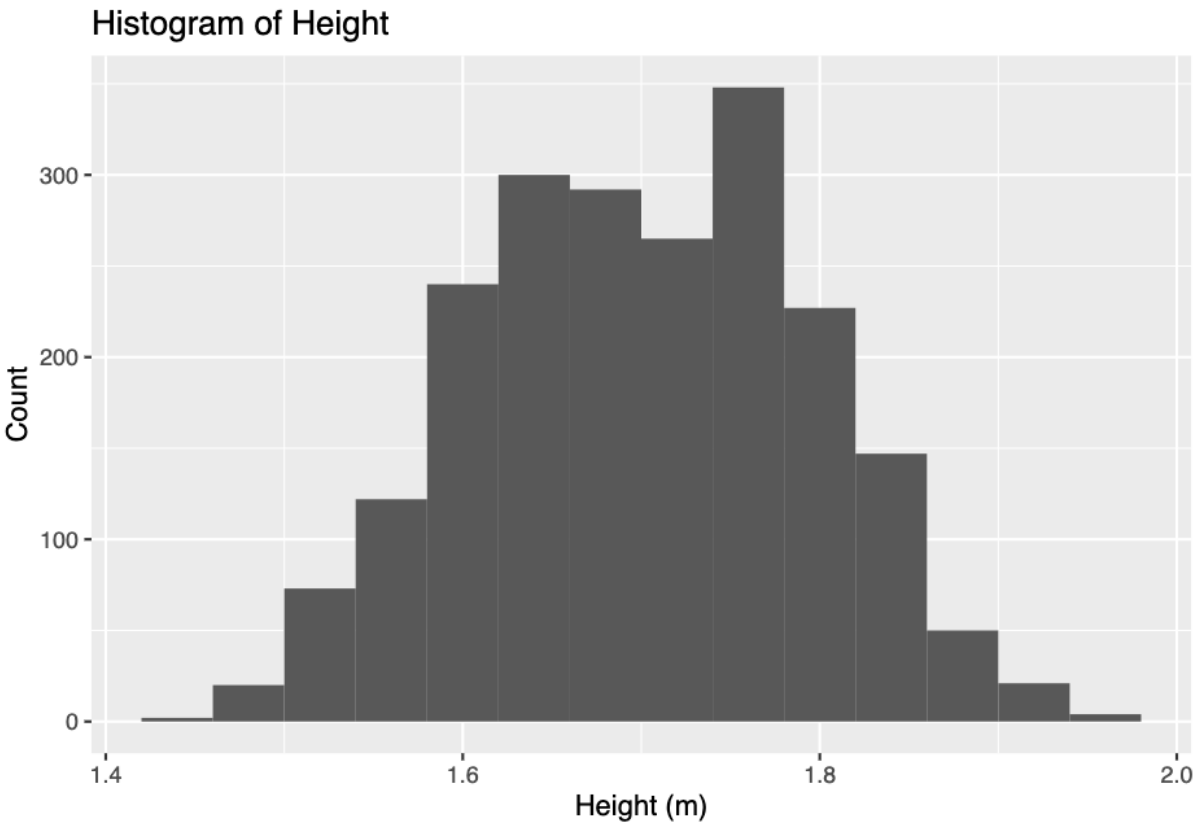
The dataset includes individuals ranging in age from 14 to 61 years old. The age values are not normally distributed, as evidenced by the histogram which is positively skewed. The mean age is 24.31, while the median age is 22.78, indicating that both are closer to the younger end of the age spectrum.

#Height

```
summary(obesity$Height)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.450   1.630   1.700   1.702   1.768   1.980
```

```
ggplot(obesity, aes(x = Height)) +
     geom_histogram(binwidth = 0.04) +
     labs(title = "Histogram of Height", x = "Height (m)", y = "Count")
```



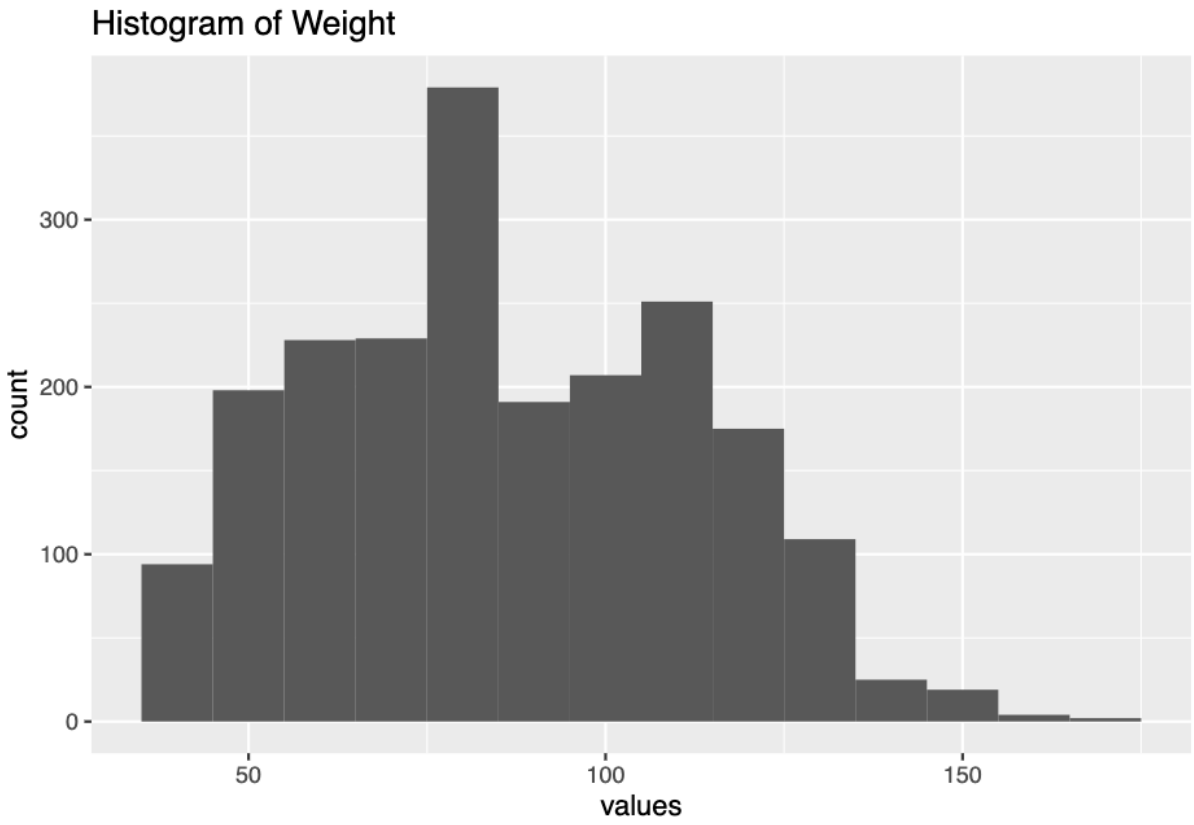
The majority of individuals are between 1.60 m and 1.85 m tall. The mean and median heights are approximately 1.70 m. However, the height values do not appear to follow a normal distribution.

#Weight

```
summary(obesity$Weight)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   39.00   65.47   83.00   86.59  107.43  173.00
```

```
ggplot(obesity, aes(x = Weight)) +
     geom_histogram(binwidth = 10,bins = length(unique(obesity$Weight))) +
     labs(title = "Histogram of Weight", x = "values", y = "count")
```



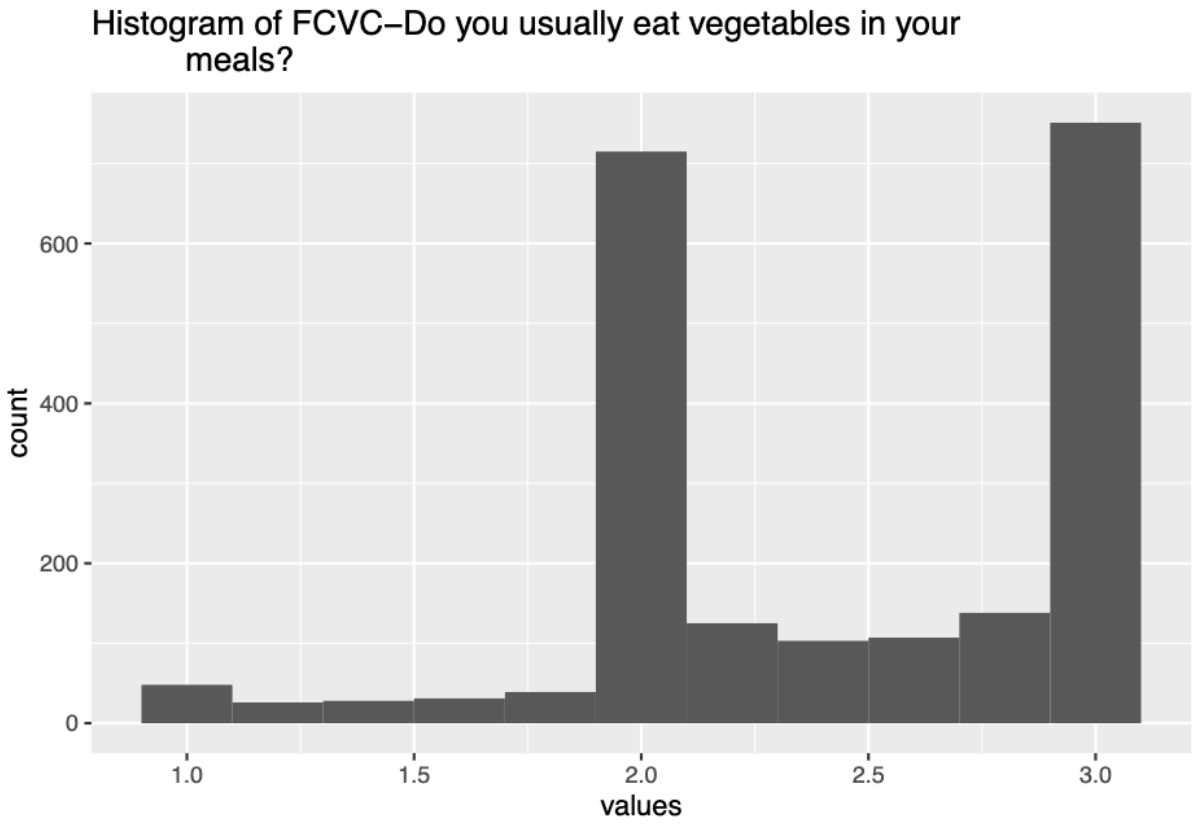
The weight distribution seems to be bi-modal; Majority of induviduals weigh around 70-80 kg.

#FCVC-Do you usually eat vegetables in your meals?

```
summary(obesity$FCVC)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   2.386   2.419   3.000   3.000
```

```
ggplot(obesity, aes(x = FCVC)) +
     geom_histogram(binwidth =0.2) +
     labs(title = "Histogram of FCVC-Do you usually eat vegetables in your
          meals? ", x = "values", y = "count")
```

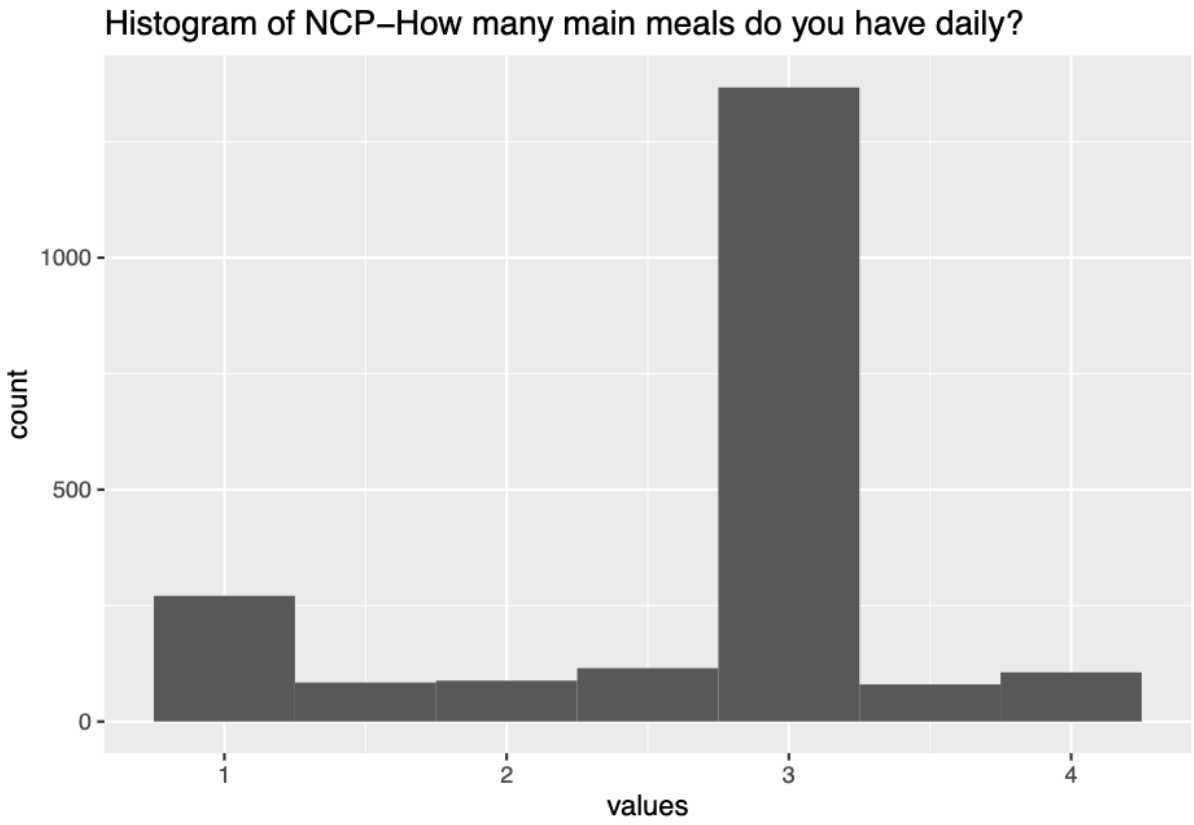


#NPC Number of meals had per day

```
summary(obesity$NCP)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.659   3.000   2.686   3.000   4.000
```

```
ggplot(obesity, aes(x = NCP)) +
     geom_histogram(binwidth = 0.5) +
     labs(title = "Histogram of NCP-How many main meals do you have daily?", x = "values", y = "count")
```

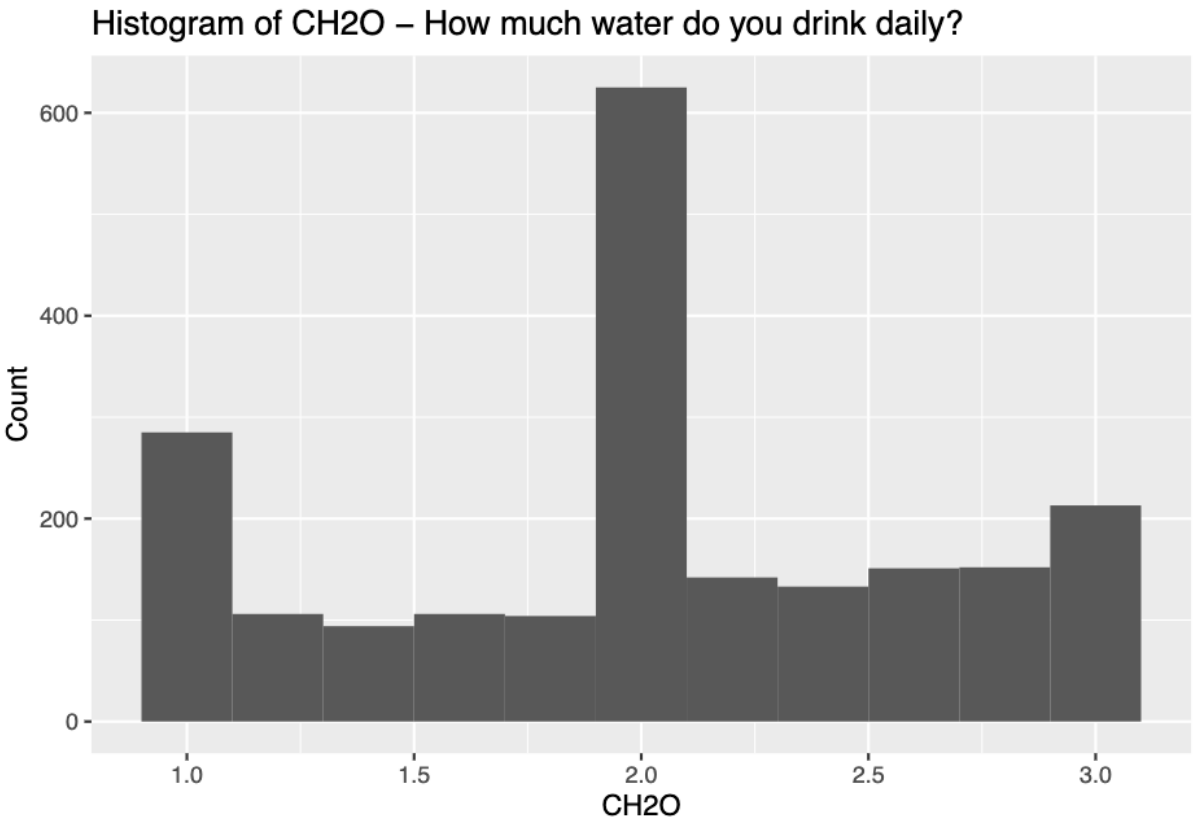


#CH20 Drinking water

```
summary(obesity$CH20)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   1.585   2.000   2.008   2.477   3.000
```

```
ggplot(obesity, aes(x = CH20)) +
     geom_histogram(binwidth = 0.2) +
     labs(title = "Histogram of CH20 - How much water do you drink daily?",
          x = "CH20", y = "Count")
```

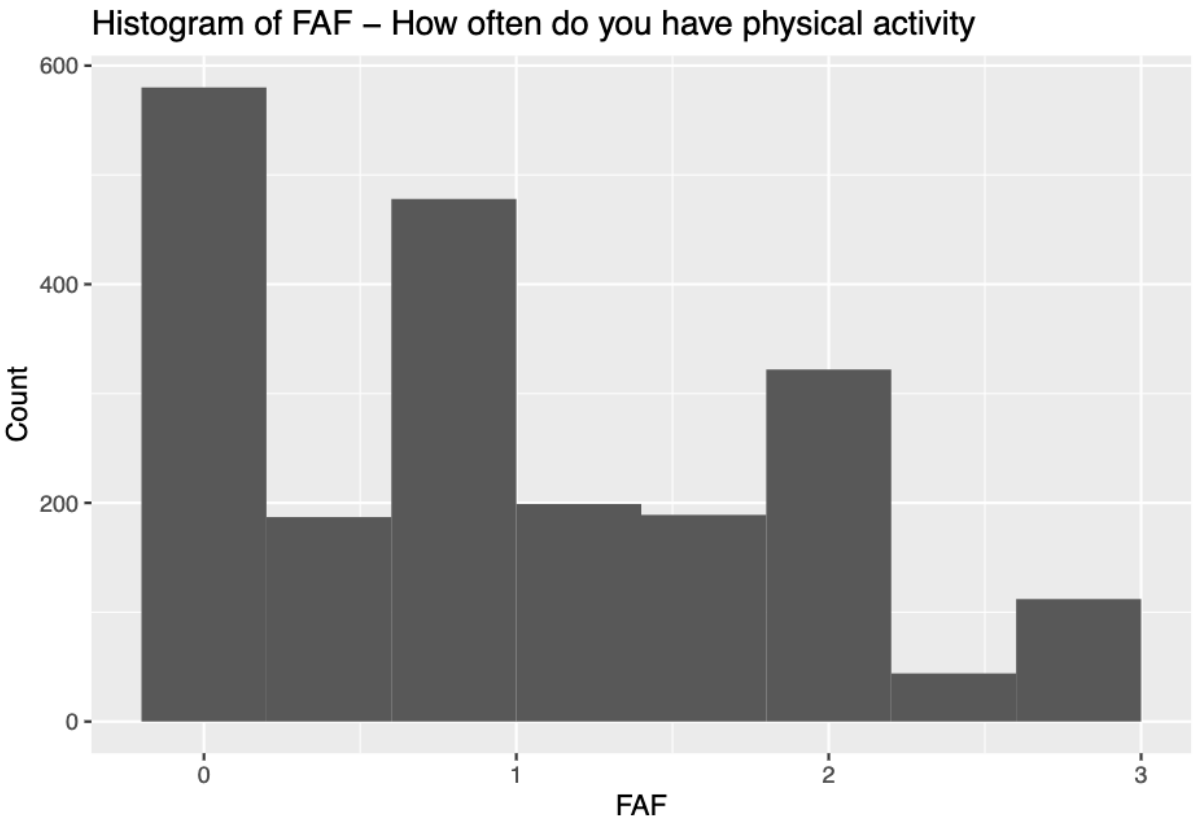


#FAF Amount of exercise

```
summary(obesity$FAF)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.1245  1.0000  1.0103  1.6667  3.0000
```

```
ggplot(obesity, aes(x = FAF)) +
     geom_histogram(binwidth = 0.4) +
     labs(title = "Histogram of FAF - How often do you have physical activity",
          x = "FAF", y = "Count")
```

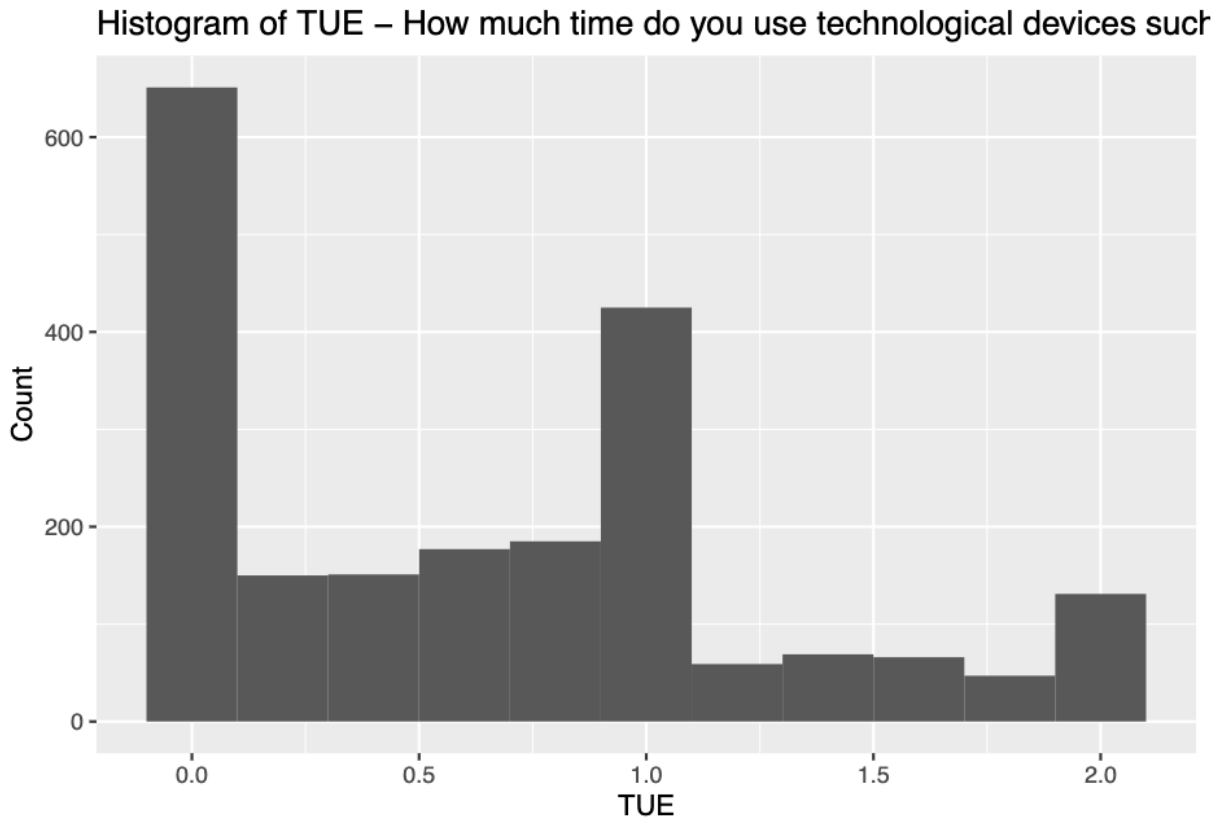


#TUE amount of technology used

```
summary(obesity$TUE)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.6253  0.6579  1.0000  2.0000
```

```
ggplot(obesity, aes(x = TUE)) +
     geom_histogram(binwidth = 0.2) +
     labs(title = "Histogram of TUE - How much time do you use technological devices such as cell phone
```



##Categorical Variables:

#Gender

The dataset contains nearly equal proportions of males and females, with slightly more data available for men than for women. However, this distribution does not indicate an imbalance in the dataset.

