MENTAL HEALTH - EXPLORATORY DATA ANALYSIS , CLASSIFICATION AND REGRESSION

Presented by :

22BRS1021 S Mukunth

22BRS1095 Kishore K G

22BRS1099 Aravind N

22BRS1357 Abhinav Balakrishnan

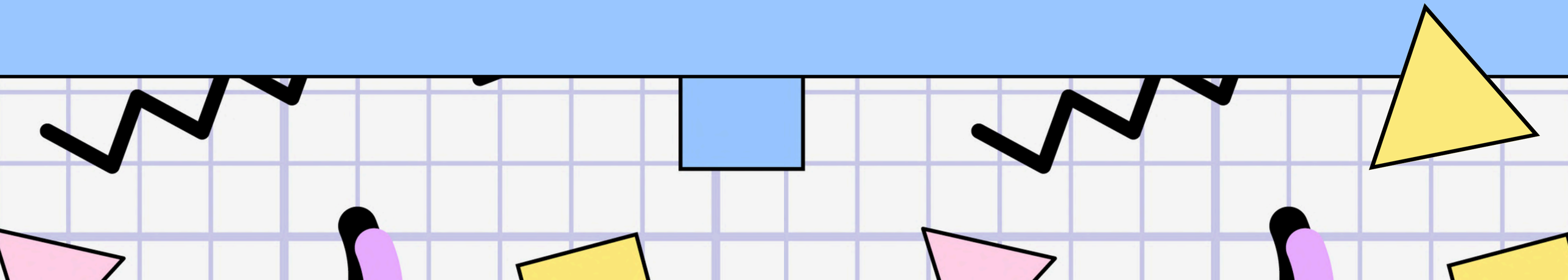
BMAT202L - Probability & Statistics

Slot: E1+TE1

Submitted to : Dr. Amit Kumar Rahul

AIM

The aim of this project is to perform a comprehensive Exploratory Data Analysis (EDA) on technology usage and mental health data, aiming to understand relationships between daily screen time, stress levels, sleep quality, and self-reported mental health. Through this analysis, we aim to uncover patterns that may impact mental well-being, such as the effect of prolonged screen time or poor sleep quality on mental health scores. Following EDA, we will develop a predictive model to estimate mental health scores based on these variables, offering a data-driven perspective to help inform strategies for better mental health management.



ABOUT THE DATASET

This dataset provides a detailed view of individual daily technology usage and mental health indicators. Each record includes a unique identifier (User_ID), age, average daily screen time in hours, a self-reported mental health score (1-4), a self-reported stress level (1-3).

These factors allow for in-depth analysis of potential correlations and influences between technology habits and mental health outcomes. This dataset is particularly valuable for studying behavioral patterns, identifying high-risk factors for mental health issues, and creating a predictive model that estimates mental health scores, potentially serving as a tool for mental health researchers, practitioners, and policymakers.

OBJECTIVE

To analyze the relationship between technology usage patterns (screen time, gaming hours) and mental health indicators through statistical distributions and visualization techniques, establishing significant correlations between digital behavior and psychological well-being

To develop and compare multiple predictive models (Linear Regression, Random Forest, XGBoost) for forecasting sleep scores based on technology usage metrics, identifying the most accurate model for prediction

To evaluate the effectiveness of various classification algorithms in categorizing mental health states, utilizing feature importance analysis to determine which technological and behavioral factors have the strongest impact on mental health outcomes

DATASET ANALYSIS

Importing Libraries and Dataset

```r
# Load necessary libraries
library(caret)
library(dplyr)
library(fastDummies)
library(caTools)
library(randomForest)
library(xgboost)
library(readr)
library(ggplot2)
library(rsample)
library(tidyverse)
library(skimr)
library(moments)
library(ggplot2)
library(dplyr)
library(gmodels)
```

```r
library(scales)
library(nortest)
library(fitdistrplus)
library(MASS)
library(gridExtra)
library(caret)
library(randomForest)
library(xgboost)
library(nnet)
library(e1071)
library(ROSE)
library(rpart)
library(gbm)
library(MLmetrics)
library(kernlab)
```

```r
# Read the CSV file
data <- read.csv("C:/Users/kisho/Desktop/prob-project/mental_health_with_sleep_quality.csv")
```

