American International University-Bangladesh (AIUB)

Faculty of Science & Technology (FST)

Department of Computer Science

Introduction to Data Science

Mid-Term Project Report

Summer 2024-2025

Section: B

Group: 09

SL #	**Student Name**	**Student ID**	**Contribution (%)**
1.	MD Tasmim Al Tahsin	22-46299-1	25%
2.	Abrar Shakil Oishik	22-46257-1	25%
3.	Aishee Debnath	22-46416-1	25%
4.	Mahir Chowdhury	22-46162-1	25%

Depression Student Dataset

About Dataset:

The dataset contains 201 records and 11 attributes related to students' demographic information, lifestyle habits, academic environment, mental health indicators, and depression status. It includes gender, age, academic pressure, study satisfaction, sleep duration, dietary habits, and study hours. Additional attributes record financial stress levels, family history of mental illness, depression and whether the student has ever experienced suicidal thoughts. Demographic information includes gender and age. Lifestyle and academic factors cover academic pressure, study satisfaction, sleep duration, dietary habits, and study hours. Mental health indicators include financial stress, family history of mental illness, depression and suicidal thoughts. The target variable, *Depression*, is a binary label showing whether a student is experiencing depression (*Yes*) or not (*No*). The dataset contains both numerical and categorical data, with some missing values, outliers, Invalid data etc.

Project Implementation Detail

Data Exploration:

Data exploration helps us understand the dataset by examining its structure, summary statistics, and distribution of values. This step allows us to detect missing values, spot unusual data points, and gain initial insights before modeling.

```
data <- read_excel("E:\\Data Science\\Dataset\\Depression Student Dataset.xlsx")
```

```
> print(data)
# A tibble: 201 × 11
   Gender   Age `Academic Pressure` `Study Satisfaction` `Sleep Duration`  `Dietary Habits`
   <chr>  <dbl>               <dbl>                <dbl> <chr>             <chr>
 1 Male      28                   2                    4 7-8 hours         Moderate
 2 Male      28                   4                    5 5-6 hours         Healthy
 3 Male      25                   1                    3 5-6 hours         Unhealthy
 4 Male      23                   1                    4 More than 8 hours Unhealthy
 5 NA        31                   1                    5 More than 8 hours Healthy
 6 Male      19                   4                    4 5-6 hours         Unhealthy
 7 Female    34                   4                    2 NA                Moderate
 8 Female    20                   4                    1 More than 8 hours Healthy
 9 Female    NA                   1                    4 More than 8 hours Moderate
10 Male      33                   4                    3 Less than 5 hours Unhealthy
# i 191 more rows
# i 5 more variables: `Have you ever had suicidal thoughts ?` <chr>, `Study Hours` <dbl>,
#   `Financial Stress` <dbl>, `Family History of Mental Illness` <chr>, Depression <chr>
# i Use `print(n = ...)` to see more rows
>
```

Quick Overview of the Dataset

str (data)

```
> str (data)
tibble [201 × 11] (S3: tbl_df/tbl/data.frame)
 $ Gender                                : chr [1:201] "Male" "Male" "Male" "Male" ...
 $ Age                                   : num [1:201] 28 28 25 23 31 19 34 20 NA 33 ...
 $ Academic Pressure                     : num [1:201] 2 4 1 1 1 4 4 4 1 4 ...
 $ Study Satisfaction                    : num [1:201] 4 5 3 4 5 4 2 1 4 3 ...
 $ Sleep Duration                        : chr [1:201] "7-8 hours" "5-6 hours" "5-6 hours" "More than 8 hours" ...
 $ Dietary Habits                        : chr [1:201] "Moderate" "Healthy" "Unhealthy" "Unhealthy" ...
 $ Have you ever had suicidal thoughts ?: chr [1:201] "Yes" "Yes" "Yes" "Yes" ...
 $ Study Hours                           : num [1:201] 9 7 10 7 4 1 6 3 10 10 ...
 $ Financial Stress                      : num [1:201] 2 1 4 2 2 4 2 4 3 1 ...
 $ Family History of Mental Illness      : chr [1:201] "Yes" "Yes" "No" "Yes" ...
 $ Depression                            : chr [1:201] "No" "No" "Yes" "No" ...
>
```

summary(data)

```
> summary(data)
    Gender               Age         Academic Pressure Study Satisfaction Sleep Duration     Dietary Habits
 Length:201         Min.   : 18.00   Min.   : 1.000    Min.   :1.000      Length:201         Length:201
 Class :character   1st Qu.: 22.00   1st Qu.: 2.000    1st Qu.:2.000      Class :character   Class :character
 Mode  :character   Median : 26.00   Median : 3.000    Median :3.000      Mode  :character   Mode  :character
                    Mean   : 28.25   Mean   : 3.154    Mean   :3.189
                    3rd Qu.: 30.00   3rd Qu.: 4.000    3rd Qu.:4.000
                    Max.   :230.00   Max.   :20.000    Max.   :5.000
                    NA's   :3
 Have you ever had suicidal thoughts ?  Study Hours     Financial Stress Family History of Mental Illness
 Length:201                             Min.   : 0.000   Min.   :1.00     Length:201
 Class :character                       1st Qu.: 3.000   1st Qu.:2.00     Class :character
 Mode  :character                       Median : 7.000   Median :3.00     Mode  :character
                                        Mean   : 6.332   Mean   :2.93
                                        3rd Qu.:10.000   3rd Qu.:4.00
                                        Max.   :12.000   Max.   :5.00
                                        NA's   :2
  Depression
 Length:201
 Class :character
 Mode  :character
```

colsums(is.na(data))

```
> colSums(is.na(data))
                                Gender                                    Age                       Academic Pressure
                                     3                                      3                                       0
                    Study Satisfaction                         Sleep Duration                          Dietary Habits
                                     0                                      3                                       0
Have you ever had suicidal thoughts ?                            Study Hours                       Financial Stress
                                     0                                      2                                       0
     Family History of Mental Illness                             Depression
```

Finding and Handling Missing value:

The task is to identify missing values in the dataset. This will be solved by using is.na() to locate missing entries and colSums() to count them for each column.

```
missing_summary <-colSums(is.na(data))
missing_summary
```

```
> colSums(is.na(data))
                                  Gender                                      Age                        Academic Pressure
                                       3                                        3                                        0
                     Study Satisfaction                         Sleep Duration                          Dietary Habits
                                       0                                        3                                        0
Have you ever had suicidal thoughts ?                            Study Hours                        Financial Stress
                                       0                                        2                                        0
     Family History of Mental Illness                             Depression
                                       0                                        3
>
```

Code Description: is.na(data) returns a table of TRUE/FALSE values showing where data is missing. colSums() then counts how many missing values are in each column, giving you the total number of missing entries per column. The result is stored in missing summary. We will work next to replace those missing values.