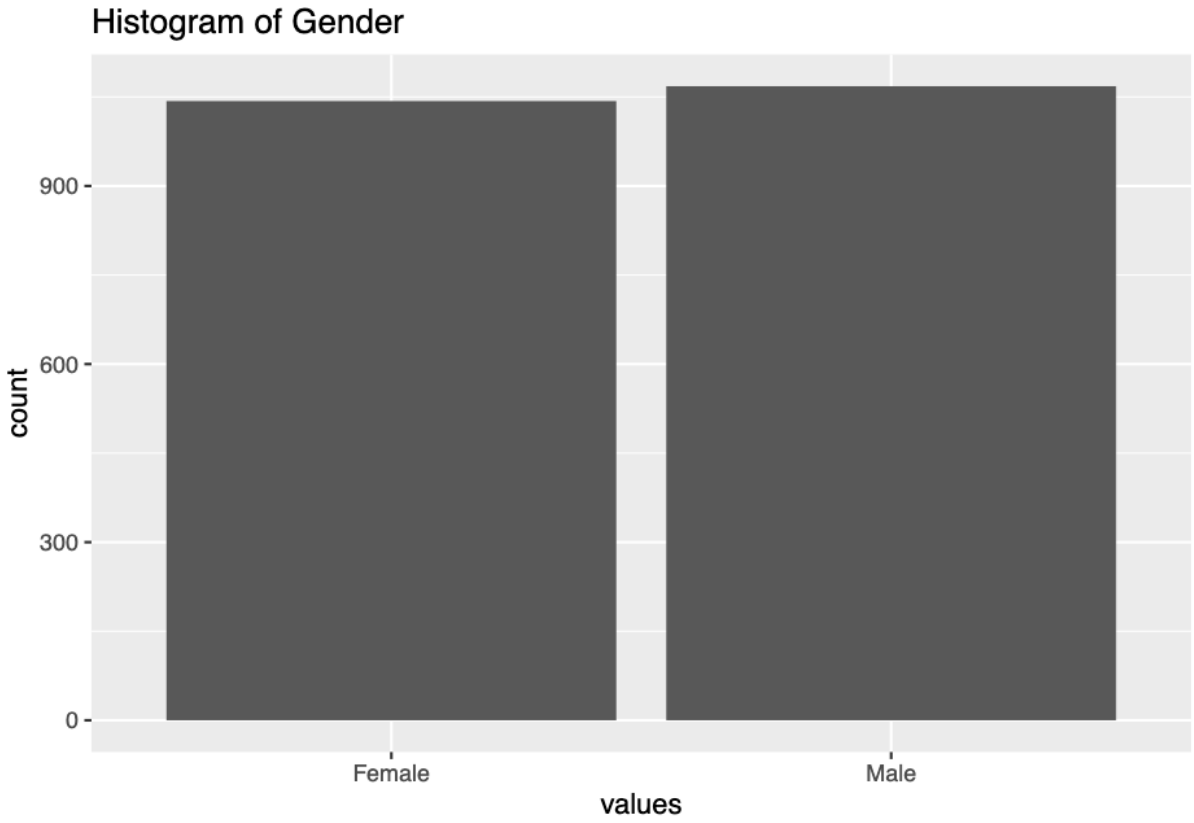
```
library(dplyr)
```

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

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

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

```
library(tidyr)

obesity %>%
  group_by(Gender) %>%
  summarise(count = n())
```

```
## # A tibble: 2 x 2
##   Gender count
##   <chr>  <int>
## 1 Female  1043
## 2 Male    1068
```

```
ggplot(obesity, aes(x=Gender)) + geom_bar()+
  labs(title = "Histogram of Gender", x = "values", y = "count")
```



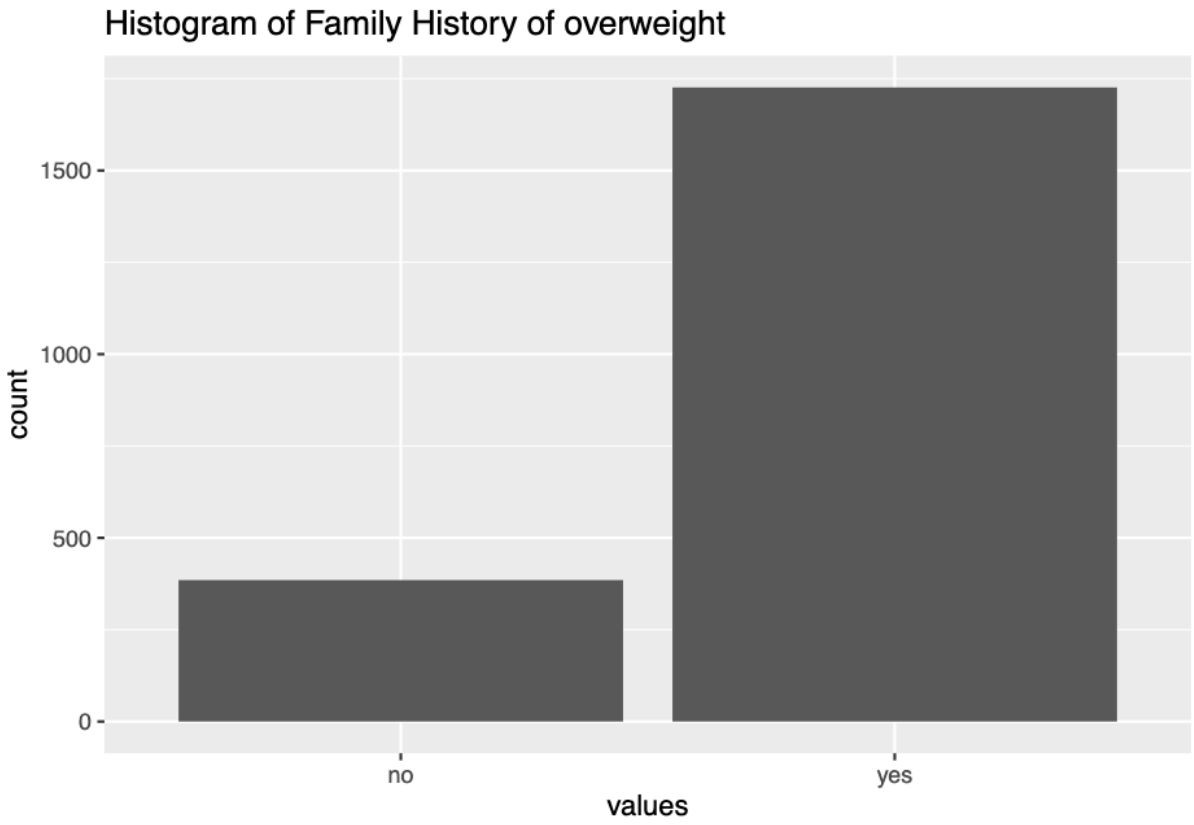
#Family History

Data is imbalances. More people have family history with obesity.

```
obesity %>%
  group_by(family_history_with_overweight) %>%
  summarise(count = n())
```

```
## # A tibble: 2 x 2
##   family_history_with_overweight count
##   <chr>                          <int>
## 1 no                               385
## 2 yes                             1726
```

```
ggplot(obesity, aes(x=family_history_with_overweight)) + geom_bar()+
  labs(title = "Histogram of Family History of overweight"
       , x = "values", y = "count")
```

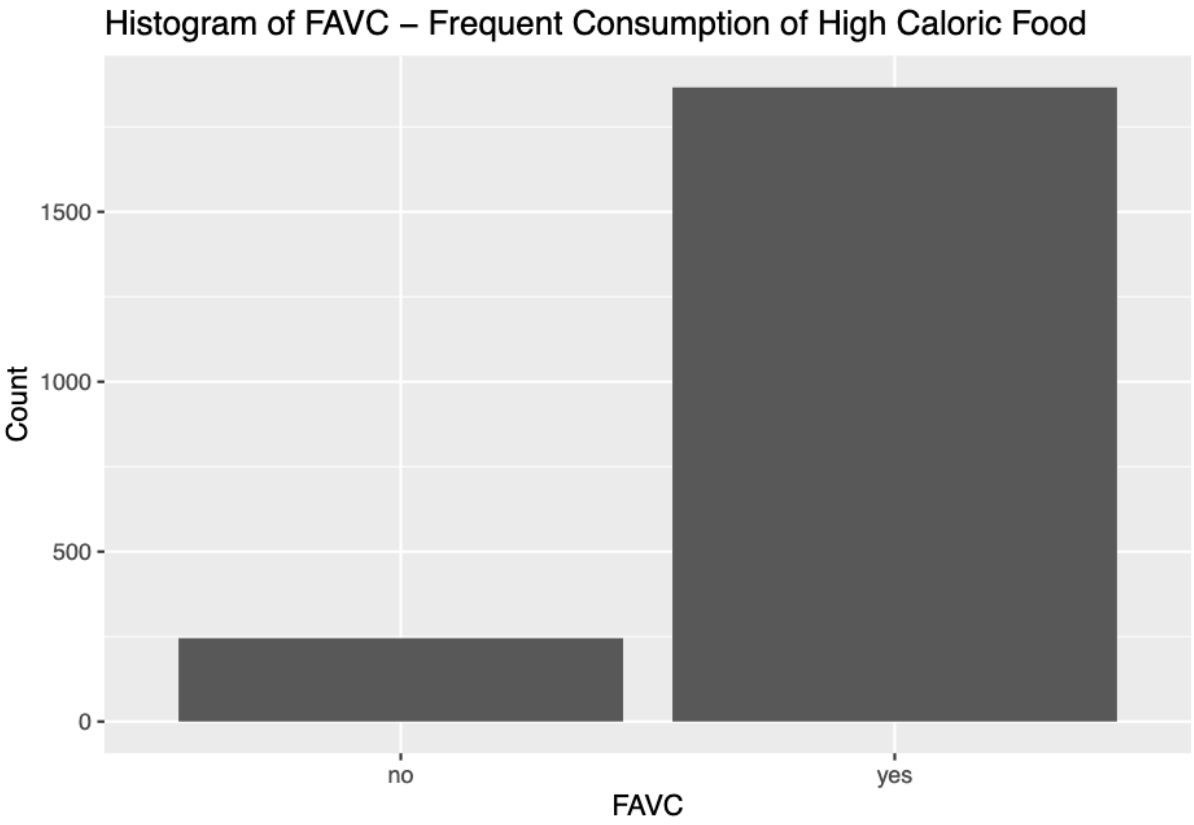


#FAVC - Frequent Consumption of High Caloric Food

```
obesity %>%
  group_by(FAVC) %>%
  summarise(count = n())
```

```
## # A tibble: 2 x 2
##   FAVC  count
##   <chr> <int>
## 1 no      245
## 2 yes    1866
```

```
ggplot(obesity, aes(x = FAVC)) +
  geom_bar() +
  labs(title = "Histogram of FAVC - Frequent Consumption of High Caloric Food", x = "FAVC", y = "Count"
```



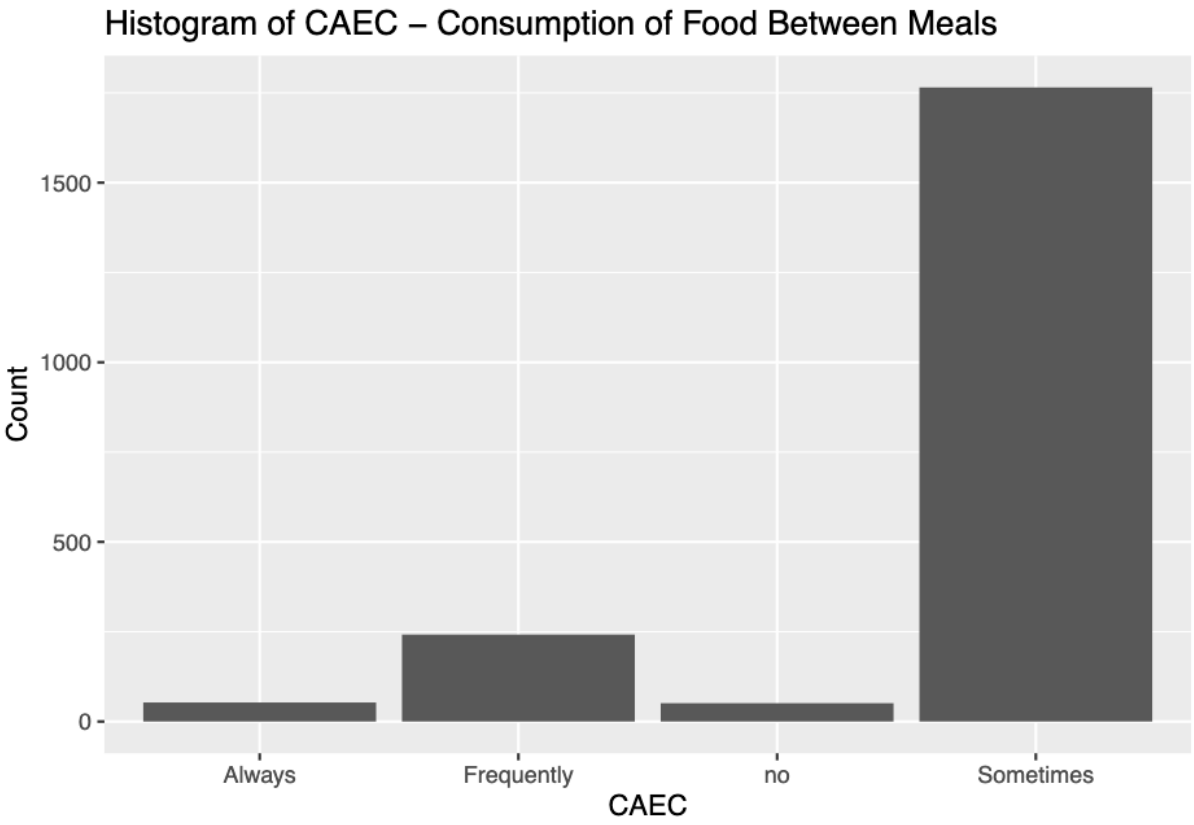
#CAEC - Consumption of Food Between Meals

The data shows that "Always" was reported 53 times, "Frequently" 242 times, "Sometimes" 1765 times, and "no" 51 times in the "CAEC" category. This indicates that the majority of responses fall under the "Sometimes" category, with fewer responses in the "Always," "Frequently," and "no" categories.

```
obesity %>%
  group_by(CAEC) %>%
  summarise(count = n())
```

```
## # A tibble: 4 x 2
##   CAEC       count
##   <chr>      <int>
## 1 Always        53
## 2 Frequently   242
## 3 Sometimes   1765
## 4 no            51
```

```
ggplot(obesity, aes(x = CAEC)) +
  geom_bar() +
  labs(title = "Histogram of CAEC - Consumption of Food Between Meals", x = "CAEC", y = "Count")
```



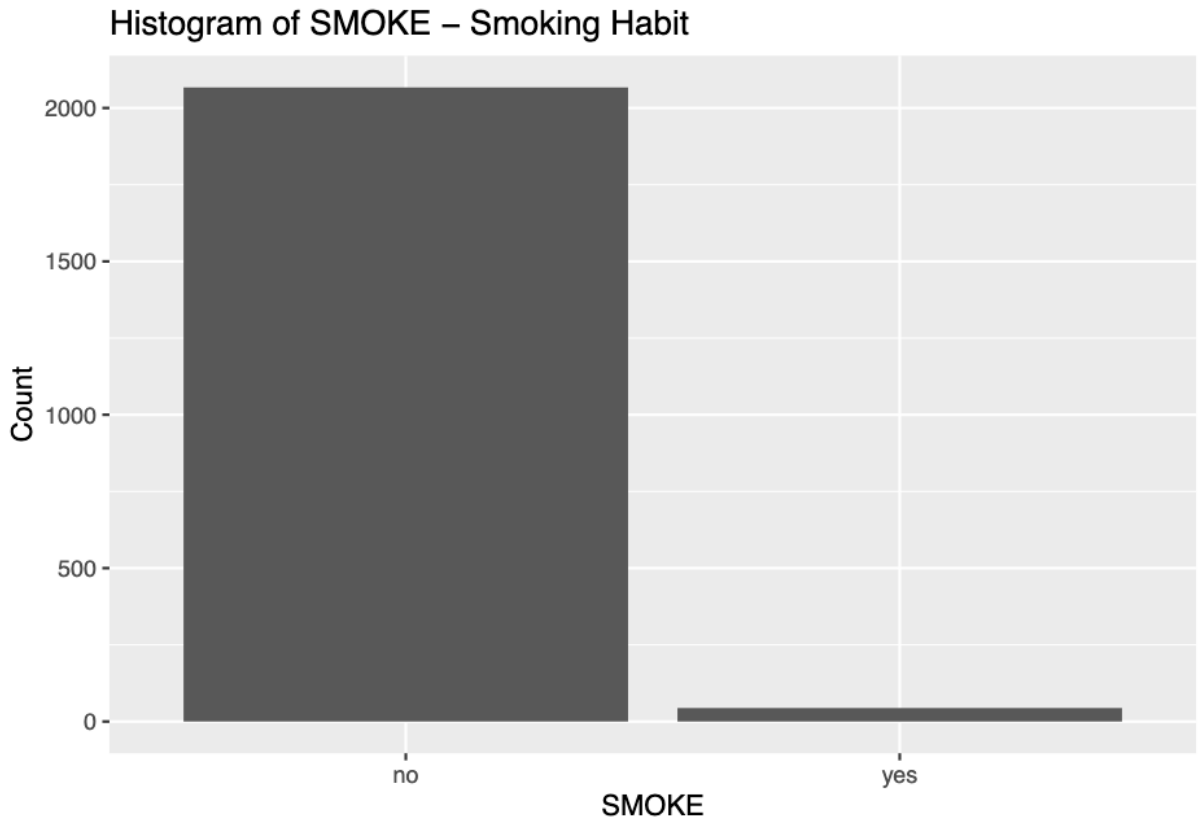
#Smoke

The "SMOKE" category indicates that there were 2067 responses indicating "no" and 44 responses indicating "yes" regarding smoking habits. This suggests that the majority of respondents do not smoke, with only a small minority reporting that they do.

```
obesity %>%
  group_by(SMOKE) %>%
  summarise(count = n())
```

```
## # A tibble: 2 x 2
##   SMOKE count
##   <chr> <int>
## 1 no     2067
## 2 yes      44
```

```
ggplot(obesity, aes(x = SMOKE)) +
  geom_bar() +
  labs(title = "Histogram of SMOKE - Smoking Habit", x = "SMOKE", y = "Count")
```

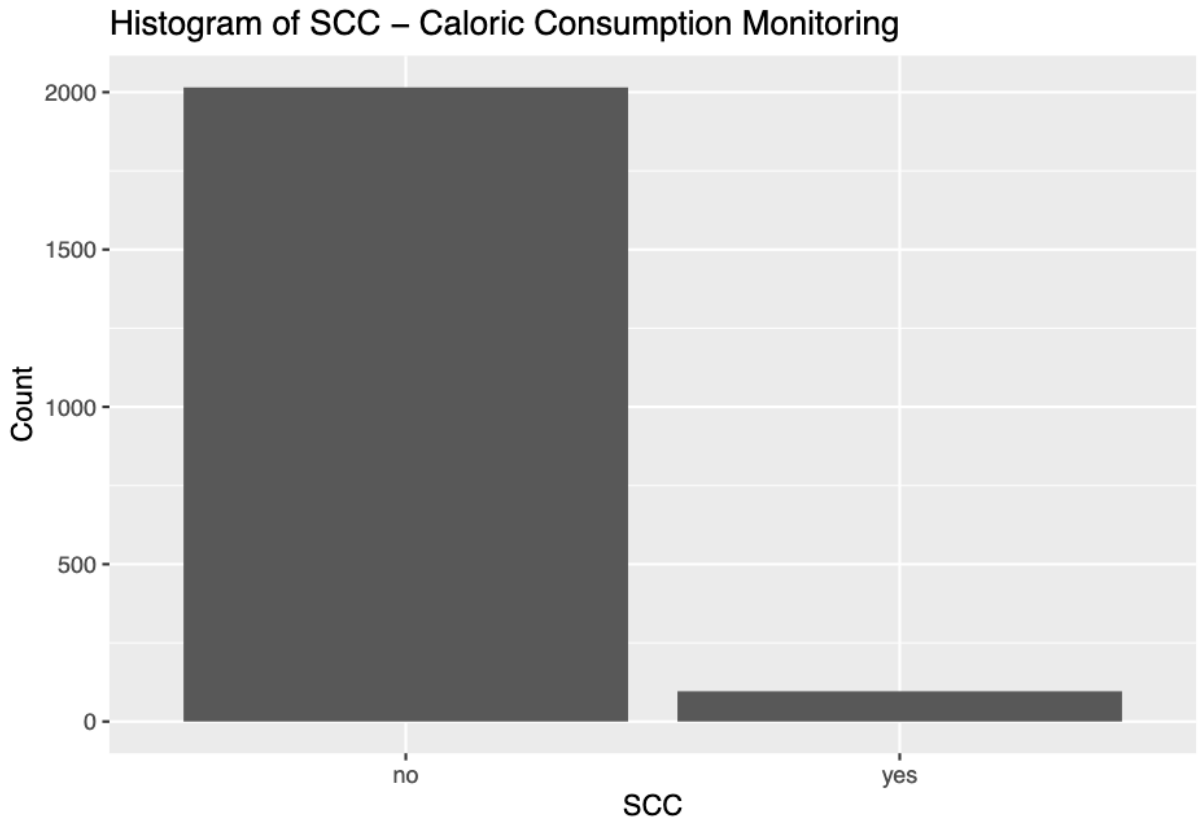


#SCC - Caloric Consumption Monitoring The distribution of responses for "SCC" (Caloric Consumption Monitoring) shows that 2015 respondents answered "no" and 96 answered "yes." This indicates that the majority of respondents do not monitor their caloric consumption, while a smaller portion do.

```
obesity %>%
  group_by(SCC) %>%
  summarise(count = n())
```

```
## # A tibble: 2 x 2
##   SCC   count
##   <chr> <int>
## 1 no     2015
## 2 yes      96
```

```
ggplot(obesity, aes(x = SCC)) +
  geom_bar() +
  labs(title = "Histogram of SCC - Caloric Consumption Monitoring", x = "SCC", y = "Count")
```

