IE6200: Engineernig Probability and Statistics

LAB 03: Summary Statistics

Prof. Mohammad Dehghani

Contents

1 R Packages

R packages are collections of functions and data sets developed by the **R** community. They increase the power of R by improving existing base **R** functionalities, or by adding new ones. In this lab, we will make use of tidyverse package, which is a coherent system of packages for data manipulation, exploration, and visualization that share a common design philosophy. It comprises of the following packages:

- ggplot2: A system for declaratively creating graphics.
- tidyr: It provides a set of functions that help you get to tidy data. Briefly, it means, every variable goes in a column, and every column is a variable.
- dplyr: It is used for data manipulation, like creating new variables.
- readr: It provides a fast and friendly way to read rectangular data (like csv, tsv, and fwf).
- purrr: A toolkit for working with functions and vectors.
- stringr: It provides a cohesive set of functions designed to make working with strings as easy as possible.
- forcats: It provides a suite of useful tools that solve common problems with factors.
- tibble: Tibbles are data.frames that are lazy and surly: they do less and complain more forcing you to confront problems earlier.

In this lab, we will make use of ggplot2 to create visualizations to get better insights into our data.

Along with tidyverse package, another package that we will use is the e1071 package for computing descriptive statistics. It is used for building machine learning models such as support vector machines, bagged clustering, naive Bayes classifier.

ϟ To know more about the tidyverse package visit www.tidyverse.org ϟ To know more about the e1071 package visit https://cran.r-project.org/web/packages/e1071/e1071.pdf

1.1 Installing and Loading Packages

We can install the package usinge the install.packages() function. This downloads the package from an online repository such as CRAN and installs it to your local machine. Installing tidyverse also installs ggplot2 since it is a part of the tidyverse collection.

```r
install.packages('tidyverse') # install the tidyverse package
install.packages('e1071') # install the e1071 package
```

You only need to install a package once, but you need to *reload* it every time you start a new session.

In order to load the package in the R session, we use the library() function. This loads the package in R so that the functions associated with it can be used.

```r
library(tidyverse) # load the tidyverse package
library(e1071) # load the e1071 package
```

2 Importing Data in R

Importing data into R is fairly simple, and a variety of file types can be imported into R, such as **CSV, TXT, HTML, JSON, etc.** Different file types usually require different functions, but once you have the right function, the rest of the process is pretty straight forward. We will be working with a **.csv* file, which can be imported using read.table()(this function does not require installlation of any additional package.) function has the following template:

```
DataName <- read.table(file = 'file path', header = TRUE/FALSE, sep = '')
```

where:

- **file:** The name of the file which the data are to be read from. Each row of the table appears as one line of the file
- **header:** A logical value indicating whether the file contains the names of the variables as its first line.
- **sep:** The field separator character. Values on each line of the file are separated by this character, example - ',' (a comma)

In order to read an excel file, we can use the read_excel() function from the readxl package.

2.1 Health Dataset

2.1.1 Description:

This is a sample dataset which was extracted from a healthcare database with an objective to find out common characteristics for a person suffering from 'Stroke', 'Hypertension' or 'Heart Diseases'. The data comprises of 1000 observations of 6 attributes. (Source: https://www.kaggle.com/asaumya/healthcare-data)

2.1.2 Attributes:

- **Avg_Glucose_Level** (numerical) - It can be defined as amount of glucose (sugar) in the blood of a person and how it changes throughout the day. The changes in these levels depends upon what type of food consumed, when was the food consumed, how much a person has eaten, and whether a person have exercised. The blood sugar level is usually computed when a person is fasting and 2 hours after a meal. For our dataset, we will assume the **Avg_Glucose_Level** to be fasting blood sugar level. Unit for this data type is mg/dL (milligrams per deciliter). Blood sugar levels can be categorised as following:

Table 1: Blood Sugar Level Chart

Category	Fasting (mg/dl)	2 *hours after meals* (mg/dl)
Normal Person	70–99	< 140
Diabetic Person	80–130	< 180

- **BMI Index** (numerical) - Body Mass Index (BMI) can be defined as a person's weight (in kg) divided by the square of the height (in mtr). Unit for this data type is kg/m^2. BMI index can be used to screen the weight categories that may lead to health problems. A high BMI index is an indicator of high body fatness. According to American Cancer Society, BMI index can be categorized as:

Table 2: BMI Chart

BMI Range (kg/m^2)	Category
< 18.5	Underweight
18.5 - 24.9	Normal
25 - 29.9	Overweight
> 30	Obese

- **Age** (numerical) - The age of a person in years.
- **Stroke** (logical) - Indicates whether a person suffered from a stroke or not.
- **Hypertension** (logical) - Indicates whether a person suffered from hypertension(high blood pressure) or not.
- **Heart__Diseases** (logical) - Indicates whether a person suffered from a heart_disease or not.