Handling Age Missing Values:

The task is to fill in the missing values of the Age column. This will be done by calculating the average age from the available data, replacing all missing entries with this average, and then rounding the results to whole numbers for uniformity.

```
data$Age[is.na(data$Age)] <- mean(data$Age, na.rm =TRUE) data$Age
data$Age <- as.integer(round(data$Age))
head(data)
```

Code Description: This code detects missing ages using is.na (), replaces them with the mean age,

```
> data$Age[is.na(data$Age)] <- mean(data$Age, na.rm = TRUE)
> data$Age
  [1]  28.00000  28.00000  25.00000  23.00000  31.00000  19.00000  34.00000  20.00000  28.25253  33.00000  31.00000
 [12]  24.00000  23.00000  25.00000  21.00000  28.00000  23.00000  23.00000  20.00000  29.00000  31.00000  24.00000
 [23]  31.00000  33.00000  33.00000  31.00000  30.00000  21.00000  29.00000  34.00000  20.00000  33.00000  32.00000
 [34]  21.00000  26.00000  26.00000  28.25253  26.00000  25.00000  21.00000  29.00000  22.00000  21.00000  31.00000
 [45]  24.00000  20.00000  20.00000  31.00000  21.00000  24.00000  34.00000  25.00000  27.00000  28.25253  26.00000
 [56]  23.00000  22.00000  29.00000  20.00000  28.00000  30.00000  29.00000  29.00000  24.00000  19.00000  29.00000
 [67]  20.00000  31.00000  27.00000  27.00000  30.00000  21.00000  27.00000  28.00000  26.00000  33.00000  31.00000
 [78]  32.00000  28.00000  24.00000  19.00000  31.00000  22.00000  24.00000  33.00000  34.00000  18.00000  32.00000
 [89]  18.00000  30.00000  25.00000  33.00000  22.00000  23.00000  26.00000  27.00000  32.00000  26.00000  33.00000
[100]  21.00000  30.00000  24.00000  26.00000  20.00000  29.00000  19.00000  19.00000  25.00000  18.00000  22.00000
[111]  18.00000  20.00000  28.00000  21.00000  20.00000 230.00000  31.00000  22.00000  25.00000  30.00000  30.00000
[122] 226.00000  28.00000  30.00000  26.00000  29.00000  28.00000  20.00000  34.00000  33.00000  19.00000  27.00000
[133]  22.00000  25.00000  20.00000  29.00000  34.00000  27.00000  26.00000  34.00000  24.00000  18.00000  28.00000
[144]  18.00000  19.00000  33.00000  20.00000  29.00000  33.00000  34.00000  24.00000  24.00000  25.00000  28.00000
[155]  30.00000  28.00000  29.00000  34.00000  32.00000  24.00000  29.00000  26.00000  29.00000  25.00000  28.00000
[166]  19.00000  24.00000  20.00000  33.00000  27.00000  24.00000  32.00000  33.00000  27.00000  25.00000  21.00000
[177]  20.00000  33.00000  27.00000  31.00000  26.00000  33.00000  18.00000  22.00000  19.00000  22.00000  34.00000
[188]  33.00000  20.00000  22.00000  29.00000  27.00000  28.00000  30.00000  34.00000  18.00000  20.00000  19.00000
[199]  26.00000  25.00000  32.00000
> data$Age <- as.integer(round(data$Age))
> head(data)
# A tibble: 6 × 11
  Gender   Age `Academic Pressure` `Study Satisfaction` `Sleep Duration`  `Dietary Habits` Have you ever had suicidal t…'
  <chr>  <int>               <dbl>                <dbl> <chr>             <chr>            <chr>
1 Male      28                   2                    4 7-8 hours         Moderate         Yes
2 Male      28                   4                    5 5-6 hours         Healthy          Yes
3 Male      25                   1                    3 5-6 hours         Unhealthy        Yes
4 Male      23                   1                    4 More than 8 hours Unhealthy        Yes
5 NA        31                   1                    5 More than 8 hours Healthy          Yes
6 Male      19                   4                    4 5-6 hours         Unhealthy        Yes
```

then rounds and converts the values to integers for consistency.

Handling Gender Column:

The task is to handle missing values in the Gender column of the dataset. Missing values will be replaced with the most frequent category (mode) to maintain data consistency and ensure the column has no gaps.

```
table(data$Gender)
mode_gender <- names(sort(table(data$Gender), decreasing = TRUE))[1]
data$Gender[is.na(data$Gender)] <- mode_gender
data[!complete.cases(data),]
```

	Gender	Age	Academic Pressure	Study Satisfaction	Sleep Duration	Dietary Habits	Have you ever had suicidal t…'
	<chr>	<int>	<dbl>	<dbl>	<chr>	<chr>	<chr>
1	Female	34	4	2	NA	Moderate	Yes
2	Male	31	5	4	5-6 hours	Healthy	Yes
3	Female	23	5	5	NA	Unhealthy	Yes
4	Male	23	5	2	More than 8 hours	Moderate	No
5	Female	23	1	3	NA	Healthy	Yes
6	Female	33	2	3	7-8 hours	Moderate	Yes

Code Description: The code first displays the frequency of each category in the Gender column using table (). It then calculates the mode by sorting the frequency table in decreasing order and selecting the first element. Finally, it replaces all missing (NA) values in the Gender column with this mode, ensuring that the column has complete and consistent data.

Handling Sleep Duration Column:

This task fills missing values in the Sleep Duration column by replacing them with the most frequent value (mode) to ensure the dataset has no gaps in this feature.

```
unique(data$`Sleep Duration`)
mode_sleep <- names(sort(table(data$`Sleep Duration`), decreasing = TRUE))[1]
data$`Sleep Duration`[is.na(data$`Sleep Duration`)] <- mode_sleep
data[!complete.cases(data),]
```

	Gender	Age	`Academic Pressure`	`Study Satisfaction`	`Sleep Duration`	`Dietary Habits`	Have you ever had suicidal t…
	<chr>	<int>	<dbl>	<dbl>	<chr>	<chr>	<chr>
1	Male	31	5	4	5-6 hours	Healthy	Yes
2	Female	23	5	5	More than 8 hours	Unhealthy	Yes
3	Male	23	5	2	More than 8 hours	Moderate	No
4	Female	33	2	3	7-8 hours	Moderate	Yes

Code Description: The code first checks all distinct values in the Sleep Duration column using unique(). It calculates the most frequent value (mode) with
mode_sleep <- names(sort(table(data$`Sleep Duration`), decreasing = TRUE))[1].
All missing values are replaced with this mode using data$`Sleep Duration`[is.na(data$`Sleep Duration`)] <- mode_sleep.

Handling Study Hours Column:

This task handles missing values in the Study Hours column by replacing them with the mean of the column. The values are then rounded and converted to integers to maintain consistency and make the data ready for analysis.

```
hist(data$`Study Hours`)
data$`Study Hours`[is.na(data$`Study Hours`)] <- mean(data$`Study Hours`, na.rm =
TRUE)
data$`Study Hours` <- as.integer(round(data$`Study Hours`))
data[!complete.cases(data),]
data$`Study Hours`
```

```
> data$`Study Hours`
  [1]  9  7 10  7  4  1  6  3 10 10  6 11  2 12  3  8  6  0  2  1  3  1 10 11 12  2  0  6  4 12  2  3 12  9 10  3  2  4
 [39]  8  8  6  6  1 12  3 11 11 12  1 12  8  7 10 10  9  8 12  4  5  1  6  0  6  3  5 11  8  5 11  8  8  5  0  8 11  8
 [77] 10  2 10  6  3  8  7  3 10  0  9  3  9 10  4  7  7  8 10  5  0  7 11  5  0  8  7  4  5  9 10 11  3  7  9  0  4  8
[115]  2  8 10 10  2  2  0  0  6 10 10  6  3  4 11 12  5  4  7  9  0 12  6 12  6 12  5  9  2  8  2  1 11 11 12  8  0  8
[153]  0  3 10 12  5  8  6  4  4 12  6  1  3  1  5 10  0  9 12  7 12 12  4  2  3  7  2  9  8  6  2 12 10  3  3  0  0 11
[191]  8  3  2 10 12 10  3  7  5  9  2
```

Code Description: The code first visualizes the distribution of Study Hours using hist(). It then calculates the mean of the column and replaces all missing values with this mean. The values are rounded and converted to integers using round() and as.integer() for consistency. Finally, data[!complete.cases(data),] checks for any remaining missing entries, and data$Study Hours`` displays the cleaned column in the console.