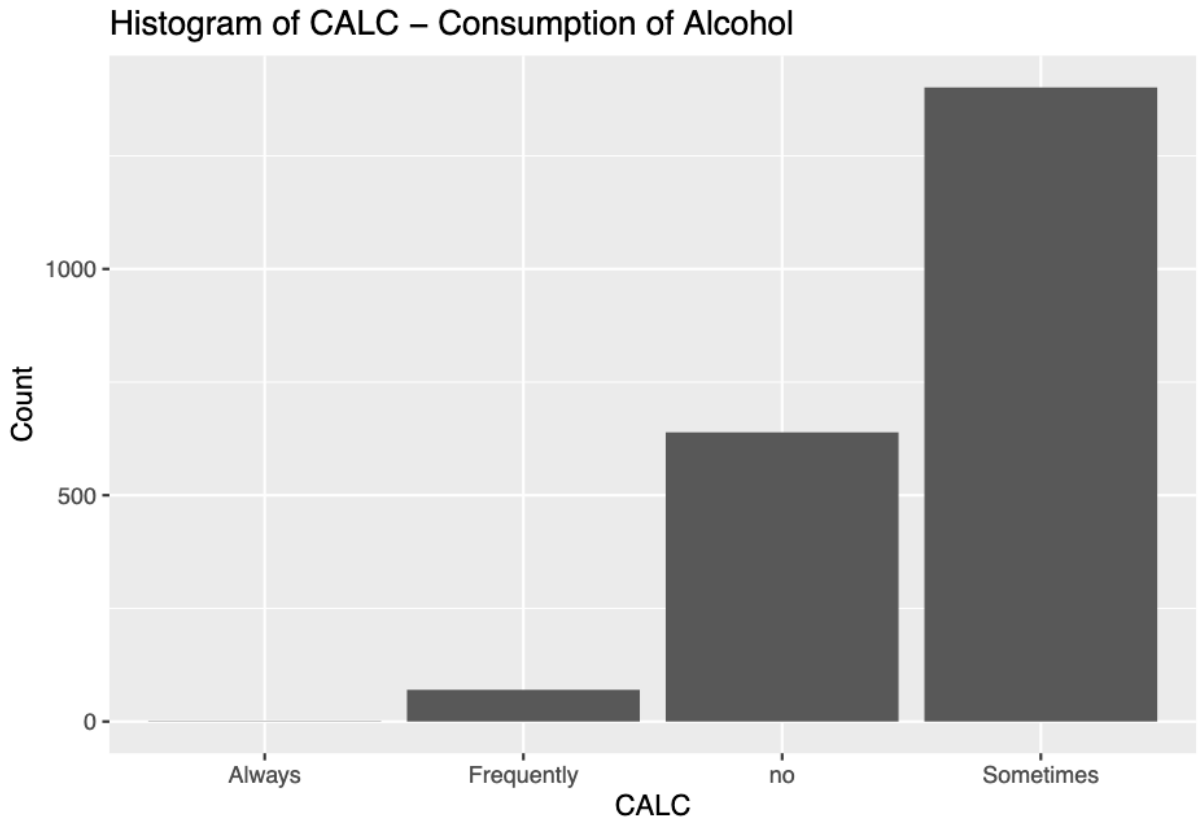


#CALC - Consumption of Alcohol

```
obesity %>%
  group_by(CALC) %>%
  summarise(count = n())
```

```
## # A tibble: 4 x 2
##   CALC       count
##   <chr>      <int>
## 1 Always         1
## 2 Frequently    70
## 3 Sometimes   1401
## 4 no           639
```

```
ggplot(obesity, aes(x = CALC)) +
  geom_bar() +
  labs(title = "Histogram of CALC - Consumption of Alcohol", x = "CALC", y = "Count")
```

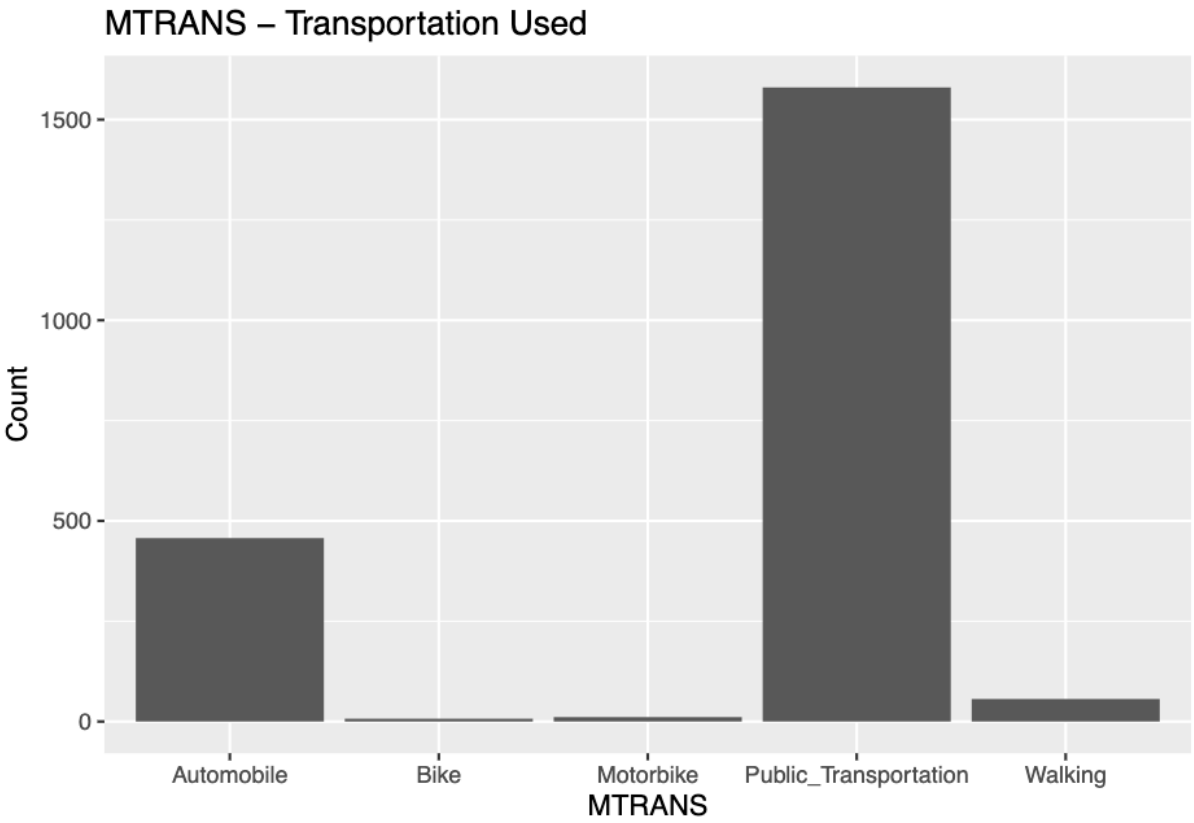


#MTRANS - Transportation Used

```
obesity %>%
  group_by(MTRANS) %>%
  summarise(count = n())
```

```
## # A tibble: 5 x 2
##   MTRANS                count
##   <chr>                 <int>
## 1 Automobile              457
## 2 Bike                      7
## 3 Motorbike                11
## 4 Public_Transportation  1580
## 5 Walking                  56
```

```
ggplot(obesity, aes(x = MTRANS)) +
  geom_bar() +
  labs(title = "MTRANS - Transportation Used", x = "MTRANS", y = "Count")
```

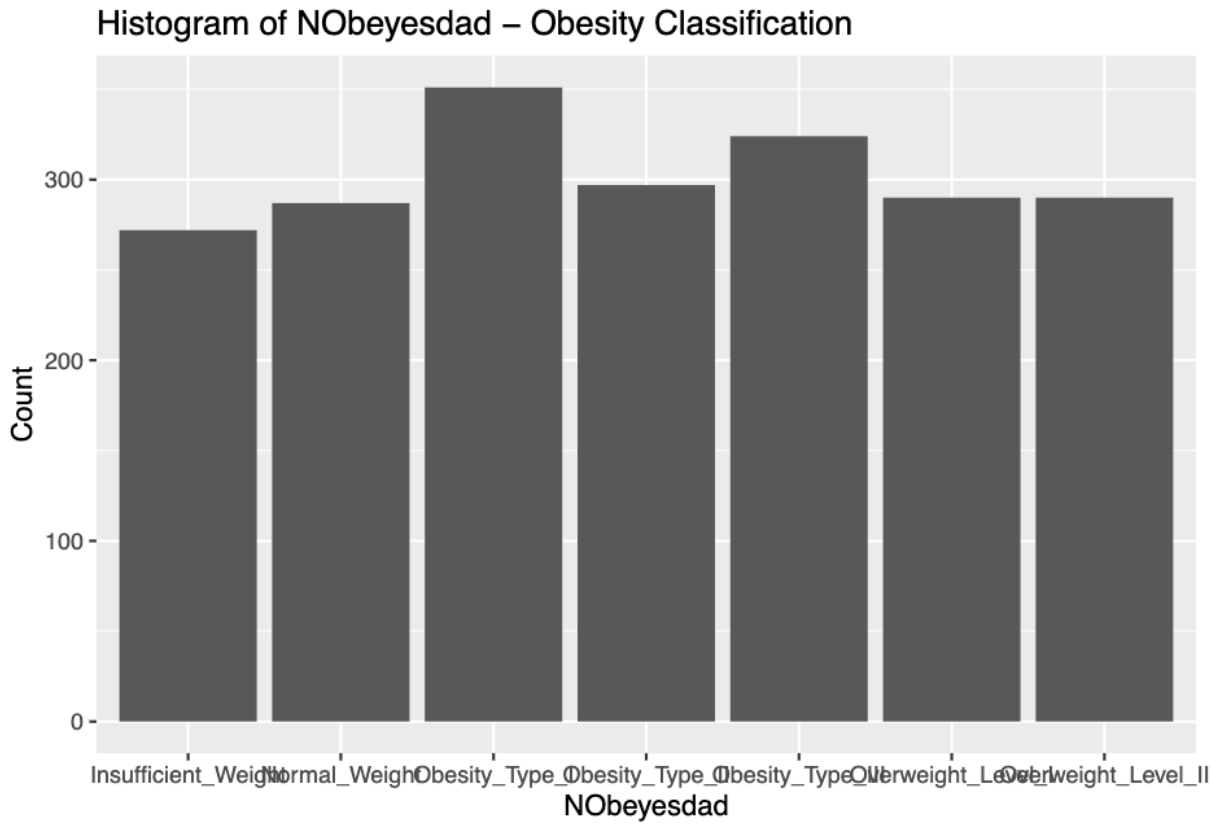


#NObeyesdad This indicates the count of individuals falling into each weight category based on the dataset.

```
obesity %>%
  group_by(NObeyesdad) %>%
  summarise(count = n())
```

```
## # A tibble: 7 x 2
##   NObeyesdad          count
##   <chr>               <int>
## 1 Insufficient_Weight   272
## 2 Normal_Weight         287
## 3 Obesity_Type_I        351
## 4 Obesity_Type_II       297
## 5 Obesity_Type_III      324
## 6 Overweight_Level_I    290
## 7 Overweight_Level_II   290
```

```
ggplot(obesity, aes(x = NObeyesdad)) +
  geom_bar() +
  labs(title = "Histogram of NObeyesdad - Obesity Classification", x = "NObeyesdad", y = "Count")
```



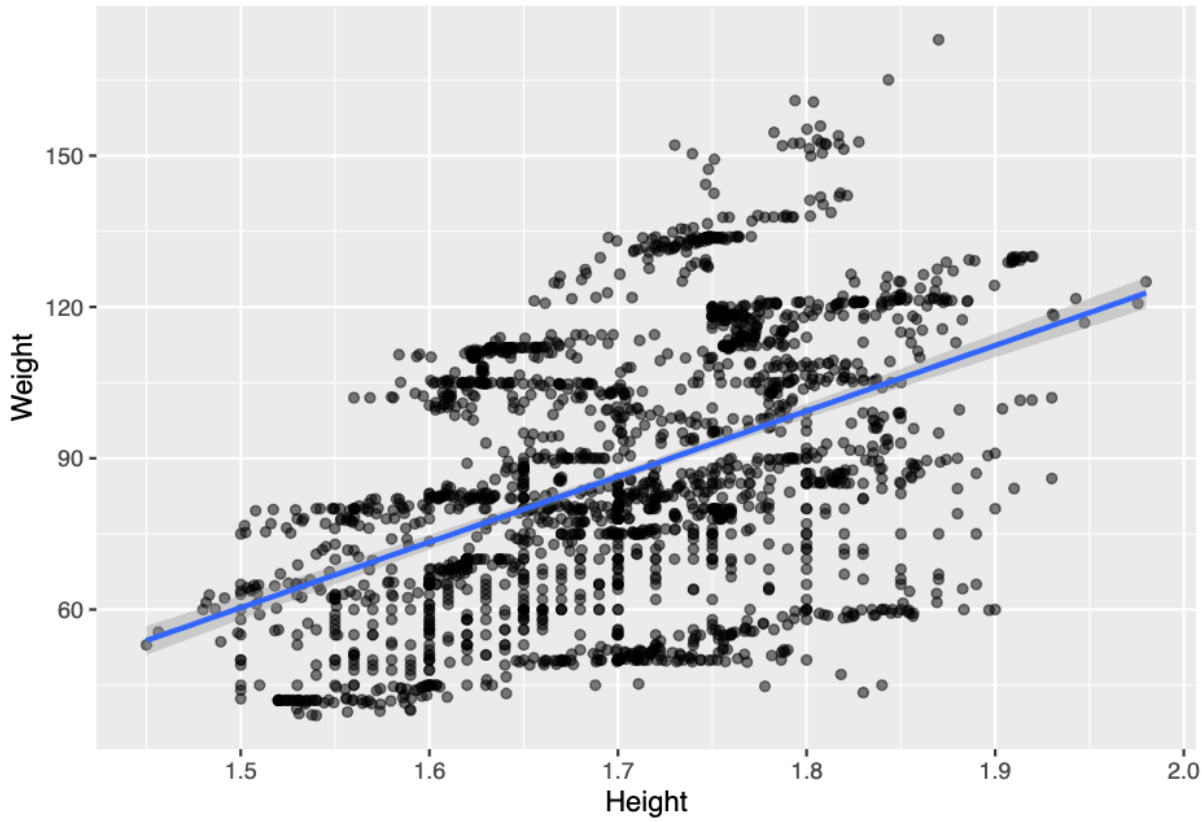
##Relationship between Numerivcal Variables:

#Height/Weight

The most important relationship is between height and weight. We calculate BMI - body mass index with the formula bmi = weight/ height^2. This determines the obesity class. Here is a scatter plot depicting the relationship:

```
ggplot(obesity, aes(Height,Weight)) + geom_point(alpha = 0.5) + geom_smooth(method = lm)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

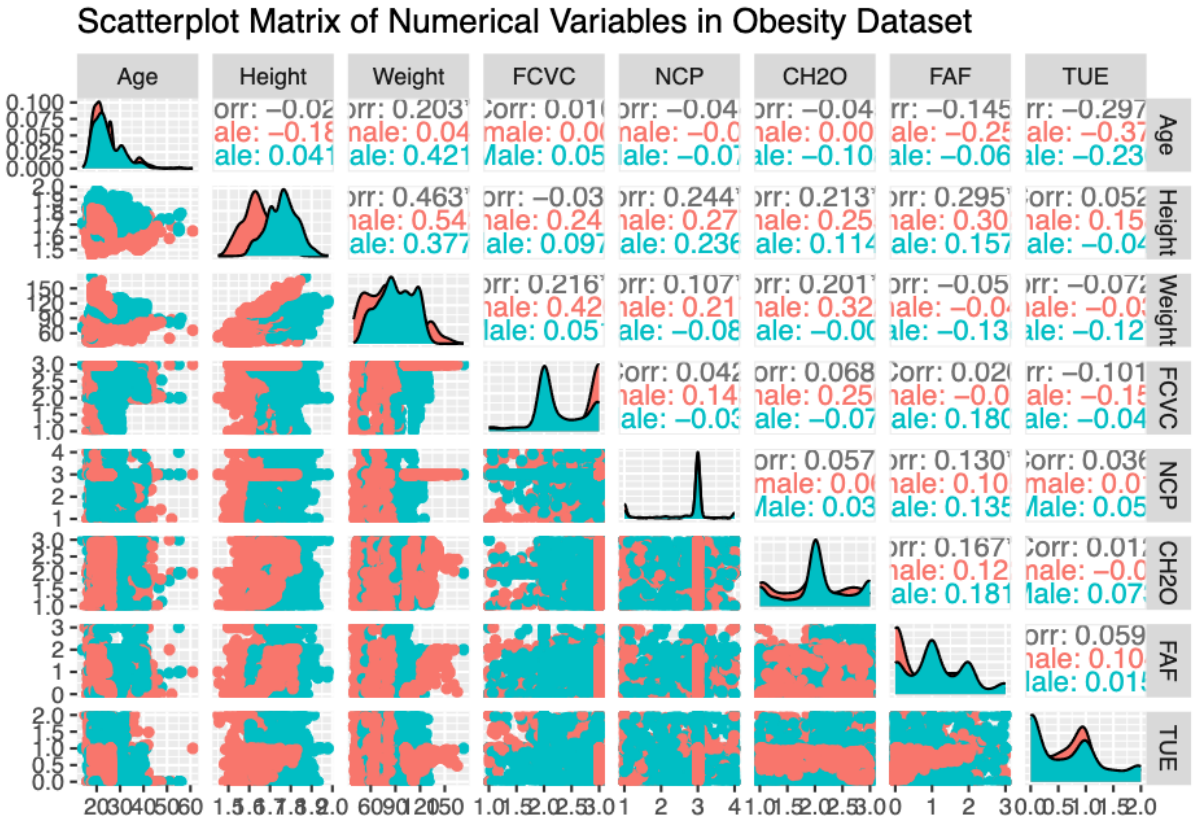


#Scatterplot matrix - Relationship between all numerical variables

```
# Load necessary libraries
library(ggplot2)
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
# Creating a subset of the dataset with only numerical variables
numerical_vars <- obesity[, c("Age", "Height", "Weight", "FCVC", "NCP", "CH2O", "FAF", "TUE")]

# Using ggpairs to create scatterplots of the numerical variables
ggpairs(numerical_vars,
        title = "Scatterplot Matrix of Numerical Variables in Obesity Dataset",
        mapping = ggplot2::aes(color = obesity$Gender)) # Color by Gender or any other categorical vari
```



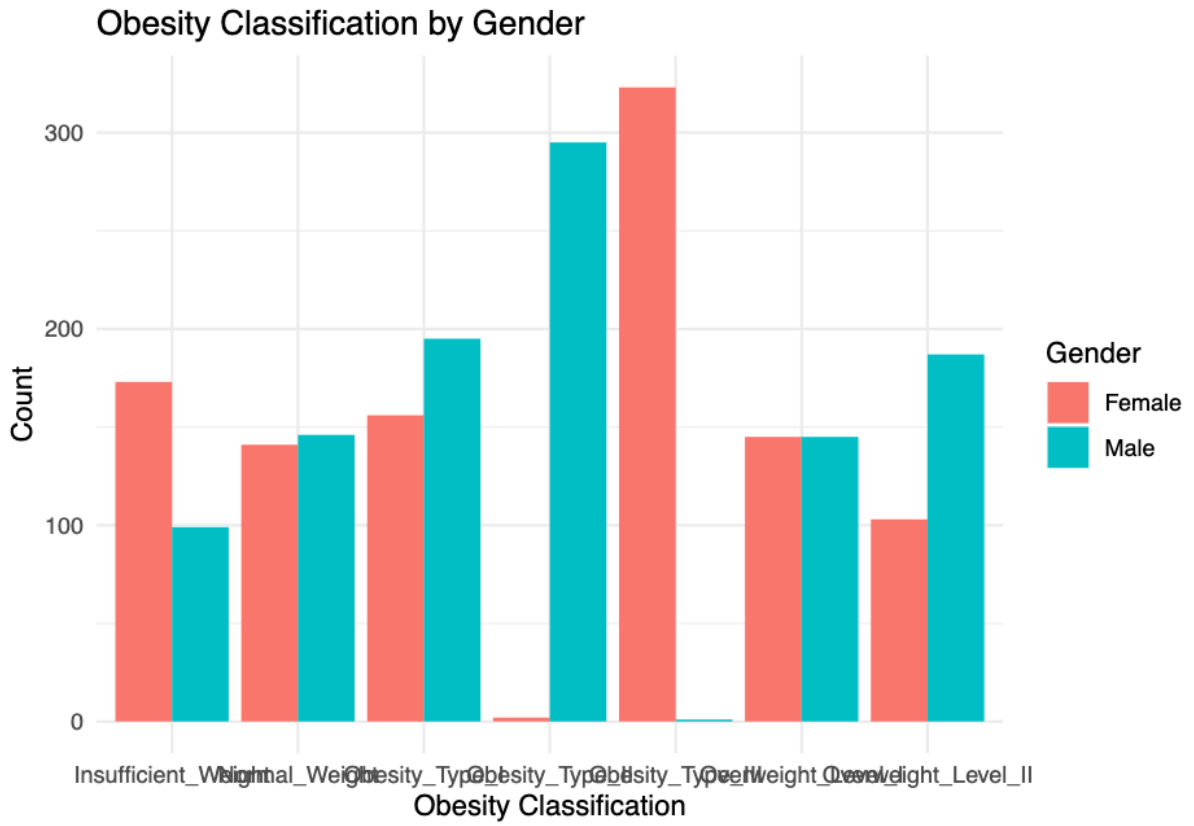
##Relationship between Categorical variables:

#Obesity vs Gender In "Insufficient weight" there are more women. Conversely, there are more obese men than women, except for the extreme obesity category, where the numbers are similar.

```
# Creating cross-tabulations between pairs of categorical variables
# NObeyesdad vs Gender
table(obesity$NObeyesdad, obesity$Gender)
```

```
##
##                      Female Male
##   Insufficient_Weight    173   99
##   Normal_Weight          141  146
##   Obesity_Type_I         156  195
##   Obesity_Type_II          2  295
##   Obesity_Type_III       323    1
##   Overweight_Level_I     145  145
##   Overweight_Level_II    103  187
```

```
# Create the bar graph
ggplot(obesity, aes(x = NObeyesdad, fill = Gender)) +
  geom_bar(position = position_dodge()) +
  labs(title = "Obesity Classification by Gender", x = "Obesity Classification", y = "Count") +
  theme_minimal()
```

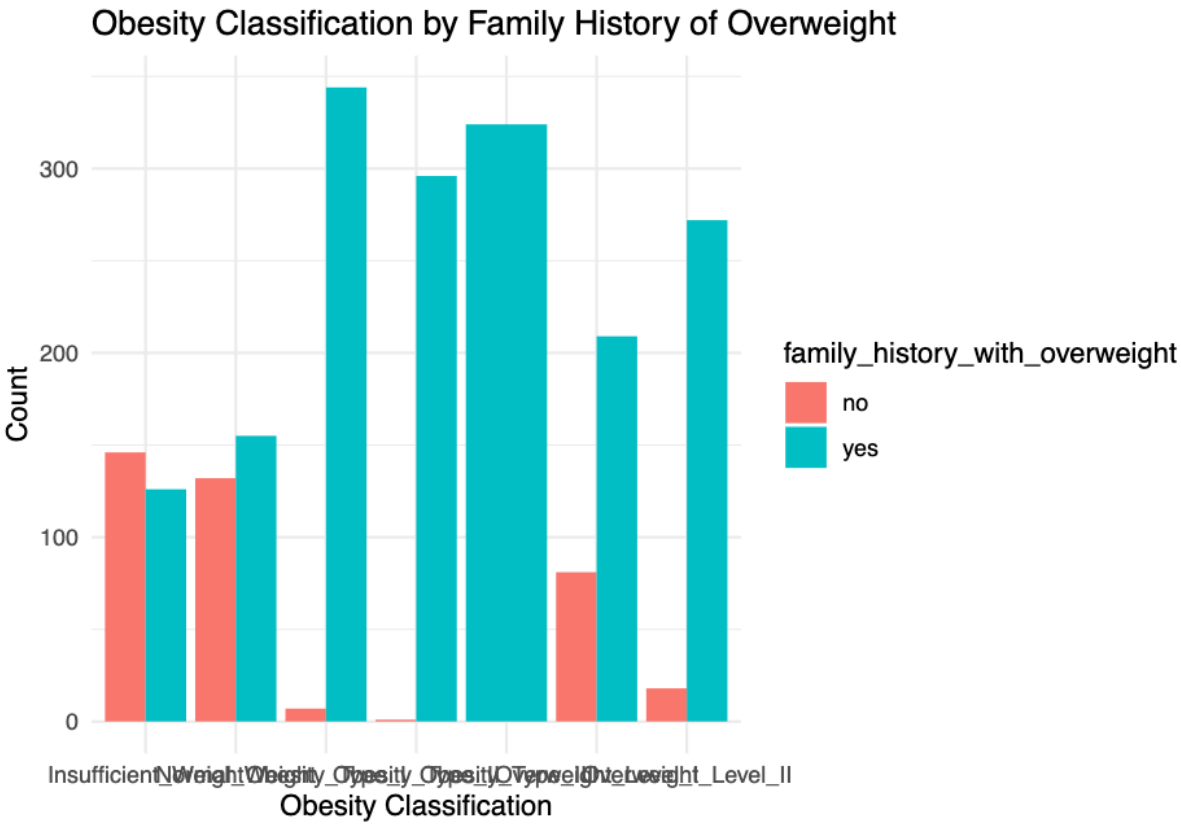


#Obesity vs Family History of Overweight

```
# NObeyesdad vs Family History of Overweight
table(obesity$NObeyesdad, obesity$family_history_with_overweight)
```

```
##
##                        no yes
##   Insufficient_Weight 146 126
##   Normal_Weight       132 155
##   Obesity_Type_I        7 344
##   Obesity_Type_II       1 296
##   Obesity_Type_III      0 324
##   Overweight_Level_I   81 209
##   Overweight_Level_II  18 272
```