The charts above are over simplified for the purpose of this lab, and **NOT** for reference purposes.

2.2 Importing the Health Dataset

Now, let's use the read.table() function to import the data into R.

```
# Loading the dataset into a data frame

health_df <- read.table('Health_Dataset.csv', # file to be imported
                        header = TRUE,  # the dataset has header columns
                        sep = ',')  # the field separator character. Values on each line
                                    # of the file are separated by this character
```

Before we start calculating the statistics, first, let us look at the structure of the data using the str() function.

```
str(health_df) # display the structure of the data
```

```
## 'data.frame':    1000 obs. of  6 variables:
##  $ Age             : num  18 32 20 60 61 18 28 27 34 11 ...
##  $ Avg_Glucose_Level: num  107.4 67.2 83.5 102.8 96.1 ...
##  $ BMI             : num  31.4 31.2 23.6 29.7 24.4 25.7 38.7 49.2 23.5 19.5 ...
##  $ Stroke          : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ Hypertension    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ Heart_Diseases  : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

We can see that **Age**, **Avg__Glucose__Level** and **BMI** are numerical attributes, while **Stroke**, **Hypertension** and **Heart__Diseases** are logical attributes.

3 Descriptive Statistics

Descriptive Statistics are used to describe the basic features of the data. They provide simple summaries about the sample and measures. Together with graphical analysis, these form the basis of quantitative anlysis of the data. There is a wide range of functions for obtaining these statistics in **R** and we will go over some of the basic functions that calculate these statistics.

3.1 Measures of Location

Measures of Location can be described as a single value that can be used to describe a set of data by identifying central position within that set of data. The three common measures of location are: Mean, Median, and Mode. These measures can be used to capture with a single number what is typical of the data. (Source: https://statistics.laerd.com/statistical-guides/measures -central-tendency-mean-mode-median.php)

3.1.1 Mean

The mean of a set of observed data is defined as the sum of the numerical values of each and every observation divided by the total number of observations.

Suppose that the observations in a sample are $x_1, x_2, \ldots, x_n$. The sample mean, denoted by $\bar{x}$ is calculated as:

$$\boxed{\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{x_1 + x_2 + \cdots + x_n}{n}} \tag{1}$$

In **R**, mean can be caluated using the mean() function. We will try to calculate the mean age of people who have suffered a *stroke*. We will do this in two steps

```r
# the first step is to get data about all the persons who have suffered a stroke
# before that let us take a look at the unfiltered data
head(health_df)
```

```
##   Age Avg_Glucose_Level  BMI Stroke Hypertension Heart_Diseases
## 1  18            107.38 31.4  FALSE        FALSE          FALSE
## 2  32             67.19 31.2  FALSE        FALSE          FALSE
## 3  20             83.45 23.6  FALSE        FALSE          FALSE
## 4  60            102.77 29.7  FALSE        FALSE          FALSE
## 5  61             96.07 24.4  FALSE        FALSE          FALSE
## 6  18             87.28 25.7  FALSE        FALSE          FALSE
```

```r
# now we take our data frame, and in the first index we mention get all the rows where
# the person suffered from stroke and we leave the second index blank so as to get all the
# columns and store it in a new data frame called stroke
stroke_df <- health_df[health_df$Stroke == TRUE, ]
head(stroke_df)
```

```
##     Age Avg_Glucose_Level      BMI Stroke Hypertension Heart_Diseases
## 60   79            129.98 22.60000   TRUE        FALSE           TRUE
## 134  62            216.62 38.20000   TRUE        FALSE          FALSE
## 215  62            115.70 44.60000   TRUE        FALSE            TRUE
## 357  68            243.27 23.70000   TRUE        FALSE           FALSE
## 375  73            200.53 29.87061   TRUE         TRUE           FALSE
## 418  57             78.92 27.70000   TRUE         TRUE           FALSE
```