Handling Depression Column:

This task handles missing values in the Depression column by replacing them with the most frequent category (mode) to ensure the data is complete and ready for analysis.

```
mode_depression <- names(sort(table(data$Depression), decreasing = TRUE))[1]
data$Depression[is.na(data$Depression)] <- mode_depression
data[!complete.cases(data),]
data$`Depression`
```

```
> data$`Depression`
  [1] "No"  "No"  "Yes" "No"  "No"  "Yes" "Yes" "Yes" "No"  "Yes" "Yes" "Yes" "No"  "Yes" "Yes" "Yes" "No"  "No"  "Yes"
 [20] "Yes" "No"  "No"  "No"  "No"  "No"  "No"  "No"  "Yes" "No"  "No"  "No"  "No"  "Yes" "Yes" "Yes" "No"  "No"  "No"
 [39] "No"  "Yes" "No"  "No"  "Yes" "No"  "No"  "No"  "No"  "No"  "No"  "No"  "No"  "No"  "No"  "No"  "No"  "No"  "No"
 [58] "Yes" "Yes" "No"  "Yes" "No"  "Yes" "No"  "Yes" "No"  "No"  "Yes" "No"  "No"  "No"  "No"  "Yes" "No"  "Yes" "No"
 [77] "No"  "No"  "Yes" "Yes" "Yes" "No"  "Yes" "Yes" "Yes" "Yes" "Yes" "Yes" "No"  "No"  "No"  "No"  "No"  "Yes" "No"
 [96] "Yes" "No"  "No"  "No"  "No"  "Yes" "Yes" "No"  "No"  "Yes" "No"  "Yes" "No"  "Yes" "Yes" "Yes" "No"  "No"  "No"
[115] "No"  "No"  "Yes" "Yes" "No"  "No"  "No"  "Yes" "Yes" "No"  "Yes" "Yes" "No"  "Yes" "Yes" "Yes" "Yes" "Yes" "Yes"
[134] "Yes" "Yes" "Yes" "No"  "Yes" "Yes" "No"  "Yes" "Yes" "No"  "Yes" "Yes" "No"  "Yes" "Yes" "Yes" "No"  "Yes" "Yes"
[153] "No"  "No"  "No"  "Yes" "No"  "No"  "No"  "Yes" "No"  "No"  "No"  "No"  "Yes" "No"  "No"  "Yes" "No"  "Yes" "Yes"
[172] "No"  "Yes" "No"  "Yes" "No"  "No"  "No"  "No"  "No"  "No"  "No"  "Yes" "Yes" "No"  "No"  "No"  "Yes" "No"  "Yes"
[191] "Yes" "No"  "No"  "No"  "No"  "No"  "No"  "No"  "No"  "No"  "No"
>
```

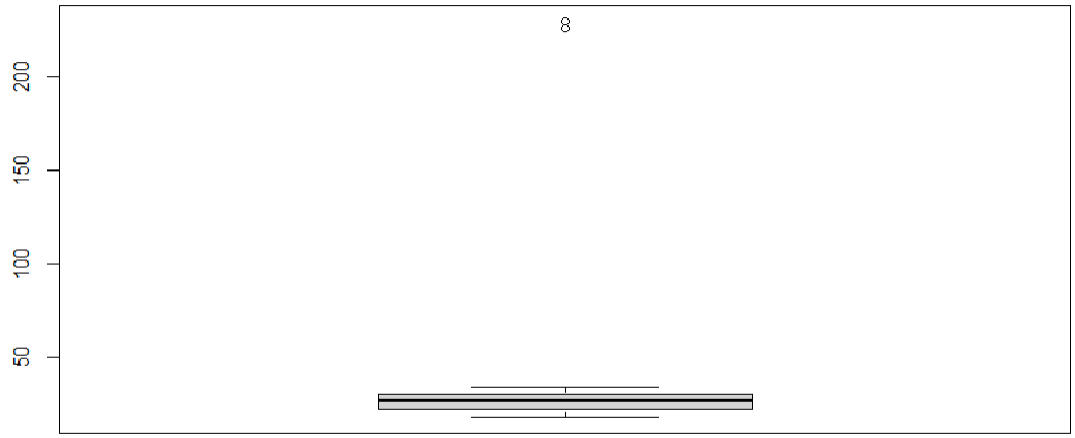
Code Description: The first line calculates the most frequent category (mode) of the Depression column using table() and sort(). The second line replaces all missing values (NA) with this mode. data[!complete.cases(data),] checks for any remaining missing entries in the dataset, and data$Depression displays the cleaned column in the console.

Outliers Detection and Handling:

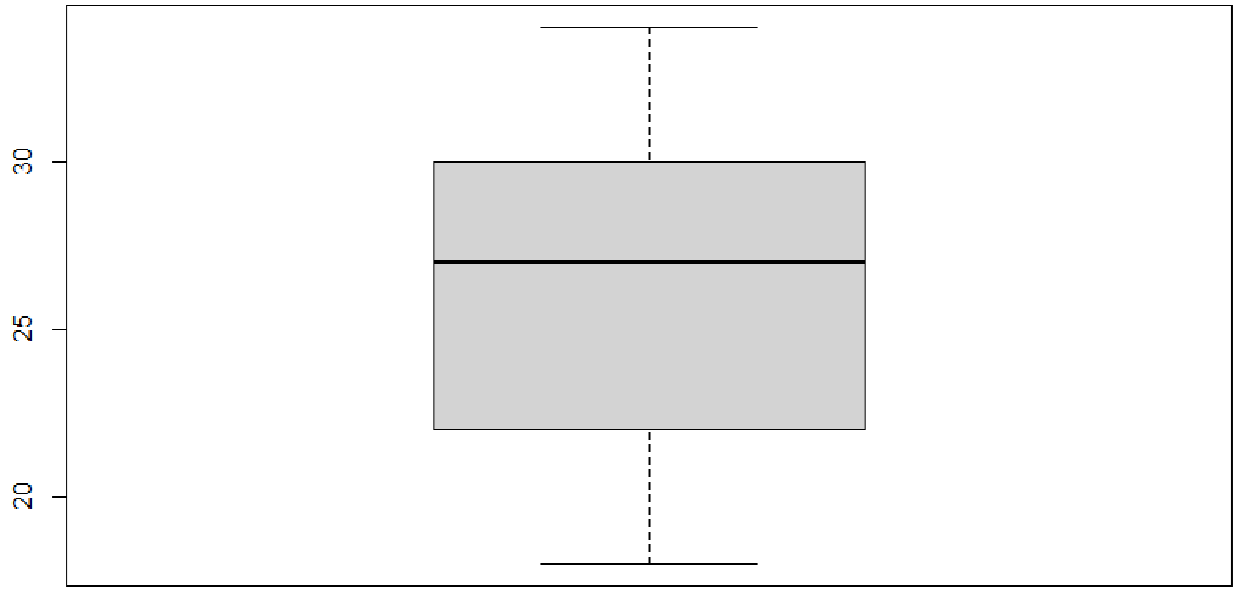
Age Column:

In this task, the goal is to detect and handle outliers in the Age column of the dataset. Outliers can skew the distribution and affect statistical measures like the mean. We will use the Interquartile Range (IQR) method to detect outliers and replace them with the median value of the column to maintain data consistency.

```
boxplot(data$Age)
```



```
boxplot(data$Age)
quantile(data$Age)
x <- data$Age
x
iqr <- IQR(x)
iqr
lower_bound <-  22 -  1.5 * iqr
upper_bound <-  30 + 1.5 * iqr
x[x < lower_bound | x > upper_bound] <- mean(x)
x <- as.integer(round(x))
boxplot(x)
data$Age <- x
```

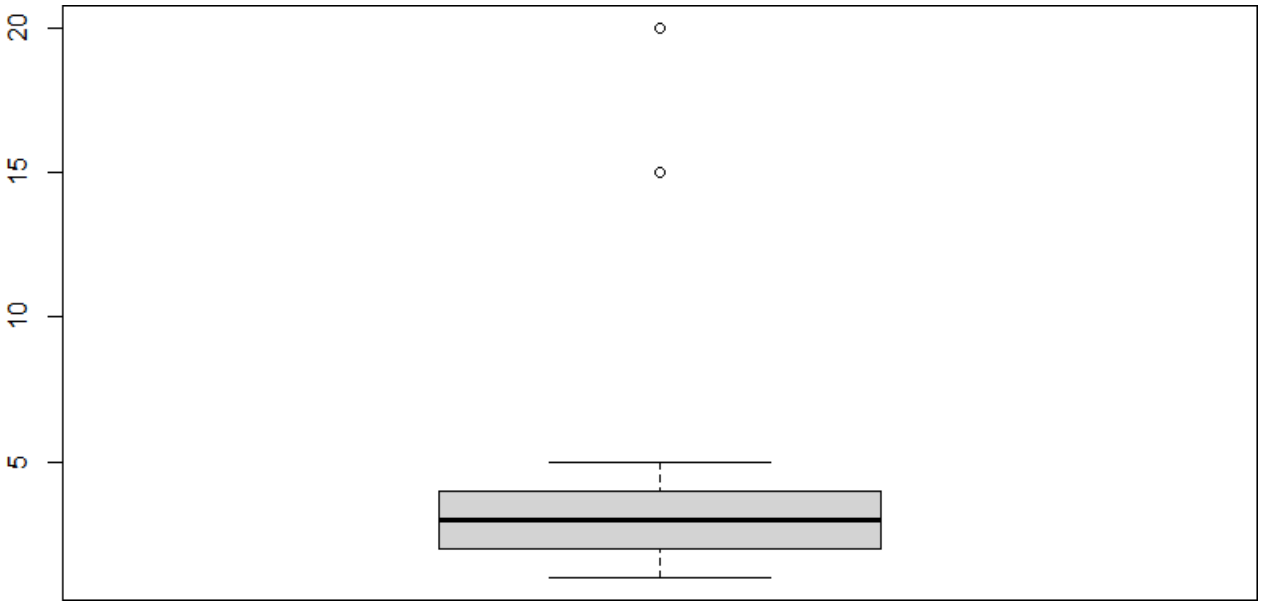


Code Description: The code first uses a boxplot to visualize the age distribution and detect possible outliers. The mean and quantiles are then calculated to understand data spread, and the interquartile range (IQR) is computed. Using the IQR, lower and upper bounds are set (Q1 – 1.5×IQR and Q3 + 1.5×IQR). Any age values outside these limits are treated as outliers and replaced with the median. The values are then rounded to integers, a new boxplot is plotted to confirm the cleaning, and finally the updated ages are reassigned to the dataset.

Academic Pressure Column:

In this task, the goal is to handle outliers in the *Academic Pressure* column of the dataset. Outliers are detected using the Interquartile Range (IQR) method, where values below the lower bound or above the upper bound are considered outliers. These outliers are then replaced with the median value to make the data cleaner and more reliable for further analysis.

```
boxplot (data$`Academic Pressure`)
```



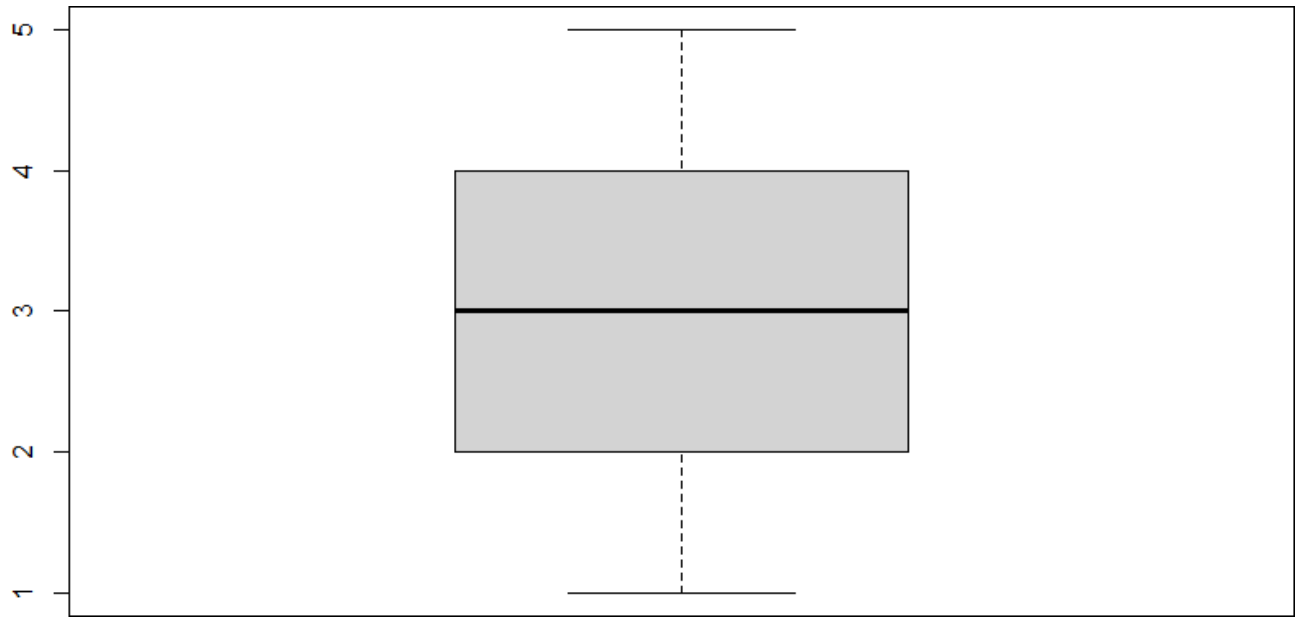

```
boxplot(data$`Academic Pressure`)
quantile(data$`Academic Pressure`)

y <- data$`Academic Pressure`
y

iqr <- IQR(y)
iqr

lower_bound_y <-  2 -  1.5 * iqr
upper_bound_y <-  4 + 1.5 * iqr

y[y < lower_bound_y | y > upper_bound_y] <- median(y)
y <- as.integer(round(y))
boxplot(y)
data$`Academic Pressure` <- y
```



Code Description: The main function used in this task is the IQR() function, which calculates the Interquartile Range and helps to determine the spread of the data by finding the difference between the third quartile (Q3) and the first quartile (Q1). Based on this IQR, the code sets the lower and upper bounds to identify outliers in the *Academic Pressure* column. Another important part is the replacement step y[y < lower_bound_y | y > upper_bound_y] <- median(y), where all detected outliers are replaced with the median value of the data. This ensures that extreme values do not distort the analysis, while still maintaining the overall central tendency of the dataset.