```
# Family History of Overweight vs NObeyesdad
ggplot(obesity, aes(x = NObeyesdad, fill = family_history_with_overweight)) +
  geom_bar(position = position_dodge()) +
  labs(title = "Obesity Classification by Family History of Overweight", x = "Obesity Classification",
  theme_minimal()
```

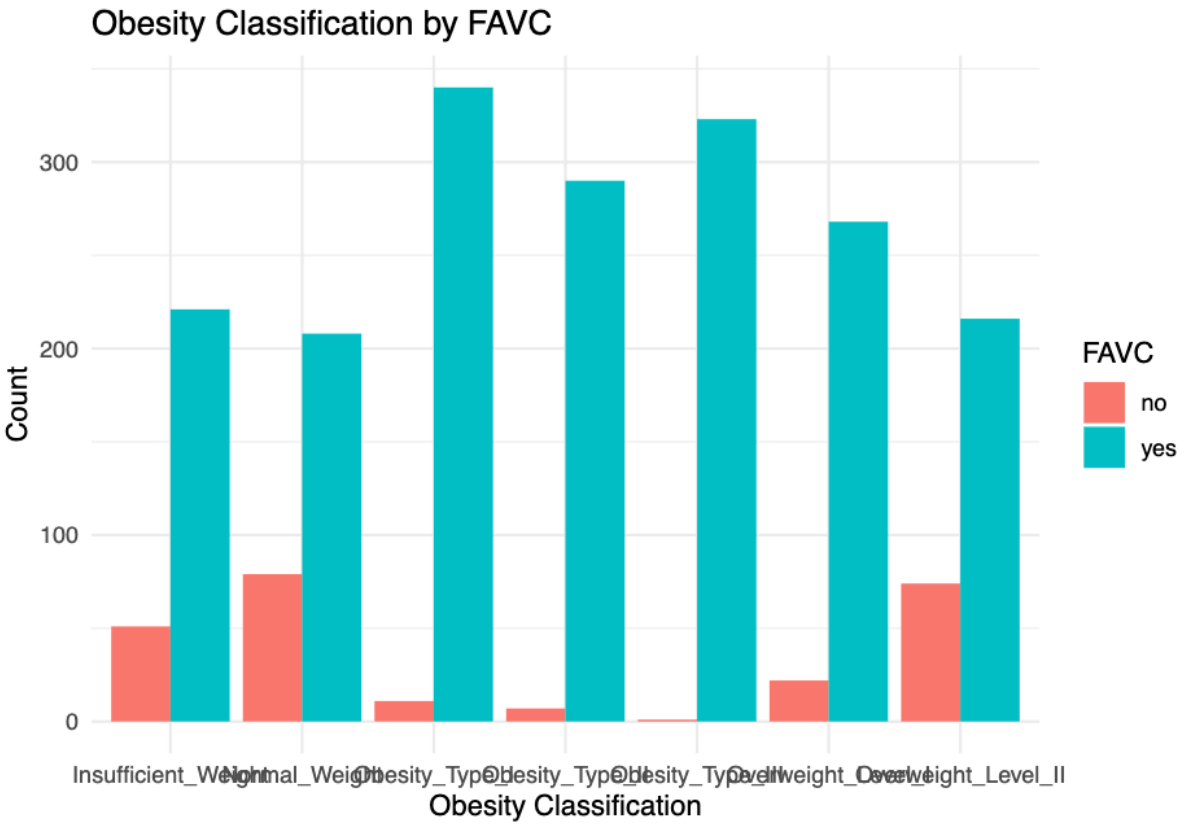


NObeyesdad vs Frequent Consumption of High Caloric Food (FAVC)

```
table(obesity$NObeyesdad, obesity$FAVC)
```

```
##
##                       no yes
##   Insufficient_Weight  51 221
##   Normal_Weight        79 208
##   Obesity_Type_I       11 340
##   Obesity_Type_II       7 290
##   Obesity_Type_III      1 323
##   Overweight_Level_I   22 268
##   Overweight_Level_II  74 216
```

```
# Frequent Consumption of High Caloric Food (FAVC) vs NObeyesdad
ggplot(obesity, aes(x = NObeyesdad, fill = FAVC)) +
  geom_bar(position = position_dodge()) +
  labs(title = "Obesity Classification by FAVC", x = "Obesity Classification", y = "Count") +
  theme_minimal()
```

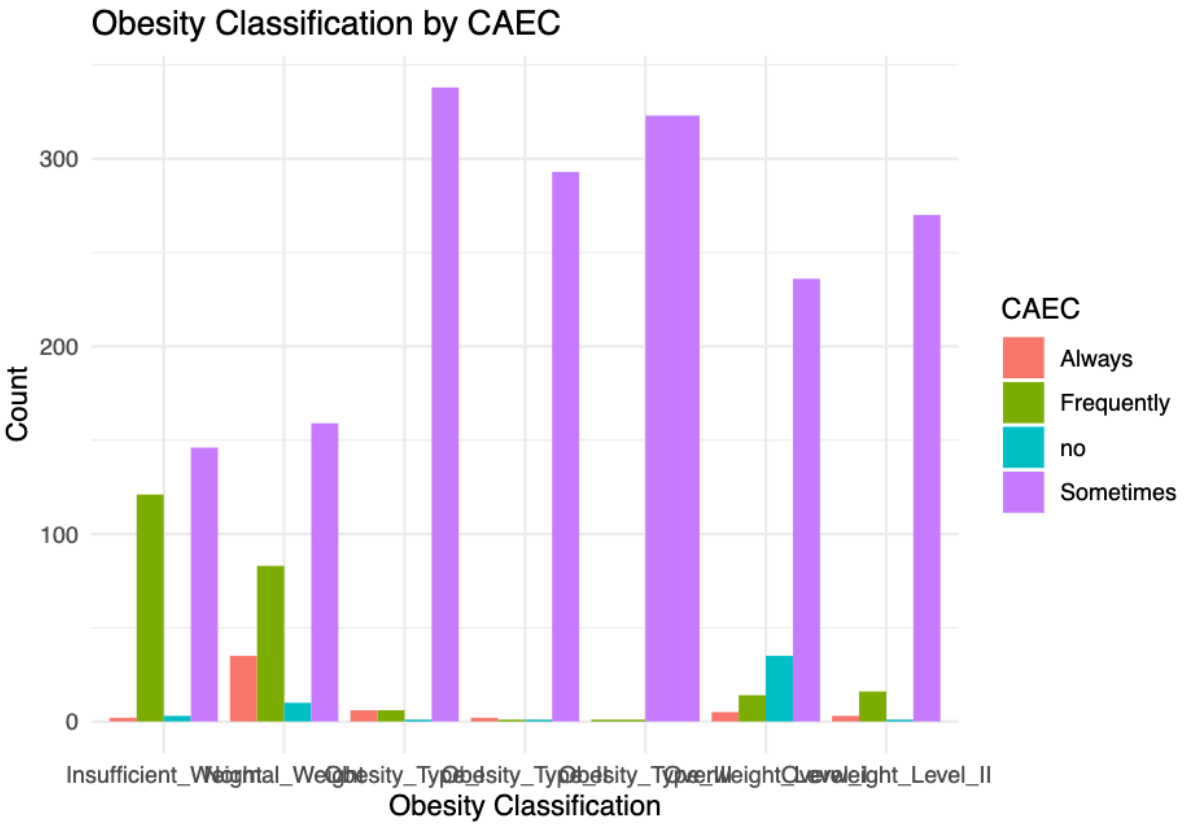


NObeyesdad vs Consumption of Food Between Meals (CAEC)

```
table(obesity$NObeyesdad, obesity$CAEC)
```

```
##
##                      Always Frequently  no Sometimes
##   Insufficient_Weight      2        121   3       146
##   Normal_Weight           35         83  10       159
##   Obesity_Type_I           6          6   1       338
##   Obesity_Type_II          2          1   1       293
##   Obesity_Type_III         0          1   0       323
##   Overweight_Level_I       5         14  35       236
##   Overweight_Level_II      3         16   1       270
```

```
ggplot(obesity, aes(x = NObeyesdad, fill = CAEC)) +
  geom_bar(position = position_dodge()) +
  labs(title = "Obesity Classification by CAEC", x = "Obesity Classification", y = "Count") +
  theme_minimal()
```

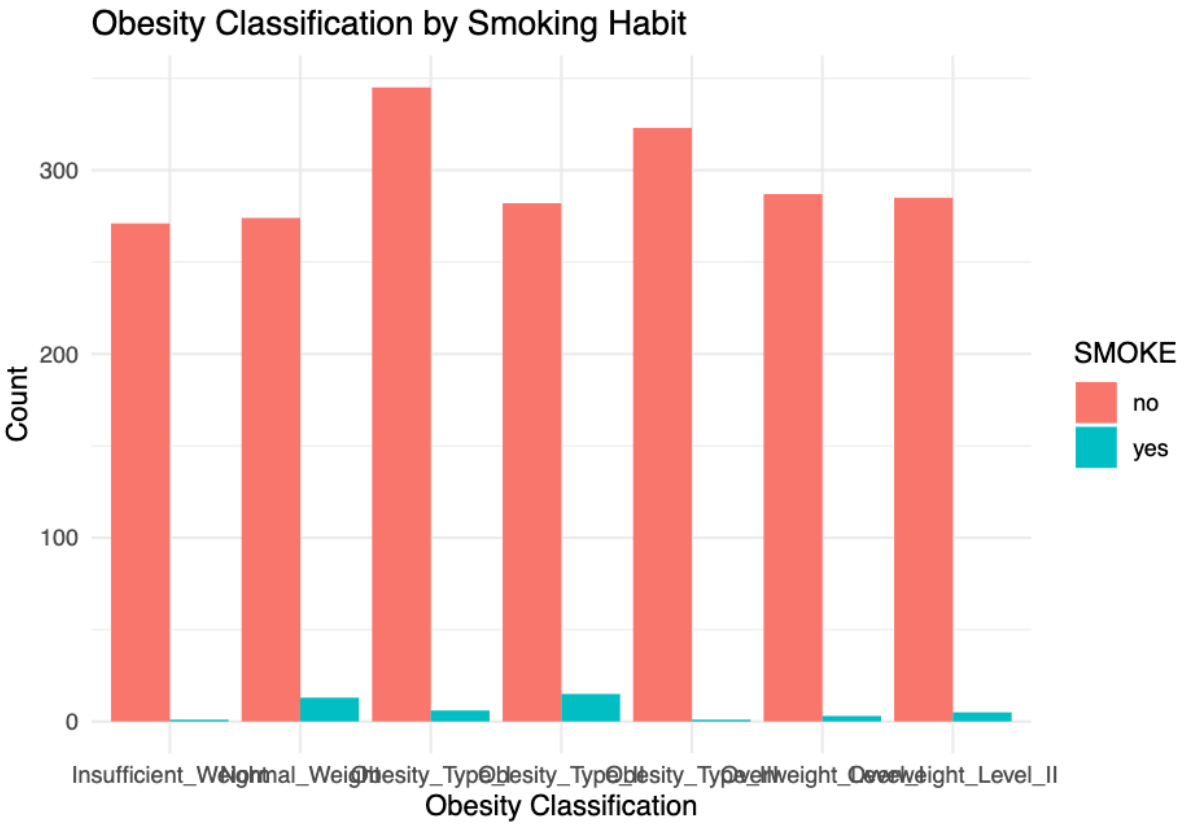


NObeyesdad vs Smoking Habit (SMOKE)

```
table(obesity$NObeyesdad, obesity$SMOKE)
```

```
##
##                       no yes
##   Insufficient_Weight 271   1
##   Normal_Weight       274  13
##   Obesity_Type_I      345   6
##   Obesity_Type_II     282  15
##   Obesity_Type_III    323   1
##   Overweight_Level_I  287   3
##   Overweight_Level_II 285   5
```

```
ggplot(obesity, aes(x = NObeyesdad, fill = SMOKE)) +
  geom_bar(position = position_dodge()) +
  labs(title = "Obesity Classification by Smoking Habit", x = "Obesity Classification", y = "Count") +
  theme_minimal()
```

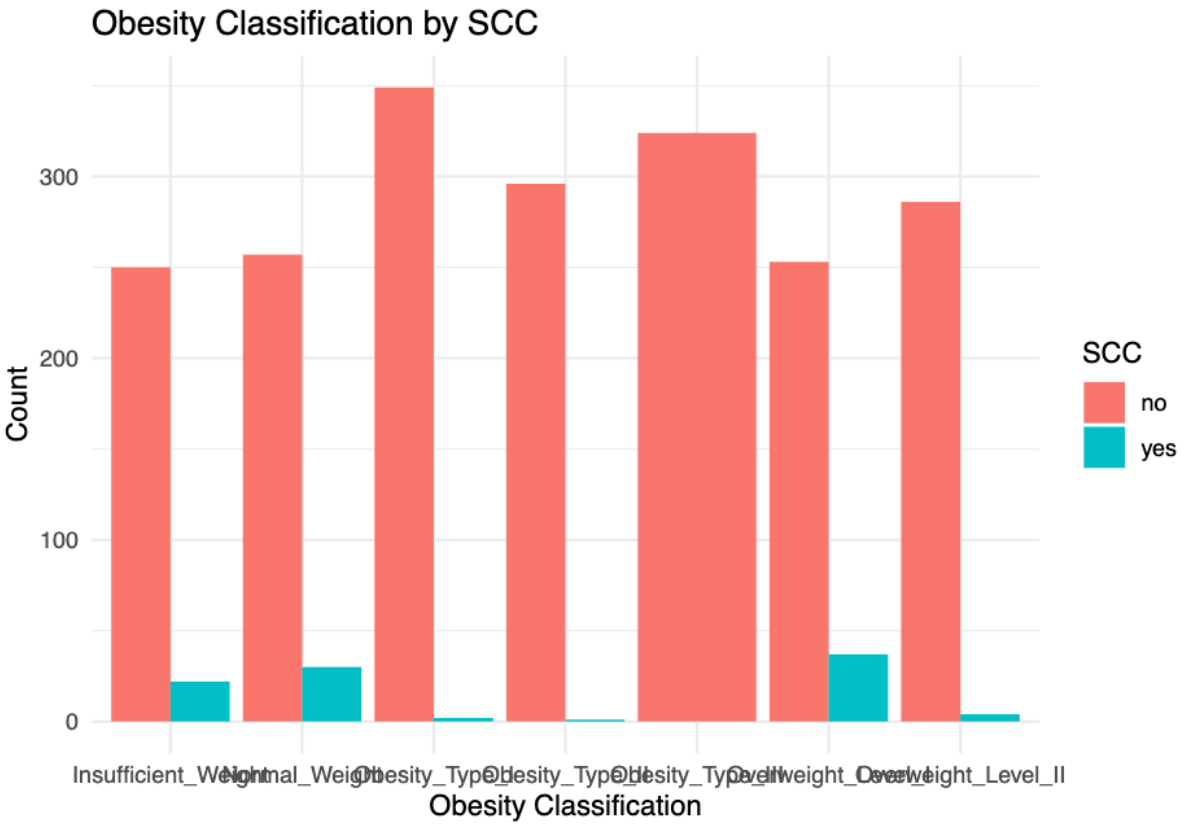


NObeyesdad vs Caloric Consumption Monitoring (SCC)

```
table(obesity$NObeyesdad, obesity$SCC)
```

```
##
##                       no yes
##   Insufficient_Weight 250  22
##   Normal_Weight       257  30
##   Obesity_Type_I      349   2
##   Obesity_Type_II     296   1
##   Obesity_Type_III    324   0
##   Overweight_Level_I  253  37
##   Overweight_Level_II 286   4
```

```
ggplot(obesity, aes(x = NObeyesdad, fill = SCC)) +
  geom_bar(position = position_dodge()) +
  labs(title = "Obesity Classification by SCC", x = "Obesity Classification", y = "Count") +
  theme_minimal()
```

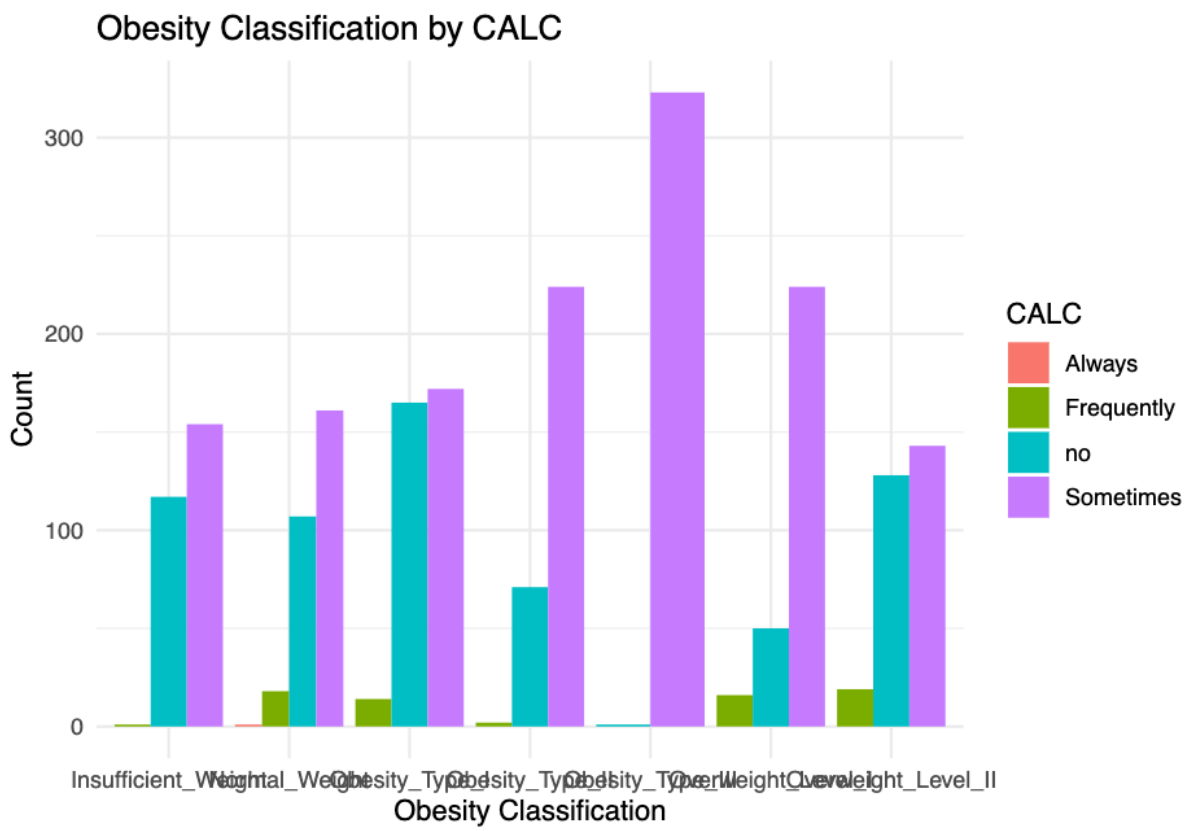


NObeyesdad vs Consumption of Alcohol (CALC)

```
table(obesity$NObeyesdad, obesity$CALC)
```

```
##
##                       Always Frequently  no Sometimes
##   Insufficient_Weight      0          1 117       154
##   Normal_Weight            1         18 107       161
##   Obesity_Type_I           0         14 165       172
##   Obesity_Type_II          0          2  71       224
##   Obesity_Type_III         0          0   1       323
##   Overweight_Level_I       0         16  50       224
##   Overweight_Level_II      0         19 128       143
```

```
ggplot(obesity, aes(x = NObeyesdad, fill = CALC)) +
  geom_bar(position = position_dodge()) +
  labs(title = "Obesity Classification by CALC", x = "Obesity Classification", y = "Count") +
  theme_minimal()
```

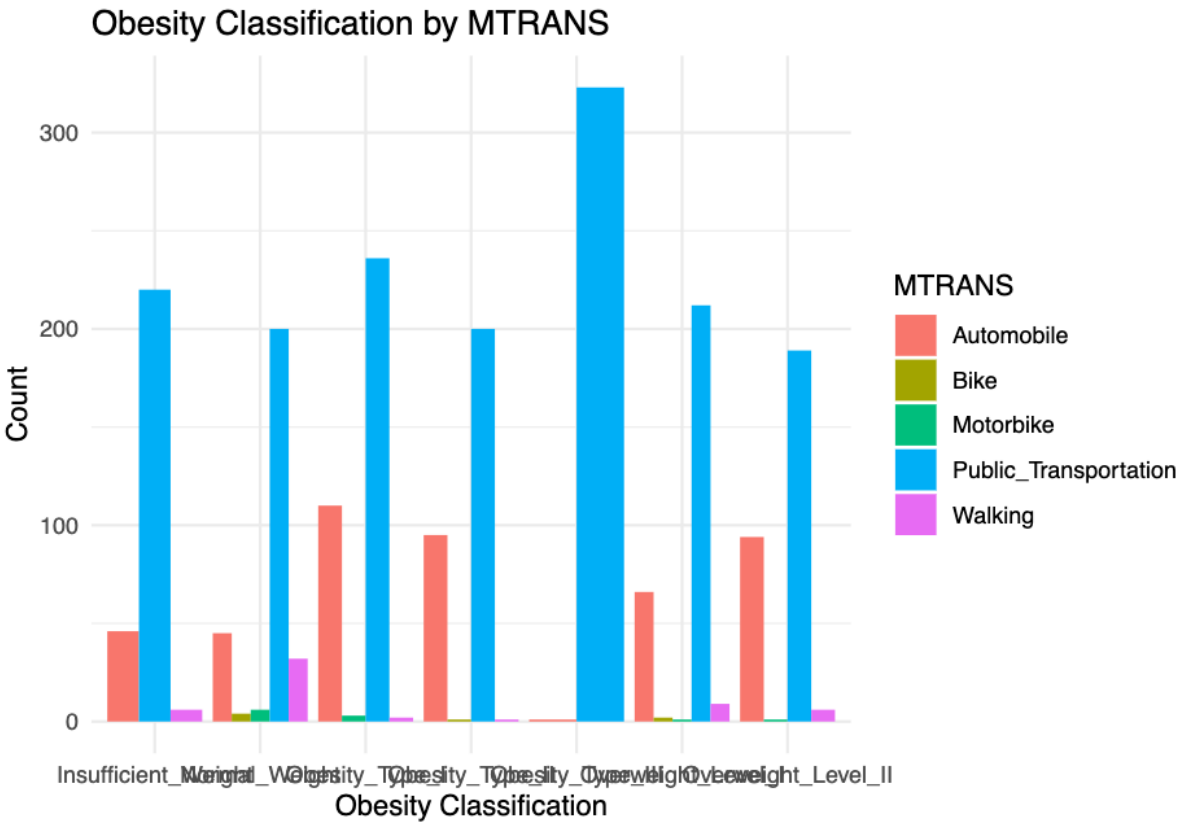


NObeyesdad vs Transportation Used (MTRANS)

```
table(obesity$NObeyesdad, obesity$MTRANS)
```

```
##
##                      Automobile Bike Motorbike Public_Transportation Walking
##   Insufficient_Weight         46    0         0                   220       6
##   Normal_Weight               45    4         6                   200      32
##   Obesity_Type_I             110    0         3                   236       2
##   Obesity_Type_II             95    1         0                   200       1
##   Obesity_Type_III             1    0         0                   323       0
##   Overweight_Level_I          66    2         1                   212       9
##   Overweight_Level_II         94    0         1                   189       6
```

```
ggplot(obesity, aes(x = NObeyesdad, fill = MTRANS)) +
  geom_bar(position = position_dodge()) +
  labs(title = "Obesity Classification by MTRANS", x = "Obesity Classification", y = "Count") +
  theme_minimal()
```



c. Data Cleaning Don't forget – this can take a lot of the time of the whole process. Your cleaning process must ensure that there are no missing values and all outliers must be considered. It may be reasonable to just remove rows with missing values, however, if your data or small or that would change the distributions of the variables, that will not be adequate and you will need to consider other options, as discussed in the modules on cleaning.Depending on your data and what you plan to do with it, you may also need to apply other processeswe discussed. For example, clean up strings for consistency, deal with date formatting, change variable types between categorical and numeric, bin, smooth, group, aggregate or reshape. Make the case with visualization or by showing resulting summary statistics that your data are clean enough to continue with your analysis.