```
# Now, we calculate the mean age of people suffering from stroke using the mean() function
mean_stroke_age <- mean(stroke_df$Age)
mean_stroke_age
```

```
## [1] 70.30769
```

```
# we can do the entire operation in a single line as well
mean_stroke_age2 <- mean(health_df[health_df$Stroke == TRUE, ]$Age)
mean_stroke_age2
```

```
## [1] 70.30769
```

📢 Can you use the mean() function for a logical variable like **Hypertension**? If yes, what does the output represent?

3.1.2 Median

Median is the value separating the higher half from the lower half of a data sample. For a data set, it may be thought of as the "middle" value when the data is sorted in ascending order. The purpose of the median is to reflect the central tendency of the sample in such a way that it is uninfluenced by extreme values or outliers.

In order to get *median*, we can use the median() function. Let us see what the median age is in our dataset

```
# calculate median age
median_age <- median(health_df$Age)
median_age
```

```
## [1] 46
```

📢 One of the main disadvantages of mean is that it is susceptible to the influence of outliners. These are the values which are unusual compared to the rest of the data set by being exceptionally small or large numerical value. In such situations, median would be a better measure of central tendency. Another advantage of median is when the given data set is skewed, median retains this position and is not as strongly influenced by the skewed values. (Source: https://statistics.laerd.com/statistical-guides/measures-central- tendency-mean-mode-median.php)

3.1.3 Minimum and Maximum

Minimum and maximum are the smallest and largest values of a function respectively. To get the minimum and maximum values of a particular column, we can use the min() and max() functions respectively. Again, let us see what the maximum and minimum age is in our dataset.

```r
# Find the minimum age
min_age <- min(health_df$Age)
min_age
```

```
## [1] 0.16
```

```r
# Find the maximum age
max_age <- max(health_df$Age)
max_age
```

```
## [1] 82
```

3.2 Measures of Variablility

Measures of Variablility can be described as a summary statistic that represents the amount of dispersion in a given data set. The goal is to obtain a measure of how far away the data points tends to fall from the center. The five common measures of variability are: Range, Interquartile Ranges (IQR), Variance, Standard Deviation, and Coefficient of Variance. These measures of variability help us to understand the width of the distribution and helps us to grasp the likelihood of unusual events. (Source: http://statisticsbyjim.com/basics/variability-range- interquartile-variance-standard-deviation/)

3.2.1 Range

Range of a set of data is the difference between the largest and smallest values. It is one of the most straightforward measure of variability to calculate. It can be understood as wider the range, greater the variability in a given data set. The range() function in R does *not* actually return the range, but returns a vector containing the minimum and maximum of the given arguement.

```
# Get the range for age of people
range_age <- range(health_df$Age)
range_age
```

```
## [1]  0.16 82.00
```

3.2.2 Quantiles

Quantiles are cut points that divide the range of a distribution into continuous intervals. Using the built-in quantile() function, we can get the required quantiles. By default, the function displays the *quartiles*, which divide the data into quarters, which are 25%, 50% and 75% along with 0% and 100%. The *interquartile range* is in between the upper and lower quartile. It can also be described as middle half of the data and includes 50% of data points that fall in Q1 and Q3.

```
# Finding the age quartiles.
quantile(health_df$Age)
```

```
##    0%   25%   50%   75%  100%
##  0.16 26.00 46.00 60.25 82.00
```

Quartiles are important because they let us see what is the range where most of the data points lie and is less sensitive and more resilient to the presence of outliners. Looking at the above example we can see that the interquartile range for age is between 26 and 60.25 years.

We can get any specific quantile by explicitly mentioning it in the following manner:

```
# To get the point till which the first 10% of data lies, we can do the following
quantile(health_df$Age, 0.10)
```

```
## 10%
##  11
```

ϟ What value do you need to mention to get the upper 10% quantile?

3.2.3 Variance

Variance is defined as the average of the squared differences from the mean. It measures how far a set of numbers are spread out from their average value. One of the main advantages of variance is that it includes all the values of a data set in the calculation by comparing each value to the mean. Another advantage is that it acts as a description of how balance is our data set.

Suppose again that the observations in a sample are $x_1, x_2, \ldots, x_n$. The variance, represented as s^2 can be computed using the following equations:

$$\boxed{s^2 = \sum_{i=1}^{n} \frac{(x_i - \overline{x})^2}{n-1}} \tag{2}$$

or

$$\boxed{s^2 = \frac{\sum x^2 - \frac{\left(\sum x\right)^2}{n}}{n-1}} \tag{3}$$

Variance can be calculated using the var() function.