DATASET ANALYSIS

Dataset Values

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

```
##      User_ID Age Gender Technology_Usage_Hours Social_Media_Usage_Hours
## 1 USER-00001  23 Female                  6.57                     6.00
## 2 USER-00002  21   Male                  3.01                     2.57
## 3 USER-00003  51   Male                  3.04                     6.14
## 4 USER-00004  25 Female                  3.84                     4.48
## 5 USER-00005  53   Male                  1.20                     0.56
## 6 USER-00006  58   Male                  5.59                     5.74
##   Gaming_Hours Screen_Time_Hours Mental_Health_Status Stress_Level Sleep_Hours
## 1         0.68             12.36                Good            1        8.01
## 2         3.74              7.61                Poor            3        7.28
## 3         1.26              3.16                Fair            3        8.04
## 4         2.59             13.08           Excellent            2        5.62
## 5         0.29             12.63                Good            1        5.55
## 6         0.11              1.34                Poor            1        8.61
```

```
##   Physical_Activity_Hours Support_Systems_Access Work_Environment_Impact
## 1                    6.71                     No                Negative
## 2                    5.88                    Yes                Positive
## 3                    9.81                     No                Negative
## 4                    5.28                    Yes                Negative
## 5                    4.00                     No                Positive
## 6                    6.54                    Yes                 Neutral
##   Online_Support_Usage Sleep_Quality
## 1                  Yes          Good
## 2                   No          Fair
## 3                   No          Good
## 4                  Yes          Poor
## 5                  Yes          Fair
## 6                  Yes          Good
```

DATASET ANALYSIS

Dataset Overview

```
# 1. Dataset dimensions
cat("Dataset Dimensions: ", paste(dim(data), collapse = " x "), "\n")
```

```
## Dataset Dimensions:  10000 x 15
```

```
# 2. Variable types
cat("Variable Types:\n", paste(names(data), sapply(data, class), sep = ": ", collapse = "\n"), "\n")
```

```
## Variable Types:
##  User_ID: character
## Age: integer
## Gender: character
## Technology_Usage_Hours: numeric
## Social_Media_Usage_Hours: numeric
## Gaming_Hours: numeric
## Screen_Time_Hours: numeric
## Mental_Health_Status: character
## Stress_Level: integer
## Sleep_Hours: numeric
## Physical_Activity_Hours: numeric
## Support_Systems_Access: character
## Work_Environment_Impact: character
## Online_Support_Usage: character
## Sleep_Quality: character
```

DATASET ANALYSIS

Cleaning Dataset

```r
# 3. Missing values analysis
missing_values <- colSums(is.na(data))
cat("Missing Values Analysis:\n", paste(names(missing_values), missing_values, sep = ": ", collapse = "\n"), "\n")
```

```
## Missing Values Analysis:
##  User_ID: 0
## Age: 0
## Gender: 0
## Technology_Usage_Hours: 0
## Social_Media_Usage_Hours: 0
## Gaming_Hours: 0
## Screen_Time_Hours: 0
## Mental_Health_Status: 0
## Stress_Level: 0
## Sleep_Hours: 0
## Physical_Activity_Hours: 0
## Support_Systems_Access: 0
## Work_Environment_Impact: 0
## Online_Support_Usage: 0
## Sleep_Quality: 0
```

```r
# Check for duplicate rows
duplicates <- sum(duplicated(data))
cat("Number of duplicate rows: ", duplicates, "\n")
```

```
## Number of duplicate rows:  0
```