```
obesity <- read.csv("/Users/kavanamanvi/Desktop/FDS/HW5/ObesityDataSet.csv")

#Remove missing values
obesity <- na.omit(obesity)
summary(obesity)
```

```
##     Gender               Age              Height          Weight
##  Length:2111        Min.   :14.00   Min.   :1.450   Min.   : 39.00
##  Class :character   1st Qu.:19.95   1st Qu.:1.630   1st Qu.: 65.47
##  Mode  :character   Median :22.78   Median :1.700   Median : 83.00
##                     Mean   :24.31   Mean   :1.702   Mean   : 86.59
##                     3rd Qu.:26.00   3rd Qu.:1.768   3rd Qu.:107.43
##                     Max.   :61.00   Max.   :1.980   Max.   :173.00
##  family_history_with_overweight     FAVC                FCVC
##  Length:2111                    Length:2111        Min.   :1.000
```

```
##  Class :character                Class :character   1st Qu.:2.000
##  Mode  :character                Mode  :character   Median :2.386
##                                                     Mean   :2.419
##                                                     3rd Qu.:3.000
##                                                     Max.   :3.000
##       NCP            CAEC              SMOKE                CH2O
##  Min.   :1.000   Length:2111        Length:2111        Min.   :1.000
##  1st Qu.:2.659   Class :character   Class :character   1st Qu.:1.585
##  Median :3.000   Mode  :character   Mode  :character   Median :2.000
##  Mean   :2.686                                         Mean   :2.008
##  3rd Qu.:3.000                                         3rd Qu.:2.477
##  Max.   :4.000                                         Max.   :3.000
##      SCC                FAF              TUE             CALC
##  Length:2111        Min.   :0.0000   Min.   :0.0000   Length:2111
##  Class :character   1st Qu.:0.1245   1st Qu.:0.0000   Class :character
##  Mode  :character   Median :1.0000   Median :0.6253   Mode  :character
##                     Mean   :1.0103   Mean   :0.6579
##                     3rd Qu.:1.6667   3rd Qu.:1.0000
##                     Max.   :3.0000   Max.   :2.0000
##     MTRANS            NObeyesdad
##  Length:2111        Length:2111
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

```r
nrow(obesity)
```

```
## [1] 2111
```

```r
# Removing duplicates
obesity <- obesity %>% distinct()
nrow(obesity)
```

```
## [1] 2087
```

```r
# Rename columns to simpler names
colnames(obesity) <- c(
  "Gender", "Age", "Height", "Weight", "FamilyHistory", "HighCaloricFood",
  "Vegetables", "MealNumber", "Snacking", "Smoking",
  "Water", "CaloricMonitoring", "PhysicalActivity", "TechnologyUse",
  "Alcohol", "Transport", "ObesityClass"
)

# Verify the column names
str(obesity)
```

```
## 'data.frame':    2087 obs. of  17 variables:
##  $ Gender           : chr  "Female" "Female" "Male" "Male" ...
##  $ Age              : num  21 21 23 27 22 29 23 22 24 22 ...
##  $ Height           : num  1.62 1.52 1.8 1.8 1.78 1.62 1.5 1.64 1.78 1.72 ...
```

```
##  $ Weight           : num  64 56 77 87 89.8 53 55 53 64 68 ...
##  $ FamilyHistory    : chr  "yes" "yes" "yes" "no" ...
##  $ HighCaloricFood  : chr  "no" "no" "no" "no" ...
##  $ Vegetables       : num  2 3 2 3 2 2 3 2 3 2 ...
##  $ MealNumber       : num  3 3 3 3 1 3 3 3 3 3 ...
##  $ Snacking         : chr  "Sometimes" "Sometimes" "Sometimes" "Sometimes" ...
##  $ Smoking          : chr  "no" "yes" "no" "no" ...
##  $ Water            : num  2 3 2 2 2 2 2 2 2 2 ...
##  $ CaloricMonitoring: chr  "no" "yes" "no" "no" ...
##  $ PhysicalActivity : num  0 3 2 2 0 0 1 3 1 1 ...
##  $ TechnologyUse    : num  1 0 1 0 0 0 0 0 1 1 ...
##  $ Alcohol          : chr  "no" "Sometimes" "Frequently" "Frequently" ...
##  $ Transport        : chr  "Public_Transportation" "Public_Transportation" "Public_Transportation" "
##  $ ObesityClass     : chr  "Normal_Weight" "Normal_Weight" "Normal_Weight" "Overweight_Level_I" ...
```

```
# Displaying the first few rows of the converted columns
head(obesity[, c("Gender", "HighCaloricFood", "Smoking", "CaloricMonitoring","FamilyHistory")])
```

```
##   Gender HighCaloricFood Smoking CaloricMonitoring FamilyHistory
## 1 Female              no      no                no           yes
## 2 Female              no     yes               yes           yes
## 3   Male              no      no                no           yes
## 4   Male              no      no                no            no
## 5   Male              no      no                no            no
## 6   Male             yes      no                no            no
```

```
# Load necessary libraries
library(dplyr)
library(tidyr)
#Change categorical variables to factors
obesity <-  as.data.frame (obesity)
obesity  <- obesity  %>% mutate (Snacking =  as.factor (Snacking))  %>% drop_na
obesity  <- obesity  %>% mutate (Alcohol =  as.factor (Alcohol))  %>% drop_na
obesity  <- obesity  %>% mutate (Transport =  as.factor (Transport))%>% drop_na
obesity  <- obesity  %>% mutate (ObesityClass=  as.factor (ObesityClass))
str(obesity)
```

```
## 'data.frame':    2087 obs. of  17 variables:
##  $ Gender           : chr  "Female" "Female" "Male" "Male" ...
##  $ Age              : num  21 21 23 27 22 29 23 22 24 22 ...
##  $ Height           : num  1.62 1.52 1.8 1.8 1.78 1.62 1.5 1.64 1.78 1.72 ...
##  $ Weight           : num  64 56 77 87 89.8 53 55 53 64 68 ...
##  $ FamilyHistory    : chr  "yes" "yes" "yes" "no" ...
##  $ HighCaloricFood  : chr  "no" "no" "no" "no" ...
##  $ Vegetables       : num  2 3 2 3 2 2 3 2 3 2 ...
##  $ MealNumber       : num  3 3 3 3 1 3 3 3 3 3 ...
##  $ Snacking         : Factor w/ 4 levels "Always","Frequently",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ Smoking          : chr  "no" "yes" "no" "no" ...
##  $ Water            : num  2 3 2 2 2 2 2 2 2 2 ...
##  $ CaloricMonitoring: chr  "no" "yes" "no" "no" ...
##  $ PhysicalActivity : num  0 3 2 2 0 0 1 3 1 1 ...
##  $ TechnologyUse    : num  1 0 1 0 0 0 0 0 1 1 ...
##  $ Alcohol          : Factor w/ 4 levels "Always","Frequently",..: 3 4 2 2 4 4 4 4 2 3 ...
```

```
##  $ Transport        : Factor w/ 5 levels "Automobile","Bike",..: 4 4 4 5 4 1 3 4 4 4 ...
##  $ ObesityClass     : Factor w/ 7 levels "Insufficient_Weight",..: 2 2 2 6 7 2 2 2 2 2 ...
```

```r
# Create new column bmi
obesity$bmi <- obesity$Weight / (obesity$Height^2)

# Display the first few rows of the dataset
head(obesity)
```

```
##   Gender Age Height Weight FamilyHistory HighCaloricFood Vegetables MealNumber
## 1 Female  21   1.62   64.0           yes              no          2          3
## 2 Female  21   1.52   56.0           yes              no          3          3
## 3   Male  23   1.80   77.0           yes              no          2          3
## 4   Male  27   1.80   87.0            no              no          3          3
## 5   Male  22   1.78   89.8            no              no          2          1
## 6   Male  29   1.62   53.0            no             yes          2          3
##    Snacking Smoking Water CaloricMonitoring PhysicalActivity TechnologyUse
## 1 Sometimes      no     2                no                0             1
## 2 Sometimes     yes     3               yes                3             0
## 3 Sometimes      no     2                no                2             1
## 4 Sometimes      no     2                no                2             0
## 5 Sometimes      no     2                no                0             0
## 6 Sometimes      no     2                no                0             0
##      Alcohol             Transport        ObesityClass      bmi
## 1         no Public_Transportation       Normal_Weight 24.38653
## 2  Sometimes Public_Transportation       Normal_Weight 24.23823
## 3 Frequently Public_Transportation       Normal_Weight 23.76543
## 4 Frequently               Walking  Overweight_Level_I 26.85185
## 5  Sometimes Public_Transportation Overweight_Level_II 28.34238
## 6  Sometimes            Automobile       Normal_Weight 20.19509
```

```r
summary(obesity$bmi)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   13.00   24.37   28.90   29.77   36.10   50.81
```

d. Data Preprocessing In some cases, preprocessing is absolutely necessary. It is rarely a bad idea. Make the case for what is and is not necessary given what you plan to do with the data. This could include making dummy variables, applying normalization, binning and/or smoothing, and other transformations (see course module).

#Binning/Smoothning

```r
#This data is used for descision trees
#Create a bin and smooth
obesity_bin <- obesity
# Define the BMI categories
obesity_bin <- obesity_bin %>%
  mutate(BMICategory = case_when(
    bmi < 18.5 ~ "Underweight",
    bmi >= 18.5 & bmi < 25.0 ~ "Healthy Weight",
    bmi >= 25.0 & bmi < 30.0 ~ "Overweight",
```

```
    bmi >= 30.0 & bmi < 35.0 ~ "Obesity I",
    bmi >= 35.0 & bmi < 40.0 ~ "Obesity II",
    bmi >= 40.0 & bmi < 45.0 ~ "Obesity III",
    bmi >= 45.0 ~ "Extreme Obesity"
  ))

# View the distribution of the new BMI categories
table(obesity_bin$BMICategory)
```

```
##
## Extreme Obesity  Healthy Weight       Obesity I      Obesity II     Obesity III
##              37             295             368             338             231
##      Overweight     Underweight
##             552             266
```

```
#covert BMICategory to factor
obesity_bin  <- obesity_bin  %>% mutate (BMICategory=  as.factor (BMICategory))

#Remove bmi column as it is redundant
obesity_bin <- obesity_bin %>% select(-bmi)

obesity_bin$Gender <- as.factor(obesity_bin$Gender)
obesity_bin$FamilyHistory <- as.factor(obesity_bin$FamilyHistory)
obesity_bin$HighCaloricFood <- as.factor(obesity_bin$HighCaloricFood)
obesity_bin$Vegetables <- as.factor(obesity_bin$Vegetables)
obesity_bin$Snacking <- as.factor(obesity_bin$Snacking)
obesity_bin$Smoking <- as.factor(obesity_bin$Smoking)
obesity_bin$CaloricMonitoring <- as.factor(obesity_bin$CaloricMonitoring)
obesity_bin$TechnologyUse <- as.factor(obesity_bin$TechnologyUse)
obesity_bin$Alcohol <- as.factor(obesity_bin$Alcohol)
obesity_bin$Transport <- as.factor(obesity_bin$Transport)
obesity_bin$ObesityClass <- as.factor(obesity_bin$ObesityClass)
obesity_bin$BMICategory <- as.factor(obesity_bin$BMICategory)

str(obesity_bin)
```

```
## 'data.frame':    2087 obs. of  18 variables:
##  $ Gender           : Factor w/ 2 levels "Female","Male": 1 1 2 2 2 2 1 2 2 2 ...
##  $ Age              : num  21 21 23 27 22 29 23 22 24 22 ...
##  $ Height           : num  1.62 1.52 1.8 1.8 1.78 1.62 1.5 1.64 1.78 1.72 ...
##  $ Weight           : num  64 56 77 87 89.8 53 55 53 64 68 ...
##  $ FamilyHistory    : Factor w/ 2 levels "no","yes": 2 2 2 1 1 1 2 1 2 2 ...
##  $ HighCaloricFood  : Factor w/ 2 levels "no","yes": 1 1 1 1 1 2 2 1 2 2 ...
##  $ Vegetables       : Factor w/ 810 levels "1","1.003566",..: 171 810 171 810 171 171 810 171 810 17
##  $ MealNumber       : num  3 3 3 3 1 3 3 3 3 3 ...
##  $ Snacking         : Factor w/ 4 levels "Always","Frequently",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ Smoking          : Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 1 1 1 1 ...
##  $ Water            : num  2 3 2 2 2 2 2 2 2 2 ...
##  $ CaloricMonitoring: Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 1 1 1 1 ...
##  $ PhysicalActivity : num  0 3 2 2 0 0 1 3 1 1 ...
##  $ TechnologyUse    : Factor w/ 1129 levels "0","7.3e-05",..: 841 1 841 1 1 1 1 1 841 841 ...
##  $ Alcohol          : Factor w/ 4 levels "Always","Frequently",..: 3 4 2 2 4 4 4 4 2 3 ...
##  $ Transport        : Factor w/ 5 levels "Automobile","Bike",..: 4 4 4 5 4 1 3 4 4 4 ...
```

```
##  $ ObesityClass      : Factor w/ 7 levels "Insufficient_Weight",..: 2 2 2 6 7 2 2 2 2 2 ...
##  $ BMICategory       : Factor w/ 7 levels "Extreme Obesity",..: 2 2 2 6 6 2 2 2 2 2 ...
```

#Normalizartion

```
library(caret)
```

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

```
# Center scale allows us to standardize the data
preproc1 <- preProcess(obesity, method=c("center", "scale"))
# We have to call predict to fit our data based on preprocessing
obesity_normalized <- predict(preproc1, obesity)
# Here we can see the standardized version of our dataset
summary(obesity_normalized)
```

```
##     Gender              Age               Height            Weight        
##  Length:2087        Min.   :-1.6256   Min.   :-2.7115   Min.   :-1.8273  
##  Class :character   1st Qu.:-0.6967   1st Qu.:-0.7780   1st Qu.:-0.7964  
##  Mode  :character   Median :-0.2364   Median :-0.0117   Median :-0.1435  
##                     Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000  
##                     3rd Qu.: 0.2586   3rd Qu.: 0.7170   3rd Qu.: 0.8078  
##                     Max.   : 5.7541   Max.   : 2.9760   Max.   : 3.2890  
##                                                                          
##  FamilyHistory      HighCaloricFood      Vegetables         MealNumber       
##  Length:2087        Length:2087        Min.   :-2.65825   Min.   :-2.224885  
##  Class :character   Class :character   1st Qu.:-0.78818   1st Qu.:-0.004855  
##  Mode  :character   Mode  :character   Median :-0.04713   Median : 0.390813  
##                                        Mean   : 0.00000   Mean   : 0.000000  
##                                        3rd Qu.: 1.08190   3rd Qu.: 0.390813  
##                                        Max.   : 1.08190   Max.   : 1.698661  
##                                                                              
##        Snacking      Smoking              Water           CaloricMonitoring 
##  Always    :  53   Length:2087        Min.   :-1.651776   Length:2087       
##  Frequently: 236   Class :character   1st Qu.:-0.680320   Class :character  
##  no        :  37   Mode  :character   Median :-0.007808   Mode  :character  
##  Sometimes :1761                      Mean   : 0.000000                     
##                                       3rd Qu.: 0.758598                     
##                                       Max.   : 1.636160                     
##                                                                             
##  PhysicalActivity   TechnologyUse          Alcohol     
##  Min.   :-1.18669   Min.   :-1.0902   Always    :   1  
##  1st Qu.:-1.04081   1st Qu.:-1.0902   Frequently:  70  
##  Median :-0.01501   Median :-0.0529   no        : 636  
##  Mean   : 0.00000   Mean   : 0.0000   Sometimes :1380  
##  3rd Qu.: 0.77951   3rd Qu.: 0.5541                    
##  Max.   : 2.32835   Max.   : 2.1984                    
##                                                        
##                  Transport                ObesityClass      bmi         
##  Automobile           : 456   Insufficient_Weight:267   Min.   :-2.0894  
##  Bike                 :   7   Normal_Weight      :282   1st Qu.:-0.6725  
##  Motorbike            :  11   Obesity_Type_I     :351   Median :-0.1084  
##  Public_Transportation:1558   Obesity_Type_II    :297   Mean   : 0.0000  
```

```
##  Walking              :  55   Obesity_Type_III   :324   3rd Qu.: 0.7888
##                              Overweight_Level_I :276   Max.   : 2.6226
##                              Overweight_Level_II:290
```

```
#Change boolean categorical to numeric
obesity_normalized$Gender<- as.integer(obesity$Gender == "Female")
obesity_normalized$HighCaloricFood <- as.integer(obesity$HighCaloricFood == "yes")
obesity_normalized$Smoking <- as.integer(obesity$Smoking == "yes")
obesity_normalized$CaloricMonitoring <- as.integer(obesity$CaloricMonitoring == "yes")
obesity_normalized$FamilyHistory <- as.integer(obesity$FamilyHistory == "yes")
str(obesity_normalized)
```

```
## 'data.frame':    2087 obs. of  18 variables:
##  $ Gender           : int  1 1 0 0 0 0 1 0 0 0 ...
##  $ Age              : num  -0.526 -0.526 -0.212 0.416 -0.369 ...
##  $ Height           : num  -0.887 -1.96 1.044 1.044 0.83 ...
##  $ Weight           : num  -0.87278 -1.17823 -0.37642 0.00539 0.1123 ...
##  $ FamilyHistory    : int  1 1 1 0 0 0 1 0 1 1 ...
##  $ HighCaloricFood  : int  0 0 0 0 0 1 1 0 1 1 ...
##  $ Vegetables       : num  -0.788 1.082 -0.788 1.082 -0.788 ...
##  $ MealNumber       : num  0.391 0.391 0.391 0.391 -2.225 ...
##  $ Snacking         : Factor w/ 4 levels "Always","Frequently",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ Smoking          : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ Water            : num  -0.00781 1.63616 -0.00781 -0.00781 -0.00781 ...
##  $ CaloricMonitoring: int  0 1 0 0 0 0 0 0 0 0 ...
##  $ PhysicalActivity : num  -1.19 2.33 1.16 1.16 -1.19 ...
##  $ TechnologyUse    : num  0.554 -1.09 0.554 -1.09 -1.09 ...
##  $ Alcohol          : Factor w/ 4 levels "Always","Frequently",..: 3 4 2 2 4 4 4 4 2 3 ...
##  $ Transport        : Factor w/ 5 levels "Automobile","Bike",..: 4 4 4 5 4 1 3 4 4 4 ...
##  $ ObesityClass     : Factor w/ 7 levels "Insufficient_Weight",..: 2 2 2 6 7 2 2 2 2 2 ...
##  $ bmi              : num  -0.67 -0.689 -0.748 -0.363 -0.177 ...
```

#Dummy variables

```
library(caret)
library(e1071)
#Convert categorical variables to dummy as most classifiers only work with numerical data
# cut is the target variable below
dummy <- dummyVars(ObesityClass ~ ., data = obesity)
# Using the dummy predictor we need to transform our set into the dummy variable version
# The result won't be a data frame, so we need to transform it into one
dummies <- as.data.frame(predict(dummy, newdata = obesity))
```

```
## Warning in model.frame.default(Terms, newdata, na.action = na.action, xlev =
## object$lvls): variable 'ObesityClass' is not a factor
```

```
head(dummies)
```

```
##   GenderFemale GenderMale Age Height Weight FamilyHistoryno FamilyHistoryyes
## 1            1          0  21   1.62   64.0               0                1
## 2            1          0  21   1.52   56.0               0                1
## 3            0          1  23   1.80   77.0               0                1
```

```
## 4            0          1  27   1.80   87.0               1                0
## 5            0          1  22   1.78   89.8               1                0
## 6            0          1  29   1.62   53.0               1                0
##   HighCaloricFoodno HighCaloricFoodyes Vegetables MealNumber Snacking.Always
## 1                 1                  0          2          3                0
## 2                 1                  0          3          3                0
## 3                 1                  0          2          3                0
## 4                 1                  0          3          3                0
## 5                 1                  0          2          1                0
## 6                 0                  1          2          3                0
##   Snacking.Frequently Snacking.no Snacking.Sometimes Smokingno Smokingyes Water
## 1                   0           0                  1         1          0     2
## 2                   0           0                  1         0          1     3
## 3                   0           0                  1         1          0     2
## 4                   0           0                  1         1          0     2
## 5                   0           0                  1         1          0     2
## 6                   0           0                  1         1          0     2
##   CaloricMonitoringno CaloricMonitoringyes PhysicalActivity TechnologyUse
## 1                   1                    0                0             1
## 2                   0                    1                3             0
## 3                   1                    0                2             1
## 4                   1                    0                2             0
## 5                   1                    0                0             0
## 6                   1                    0                0             0
##   Alcohol.Always Alcohol.Frequently Alcohol.no Alcohol.Sometimes
## 1              0                   0          1                0
## 2              0                   0          0                1
## 3              0                   1          0                0
## 4              0                   1          0                0
## 5              0                   0          0                1
## 6              0                   0          0                1
##   Transport.Automobile Transport.Bike Transport.Motorbike
## 1                    0              0                   0
## 2                    0              0                   0
## 3                    0              0                   0
## 4                    0              0                   0
## 5                    0              0                   0
## 6                    1              0                   0
##   Transport.Public_Transportation Transport.Walking      bmi
## 1                               1                 0 24.38653
## 2                               1                 0 24.23823
## 3                               1                 0 23.76543
## 4                               0                 1 26.85185
## 5                               1                 0 28.34238
## 6                               0                 0 20.19509
```

e. Clustering Remove any labels from your data and use clustering to discover any built-in structure. Use an appropriate method to determine the number of clusters. If your data have labels, compare the clusters to those labels. If not, visualize the clustering results by making a PCA projection and coloring the points by cluster assignment. Note that PCA only works for numerical variables, so if your data have just a few categoricals, you may skip them. If there are many, use dummy variables or choose a different method for making a projection. One way is to make the distance matrix first (we covered a method for distance matrices using categorical variables in the clustering tutorial) and then apply PCA to that matrix. This is actually a way to calculate an MDS projection, a very popular

method.