```r
# Calculating the variance of age
var_age <- var(health_df$Age)
var_age
```

```
## [1] 499.7583
```

3.2.4 Standard Deviation

Standard Deviation is defined as amount of variability or dispersion around an average. It is typical difference between each data point and the mean. One of main advantages of standard deviation is that it makes it easier for us to ignore small deviations and see the larger ones clearly. If the given set of data is normally distributed, standard deviation is considered as the best measure of

---

It is the positive square root of s^2 variation, and given the same sample observartions from x_1, $x_2, \ldots, x_n$, it can be represented as:

$$\boxed{s = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n-1}}} \tag{4}$$

In **R**, standard deviation is calculated using the sd() function.

```r
# Standard Deviation of age for people that suffered a stroke
sd_stroke_age <- sd(health_df[health_df$Stroke == TRUE, ]$Age)
sd_stroke_age
```

```
## [1] 7.696153
```

3.2.5 Coefficient of Variation

Coefficient of Variation (CV) is defined as the ratio of the standard deviation to the mean. It is often expressed as a percentage and is a standardized measure of dispersion of a probability distribution or frequency distribution. One of the main advantages of coefficient of variance is that it is dimensionless number. So, if we want to compare different data set with different units or widely different means, we must choose coefficient of variance over standard deviation as it is considered as the best measure of variability for such situations.

The coefficient of variation (CV) is defined as the ratio of the standard deviation, it can be represented as:

$$\boxed{c_{\mathrm{v}} = \frac{\sigma}{\mu}.} \tag{5}$$

R does not have a function that automatically caluclates CV, so we need to do it manually using standard deviation and mean.

```
cv_stroke_age <- (sd_stroke_age/mean_stroke_age)*100
cv_stroke_age
```

```
## [1] 10.94639
```

3.3 Measures of Symmetry

Measures of Symmetry can be described as the distribution or pattern of the data points within a data set. A distribution of a data points can be symmetric or an asymmetric. A normal distribution is a common example of symmetric distribution of data points. Measures of symmetric is important because it can assist us in identifying which measures of location can be considered appropriate to use. (Source: http://www.abs.gov.au/websitedbs/a3121120.nsf/home/statistical+language+-+measures+of+shape)

3.3.1 Skewness

Skewness can be defined as the tendency for the data points to be more frequent around the high or low ends of the x-axis. There are two types of skewness:

- *Positive Skewness* means when the tail on the right side of the distribution is longer. The mean and median will be greater than the mode.
- *Negative Skewness* is when the tail on the left side of the distribution is longer than the tail on the right side. The mean and median will be less than the mode.

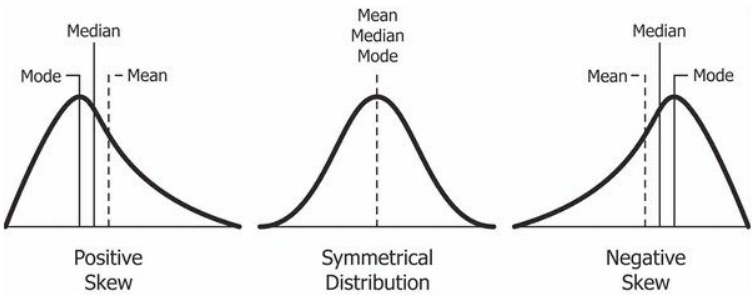


Figure 1: Skewness

When is the skewness too much?

The rule of thumb seems to be:

- If the skewness is between -0.5 and 0.5, the data are fairly symmetrical
- If the skewness is between -1 and -0.5(negatively skewed) or between 0.5 and 1(positively skewed), the data are moderately skewed.
- If the skewness is less than -1(negatively skewed) or greater than 1(positively skewed), the data are highly skewed.

The skewness() function can be used tocalculate skewness.