Noisy Value Checking and Handling:

Suicidal Thought column:

The task is to handle noisy values in the dataset column *"Have you ever had suicidal thoughts ? "*. The column contains inconsistent entries such as "yes", "yess", "no", and "noo", which can cause errors in analysis. To solve this problem, we first visualize the noisy values using a bar plot and then standardize them into consistent categories ("Yes" and "No") using data cleaning techniques.

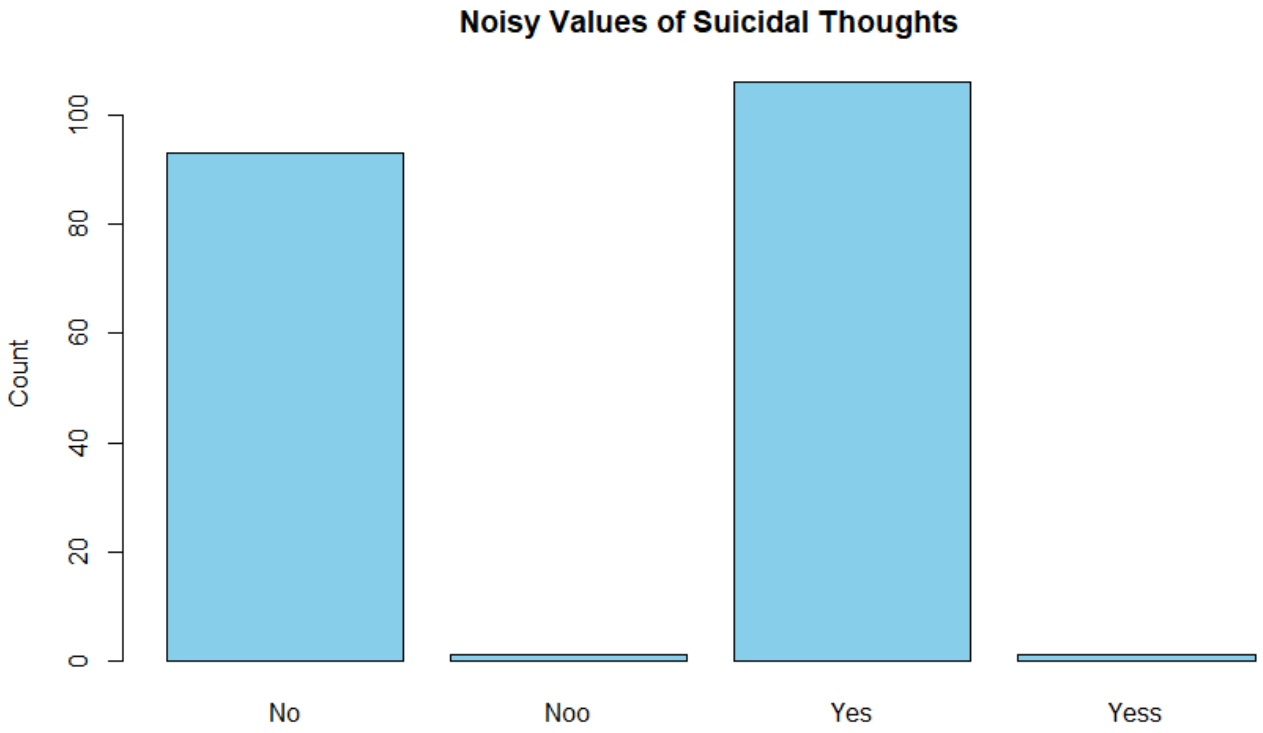
```
unique(data$`Have you ever had suicidal thoughts ?`)

barplot(table(data$`Have you ever had suicidal thoughts ?`),
        main = "Noisy Values of Suicidal Thoughts",
        col = "skyblue",
        ylab = "Count")

data <- data %>%
  mutate(suicidal_thoughts_clean = case_when(
    tolower(`Have you ever had suicidal thoughts ?`) %in% c("yes", "yess") ~ "Yes",
    tolower(`Have you ever had suicidal thoughts ?`) %in% c("no", "noo") ~ "No"
  ))

data$`Have you ever had suicidal thoughts ?` <-data$suicidal_thoughts_clean

data$suicidal_thoughts_clean <- NULL
```



Code Description: The code first uses unique() and barplot(table(...)) to visualize and identify noisy values in the *Suicidal Thoughts* column. Then, mutate() along with case_when() is applied to clean the data by mapping similar values (like "yes" and "yess") to a standard category ("Yes") and ("no", "noo") to ("No"). Finally, the cleaned column replaces the original one, and the temporary column is removed. This ensures that the dataset is free from inconsistencies and can be used reliably for analysis.

Categorical to Numerical Convertion:

Gender Column:

In this task, we aim to convert categorical variables into numerical values. Categorical data (like "Male"/"Female") cannot be directly used in mathematical calculations, so we encode them into numbers. we will use the case_when() function from the **dplyr** package to map categorical values to numerical values.

```
data <- data %>%
  mutate(gender_numeric = case_when(
    tolower(Gender) == "male" ~ 1,
    tolower(Gender) == "female" ~ 0
  ))
data$Gender <- data$gender_numeric
data$gender_numeric <- NULL
table(data$Gender)
```

```
> data <- data %>%
+   mutate(gender_numeric = case_when(
+     tolower(Gender) == "male" ~ 1,
+     tolower(Gender) == "female" ~ 0
+   ))
> data$Gender <- data$gender_numeric
> data$gender_numeric <- NULL
> table(data$Gender)

  0   1
 88 113
>
```

Code Description: The dataset was created using data.frame() with ID and Gender columns. The mutate () and case_when() functions converted Gender values, mapping Male to 1 and Female to 0, while tolower() handled case sensitivity. The original Gender column was then replaced with numeric values, verified using is.numeric(), and finally displayed with print(data).

Family History of Mental Illness Column:

This task converts the categorical values in the Family History of Mental Illness column into numerical form. Since the column contains "Yes" and "No" responses, we encode "Yes" as 1 and "No" as 0. This makes the data suitable for statistical analysis that requires numeric inputs.

```
data <- data %>%
  mutate(history_numeric = case_when(
    tolower(`Family History of Mental Illness`) == "yes" ~ 1,
    tolower(`Family History of Mental Illness`) == "no" ~ 0
  ))
data$`Family History of Mental Illness` <- data$history_numeric
data$history_numeric <- NULL
table(data$`Family History of Mental Illness`)
```

```
> data <- data %>%
+   mutate(history_numeric = case_when(
+     tolower(`Family History of Mental Illness`) == "yes" ~ 1,
+     tolower(`Family History of Mental Illness`) == "no" ~ 0
+   ))
> data$`Family History of Mental Illness` <- data$history_numeric
> data$history_numeric <- NULL
> table(data$`Family History of Mental Illness`)

  0   1
108  93
>
```

Code Description: The column *Family History of Mental Illness* was transformed into numeric values using mutate() and case_when(), where "Yes" was mapped to 1 and "No" to 0. The tolower() function ensured that case differences did not cause mismatches. The original column was then replaced with these numeric values, and the table() function was used to display the frequency distribution of responses.