DATASET ANALYSIS

Datatypes of Columns

```
print(skim(data))
```

```
## ── Data Summary ────────────────────────
##                            Values
## Name                       data
## Number of rows             10000
## Number of columns          15
## _________________________
## Column type frequency:
##   character                7
##   numeric                  8
## _________________________
## Group variables            None
##
```

```
## ── Variable type: character ──────────────────────────────────────────────────
##   skim_variable           n_missing complete_rate min max empty n_unique
## 1 User_ID                         0             1  10  10     0    10000
## 2 Gender                          0             1   4   6     0        3
## 3 Mental_Health_Status            0             1   4   9     0        4
## 4 Support_Systems_Access          0             1   2   3     0        2
## 5 Work_Environment_Impact         0             1   7   8     0        3
## 6 Online_Support_Usage            0             1   2   3     0        2
## 7 Sleep_Quality                   0             1   4   4     0        3
##   whitespace
## 1          0
## 2          0
## 3          0
## 4          0
## 5          0
## 6          0
## 7          0
```

DATASET ANALYSIS

Datatypes of Columns

```
##
## ── Variable type: numeric ──────────────────────────────────────────────────────
##   skim_variable            n_missing complete_rate  mean     sd p0   p25   p50
## 1 Age                              0             1 41.5  13.9   18 29    42
## 2 Technology_Usage_Hours           0             1  6.47  3.17   1  3.76  6.42
## 3 Social_Media_Usage_Hours         0             1  3.97  2.31   0  1.98  3.95
## 4 Gaming_Hours                     0             1  2.52  1.45   0  1.26  2.52
## 5 Screen_Time_Hours                0             1  7.98  4.04   1  4.52  7.9
## 6 Stress_Level                     0             1  2.00  0.816  1  1     2
## 7 Sleep_Hours                      0             1  6.50  1.45   4  5.26  6.5
## 8 Physical_Activity_Hours          0             1  5.00  2.91   0  2.49  4.99
```

```
##      p75 p100 hist
## 1 54       65 ▇▇▇▇▇
## 2  9.21    12 ▇▇▇▇▇
## 3  5.99     8 ▇▇▇▇▇
## 4  3.79     5 ▇▇▇▇▇
## 5 11.5     15 ▇▇▇▇▇
## 6  3        3 ▇▁▇▁▇
## 7  7.76     9 ▇▇▇▇▇
## 8  7.54    10 ▇▇▇▇▇
```

DATASET ANALYSIS

Datatypes of Columns

```
## 'data.frame':    10000 obs. of  15 variables:
##  $ User_ID                : chr  "USER-00001" "USER-00002" "USER-00003" "USER-00004" ...
##  $ Age                    : int  23 21 51 25 53 58 63 51 57 31 ...
##  $ Gender                 : chr  "Female" "Male" "Male" "Female" ...
##  $ Technology_Usage_Hours  : num  6.57 3.01 3.04 3.84 1.2 ...
##  $ Social_Media_Usage_Hours: num  6 2.57 6.14 4.48 0.56 5.74 2.55 4.1 4.11 7.23 ...
##  $ Gaming_Hours           : num  0.68 3.74 1.26 2.59 0.29 0.11 3.79 4.74 0.08 0.81 ...
##  $ Screen_Time_Hours      : num  12.36 7.61 3.16 13.08 12.63 ...
##  $ Mental_Health_Status   : chr  "Good" "Poor" "Fair" "Excellent" ...
##  $ Stress_Level           : int  1 3 3 2 1 1 2 2 2 3 ...
##  $ Sleep_Hours            : num  8.01 7.28 8.04 5.62 5.55 8.61 8.61 7.11 7.19 5.09 ...
##  $ Physical_Activity_Hours : num  6.71 5.88 9.81 5.28 4 6.54 1.34 5.27 5.22 0.47 ...
##  $ Support_Systems_Access  : chr  "No" "Yes" "No" "Yes" ...
##  $ Work_Environment_Impact : chr  "Negative" "Positive" "Negative" "Negative" ...
##  $ Online_Support_Usage    : chr  "Yes" "No" "No" "Yes" ...
##  $ Sleep_Quality           : chr  "Good" "Fair" "Good" "Poor" ...
```