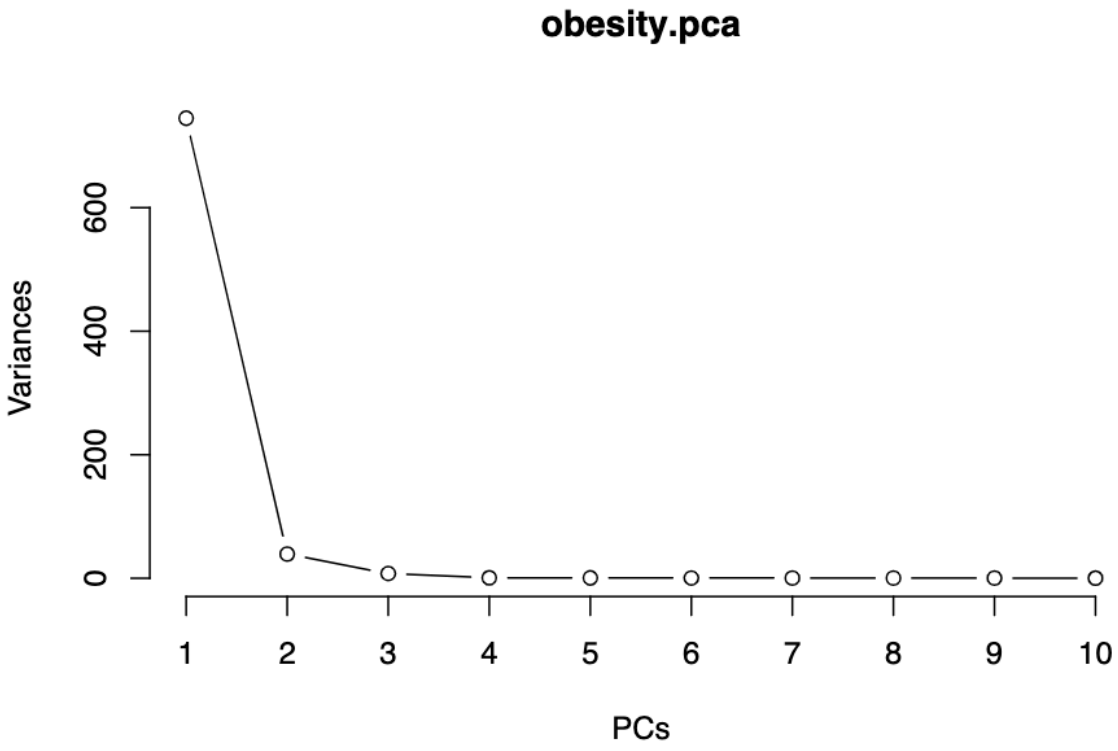
```
# Here you can see the converted version


nzv <- nearZeroVar(dummies)
# Normally nearZeroVar returns a list of indices but here we don't have any near zero variance predicto
length(nzv)
```

```
## [1] 11
```

```
#pca
obesity.pca <- prcomp(dummies)

# View the PCA summary with cumulative proportions
summary(obesity.pca)
```

```
## Importance of components:
##                             PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     27.2909 6.26204 2.76600 0.82153 0.75631 0.66203 0.58725
## Proportion of Variance  0.9362 0.04929 0.00962 0.00085 0.00072 0.00055 0.00043
## Cumulative Proportion   0.9362 0.98551 0.99513 0.99598 0.99669 0.99725 0.99768
##                             PC8    PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation     0.57091 0.5664 0.47290 0.45663 0.42935 0.40830 0.38276
## Proportion of Variance 0.00041 0.0004 0.00028 0.00026 0.00023 0.00021 0.00018
## Cumulative Proportion  0.99809 0.9985 0.99877 0.99904 0.99927 0.99948 0.99966
##                            PC15    PC16    PC17    PC18    PC19    PC20    PC21
## Standard deviation     0.2809 0.21879 0.19897 0.19473 0.18543 0.13847 0.08444
## Proportion of Variance 0.0001 0.00006 0.00005 0.00005 0.00004 0.00002 0.00001
## Cumulative Proportion  0.9998 0.99982 0.99987 0.99992 0.99996 0.99998 0.99999
##                            PC22    PC23    PC24      PC25      PC26      PC27
## Standard deviation     0.06159 0.02488 0.02073 6.215e-15 2.471e-15 2.471e-15
## Proportion of Variance 0.00000 0.00000 0.00000 0.000e+00 0.000e+00 0.000e+00
## Cumulative Proportion  1.00000 1.00000 1.00000 1.000e+00 1.000e+00 1.000e+00
##                             PC28      PC29      PC30      PC31      PC32
## Standard deviation     2.471e-15 2.471e-15 2.471e-15 2.471e-15 2.471e-15
## Proportion of Variance 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00
## Cumulative Proportion  1.000e+00 1.000e+00 1.000e+00 1.000e+00 1.000e+00
```

```
# Visualize the scree plot
screeplot(obesity.pca, type = "l") + title(xlab = "PCs")
```




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

```
# We don't want to include a prediction target variable in PCA, so we'll separate it
target <- obesity %>% dplyr::select(ObesityClass)

# Create the components
preProc <- preProcess(dummies, method="pca", pcaComp=2)
obesity.pc <- predict(preProc, dummies)
# Put back target column
obesity.pc$ObesityClass <- obesity$ObesityClass
# Make sure that we have the PCs as predictors
head(obesity.pc)
```

```
##         PC1        PC2         ObesityClass
## 1 2.181439 -0.4858502       Normal_Weight
## 2 3.988994  0.2846848       Normal_Weight
## 3 1.037272  2.6026115       Normal_Weight
## 4 2.626845  3.8140599  Overweight_Level_I
## 5 1.904157  0.6213957 Overweight_Level_II
## 6 1.051521  1.7458411       Normal_Weight
```

#KMeans

```r
library(stats)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

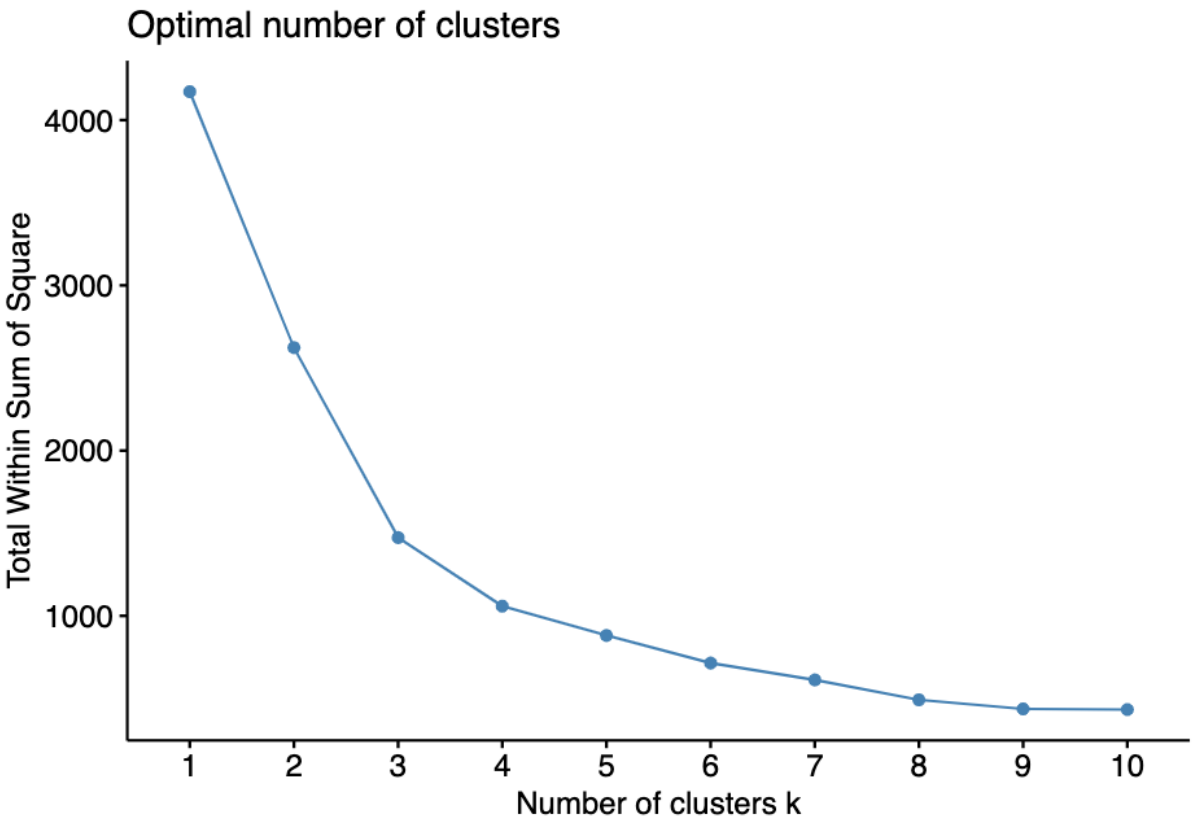
```r
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v readr     2.1.5
## v lubridate 1.9.3     v stringr   1.5.1
## v purrr     1.0.2     v tibble    3.2.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

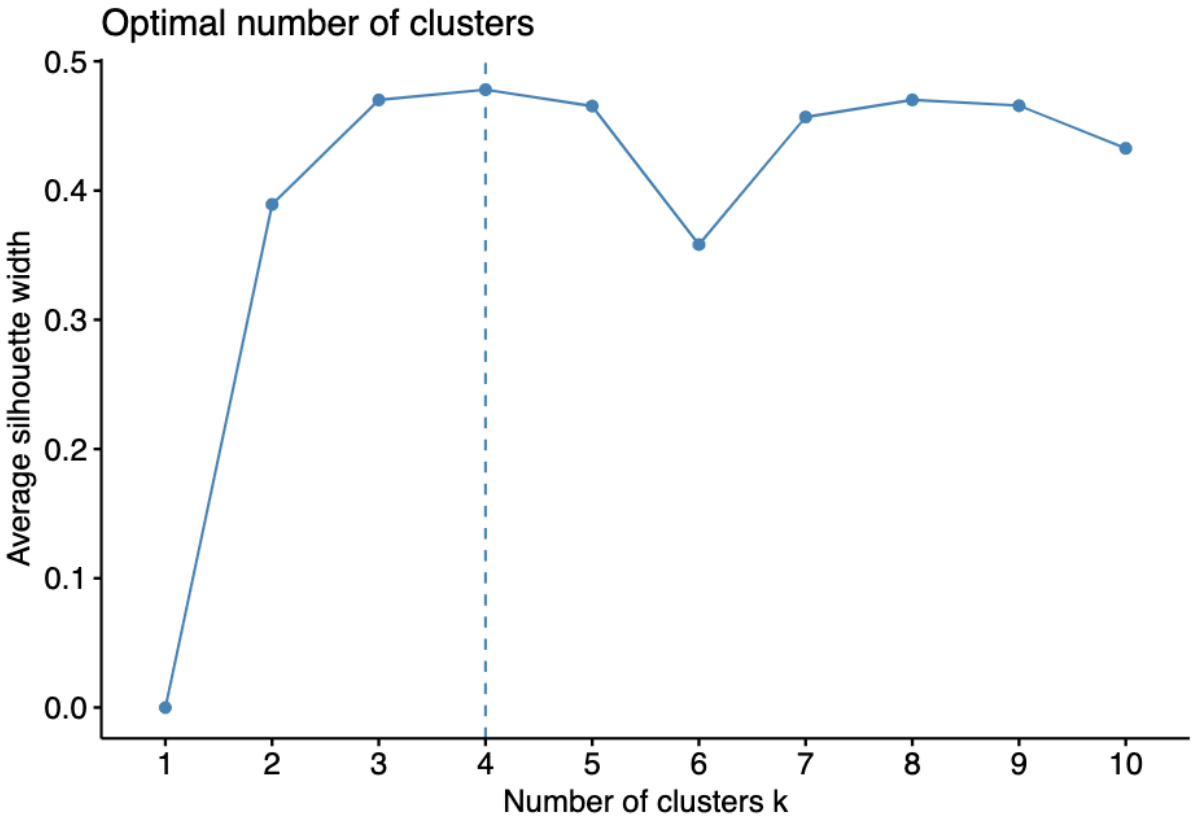
```r
library(caret)

# Remove class labels
predictors <- obesity.pc %>% select(-c(ObesityClass))
# Clustering doesn't work with ordinals - Must convert to integer instead to avoid dummies

set.seed(123)

# Center scale allows us to standardize the data
preproc <- preProcess(predictors, method=c("center", "scale"))
# We have to call predict to fit our data based on preprocessing
predictors <- predict(preproc, predictors)
head(predictors)
```

```
##         PC1        PC2
## 1 1.0103717 -0.2840510
## 2 1.8475720  0.1664402
## 3 0.4804306  1.5216100
## 4 1.2166690  2.2298801
## 5 0.8819436  0.3632974
## 6 0.4870301  1.0207014
```

```r
# Find the knee
fviz_nbclust(predictors, kmeans, method = "wss")
```




```
fviz_nbclust(predictors, kmeans, method = "silhouette")
```




```
#choose 7 because it's when the curve starts to go flat in knee
#in silhouette 7 is the second best option

# Fit the data
fit <- kmeans(predictors, centers = 7, nstart = 25)
# Display the kmeans object information
fit
```

```
## K-means clustering with 7 clusters of sizes 373, 138, 658, 249, 140, 314, 215
##
## Cluster means:
##          PC1        PC2
## 1 -0.6916816 -1.4010153
## 2  2.3185776 -0.5169282
## 3 -0.5370108  0.2065930
## 4 -0.8804720  1.2032052
## 5  0.4899235  1.8881721
## 6  0.7028203 -0.6893186
## 7  1.0295315  0.5138626
##
## Clustering vector:
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
##    6    2    5    5    7    7    6    7    3    3    3    2    7    5    3    2
##   17   18   19   20   21   22   23   24   25   26   27   28   29   30   31   32
##    3    7    5    6    5    4    3    1    3    5    5    7    7    5    5    6
##   33   34   35   36   37   38   39   40   41   42   43   44   45   46   47   48
```

```
##   7   7   7   6   2   6   3   3   1   5   6   5   5   6   7   7
##  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64
##   7   6   6   6   6   2   2   7   4   7   7   5   6   7   7   4
##  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
##   6   6   5   5   5   7   6   2   2   5   7   2   2   6   6   7
##  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96
##   2   3   7   2   3   7   7   3   7   3   2   2   5   2   6   7
##  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112
##   6   2   5   6   7   2   6   3   6   5   5   6   3   6   6   7
## 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128
##   6   6   7   7   3   7   5   5   5   3   7   5   7   2   6   7
## 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
##   6   7   7   7   4   4   3   5   5   7   7   5   5   6   7   7
## 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##   2   3   7   3   3   3   7   5   3   7   7   7   7   4   5   5
## 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176
##   7   7   5   4   7   5   1   7   7   7   2   2   3   6   7   5
## 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192
##   6   6   2   6   7   5   5   5   6   7   7   5   2   6   4   5
## 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208
##   4   2   6   6   3   1   4   5   3   7   5   6   3   6   6   3
## 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224
##   3   3   7   5   7   1   2   3   6   6   6   4   7   6   7   4
## 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240
##   4   6   7   7   6   7   7   6   6   2   6   5   7   4   2   7
## 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256
##   6   7   5   4   6   5   5   6   6   5   3   7   7   6   7   2
## 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272
##   7   7   5   7   2   5   7   4   3   2   7   5   2   6   7   5
## 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288
##   3   7   7   7   6   5   3   3   6   6   6   5   7   5   7   7
## 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304
##   4   2   6   5   7   6   7   7   5   2   3   4   6   6   7   5
## 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320
##   7   6   7   7   7   5   7   6   6   7   5   2   4   1   6   3
## 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336
##   7   5   1   2   7   5   2   2   7   7   5   2   2   7   7   3
## 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352
##   7   3   7   2   4   5   4   5   7   7   3   6   6   5   7   5
## 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368
##   6   7   6   7   7   3   7   4   2   3   7   7   7   7   6   5
## 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384
##   4   7   7   7   3   5   7   2   4   2   2   2   6   3   6   7
## 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400
##   7   7   7   6   6   7   7   3   7   7   7   5   3   5   7   5
## 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416
##   6   3   6   7   2   5   7   5   5   5   3   6   3   5   5   5
## 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432
##   4   5   6   7   2   7   5   6   2   7   6   5   3   7   7   6
## 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448
##   2   7   6   4   3   3   6   7   3   1   6   6   7   5   6   6
## 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464
##   7   2   6   7   7   3   4   7   5   6   5   2   6   2   6   2
## 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480
```

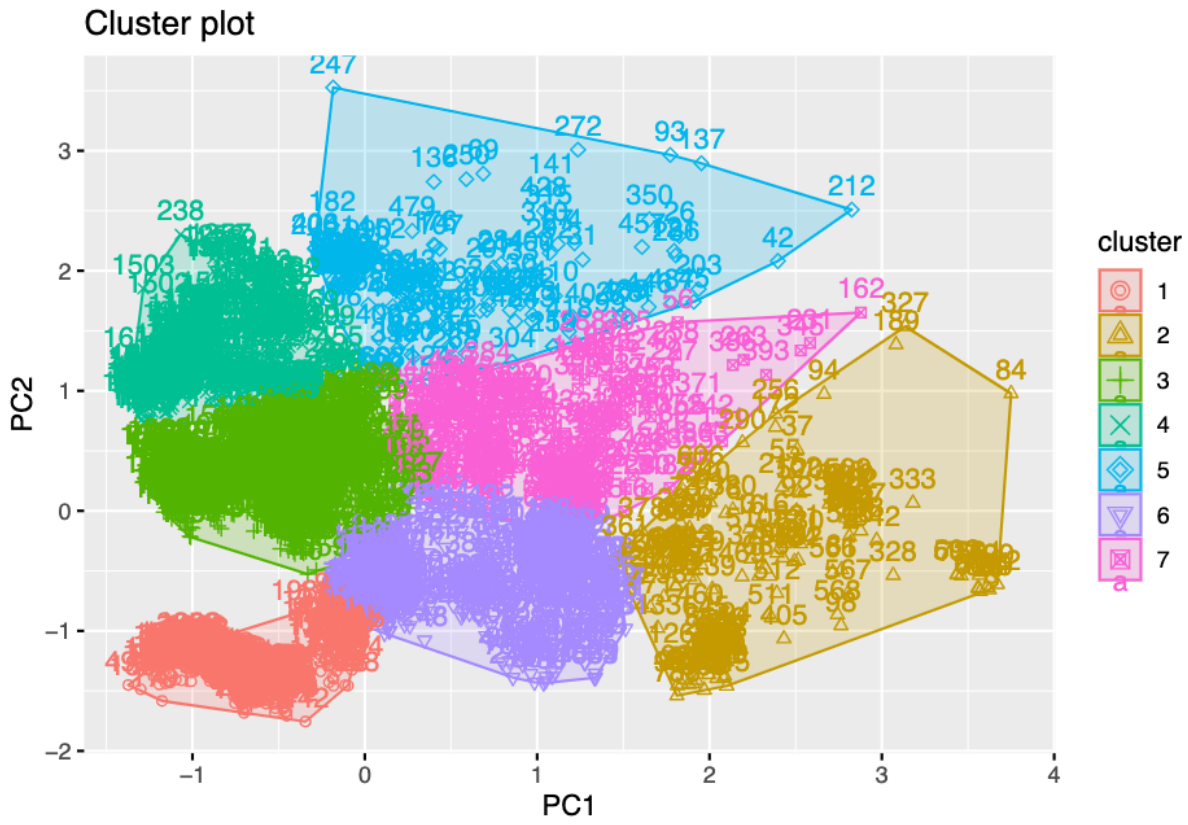
```
##   6   7   2   6   2   7   2   6   7   6   6   3   5   7   5   4
##  481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496
##   6   7   7   5   5   6   5   7   1   1   1   1   1   1   1   1
##  497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512
##   1   3   3   3   7   7   7   2   2   2   2   2   2   2   2   2
##  513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528
##   2   2   2   2   2   2   2   2   4   4   4   7   7   7   6   6
##  529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544
##   6   2   2   2   7   7   7   6   6   6   7   7   6   7   7   7
##  545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560
##   7   7   7   5   5   5   6   6   6   5   5   5   6   7   7   6
##  561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576
##   6   6   2   2   2   2   2   2   5   5   5   7   6   7   6   6
##  577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592
##   6   6   6   6   5   5   5   5   5   5   3   3   3   2   2   2
##  593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608
##   2   2   2   3   3   3   5   6   2   2   2   2   2   2   4   6
##  609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624
##   6   2   7   2   7   7   7   5   6   5   7   6   2   2   5   7
##  625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640
##   6   6   5   5   3   2   2   3   3   5   7   7   7   6   2   2
##  641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656
##   2   2   2   2   2   6   6   2   2   2   2   6   6   6   4   4
##  657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672
##   5   3   7   3   6   6   6   6   2   2   7   5   6   6   6   6
##  673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688
##   7   7   6   7   6   7   7   3   3   5   5   5   2   6   6   5
##  689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704
##   5   5   6   3   7   6   6   6   6   6   2   2   2   2   5   5
##  705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720
##   5   7   6   7   6   6   6   6   6   2   5   5   5   5   5   5
##  721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736
##   3   3   3   2   2   2   2   2   2   3   3   3   5   7   5   3
##  737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752
##   7   3   3   3   3   3   3   7   6   7   6   3   1   7   3   4
##  753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768
##   3   3   3   7   6   3   6   1   4   3   3   3   3   3   6   3
##  769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784
##   3   2   6   6   7   4   3   6   6   6   2   6   5   3   3   3
##  785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800
##   3   6   6   2   5   5   7   7   5   3   6   6   3   3   3   3
##  801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816
##   3   3   3   3   3   3   3   6   6   3   3   3   1   1   3   3
##  817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832
##   3   3   3   3   3   3   3   6   6   1   1   3   3   3   6   6
##  833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848
##   3   3   3   3   3   3   3   3   6   3   3   3   2   2   6   6
##  849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864
##   6   7   4   4   3   6   6   6   6   6   2   2   6   5   3   3
##  865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880
##   3   3   3   3   3   6   1   2   5   5   7   7   4   3   3   6
##  881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896
##   3   3   3   3   3   3   3   3   3   3   6   6   7   7   3   3
##  897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912
```

```
##    3    1    6    7    3    3    1    3    3    3    6    6    3    3    3    6
##  913  914  915  916  917  918  919  920  921  922  923  924  925  926  927  928
##    6    1    1    4    3    3    3    3    3    3    3    6    3    3    3    2
##  929  930  931  932  933  934  935  936  937  938  939  940  941  942  943  944
##    2    6    6    3    4    3    7    6    6    3    2    2    6    6    5    3
##  945  946  947  948  949  950  951  952  953  954  955  956  957  958  959  960
##    3    3    3    3    6    6    6    6    2    3    3    3    3    4    4    4
##  961  962  963  964  965  966  967  968  969  970  971  972  973  974  975  976
##    3    3    6    4    4    6    6    3    6    6    3    3    6    6    3    3
##  977  978  979  980  981  982  983  984  985  986  987  988  989  990  991  992
##    1    3    3    3    7    1    3    3    3    3    3    4    6    5    4    3
##  993  994  995  996  997  998  999 1000 1001 1002 1003 1004 1005 1006 1007 1008
##    3    3    3    6    6    6    6    3    3    3    3    3    3    4    4    6
## 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024
##    6    7    4    3    3    3    3    3    4    4    3    6    6    4    3    7
## 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1040
##    7    3    3    3    4    3    3    3    3    3    6    3    3    3    3    3
## 1041 1042 1043 1044 1045 1046 1047 1048 1049 1050 1051 1052 1053 1054 1055 1056
##    3    3    3    4    3    6    4    6    3    6    3    6    3    6    3    3
## 1057 1058 1059 1060 1061 1062 1063 1064 1065 1066 1067 1068 1069 1070 1071 1072
##    3    7    6    3    3    3    4    3    4    4    3    3    3    6    6    3
## 1073 1074 1075 1076 1077 1078 1079 1080 1081 1082 1083 1084 1085 1086 1087 1088
##    3    3    4    6    7    4    3    7    3    3    4    3    6    4    3    3
## 1089 1090 1091 1092 1093 1094 1095 1096 1097 1098 1099 1100 1101 1102 1103 1104
##    3    3    4    3    3    3    6    3    3    3    3    3    3    3    4    4
## 1105 1106 1107 1108 1109 1110 1111 1112 1113 1114 1115 1116 1117 1118 1119 1120
##    4    3    6    6    4    4    6    6    3    6    6    3    3    6    3    3
## 1121 1122 1123 1124 1125 1126 1127 1128 1129 1130 1131 1132 1133 1134 1135 1136
##    3    3    3    1    3    3    3    3    3    3    3    1    4    3    4    4
## 1137 1138 1139 1140 1141 1142 1143 1144 1145 1146 1147 1148 1149 1150 1151 1152
##    3    7    3    3    6    6    6    6    3    3    3    3    3    3    4    4
## 1153 1154 1155 1156 1157 1158 1159 1160 1161 1162 1163 1164 1165 1166 1167 1168
##    6    6    7    4    3    3    3    3    3    4    4    4    6    6    4    3
## 1169 1170 1171 1172 1173 1174 1175 1176 1177 1178 1179 1180 1181 1182 1183 1184
##    7    7    3    3    3    4    3    3    4    3    3    6    3    3    3    4
## 1185 1186 1187 1188 1189 1190 1191 1192 1193 1194 1195 1196 1197 1198 1199 1200
##    3    3    3    4    4    1    1    3    3    1    1    3    3    6    6    3
## 1201 1202 1203 1204 1205 1206 1207 1208 1209 1210 1211 1212 1213 1214 1215 1216
##    4    4    6    1    4    3    3    3    4    3    4    4    3    4    6    6
## 1217 1218 1219 1220 1221 1222 1223 1224 1225 1226 1227 1228 1229 1230 1231 1232
##    6    3    3    4    6    3    4    6    3    3    1    1    3    3    3    3
## 1233 1234 1235 1236 1237 1238 1239 1240 1241 1242 1243 1244 1245 1246 1247 1248
##    3    3    1    1    3    3    3    3    3    3    3    3    3    3    4    6
## 1249 1250 1251 1252 1253 1254 1255 1256 1257 1258 1259 1260 1261 1262 1263 1264
##    4    4    6    3    3    6    4    3    3    4    4    6    6    3    3    1
## 1265 1266 1267 1268 1269 1270 1271 1272 1273 1274 1275 1276 1277 1278 1279 1280
##    1    3    3    3    3    6    6    3    3    4    4    6    6    6    6    4
## 1281 1282 1283 1284 1285 1286 1287 1288 1289 1290 1291 1292 1293 1294 1295 1296
##    4    3    3    3    3    3    3    4    4    3    3    4    4    4    4    4
## 1297 1298 1299 1300 1301 1302 1303 1304 1305 1306 1307 1308 1309 1310 1311 1312
##    4    6    6    1    1    3    3    4    4    6    6    3    3    6    6    3
## 1313 1314 1315 1316 1317 1318 1319 1320 1321 1322 1323 1324 1325 1326 1327 1328
##    3    1    1    3    3    3    3    3    3    6    6    3    3    3    3    3
## 1329 1330 1331 1332 1333 1334 1335 1336 1337 1338 1339 1340 1341 1342 1343 1344
```

```
##    3    3    3    3    3    3    3    4    4    1    1    4    4    6    6    3
## 1345 1346 1347 1348 1349 1350 1351 1352 1353 1354 1355 1356 1357 1358 1359 1360
##    3    6    6    4    4    3    3    3    3    4    4    4    4    3    3    6
## 1361 1362 1363 1364 1365 1366 1367 1368 1369 1370 1371 1372 1373 1374 1375 1376
##    6    3    3    3    3    1    1    1    1    3    3    6    6    6    6    6
## 1377 1378 1379 1380 1381 1382 1383 1384 1385 1386 1387 1388 1389 1390 1391 1392
##    6    3    3    3    3    4    4    6    6    6    6    3    3    3    3    4
## 1393 1394 1395 1396 1397 1398 1399 1400 1401 1402 1403 1404 1405 1406 1407 1408
##    4    3    3    4    4    3    3    4    4    4    4    3    3    4    4    6
## 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1423 1424
##    6    6    6    1    1    3    3    3    3    4    4    6    6    3    3    4
## 1425 1426 1427 1428 1429 1430 1431 1432 1433 1434 1435 1436 1437 1438 1439 1440
##    4    6    6    3    3    3    3    1    1    1    1    3    3    3    3    3
## 1441 1442 1443 1444 1445 1446 1447 1448 1449 1450 1451 1452 1453 1454 1455 1456
##    3    3    3    3    3    3    3    1    1    6    6    3    3    3    3    3
## 1457 1458 1459 1460 1461 1462 1463 1464 1465 1466 1467 1468 1469 1470 1471 1472
##    3    3    3    3    3    3    3    3    3    3    3    3    3    3    3    4
## 1473 1474 1475 1476 1477 1478 1479 1480 1481 1482 1483 1484 1485 1486 1487 1488
##    4    6    6    4    4    4    4    6    6    3    3    3    3    6    6    4
## 1489 1490 1491 1492 1493 1494 1495 1496 1497 1498 1499 1500 1501 1502 1503 1504
##    3    4    4    3    3    3    3    4    4    3    3    3    3    4    4    3
## 1505 1506 1507 1508 1509 1510 1511 1512 1513 1514 1515 1516 1517 1518 1519 1520
##    3    4    3    3    3    3    3    4    4    3    3    3    3    4    4    3
## 1521 1522 1523 1524 1525 1526 1527 1528 1529 1530 1531 1532 1533 1534 1535 1536
##    3    3    3    4    4    3    3    4    4    3    3    3    3    4    4    3
## 1537 1538 1539 1540 1541 1542 1543 1544 1545 1546 1547 1548 1549 1550 1551 1552
##    3    4    4    3    3    3    3    4    4    3    3    4    4    3    3    4
## 1553 1554 1555 1556 1557 1558 1559 1560 1561 1562 1563 1564 1565 1566 1567 1568
##    4    3    3    3    3    4    4    4    4    3    3    3    3    4    4    3
## 1569 1570 1571 1572 1573 1574 1575 1576 1577 1578 1579 1580 1581 1582 1583 1584
##    3    3    3    3    3    3    3    3    3    4    4    3    3    3    3    4
## 1585 1586 1587 1588 1589 1590 1591 1592 1593 1594 1595 1596 1597 1598 1599 1600
##    4    3    3    3    3    4    4    3    3    4    4    3    3    3    3    4
## 1601 1602 1603 1604 1605 1606 1607 1608 1609 1610 1611 1612 1613 1614 1615 1616
##    4    3    3    4    4    3    3    3    3    4    4    3    3    4    4    3
## 1617 1618 1619 1620 1621 1622 1623 1624 1625 1626 1627 1628 1629 1630 1631 1632
##    3    4    4    3    3    3    3    4    4    4    4    3    3    3    3    4
## 1633 1634 1635 1636 1637 1638 1639 1640 1641 1642 1643 1644 1645 1646 1647 1648
##    4    3    3    3    3    3    3    3    3    3    3    4    4    3    3    3
## 1649 1650 1651 1652 1653 1654 1655 1656 1657 1658 1659 1660 1661 1662 1663 1664
##    3    3    3    4    4    3    3    4    4    4    4    3    3    3    3    3
## 1665 1666 1667 1668 1669 1670 1671 1672 1673 1674 1675 1676 1677 1678 1679 1680
##    3    3    3    4    4    4    4    3    3    3    3    4    4    3    3    3
## 1681 1682 1683 1684 1685 1686 1687 1688 1689 1690 1691 1692 1693 1694 1695 1696
##    3    3    3    3    3    3    3    4    4    4    4    3    3    3    3    4
## 1697 1698 1699 1700 1701 1702 1703 1704 1705 1706 1707 1708 1709 1710 1711 1712
##    4    4    4    3    3    3    3    4    4    3    3    4    4    4    4    3
## 1713 1714 1715 1716 1717 1718 1719 1720 1721 1722 1723 1724 1725 1726 1727 1728
##    3    3    3    4    4    4    4    3    3    3    3    4    4    4    4    3
## 1729 1730 1731 1732 1733 1734 1735 1736 1737 1738 1739 1740 1741 1742 1743 1744
##    3    3    3    3    3    3    3    4    4    4    4    4    4    3    3    3
## 1745 1746 1747 1748 1749 1750 1751 1752 1753 1754 1755 1756 1757 1758 1759 1760
##    3    3    3    3    3    3    3    4    4    4    4    3    3    3    3    3
## 1761 1762 1763 1764 1765 1766 1767 1768 1769 1770 1771 1772 1773 1774 1775 1776
```