Normalization:

Study Hours Column:

This task applies normalization to the *Study Hours* column so that all values fall within the range [0,1]. Normalization is often used in data preprocessing to bring different scales of numerical variables into a comparable range, which helps improve performance of machine learning algorithms. We will use the min–max normalization method.

```
data <- data %>%
 mutate(StudyHours_normalized = (`Study Hours` - min(`Study Hours`)) /
(max(`Study Hours`, na.rm = TRUE) - min(`Study Hours`)))
data$`Study Hours`<- data$StudyHours_normalized
data$StudyHours_normalized <- NULL
```

```
> data$`Study Hours`
  [1] 0.75000000 0.58333333 0.83333333 0.58333333 0.33333333 0.08333333 0.50000000 0.25000000
  [9] 0.83333333 0.83333333 0.50000000 0.91666667 0.16666667 1.00000000 0.25000000 0.66666667
 [17] 0.50000000 0.00000000 0.16666667 0.08333333 0.25000000 0.08333333 0.83333333 0.91666667
 [25] 1.00000000 0.16666667 0.00000000 0.50000000 0.33333333 1.00000000 0.16666667 0.25000000
 [33] 1.00000000 0.75000000 0.83333333 0.25000000 0.16666667 0.33333333 0.66666667 0.66666667
 [41] 0.50000000 0.50000000 0.08333333 1.00000000 0.25000000 0.91666667 0.91666667 1.00000000
 [49] 0.08333333 1.00000000 0.66666667 0.58333333 0.83333333 0.83333333 0.75000000 0.66666667
 [57] 1.00000000 0.33333333 0.41666667 0.08333333 0.50000000 0.00000000 0.50000000 0.25000000
 [65] 0.41666667 0.91666667 0.66666667 0.41666667 0.91666667 0.66666667 0.66666667 0.41666667
 [73] 0.00000000 0.66666667 0.91666667 0.66666667 0.83333333 0.16666667 0.83333333 0.50000000
 [81] 0.25000000 0.66666667 0.58333333 0.25000000 0.83333333 0.00000000 0.75000000 0.25000000
 [89] 0.75000000 0.83333333 0.33333333 0.58333333 0.58333333 0.66666667 0.83333333 0.41666667
 [97] 0.00000000 0.58333333 0.91666667 0.41666667 0.00000000 0.66666667 0.58333333 0.33333333
[105] 0.41666667 0.75000000 0.83333333 0.91666667 0.25000000 0.58333333 0.75000000 0.00000000
[113] 0.33333333 0.66666667 0.16666667 0.66666667 0.83333333 0.83333333 0.16666667 0.16666667
[121] 0.00000000 0.00000000 0.50000000 0.83333333 0.83333333 0.50000000 0.25000000 0.33333333
[129] 0.91666667 1.00000000 0.41666667 0.33333333 0.58333333 0.75000000 0.00000000 1.00000000
[137] 0.50000000 1.00000000 0.50000000 1.00000000 0.41666667 0.75000000 0.16666667 0.66666667
[145] 0.16666667 0.08333333 0.91666667 0.91666667 1.00000000 0.66666667 0.00000000 0.66666667
[153] 0.00000000 0.25000000 0.83333333 1.00000000 0.41666667 0.66666667 0.50000000 0.33333333
[161] 0.33333333 1.00000000 0.50000000 0.08333333 0.25000000 0.08333333 0.41666667 0.83333333
[169] 0.00000000 0.75000000 1.00000000 0.58333333 1.00000000 1.00000000 0.33333333 0.16666667
[177] 0.25000000 0.58333333 0.16666667 0.75000000 0.66666667 0.50000000 0.16666667 1.00000000
[185] 0.83333333 0.25000000 0.25000000 0.00000000 0.00000000 0.91666667 0.66666667 0.25000000
[193] 0.16666667 0.83333333 1.00000000 0.83333333 0.25000000 0.58333333 0.41666667 0.75000000
[201] 0.16666667
```

Code Description:

A custom function mutate () was defined to perform min–max scaling. It subtracts the minimum value from each data point and then divides by the range (max − min), ensuring all values lie between 0 and 1. This function was applied to the *Study Hours* column, and the normalized values were stored in a new column named StudyHours_normalized. Finally, head() was used to display a few normalized values.

Data Filtering:

In this task, we filter the dataset according to specific conditions to focus on particular subsets of students. Filtering helps to extract meaningful groups of data for deeper analysis. Different logical operators (>, <, &, |, ==) are used to apply these conditions.

```
filtered_data1 <- data %>%
  filter(`Study Hours` > 10 | Age < 20)
```

```
> str(filtered_data1)
tibble [18 × 11] (S3: tbl_df/tbl/data.frame)
 $ Gender                                : num [1:18] 1 1 1 1 0 1 0 1 1 0 ...
 $ Age                                   : int [1:18] 19 19 19 18 18 19 19 18 18 19 ...
 $ Academic Pressure                     : int [1:18] 4 2 3 3 3 2 1 5 3 4 ...
 $ Study Satisfaction                    : num [1:18] 4 4 2 4 4 5 1 5 4 3 ...
 $ Sleep Duration                        : chr [1:18] "5-6 hours" "Less than 5 hours" "More than 8 h
ours" "7-8 hours" ...
 $ Dietary Habits                        : chr [1:18] "Unhealthy" "Moderate" "Unhealthy" "Unhealthy"
...
 $ Have you ever had suicidal thoughts ?: chr [1:18] "Yes" "Yes" "Yes" "Yes" ...
 $ Study Hours                           : num [1:18] 0.0833 0.4167 0.25 0.75 0.75 ...
 $ Financial Stress                      : num [1:18] 4 4 1 3 2 1 3 4 1 2 ...
 $ Family History of Mental Illness      : num [1:18] 1 1 0 1 0 1 0 0 0 1 ...
 $ Depression                            : chr [1:18] "Yes" "Yes" "Yes" "Yes" ...
>
```

```
filtered_data2 <- data %>%
  filter(`Financial Stress` > 3 & Age < 20)
```

```
> str(filtered_data2)
tibble [3 × 11] (S3: tbl_df/tbl/data.frame)
 $ Gender                                : num [1:3] 1 1 1
 $ Age                                   : int [1:3] 19 19 18
 $ Academic Pressure                     : int [1:3] 4 2 5
 $ Study Satisfaction                    : num [1:3] 4 4 5
 $ Sleep Duration                        : chr [1:3] "5-6 hours" "Less than 5 hours" "5-6 hours"
 $ Dietary Habits                        : chr [1:3] "Unhealthy" "Moderate" "Moderate"
 $ Have you ever had suicidal thoughts ?: chr [1:3] "Yes" "Yes" "No"
 $ Study Hours                           : num [1:3] 0.0833 0.4167 0.25
 $ Financial Stress                      : num [1:3] 4 4 4
 $ Family History of Mental Illness      : num [1:3] 1 1 0
 $ Depression                            : chr [1:3] "Yes" "Yes" "Yes"
>
```

```
filtered_data3 <- data %>%
  filter(`Have you ever had suicidal thoughts ?` == 'Yes' & `Financial Stress` > 3)
```

```
> str(filtered_data3)
tibble [38 × 11] (S3: tbl_df/tbl/data.frame)
 $ Gender                                : num [1:38] 1 1 0 1 1 1 0 0 1 0 ...
 $ Age                                   : int [1:38] 25 19 20 31 24 21 20 33 26 21 ...
 $ Academic Pressure                     : int [1:38] 1 4 4 5 2 5 5 2 5 3 ...
 $ Study Satisfaction                    : num [1:38] 3 4 1 4 1 1 5 3 1 3 ...
 $ Sleep Duration                        : chr [1:38] "5-6 hours" "5-6 hours" "More than 8 hours" "5
-6 hours" ...
 $ Dietary Habits                        : chr [1:38] "Unhealthy" "Unhealthy" "Healthy" "Healthy"
...
 $ Have you ever had suicidal thoughts ?: chr [1:38] "Yes" "Yes" "Yes" "Yes" ...
 $ Study Hours                           : num [1:38] 0.8333 0.0833 0.25 0.5 0.9167 ...
 $ Financial Stress                      : num [1:38] 4 4 4 4 5 5 5 5 5 5 ...
 $ Family History of Mental Illness      : num [1:38] 0 1 1 0 0 1 0 1 0 1 ...
 $ Depression                            : chr [1:38] "Yes" "Yes" "Yes" "Yes" ...
>
```