DATASET ANALYSIS

Central Tendency

```
  User_ID              Age           Gender          Technology_Usage_Hours
Length:10000       Min.   :18.00   Length:10000       Min.   : 1.000
Class :character   1st Qu.:29.00   Class :character   1st Qu.: 3.760
Mode  :character   Median :42.00   Mode  :character   Median : 6.425
                   Mean   :41.52                      Mean   : 6.474
                   3rd Qu.:54.00                      3rd Qu.: 9.213
                   Max.   :65.00                      Max.   :12.000
Social_Media_Usage_Hours  Gaming_Hours    Screen_Time_Hours
Min.   :0.000             Min.   :0.000   Min.   : 1.000
1st Qu.:1.980             1st Qu.:1.260   1st Qu.: 4.520
Median :3.950             Median :2.520   Median : 7.900
Mean   :3.972             Mean   :2.516   Mean   : 7.976
3rd Qu.:5.990             3rd Qu.:3.790   3rd Qu.:11.500
Max.   :8.000             Max.   :5.000   Max.   :15.000
Mental_Health_Status  Stress_Level  Sleep_Hours     Physical_Activity_Hours
Length:10000          Min.   :1     Min.   :4.000   Min.   : 0.000
Class :character      1st Qu.:1     1st Qu.:5.260   1st Qu.: 2.490
Mode  :character      Median :2     Median :6.500   Median : 4.990
                      Mean   :2     Mean   :6.501   Mean   : 5.004
                      3rd Qu.:3     3rd Qu.:7.760   3rd Qu.: 7.540
                      Max.   :3     Max.   :9.000   Max.   :10.000
```

```
Support_Systems_Access Work_Environment_Impact Online_Support_Usage
Length:10000           Length:10000            Length:10000
Class :character       Class :character        Class :character
Mode  :character       Mode  :character        Mode  :character



Sleep_Quality
Length:10000
Class :character
Mode  :character
```

DISPERSION MEASUREMENT

```r
# Dispersion Measurement
calculate_dispersion <- function(x) {
  q <- quantile(x, probs = c(0.25, 0.5, 0.75), na.rm = TRUE)
  c(
    Range = diff(range(x, na.rm = TRUE)),
    Variance = var(x, na.rm = TRUE),
    SD = sd(x, na.rm = TRUE),
    CV = sd(x, na.rm = TRUE) / mean(x, na.rm = TRUE) * 100,
    Q1 = q[1],
    Median = q[2],
    Q3 = q[3],
    IQR = IQR(x, na.rm = TRUE)
  )
}

# Calculate dispersion measures
dispersion_measures <- t(sapply(numeric_data, calculate_dispersion))
cat("\nMeasures of Dispersion for Numeric Variables:\n")
```

```
##                          Range    Variance        SD       CV Q1.25%
## Age                         47 193.7724313 13.9202166 33.52766   29.00
## Technology_Usage_Hours      11  10.0427030  3.1690224 48.94741    3.76
## Social_Media_Usage_Hours     8   5.3532384  2.3137066 58.24571    1.98
## Gaming_Hours                 5   2.0930812  1.4467485 57.51112    1.26
## Screen_Time_Hours           14  16.3426773  4.0426077 50.68614    4.52
## Stress_Level                 2   0.6662666  0.8162515 40.81666    1.00
## Sleep_Hours                  5   2.1052070  1.4509331 22.31956    5.26
## Physical_Activity_Hours     10   8.4392801  2.9050439 58.05606    2.49
##                          Median.50%  Q3.75%     IQR
## Age                          42.000 54.0000 25.0000
## Technology_Usage_Hours        6.425  9.2125  5.4525
## Social_Media_Usage_Hours      3.950  5.9900  4.0100
## Gaming_Hours                  2.520  3.7900  2.5300
## Screen_Time_Hours             7.900 11.5000  6.9800
## Stress_Level                  2.000  3.0000  2.0000
## Sleep_Hours                   6.500  7.7600  2.5000
## Physical_Activity_Hours       4.990  7.5400  5.0500
```