```r
# Get the skewness of age
skewness_age <- skewness(health_df$Age)
skewness_age
```

```
## [1] -0.1713341
```

Here, the negative value signifies that mean and median are less than mode and tail is longer on the left side

3.3.2 Kurtosis

Kurtosis can be described as a measure of whether the data are heavily-tailed or lightly-tailed relative to a normal distribution. Moreover, a data set with high kurtosis tends to have heavy tails, or outliers. A data set with low kurtosis tends to have light tails, or lack of outliers. There are two types of Kurtosis:

- *Positive Kurtosis(heavily tailed)* means distribution has longer tails which means that data is heavy-tailed or profusion of outliers.
- *Negative Kurtosis(lightly tailed)* implies distribution is shorter and tails are thinner and means data is light-tailed or lack of outliers.

(Source: https://www.itl.nist.gov/div898/handbook/eda/section3/eda35b.htm)

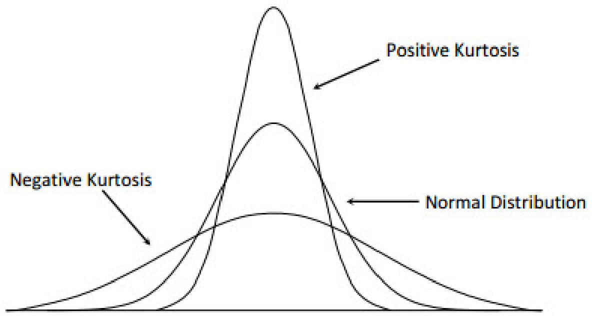


Figure 2: Kurtosis

The kurtosis() function can calculate kurtosis.

```
# Get kurtosis of age
kurtosis_age <- kurtosis(health_df$Age)
kurtosis_age
```

```
## [1] -0.9914644
```

Negative kurtosis for this subset of data means shorter tails and lack of outliers.

3.4 Summary

An easier way to get some of these descriptive statistics is by using the summary() function, which provides overall description of the dataset.