Code Description: The filtering was done using the filter() function from dplyr. In the first filter, students who either study more than 10 hours or are younger than 20 were selected using the OR (|) operator. The second filter extracted students with financial stress greater than 3 and age below 20 using the AND (&) operator. The third filter selected students who reported having suicidal thoughts and also had financial stress above 3. These filtered subsets help in targeted analysis of specific conditions.

Data Balancing:

Sometimes, categorical variables such as Gender may be imbalanced, meaning one category (e.g., Male) has many more samples than the other (e.g., Female). This imbalance can bias results in analysis and modeling. To address this, we use random undersampling, where we take an equal number of samples from each group. This ensures a balanced dataset with fair representation of both genders.

```
balanced_data <- data %>%
  group_by(Gender) %>%
  slice_sample(n = min(table(data$Gender)), replace = FALSE) %>%
  ungroup()
```

	Gender	n	percentage
1	0	88	50
2	1	88	50

Code Description: The data was grouped by Gender using group_by(). Within each group, slice_sample() randomly selected a number of rows equal to the size of the smaller group (using min(table(data$Gender))). This undersampling step ensured that both Male and Female categories had the same number of observations. Finally, ungroup () removed the grouping, and the new distribution was verified with table().

Split Train and Test data:

In this task, the dataset is divided into two subsets: training data and testing data. Splitting ensures that the model is tested on unseen data, which helps in checking its generalization ability.

```
set.seed(123)
train_data <- data %>%
  sample_frac(0.8)

test_data <- data %>%
  anti_join(train_data)
```

```
> str(train_data)
tibble [161 × 11] (S3: tbl_df/tbl/data.frame)
 $ Gender                                : num [1:161] 0 1 1 1 1 1 1 1 1 1 ...
 $ Age                                   : int [1:161] 32 27 25 34 27 24 22 21 26 19 ...
 $ Academic Pressure                     : int [1:161] 2 2 1 2 3 2 4 3 4 5 ...
 $ Study Satisfaction                    : num [1:161] 3 5 1 3 1 4 2 2 3 1 ...
 $ Sleep Duration                        : chr [1:161] "7-8 hours" "Less than 5 hours" "5-6 hours"
"5-6 hours" ...
 $ Dietary Habits                        : chr [1:161] "Moderate" "Moderate" "Moderate" "Moderate"
...
 $ Have you ever had suicidal thoughts ?: chr [1:161] "No" "No" "Yes" "No" ...
 $ Study Hours                           : num [1:161] 0.5 0.167 1 1 0.75 ...
 $ Financial Stress                      : num [1:161] 1 5 3 2 5 4 1 5 4 3 ...
 $ Family History of Mental Illness      : num [1:161] 0 1 1 0 1 1 1 0 1 0 ...
 $ Depression                            : chr [1:161] "No" "No" "Yes" "No" ...
> str(test_data)
tibble [39 × 11] (S3: tbl_df/tbl/data.frame)
 $ Gender                                : num [1:39] 1 1 0 0 1 1 0 1 1 0 ...
 $ Age                                   : int [1:39] 33 21 23 20 21 32 24 22 20 30 ...
 $ Academic Pressure                     : int [1:39] 4 5 1 5 5 5 1 1 4 5 ...
 $ Study Satisfaction                    : num [1:39] 3 1 3 5 3 2 3 5 2 3 ...
 $ Sleep Duration                        : chr [1:39] "Less than 5 hours" "More than 8 hours" "More
than 8 hours" "More than 8 hours" ...
 $ Dietary Habits                        : chr [1:39] "Unhealthy" "Unhealthy" "Healthy" "Unhealthy"
...
 $ Have you ever had suicidal thoughts ?: chr [1:39] "Yes" "Yes" "Yes" "Yes" ...
 $ Study Hours                           : num [1:39] 0.833 0.25 0 0.167 0.5 ...
 $ Financial Stress                      : num [1:39] 1 5 3 5 4 3 5 4 4 2 ...
 $ Family History of Mental Illness      : num [1:39] 0 1 0 0 1 0 0 0 1 0 ...
 $ Depression                            : chr [1:39] "Yes" "Yes" "No" "Yes" ...
>
```

Code Description: A random seed was set using set.seed(123) to ensure consistent results each time the code runs. The sample_frac(0.8) function selected 80% of the rows randomly for the training set. The remaining 20% of the data was obtained using anti_join(), which removes all rows already included in the training set.

Central Tendencies:

In this task, we analyze the central tendencies of the dataset to understand the typical values of each variable. For categorical variables we calculate the mode to identify the most frequent category. For numerical variables we calculate the mean and median to understand the average and middle values of the data.

Sleep Duration:

```
> sleep_duration_mode <-  names(sort(table(data$`Sleep Duration`), decreasing = TRUE))[1]
> sleep_duration_mode
[1] "More than 8 hours"
>
```

Dietary Habit:

```
> habit_mode <- names(sort(table(data$`Dietary Habits`), decreasing = TRUE))[1]
> habit_mode
[1] "Unhealthy"
>
```

Age:

```
> mean_age <- mean(data$Age)
> mean_age
[1] 26.24876
>
```

Academic Pressure:

```
> median_Academic_pressure <- median(data$`Academic Pressure`)
> median_Academic_pressure
[1] 3
>
```

```
summary_table <- data.frame(
  Statistic = c("Sleep Duration (Mode)",
                "Dietary Habits (Mode)",
                "Age (Mean)",
                "Academic Pressure (Median)"),
  Value = c(sleep_duration_mode,
            habit_mode,
            round(mean_age, 2),
            round(median_Academic_pressure, 2))
)
```

```
> print(summary_table)
                  Statistic             Value
1      Sleep Duration (Mode) More than 8 hours
2      Dietary Habits (Mode)         Unhealthy
3                Age (Mean)             26.26
4 Academic Pressure (Median)                 3
```

Code Description: For numerical variables like Age and Study Hours, mean and median were calculated using the built-in mean() and median() functions to describe the central location of the data And made a summary_table to show and store those values.