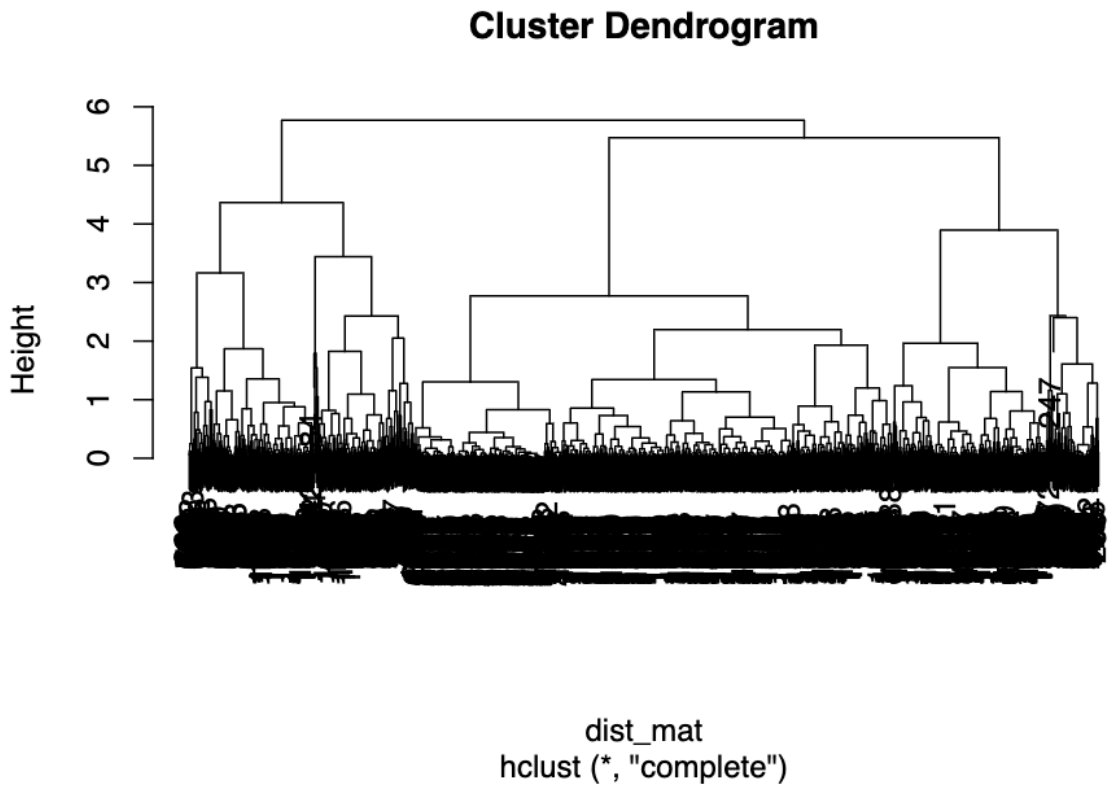
```
##    3    3    3    3    3    3    3    3    3    3    3    3    3    3    3    1
## 1777 1778 1779 1780 1781 1782 1783 1784 1785 1786 1787 1788 1789 1790 1791 1792
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1793 1794 1795 1796 1797 1798 1799 1800 1801 1802 1803 1804 1805 1806 1807 1808
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1809 1810 1811 1812 1813 1814 1815 1816 1817 1818 1819 1820 1821 1822 1823 1824
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1841 1842 1843 1844 1845 1846 1847 1848 1849 1850 1851 1852 1853 1854 1855 1856
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1857 1858 1859 1860 1861 1862 1863 1864 1865 1866 1867 1868 1869 1870 1871 1872
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1873 1874 1875 1876 1877 1878 1879 1880 1881 1882 1883 1884 1885 1886 1887 1888
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1889 1890 1891 1892 1893 1894 1895 1896 1897 1898 1899 1900 1901 1902 1903 1904
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1905 1906 1907 1908 1909 1910 1911 1912 1913 1914 1915 1916 1917 1918 1919 1920
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1921 1922 1923 1924 1925 1926 1927 1928 1929 1930 1931 1932 1933 1934 1935 1936
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1937 1938 1939 1940 1941 1942 1943 1944 1945 1946 1947 1948 1949 1950 1951 1952
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1953 1954 1955 1956 1957 1958 1959 1960 1961 1962 1963 1964 1965 1966 1967 1968
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1969 1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 2017 2018 2019 2020 2021 2022 2023 2024 2025 2026 2027 2028 2029 2030 2031 2032
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 2033 2034 2035 2036 2037 2038 2039 2040 2041 2042 2043 2044 2045 2046 2047 2048
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 2049 2050 2051 2052 2053 2054 2055 2056 2057 2058 2059 2060 2061 2062 2063 2064
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 2065 2066 2067 2068 2069 2070 2071 2072 2073 2074 2075 2076 2077 2078 2079 2080
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 2081 2082 2083 2084 2085 2086 2087
##    1    1    1    1    1    1    1
##
## Within cluster sum of squares by cluster:
## [1]  39.51117  95.24879 124.87623  54.11424  82.49535 104.26309  89.12230
##  (between_SS / total_SS =  85.9 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
# Display the cluster plot
fviz_cluster(fit, data = predictors)
```

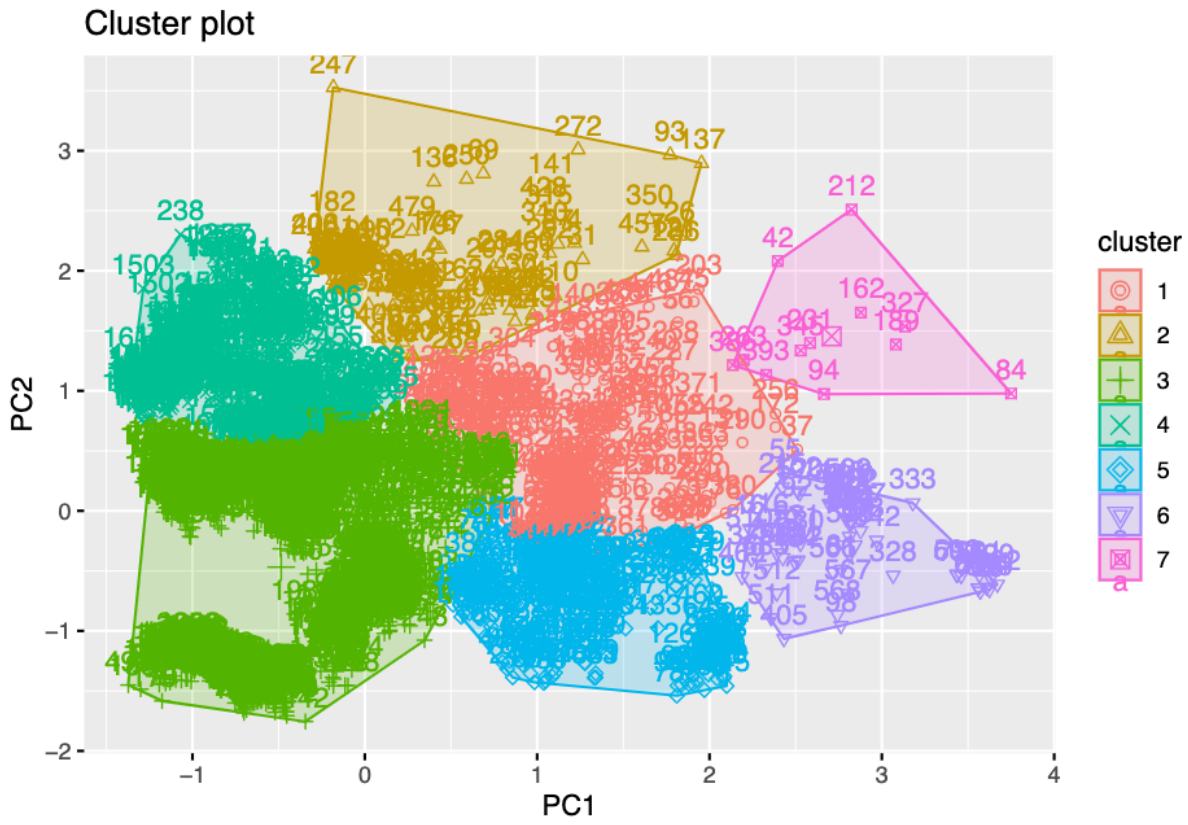


#HAC

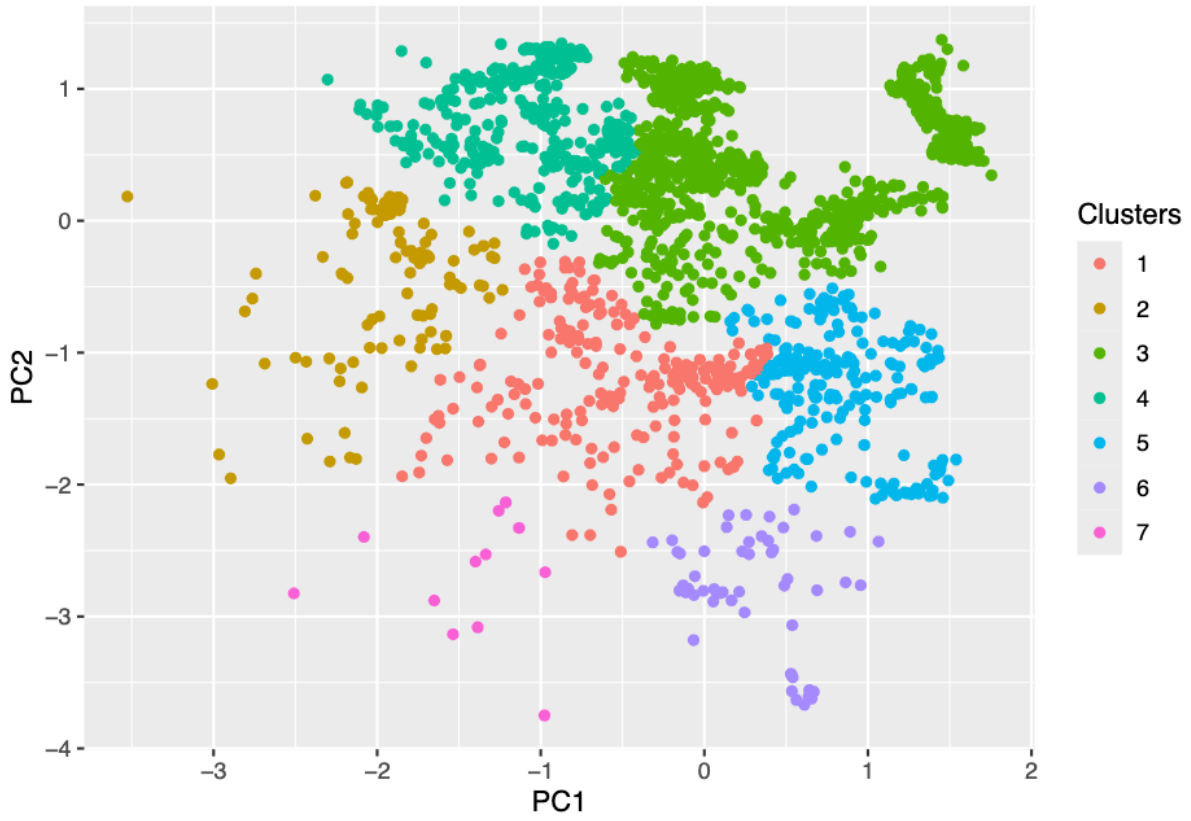
```
dist_mat <- dist(predictors, method = 'euclidean')
# Determine assembly/agglomeration method and run hclust (average uses mean)
hfit <- hclust(dist_mat, method = 'complete')
plot(hfit)
```




```
# Build the new model
h3 <- cutree(hfit, k=7)
# Visualize 2-cluster HAC
fviz_cluster(list(data = predictors, cluster = h3))
```




```
pca = prcomp(predictors)
# Save as dataframe
rotated_data = as.data.frame(pca$x)
# Add original labels as a reference
rotated_data$Color <- obesity$ObesityClass
# Assign clusters as a new column
rotated_data$Clusters = as.factor(h3)
# Plot and color by labels
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = Clusters)) +
  geom_point()
```




```r
# Create a dataframe
result <- data.frame(ObesityClass = obesity$ObesityClass, HAC3 = h3, Kmeans = fit$cluster)
# View the first 100 cases one by one
head(result, n = 100)
```

```
##            ObesityClass HAC3 Kmeans
## 1         Normal_Weight    1      6
## 2         Normal_Weight    1      2
## 3         Normal_Weight    2      5
## 4    Overweight_Level_I    2      5
## 5   Overweight_Level_II    1      7
## 6         Normal_Weight    1      7
## 7         Normal_Weight    3      6
## 8         Normal_Weight    1      7
## 9         Normal_Weight    4      3
## 10        Normal_Weight    4      3
## 11       Obesity_Type_I    3      3
## 12  Overweight_Level_II    5      2
## 13        Normal_Weight    1      7
## 14       Obesity_Type_I    2      5
## 15        Normal_Weight    3      3
## 16        Normal_Weight    6      2
## 17  Overweight_Level_II    4      3
## 18       Obesity_Type_I    3      7
## 19  Overweight_Level_II    2      5
## 20   Overweight_Level_I    5      6
```

```
## 21  Overweight_Level_II    2      5
## 22       Obesity_Type_I    4      4
## 23        Normal_Weight    3      3
## 24       Obesity_Type_I    3      1
## 25        Normal_Weight    3      3
## 26        Normal_Weight    2      5
## 27        Normal_Weight    2      5
## 28        Normal_Weight    1      7
## 29        Normal_Weight    3      7
## 30        Normal_Weight    2      5
## 31   Overweight_Level_I    2      5
## 32  Overweight_Level_II    5      6
## 33        Normal_Weight    1      7
## 34  Overweight_Level_II    1      7
## 35        Normal_Weight    1      7
## 36  Overweight_Level_II    5      6
## 37        Normal_Weight    1      2
## 38        Normal_Weight    5      6
## 39        Normal_Weight    3      3
## 40  Overweight_Level_II    3      3
## 41   Overweight_Level_I    3      1
## 42        Normal_Weight    7      5
## 43        Normal_Weight    3      6
## 44        Normal_Weight    2      5
## 45        Normal_Weight    1      5
## 46  Overweight_Level_II    5      6
## 47        Normal_Weight    1      7
## 48        Normal_Weight    1      7
## 49        Normal_Weight    1      7
## 50        Normal_Weight    5      6
## 51        Normal_Weight    3      6
## 52        Normal_Weight    1      6
## 53        Normal_Weight    5      6
## 54        Normal_Weight    6      2
## 55        Normal_Weight    6      2
## 56        Normal_Weight    1      7
## 57        Normal_Weight    4      4
## 58        Normal_Weight    1      7
## 59        Normal_Weight    3      7
## 60  Insufficient_Weight    4      5
## 61        Normal_Weight    3      6
## 62        Normal_Weight    1      7
## 63        Normal_Weight    1      7
## 64        Normal_Weight    4      4
## 65        Normal_Weight    1      6
## 66   Overweight_Level_I    1      6
## 67  Overweight_Level_II    2      5
## 68       Obesity_Type_I    2      5
## 69      Obesity_Type_II    2      5
## 70        Normal_Weight    1      7
## 71  Overweight_Level_II    5      6
## 72  Insufficient_Weight    6      2
## 73        Normal_Weight    1      2
## 74        Normal_Weight    2      5
```

```
## 75  Overweight_Level_II    1      7
## 76  Insufficient_Weight    1      2
## 77  Insufficient_Weight    1      2
## 78  Overweight_Level_II    3      6
## 79       Obesity_Type_I    3      6
## 80        Normal_Weight    3      7
## 81        Normal_Weight    6      2
## 82  Overweight_Level_II    3      3
## 83       Obesity_Type_I    1      7
## 84  Insufficient_Weight    7      2
## 85  Overweight_Level_II    3      3
## 86        Normal_Weight    1      7
## 87        Normal_Weight    1      7
## 88   Overweight_Level_I    4      3
## 89        Normal_Weight    1      7
## 90  Overweight_Level_II    4      3
## 91      Obesity_Type_II    1      2
## 92        Normal_Weight    6      2
## 93   Overweight_Level_I    2      5
## 94        Normal_Weight    7      2
## 95        Normal_Weight    5      6
## 96        Normal_Weight    1      7
## 97        Normal_Weight    5      6
## 98  Insufficient_Weight    6      2
## 99        Normal_Weight    1      5
## 100       Normal_Weight    5      6
```

```
# Crosstab for HAC
result %>% group_by(HAC3) %>% select(HAC3, ObesityClass) %>% table()
```

```
##     ObesityClass
## HAC3 Insufficient_Weight Normal_Weight Obesity_Type_I Obesity_Type_II
##    1                 54           107              3               1
##    2                 43            33             10               2
##    3                 32            50            231             174
##    4                 12            11            103             120
##    5                 83            62              4               0
##    6                 42             9              0               0
##    7                  1            10              0               0
##     ObesityClass
## HAC3 Obesity_Type_III Overweight_Level_I Overweight_Level_II
##    1                0                28                 27
##    2                0                18                 14
##    3              324               149                137
##    4                0                29                 76
##    5                0                48                 35
##    6                0                 3                  1
##    7                0                 1                  0
```

```
# Crosstab for K Means
result %>% group_by(Kmeans) %>% select(Kmeans, ObesityClass) %>% table()
```

```
##       ObesityClass
```

```
## Kmeans Insufficient_Weight Normal_Weight Obesity_Type_I Obesity_Type_II
##      1                  0             0             31              1
##      2                 81            34              0              1
##      3                 24            23            165            187
##      4                  6             6             70            106
##      5                 47            43             10              2
##      6                 60            71             70              0
##      7                 49           105              5              0
##       ObesityClass
## Kmeans Obesity_Type_III Overweight_Level_I Overweight_Level_II
##      1              322                 15                  4
##      2                0                 20                  2
##      3                2                117                140
##      4                0                 12                 49
##      5                0                 23                 15
##      6                0                 61                 52
##      7                0                 28                 28
```

#Analysis of HAC Clustering:

Cluster 1: Predominantly contains "Normal_Weight" (107), but also has a significant number of "Insufficient_Weight" (54). Cluster 2: Mix of "Insufficient_Weight" (43) and "Normal_Weight" (33) with some "Obesity_Type_I" (10). Cluster 3: Large cluster with a mix of "Obesity_Type_I" (231), "Obesity_Type_II" (174), and "Obesity_Type_III" (324), and significant "Overweight_Level_I" (149) and "Overweight_Level_II" (137). Cluster 4: Contains "Obesity_Type_I" (103), "Obesity_Type_II" (120), and some "Overweight_Level_I" (29) and "Overweight_Level_II" (76). Cluster 5: Mainly "Insufficient_Weight" (83) and "Normal_Weight" (62). Cluster 6: Primarily "Insufficient_Weight" (42) with very few instances of other classes. Cluster 7: Very small cluster, mainly "Normal_Weight" (10).