```r
summary(health_df)
```

```
##       Age        Avg_Glucose_Level      BMI          Stroke       
##  Min.   : 0.16   Min.   : 55.06    Min.   :13.20   Mode :logical  
##  1st Qu.:26.00   1st Qu.: 77.80    1st Qu.:23.90   FALSE:987      
##  Median :46.00   Median : 91.36    Median :28.85   TRUE :13       
##  Mean   :43.35   Mean   :104.53    Mean   :29.29                  
##  3rd Qu.:60.25   3rd Qu.:111.47    3rd Qu.:32.90                  
##  Max.   :82.00   Max.   :256.43    Max.   :62.90                  
##  Hypertension    Heart_Diseases 
##  Mode :logical   Mode :logical  
##  FALSE:913       FALSE:959      
##  TRUE :87        TRUE :41       
##                                 
##                                 
##                                 
```

4 Visual Analysis

A good visualisation will show you things that you did not expect, or raise new questions about the data. A good visualisation might also hint that you are asking the wrong question, or you need to collect different data. Essentailly, a visualization lets you inspect your data with a picture, see how it looks and only then think about interpreting the more vital statistics. In this section, we will create a histogram, scatterplot, stem and leaf plot and a boxplot to see how the data is distributed.

4.1 ggplot2

The ggplot2 is a coherent system for describing and building graphs. With ggplot2, you can do more and faster by learning one system and applying it in many places. The command structure of a ggplot() function follows the following template:

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

where:

- ggplot(): It initalizes a ggplot object and declares the input data frame for a graphic.
- aes(): Aesthetic mappings describe how variables in the data are mapped to visual properties (aesthetics) of geoms. Bascially, these mappings determine the apperance of a visualization.
- geom_function(): Geom functions determine what kind of object is plotted, for example, geom_histogram() will create a histogram.

ϟ To get more practice on building graphs using ggplot2, go through chapter 3 in the following link: https://r4ds.had.co.nz/index.html

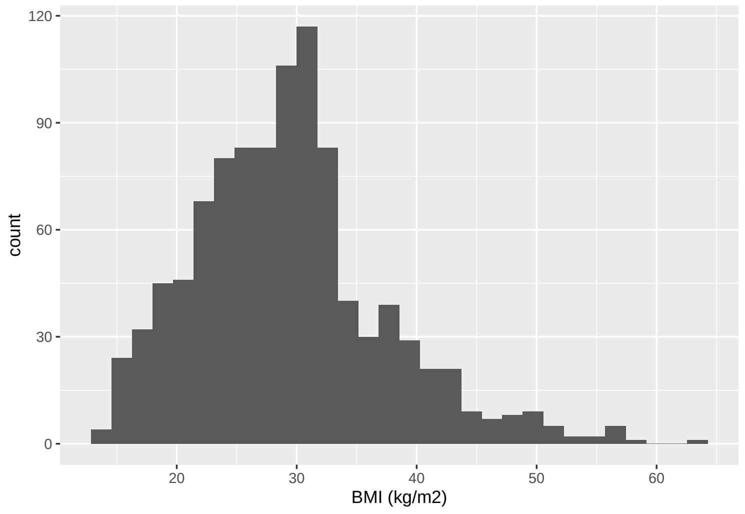
Table 3: Aesthetic Properties associated with commonly used geoms

Geom	Required	Optional
geom_bar	x variable	color, size, fill, linetype, weight, alpha
geom_point	x variable, y variable	shape, color, size, fill, alpha
geom_line	x variable, y variable	color, size, linetype, alpha
geom_smooth	x variable, y variable	color, size, fill, linetype, weight, alpha
geom_histogram	x variable	color, size, fill, linetype, weight, alpha
geom_boxplot	x variable	color, size, fill, weight, alpha
geom_hline	y intercept value	color, size, linetype, alpha
geom_vline	x intercept value	color, size, linetype, alpha

4.2 Histogram

A histogram is a representation of distribution of numerical data and can be used to estimate the probabiliy distribution of a continuous variable. A histogram can be plotted as below:

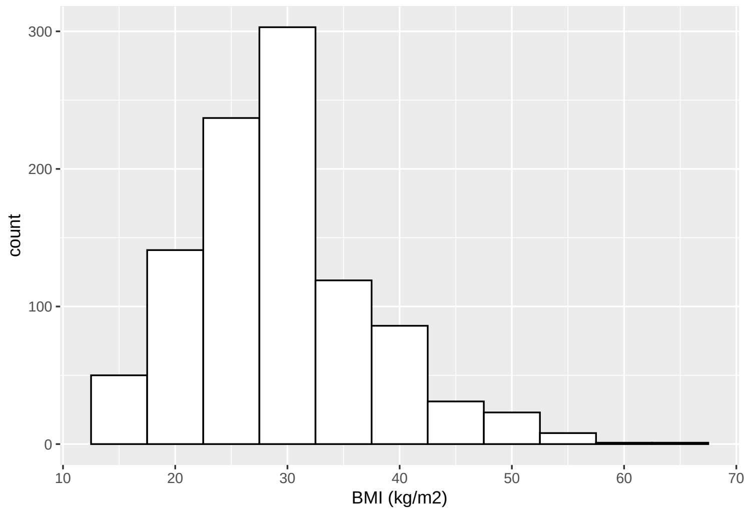
```r
ggplot(data = health_df) +
  geom_histogram(mapping = aes(x = BMI)) +
  xlab('BMI (kg/m2)')
```



Here, bins is a default parameter with value = 30, i.e., the variable is divided into 30 intervals.

We can change the intervals of a histogram using the *binwidth* parameter, which divides the range of values into a series of intervals and counts values in each interval. This allows us to better fit the data and draw better insights.