Compute the Spread:

We have to compute the spread (Range, IQR, Variance, Standard deviation) for any two attributes. We are calculating for Study Hours and Age Column only.

```
study_hours_spread <- data %>%
  summarise(
    Attribute = "Study Hours",
    Min = min(`Study Hours`),
    Max = max(`Study Hours`),
    Range = Max - Min,
    IQR = IQR(`Study Hours`),
    Variance = var(`Study Hours`),
    Std_Dev = sd(`Study Hours`)
  )

age_spread <- data %>%
  summarise(
    Attribute = "Age",
    Min = min(Age),
    Max = max(Age),
    Range = Max - Min,
    IQR = IQR(Age),
    Variance = var(Age),
    Std_Dev = sd(Age)
  )

spread_results <- bind_rows(study_hours_spread, age_spread)
```

```
> print(spread_results)
# A tibble: 2 x 7
  Attribute     Min   Max Range   IQR Variance Std_Dev
  <chr>       <dbl> <dbl> <dbl> <dbl>    <dbl>   <dbl>
1 Study Hours     0     1     1 0.583   0.0996   0.316
2 Age            18    34    16 8      22.8      4.77
>
```

Code Description: This code computes spread metrics for two attributes (Study Hours and Age) in a dataset and combines the results into a single table.

Project Code

```
install.packages(c("readxl", "dplyr"))

library(readxl)
library(dplyr)



data <- read_excel("E:\\Data Science\\Dataset\\Depression Student Dataset.xlsx")

str (data)
summary(data)
colSums(is.na(data))


show(data$Age_norm)
class(data$Gender)
data$Gender
data$Age
unique(data)
mean(data$Age)
mode(data$Age)

data[!complete.cases(data),]



missing_summary <- colSums(is.na(data))
missing_summary
colSums(is.na(data))



data$Age[is.na(data$Age)] <- mean(data$Age, na.rm = TRUE)
data$Age
data$Age <- as.integer(round(data$Age))
data[!complete.cases(data),]
head(data)


table(data$Gender)
mode_gender <- names(sort(table(data$Gender), decreasing = TRUE))[1]
data$Gender[is.na(data$Gender)] <- mode_gender
data[!complete.cases(data),]


unique(data$`Sleep Duration`)
mode_sleep <- names(sort(table(data$`Sleep Duration`), decreasing = TRUE))[1]
data$`Sleep Duration`[is.na(data$`Sleep Duration`)] <- mode_sleep
data[!complete.cases(data),]


hist(data$`Study Hours`)
data$`Study Hours`[is.na(data$`Study Hours`)] <- mean(data$`Study Hours`, na.rm =
TRUE)
data$`Study Hours` <- as.integer(round(data$`Study Hours`))
data[!complete.cases(data),]
```

```
mode_depression <- names(sort(table(data$Depression), decreasing = TRUE))[1]
data$Depression[is.na(data$Depression)] <- mode_depression
data[!complete.cases(data),]



boxplot(data$Age)
mean(data$Age)
quantile(data$Age)

x <- data$Age
x
iqr <- IQR(x)
iqr

lower_bound <-  22 -  1.5 * iqr
upper_bound <-  30 + 1.5 * iqr

x[x < lower_bound | x > upper_bound] <- mean(x)
x <- as.integer(round(x))

boxplot(x)
data$Age <- x



boxplot(data$`Academic Pressure`)
quantile(data$`Academic Pressure`)

y <- data$`Academic Pressure`
y

iqr <- IQR(y)
iqr

lower_bound_y <-  2 -  1.5 * iqr
upper_bound_y <-  4 + 1.5 * iqr

y[y < lower_bound_y | y > upper_bound_y] <- median(y)
y <- as.integer(round(y))
boxplot(y)
data$`Academic Pressure` <- y



unique(data$`Have you ever had suicidal thoughts ?`)
barplot(table(data$`Have you ever had suicidal thoughts ?`),
        main = "Noisy Values of Suicidal Thoughts",
        col = "skyblue",
        ylab = "Count")

data <- data %>%
  mutate(suicidal_thoughts_clean = case_when(
    tolower(`Have you ever had suicidal thoughts ?`) %in% c("yes", "yess") ~ "Yes",
    tolower(`Have you ever had suicidal thoughts ?`) %in% c("no", "noo") ~ "No"
  ))
data$`Have you ever had suicidal thoughts ?` <- data$suicidal_thoughts_clean
data$suicidal_thoughts_clean <- NULL
```

```
data <- data %>%
  mutate(gender_numeric = case_when(
    tolower(Gender) == "male" ~ 1,
    tolower(Gender) == "female" ~ 0
  ))
data$Gender <- data$gender_numeric
data$gender_numeric <- NULL
table(data$Gender)


data <- data %>%
  mutate(history_numeric = case_when(
    tolower(`Family History of Mental Illness`) == "yes" ~ 1,
    tolower(`Family History of Mental Illness`) == "no" ~ 0
  ))
data$`Family History of Mental Illness` <- data$history_numeric
data$history_numeric <- NULL
table(data$`Family History of Mental Illness`)


data <- data %>%
  mutate(StudyHours_normalized = (`Study Hours` - min(`Study Hours`)) /
           (max(`Study Hours`, na.rm = TRUE) - min(`Study Hours`)))
data$`Study Hours`<- data$StudyHours_normalized
data$StudyHours_normalized <- NULL
data$`Study Hours`




filtered_data1 <- data %>%
  filter(`Study Hours` > 10 | Age < 20)
str(filtered_data1)

filtered_data2 <- data %>%
  filter(`Financial Stress` > 3 & Age < 20)
str(filtered_data2)

filtered_data3 <- data %>%
  filter(`Have you ever had suicidal thoughts ?` == 'Yes' & `Financial Stress` > 3)
str(filtered_data3)



class_distribution <- data %>%
  count(Gender) %>%
  mutate(percentage = n / sum(n) * 100)

balanced_data <- data %>%
  group_by(Gender) %>%
  slice_sample(n = min(table(data$Gender)), replace = FALSE) %>%
  ungroup()

class_distribution_balanced <- balanced_data %>%
  count(Gender) %>%
  mutate(percentage = n / sum(n) * 100)




set.seed(123)
train_data <- data %>%
```

```
  sample_frac(0.8)

test_data <- data %>%
  anti_join(train_data)
str(train_data)
str(test_data)


sleep_duration_mode <-  names(sort(table(data$`Sleep Duration`), decreasing =
TRUE))[1]
sleep_duration_mode

habit_mode <- names(sort(table(data$`Dietary Habits`), decreasing = TRUE))[1]
habit_mode

mean_age <- mean(data$Age)
mean_age

median_Academic_pressure <- median(data$`Academic Pressure`)
median_Academic_pressure

summary_table <- data.frame(
  Statistic = c("Sleep Duration (Mode)",
                "Dietary Habits (Mode)",
                "Age (Mean)",
                "Academic Pressure (Median)"),
  Value = c(sleep_duration_mode,
            habit_mode,
            round(mean_age, 2),
            round(median_Academic_pressure, 2))
)

print(summary_table)



study_hours_spread <- data %>%
  summarise(
    Attribute = "Study Hours",
    Min = min(`Study Hours`),
    Max = max(`Study Hours`),
    Range = Max - Min,
    IQR = IQR(`Study Hours`),
    Variance = var(`Study Hours`),
    Std_Dev = sd(`Study Hours`)
  )


age_spread <- data %>%
  summarise(
    Attribute = "Age",
    Min = min(Age),
    Max = max(Age),
    Range = Max - Min,
    IQR = IQR(Age),
    Variance = var(Age),
    Std_Dev = sd(Age)
  )

spread_results <- bind_rows(study_hours_spread, age_spread)
print(spread_results)
```