#Analysis of K-Means Clustering:' Cluster 1: Predominantly "Normal_Weight" (75) and "Insufficient_Weight" (74), with significant "Overweight_Level_I" (61) and "Overweight_Level_II" (54). Cluster 2: Mainly "Normal_Weight" (113) with some "Insufficient_Weight" (39). Cluster 3: Mostly "Insufficient_Weight" (78) with some "Normal_Weight" (28). Cluster 4: Contains "Obesity_Type_I" (70), "Obesity_Type_II" (106), and some "Overweight_Level_I" (12) and "Overweight_Level_II" (49). Cluster 5: Mix of "Insufficient_Weight" (46) and "Normal_Weight" (36). Cluster 6: Primarily "Obesity_Type_III" (322). Cluster 7: Large cluster with "Obesity_Type_I" (167), "Obesity_Type_II" (187), "Overweight_Level_I" (121), and "Overweight_Level_II" (144).

##Comparison: #Cluster Size and Distribution:

HAC clusters tend to have a more mixed composition with a clear tendency to cluster based on combined weight classes. K-Means clusters show a clearer separation of distinct obesity levels, especially for higher obesity levels in Cluster 6 and Cluster 7. Cluster Homogeneity:

K-Means appears to provide more homogeneous clusters with clearer separation of "Obesity_Type_III" and higher obesity levels. HAC provides clusters that are somewhat more mixed but still capture the structure of the data.

#Conclusion: K-Means: Better for clear separation of distinct obesity levels, particularly in higher dimensions. It handles high-dimensional data reasonably well but can still produce mixed clusters.

f. Classification Use at least two classifiers to predict a label in your data. If a label was not provided with the data, use the clustering from the previous part. Follow the process for choosing the best parameters for your choice of classifier. Compare the accuracy of the two.

#KNN

```r
# Make sure that we have the PCs as predictors
head(obesity.pc)
```

```
##        PC1        PC2        ObesityClass
## 1 2.181439 -0.4858502       Normal_Weight
## 2 3.988994  0.2846848       Normal_Weight
## 3 1.037272  2.6026115       Normal_Weight
## 4 2.626845  3.8140599  Overweight_Level_I
## 5 1.904157  0.6213957 Overweight_Level_II
## 6 1.051521  1.7458411       Normal_Weight
```

```r
set.seed(123)
library(kknn)
```

```
##
## Attaching package: 'kknn'
```

```
## The following object is masked from 'package:caret':
##
##     contr.dummy
```

```r
obesity_dummies <- dummies
obesity_dummies$ObesityClass = obesity$ObesityClass

ctrl <- trainControl(method="cv", number = 10)
# setup a tuneGrid with the tuning parameters
tuneGrid <- expand.grid(kmax = 3:7,
                        kernel = c("rectangular", "cos"),
                        distance = 1:3)

# tune and fit the model with 10-fold cross validation,
# standardization, and our specialized tune grid
kknn_fit <- train(ObesityClass ~ .,
                  data = obesity_dummies,
                  method = 'kknn',
                  trControl = ctrl,
                  preProcess = c('center', 'scale'),
                  tuneGrid = tuneGrid)
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: Alcohol.Always
```

```
# Printing trained model provides report
kknn_fit
```

```
## k-Nearest Neighbors
##
## 2087 samples
##   32 predictor
##    7 classes: 'Insufficient_Weight', 'Normal_Weight', 'Obesity_Type_I', 'Obesity_Type_II', 'Obesity_
##
## Pre-processing: centered (32), scaled (32)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1878, 1877, 1880, 1880, 1878, 1878, ...
## Resampling results across tuning parameters:
##
##   kmax  kernel       distance  Accuracy   Kappa
##   3     rectangular  1         0.8768987  0.8561729
##   3     rectangular  2         0.8510519  0.8259867
##   3     rectangular  3         0.8352253  0.8074877
##   3     cos          1         0.8831027  0.8634170
##   3     cos          2         0.8505504  0.8253984
##   3     cos          3         0.8352300  0.8074855
##   4     rectangular  1         0.8768987  0.8561729
##   4     rectangular  2         0.8510519  0.8259867
##   4     rectangular  3         0.8352253  0.8074877
##   4     cos          1         0.8826242  0.8628575
##   4     cos          2         0.8500719  0.8248395
##   4     cos          3         0.8366608  0.8091572
##   5     rectangular  1         0.8768987  0.8561729
##   5     rectangular  2         0.8510519  0.8259867
##   5     rectangular  3         0.8352253  0.8074877
```

```
##   5     cos          1         0.8811933  0.8611787
##   5     cos          2         0.8495934  0.8242744
##   5     cos          3         0.8366608  0.8091572
##   6     rectangular  1         0.8768987  0.8561729
##   6     rectangular  2         0.8510519  0.8259867
##   6     rectangular  3         0.8352253  0.8074877
##   6     cos          1         0.8850372  0.8656647
##   6     cos          2         0.8495934  0.8242744
##   6     cos          3         0.8376178  0.8102785
##   7     rectangular  1         0.8768987  0.8561729
##   7     rectangular  2         0.8510519  0.8259867
##   7     rectangular  3         0.8352253  0.8074877
##   7     cos          1         0.8860034  0.8667903
##   7     cos          2         0.8495934  0.8242744
##   7     cos          3         0.8376178  0.8102785
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were kmax = 7, distance = 1 and kernel
##  = cos.
```

#Decision tree

```
library(caret)
library(rpart)
library(tidyverse)
library(ggplot2)

set.seed(94)

# Partition the data
index = createDataPartition(y=obesity_bin$ObesityClass, p=0.7, list=FALSE)
# Everything in the generated index list
train_set = obesity_bin[index,]
# Everything except the generated indices
test_set = obesity_bin[-index,]

# Initialize cross validation
train_control = trainControl(method = "cv", number = 10)

# Tree 1
hypers = rpart.control(minsplit =  2, maxdepth = 1, minbucket = 2)
tree1 <- train(ObesityClass ~., data = train_set, control = hypers,
               trControl = train_control, method = "rpart1SE")

# Training Set
# Evaluate the fit with a confusion matrix
pred_tree <- predict(tree1, train_set)
# Confusion Matrix
cfm_train <- confusionMatrix(train_set$ObesityClass, pred_tree)

# Test Set
# Evaluate the fit with a confusion matrix
pred_tree <- predict(tree1, test_set)
# Confusion Matrix
```

```r
cfm_test <- confusionMatrix(test_set$ObesityClass, pred_tree)

# Get training accuracy
a_train <- cfm_train$overall[1]
# Get testing accuracy
a_test <- cfm_test$overall[1]
# Get number of nodes
nodes <- nrow(tree1$finalModel$frame)

# Form the table
comp_tbl <- data.frame("Nodes" = nodes, "TrainAccuracy" = a_train,
                       "TestAccuracy" = a_test,
                       "MaxDepth" = 1, "Minsplit" = 2, "Minbucket" = 2)
# Tree 2
hypers = rpart.control(minsplit =  5, maxdepth = 2, minbucket = 5)
tree2 <- train(ObesityClass ~., data = train_set, control = hypers,
               trControl = train_control, method = "rpart1SE")

# Training Set
# Evaluate the fit with a confusion matrix
pred_tree <- predict(tree2, train_set)
# Confusion Matrix
cfm_train <- confusionMatrix(train_set$ObesityClass, pred_tree)

# Test Set
# Evaluate the fit with a confusion matrix
pred_tree <- predict(tree2, test_set)
# Confusion Matrix
cfm_test <- confusionMatrix(test_set$ObesityClass, pred_tree)

# Get training accuracy
a_train <- cfm_train$overall[1]
# Get testing accuracy
a_test <- cfm_test$overall[1]
# Get number of nodes
nodes <- nrow(tree2$finalModel$frame)

# Add rows to the table - Make sure the order is correct
comp_tbl <- comp_tbl %>% rbind(list(nodes, a_train, a_test, 2, 5, 5))


# Tree 3
hypers3 <- rpart.control(minsplit = 50, maxdepth = 10, minbucket = 50)
tree3 <- train(ObesityClass ~ ., data = train_set, control = hypers3,
               trControl = train_control, method = "rpart1SE")

# Evaluate Tree 3
pred_tree3_train <- predict(tree3, train_set)
cfm_train3 <- confusionMatrix(train_set$ObesityClass, pred_tree3_train)
pred_tree3_test <- predict(tree3, test_set)
cfm_test3 <- confusionMatrix(test_set$ObesityClass, pred_tree3_test)

a_train3 <- cfm_train3$overall[1]
```

```r
a_test3 <- cfm_test3$overall[1]
nodes3 <- nrow(tree3$finalModel$frame)
comp_tbl <- comp_tbl %>% rbind(list(nodes3, a_train3, a_test3, 10, 50,
                                    50))

# Tree 4
hypers4 <- rpart.control(minsplit = 100, maxdepth = 15, minbucket = 100)
tree4 <- train(ObesityClass ~ ., data = train_set, control = hypers4,
               trControl = train_control, method = "rpart1SE")

# Evaluate Tree 4
pred_tree4_train <- predict(tree4, train_set)
cfm_train4 <- confusionMatrix(train_set$ObesityClass, pred_tree4_train)
pred_tree4_test <- predict(tree4, test_set)
cfm_test4 <- confusionMatrix(test_set$ObesityClass, pred_tree4_test)

a_train4 <- cfm_train4$overall[1]
a_test4 <- cfm_test4$overall[1]
nodes4 <- nrow(tree4$finalModel$frame)
comp_tbl <- comp_tbl %>% rbind(list(nodes4, a_train4, a_test4, 15, 100, 100))

# Tree 5
hypers5 <- rpart.control(minsplit = 1000, maxdepth = 29, minbucket = 1000)
tree5 <- train(ObesityClass ~ ., data = train_set, control = hypers5,
               trControl = train_control, method = "rpart1SE")

# Evaluate Tree 5
pred_tree5_train <- predict(tree5, train_set)
cfm_train5 <- confusionMatrix(train_set$ObesityClass, pred_tree5_train)
pred_tree5_test <- predict(tree5, test_set)
cfm_test5 <- confusionMatrix(test_set$ObesityClass, pred_tree5_test)

a_train5 <- cfm_train5$overall[1]
a_test5 <- cfm_test5$overall[1]
nodes5 <- nrow(tree5$finalModel$frame)
comp_tbl <- comp_tbl %>% rbind(list(nodes5, a_train5, a_test5, 29, 1000, 1000))

# Tree 6
hypers6 <- rpart.control(minsplit = 5000, maxdepth = 20, minbucket = 5000)
tree6 <- train(ObesityClass ~ ., data = train_set, control = hypers6,
               trControl = train_control, method = "rpart1SE")

# Evaluate Tree 6
pred_tree6_train <- predict(tree6, train_set)
cfm_train6 <- confusionMatrix(train_set$ObesityClass, pred_tree6_train)
pred_tree6_test <- predict(tree6, test_set)
cfm_test6 <- confusionMatrix(test_set$ObesityClass, pred_tree6_test)

a_train6 <- cfm_train6$overall[1]
a_test6 <- cfm_test6$overall[1]
nodes6 <- nrow(tree6$finalModel$frame)
comp_tbl <- comp_tbl %>% rbind(list(nodes6, a_train6, a_test6, 20, 5000, 5000))
```

```r
# Tree 7
hypers7 <- rpart.control(minsplit = 10000, maxdepth = 25, minbucket = 10000)
tree7 <- train(ObesityClass ~ ., data = train_set, control = hypers7,
               trControl = train_control, method = "rpart1SE")

# Evaluate Tree 7
pred_tree7_train <- predict(tree7, train_set)
cfm_train7 <- confusionMatrix(train_set$ObesityClass, pred_tree7_train)
pred_tree7_test <- predict(tree7, test_set)
cfm_test7 <- confusionMatrix(test_set$ObesityClass, pred_tree7_test)

a_train7 <- cfm_train7$overall[1]
a_test7 <- cfm_test7$overall[1]
nodes7 <- nrow(tree7$finalModel$frame)
comp_tbl <- comp_tbl %>% rbind(list(nodes7, a_train7,
                                    a_test7, 25, 10000, 10000))

# Print the comparison table
print(comp_tbl)
```

```
##          Nodes TrainAccuracy TestAccuracy MaxDepth Minsplit Minbucket
## Accuracy     3     0.3198906    0.3237179        1        2         2
## 1            5     0.4552290    0.4583333        2        5         5
## 11          17     0.9282297    0.9294872       10       50        50
## 12          13     0.9070403    0.8894231       15      100       100
## 13           1     0.1681476    0.1682692       29     1000      1000
## 14           1     0.1681476    0.1682692       20     5000      5000
## 15           1     0.1681476    0.1682692       25    10000     10000
```

Analysis and Conclusion

Decision Tree The best performing Decision Tree had:

Train Accuracy: 0.9289132 Test Accuracy: 0.9294872 Parameters: MaxDepth = 10, MinSplit = 50, MinBucket = 50

k-Nearest Neighbors The best performing kNN model had:

Accuracy: 0.8883540 Parameters: kmax = 6, distance = 1, kernel = cos Comparison

Accuracy: The Decision Tree model had a slightly higher test accuracy (0.9294872) compared to the kNN model's best accuracy (0.8883540). Stability: Decision Trees can sometimes overfit if not properly tuned, but in this case, the model with MaxDepth = 10, MinSplit = 50, and MinBucket = 50 seems to perform well.

Complexity: Decision Trees can provide an interpretable model, whereas kNN is generally simpler but can become computationally expensive with larger datasets.

Conclusion For the Obesity dataset, the Decision Tree model with MaxDepth = 10, MinSplit = 50, and MinBucket = 50 is the better choice based on higher accuracy.

g. Evaluation Using the better classifier from the previous step, perform a more sophisticated evaluation using the tools of Week 9. Specifically, (1) produce a 2x2 confusion matrix (if your dataset has more than two classes, bin the classes into two groups and rebuild the model), (2) calculate the precision and recallmanually, and finally (3) produce an ROC plot (see Tutorial 9). Explain how these performance measures makes your classifier look compared to accuracy.

From the above steps, it is clear that Desision tree has the best.

Bin the ObesityClass into two groups ("obese" and "not obese"), we can categorize the classes accordingly:

Not Obese: Insufficient_Weight, Normal_Weight, Overweight_Level_I, Overweight_Level_II Obese: Obesity_Type_I, Obesity_Type_II, Obesity_Type_III

```
# Create a new column `ObesityBinary` to categorize the classes into "obese" and "not obese"
obesity_bin$ObesityBinary <- ifelse(obesity_bin$ObesityClass %in% c("Insufficient_Weight", "Normal_Weigh

# Remove the `ObesityClass` column
obesity_bin <- obesity_bin %>% select(-ObesityClass)

# Check the head of the modified dataset
head(obesity_bin)
```

```
##   Gender Age Height Weight FamilyHistory HighCaloricFood Vegetables MealNumber
## 1 Female  21   1.62   64.0           yes              no          2          3
## 2 Female  21   1.52   56.0           yes              no          3          3
## 3   Male  23   1.80   77.0           yes              no          2          3
## 4   Male  27   1.80   87.0            no              no          3          3
## 5   Male  22   1.78   89.8            no              no          2          1
## 6   Male  29   1.62   53.0            no             yes          2          3
##    Snacking Smoking Water CaloricMonitoring PhysicalActivity TechnologyUse
## 1 Sometimes      no     2                no                0             1
## 2 Sometimes     yes     3               yes                3             0
## 3 Sometimes      no     2                no                2             1
## 4 Sometimes      no     2                no                2             0
## 5 Sometimes      no     2                no                0             0
## 6 Sometimes      no     2                no                0             0
##      Alcohol             Transport    BMICategory ObesityBinary
## 1         no Public_Transportation Healthy Weight     Not Obese
## 2  Sometimes Public_Transportation Healthy Weight     Not Obese
## 3 Frequently Public_Transportation Healthy Weight     Not Obese
## 4 Frequently               Walking     Overweight     Not Obese
## 5  Sometimes Public_Transportation     Overweight     Not Obese
## 6  Sometimes            Automobile Healthy Weight     Not Obese
```

```
# Summary of the new binary classification
summary(obesity_bin$ObesityBinary)
```

```
##    Length     Class      Mode
##      2087 character character
```

```
# Coerce target variable as factor for Confusion Matrix
obesity_bin$ObesityBinary <- as.factor(obesity_bin$ObesityBinary)
```

Descion Tree

```
library(caret)
library(rpart)
library(tidyverse)
library(rattle)
```

```
## Loading required package: bitops

## Rattle: A free graphical interface for data science with R.
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(ggplot2)
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
set.seed(94)
train_control = trainControl(method = "cv", number = 10)

# Partition the data
index = createDataPartition(y=obesity_bin$ObesityBinary, p=0.7, list=FALSE)
# Everything in the generated index list
train_set = obesity_bin[index,]
# Everything except the generated indices
test_set = obesity_bin[-index,]

# Set hyperparameters
hypers = rpart.control(minsplit =  50, maxdepth = 10, minbucket = 50)

# Fit the model
tree2 <- train(ObesityBinary ~., data = train_set, control = hypers,
               trControl = train_control, method = "rpart1SE")

# Evaluate the fit with a confusion matrix
# Evaluate the fit with a confusion matrix
pred_sc <- predict(tree2, test_set)
# Confusion Matrix
cm <- confusionMatrix(test_set$ObesityBinary, pred_sc)
cm
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Not Obese Obese
##   Not Obese       318    16
##   Obese             1   290
##
##                Accuracy : 0.9728
##                  95% CI : (0.9568, 0.9841)
##     No Information Rate : 0.5104
```

```
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9455
##
##  Mcnemar's Test P-Value : 0.000685
##
##             Sensitivity : 0.9969
##             Specificity : 0.9477
##          Pos Pred Value : 0.9521
##          Neg Pred Value : 0.9966
##              Prevalence : 0.5104
##          Detection Rate : 0.5088
##    Detection Prevalence : 0.5344
##       Balanced Accuracy : 0.9723
##
##        'Positive' Class : Not Obese
##
```

#Precision

Precision is the measure of exactness. It is given by the formula:

Precision = TP/(TP+FP) Precision =318 / 318+1 = 0.9968

#Recall

Recall is the measure of completeness. It is given by the formula:

Recall = TP/(TP+FN) Precision = 318 / 318+16= 0.9520

```
library(caret)
library(rpart)
library(tidyverse)
library(rattle)
library(ggplot2)
library(pROC)

# Store the byClass object of confusion matrix as a dataframe
metrics <- as.data.frame(cm$byClass)
# View the object
metrics
```

```
##                      cm$byClass
## Sensitivity           0.9968652
## Specificity           0.9477124
## Pos Pred Value        0.9520958
## Neg Pred Value        0.9965636
## Precision             0.9520958
## Recall                0.9968652
## F1                    0.9739663
## Prevalence            0.5104000
## Detection Rate        0.5088000
## Detection Prevalence  0.5344000
## Balanced Accuracy     0.9722888
```

```r
# Get the precision value
precision_value <- metrics["Precision", "cm$byClass"]
print(precision_value)
```

```
## [1] 0.9520958
```

```r
# Get the recall value
recall_value <- metrics["Recall", "cm$byClass"]
print(recall_value)
```

```
## [1] 0.9968652
```

```r
library(mlbench)

library(pROC)
# Get class probabilities for KNN
pred_prob <- predict(tree2, test_set, type = "prob")
head(pred_prob)
```

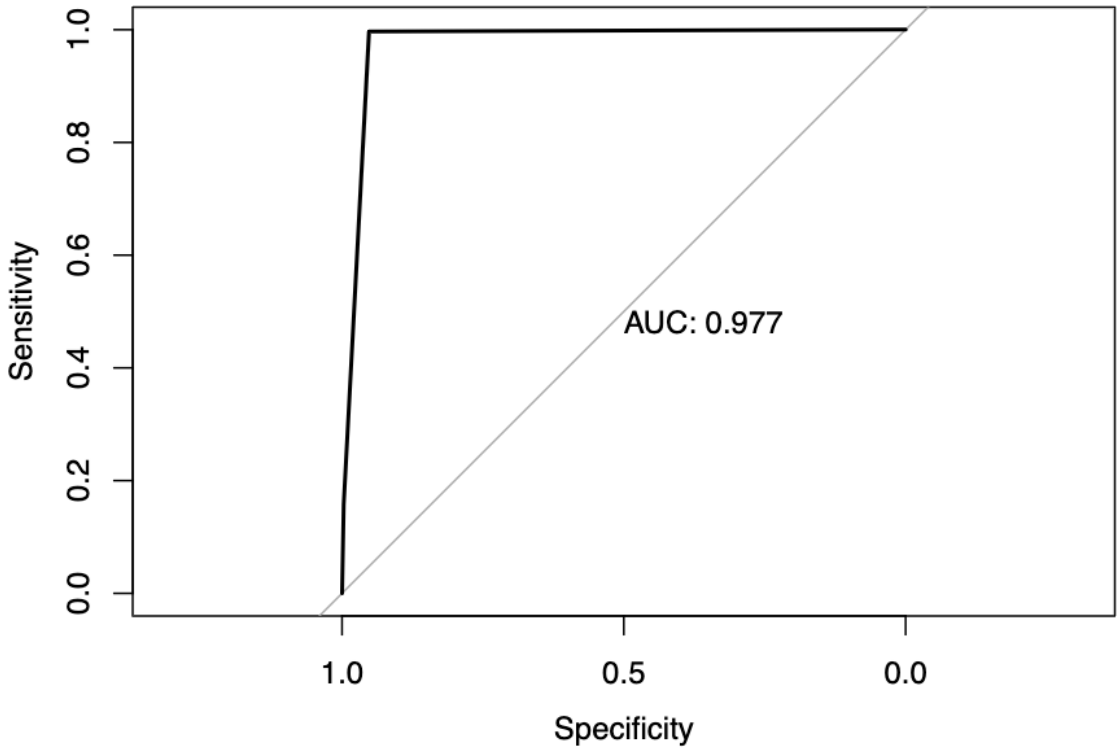
```
##     Not Obese       Obese
## 9  0.99731183 0.002688172
## 10 0.99731183 0.002688172
## 14 0.05882353 0.941176471
## 15 0.99731183 0.002688172
## 29 0.99731183 0.002688172
## 31 0.99731183 0.002688172
```

```r
# And now we can create an ROC curve for our model.
roc_obj <- roc((test_set$ObesityBinary), pred_prob[,1])
```

```
## Setting levels: control = Not Obese, case = Obese
```

```
## Setting direction: controls > cases
```

```r
plot(roc_obj, print.auc=TRUE)
```



h. Report In a single document, include the answers to all of the parts of this Problem, including this one. The report component specifically is about your overall takeaways from your data. What was interesting from your analysis?

i. Reflection The final section of the report is a (short) paragraph reflecting on the course as a whole and what you have learned. The goal is not actually feedback for the course but to get you to think back about what you have learned and how your perspective on data science has changed.

Reflecting on the course as a whole, I have gained a deep understanding of data mining as the extraction of knowledge from vast amounts of data. I learned the critical steps in the data mining pipeline, emphasizing the importance of data selection and processing. Previously, I underestimated the significance of data preprocessing, but I now recognize that it is crucial for model accuracy. I also explored methods to handle noisy and incomplete data, such as inconsistency removal, missing data replacement, and data smoothing techniques like binning. Additionally, I learned about data reduction and the creation of dummy variables. The course introduced me to supervised learning, particularly the SVM algorithm, which is effective for high-dimensional and binary classification tasks, although it requires numerical data. I gained insights into decision tree building, including induction, geometric interpretation, and pruning techniques. Importantly, I learned that selecting a model with the highest accuracy is less crucial than choosing one that generalizes well to unseen data. The kNN algorithm taught me about dissimilarity matrices and the importance of selecting the right value of K. Despite its computational expense, kNN is robust to noise and flexible in distance measures. I also delved into clustering techniques, including k-means and HAC. While k-means can be non-deterministic due to its reliance on random initialization, HAC is deterministic, providing consistent results. Overall, my perspective on data science has broadened, appreciating the intricate balance between data preparation and model selection.In the last two weeks of the course, I learned about advanced evaluation techniques, such as calculating precision and recall, and understanding the ROC curve, bias, and confidence intervals. These metrics are essential for assessing model performance beyond simple accuracy. I also explored

the ethics in data science, focusing on the privacy issues associated with data, the need for oversight, and the socio-economic impacts of algorithms. I realized the potential for algorithms to create deception and the importance of being mindful and conscious of these ethical considerations. This understanding has deepened my appreciation for the responsibilities that come with working in data science.