```r
ggplot(data = health_df) +
  geom_histogram(mapping = aes(x = BMI), binwidth = 5,
                 color = 'Black', fill = 'White') +
  xlab('BMI (kg/m2)')
```



ϟ From the two histograms we can see that BMI has somewhat of a normal distribution with positive skew. Can you explain this behavior?

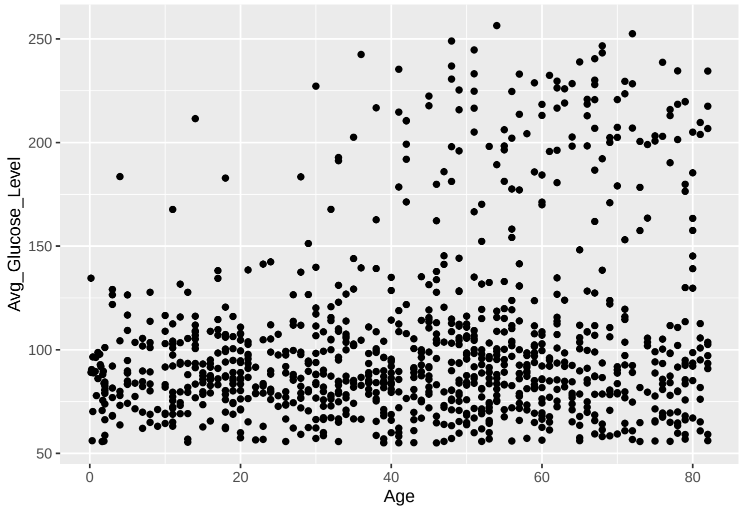
Just looking at the above histogram for BMI, and not having any prior knowledge about other variables in the data, some of the questions that might be asked are:

- What are the demographics of this distribution? Does it include a particular age range or gender?
- Is this data for a particular geographic location?
- If this is a sample data, does it mirror the distribution of the entire population given the parameters are same, such as age and gender and geographic location?
- Can the histogram bins be made better given how the BMI itself is classified?
- What are the common characteristics for the people that fall into a particular bucket?

4.3 Scatter plot

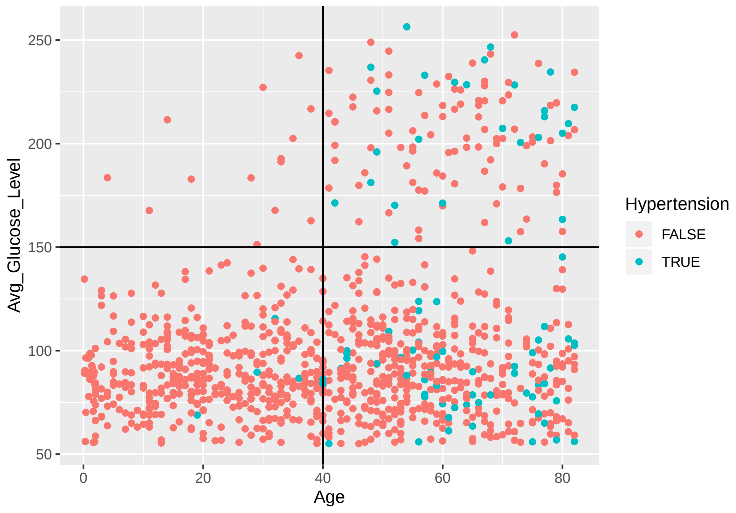
A scatter plot displays the chosen variables in form of Cartesian coordinates for a set of data. This allows the user to identify if any relationship exists between the two variables.

```r
ggplot(data = health_df) +
  geom_point(mapping = aes(x = Age, y = Avg_Glucose_Level))
```



To identify the relationships better that might exist, adding a third variable as a color element can provide additional insights into the data.

```r
ggplot(data = health_df) +
  geom_point(mapping = aes(x = Age, y = Avg_Glucose_Level, color = Hypertension)) +
  geom_vline(xintercept = 40) +
  geom_hline(yintercept = 150)
```



Adding Hypertension to Age and Glucose levels, and adding horizontal and vertical reference lines, we can see that the chances of a person suffering from Hypertension are much greater when Age is more than 40 and Glucose levels above 150.

4.4 Stem and Leaf Plot

Stem and Leaf Plots A Stem and Leaf Plot is a special table where each data value is split into a "stem" (the first digit or digits) and a "leaf" (usually the last digit). The "stem" is used to group the scores and each "leaf" shows the individual scores within each group.