VISUAL INSPECTION

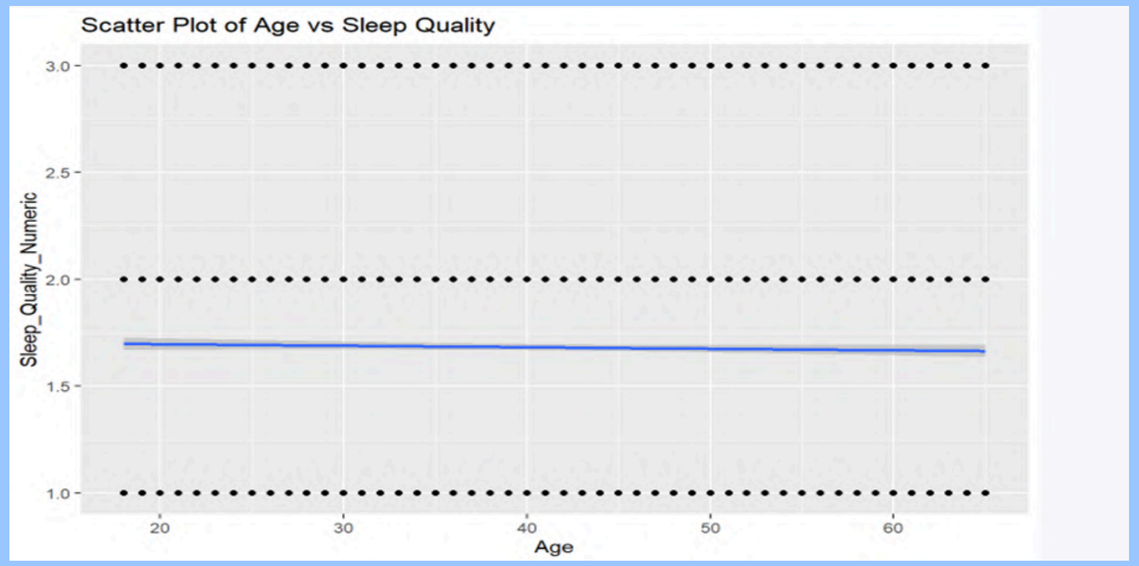
Categorical and Numerical Data Analysis

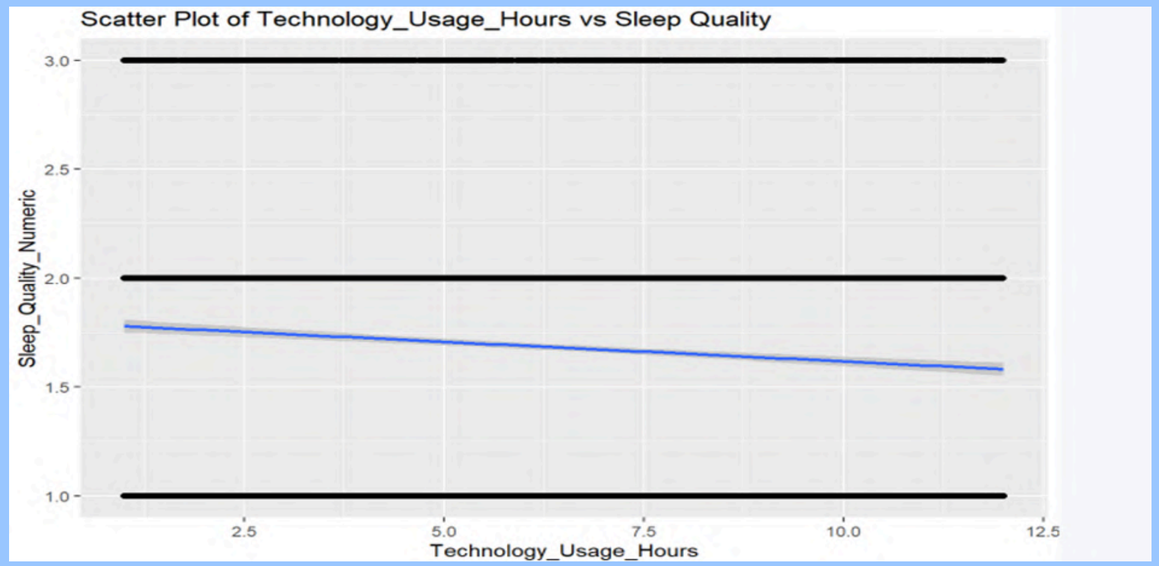
```
# Categorical and Numerical Data Analysis
data$Sleep_Quality_Numeric <- as.numeric(factor(data$Sleep_Quality, levels = c("Poor", "Fair", "Good")))
numerical_vars <- c("Age", "Technology_Usage_Hours", "Social_Media_Usage_Hours", "Gaming_Hours",
                    "Screen_Time_Hours", "Stress_Level", "Sleep_Hours", "Physical_Activity_Hours")

cat("\nScatter plots for numeric columns vs Sleep Quality Numeric:\n")
```

```
for (col in setdiff(numerical_vars, "Sleep_Quality_Numeric")) {
  plot <- ggplot(data, aes_string(x = col, y = "Sleep_Quality_Numeric")) +
    geom_point(alpha = 0.5) +
    geom_smooth(method = "lm") +
    labs(title = paste("Scatter Plot of", col, "vs Sleep Quality"))
  print(plot)
}
```