```r
stem(health_df$Age)
```

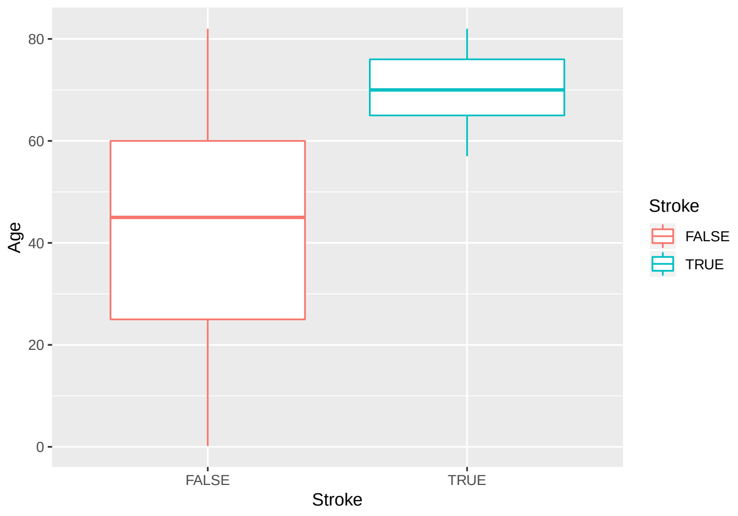
```
##
##   The decimal point is 1 digit(s) to the right of the |
##
##   0 | 00000001111111122222222222222222233333333444444
##   0 | 555555555555666677777778888888899
##   1 | 0000000001111111111111112222222223333333334444444444444
##   1 | 5555555556666666666777777777788888888888888889999999999999
##   2 | 000000000000000001111111111122223333333334444444444
##   2 | 55556666666666777777777888888888888899999999
##   3 | 00000000000000001111111111112222222222222333333333333333334444444444
##   3 | 5555555555556666677777777777888888888888888999999999999999
##   4 | 00000000000000001111111111111122222222333333333333344444444444
##   4 | 55555555555555555666666666666666677777777778888888888888888888899999999+2
##   5 | 00000000000000000111111111111111111111222222222222222222222233333333333333+9
##   5 | 55555555555555555666666666666666667777777777777777778888888888888899999
##   6 | 00000000000000000000011111111111122222222222222222222233333333344444444444
##   6 | 55555555555555666666666666666677777777777777777888888888888999999999999
##   7 | 00000000000111111111111122222222223333334444444
##   7 | 5555555555566666666666677777777777788888888888889999999999999999
##   8 | 000000000000111111111112222222222
```

4.5 Boxplot

A boxplot is standardized way of displaying the distribution of data based on a five number summary - *minimum, first quartile (Q1), median, third quartile (Q3), and maximum.* Outliers are also displayed if any exist.

```r
boxplot <- ggplot(data = health_df) +
  geom_boxplot(mapping = aes(x = Stroke, y = Age, color = Stroke))

boxplot
```

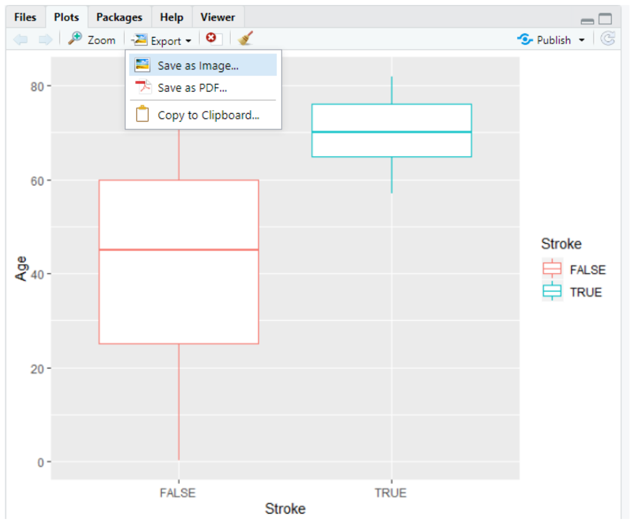


From the boxplots, we can see that the older population is at a greater risk of suffering from stroke.

A graph can also be saved as workspace variable be defining it in the same manner as a vector or a list.

4.6 Exporting Graphs

Graphs rendered in **R** can be exported by going to the **Plots** panes and clicking the ‘Export’ button and choosing the desired format.


Files
Plots
Packages
Help
Viewer
Zoom
Export
Publish
Save as Image...
Save as PDF...
Copy to Clipboard...
80
60
40
20
0
Age
FALSE
TRUE
Stroke
Stroke
FALSE
TRUE