VISUAL INSPECTION

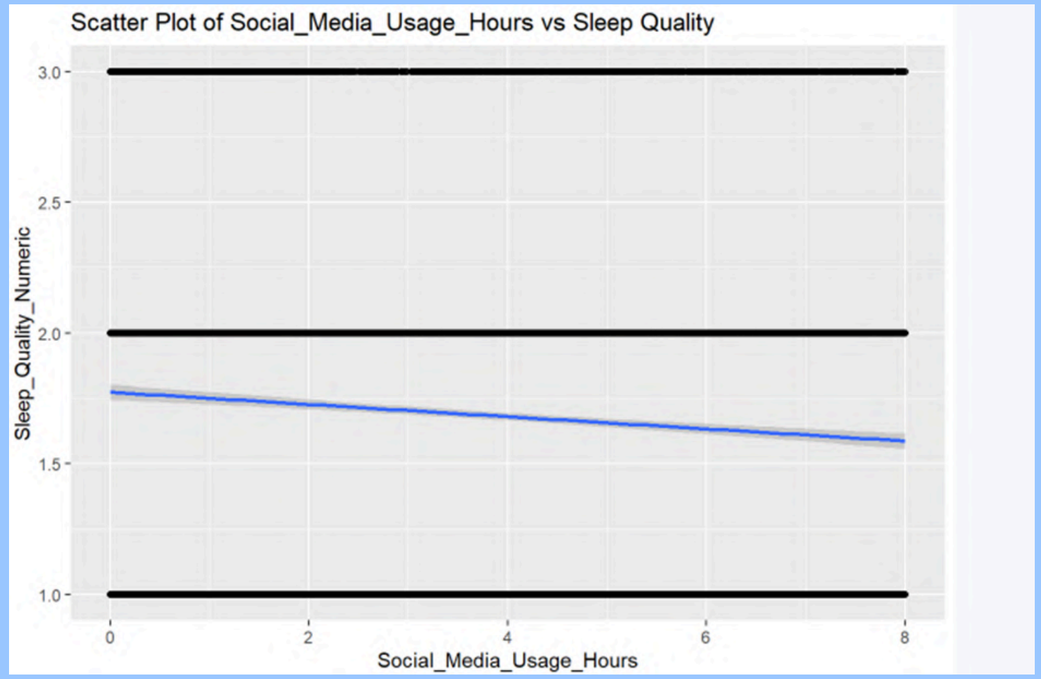
Categorical and Numerical Data Analysis

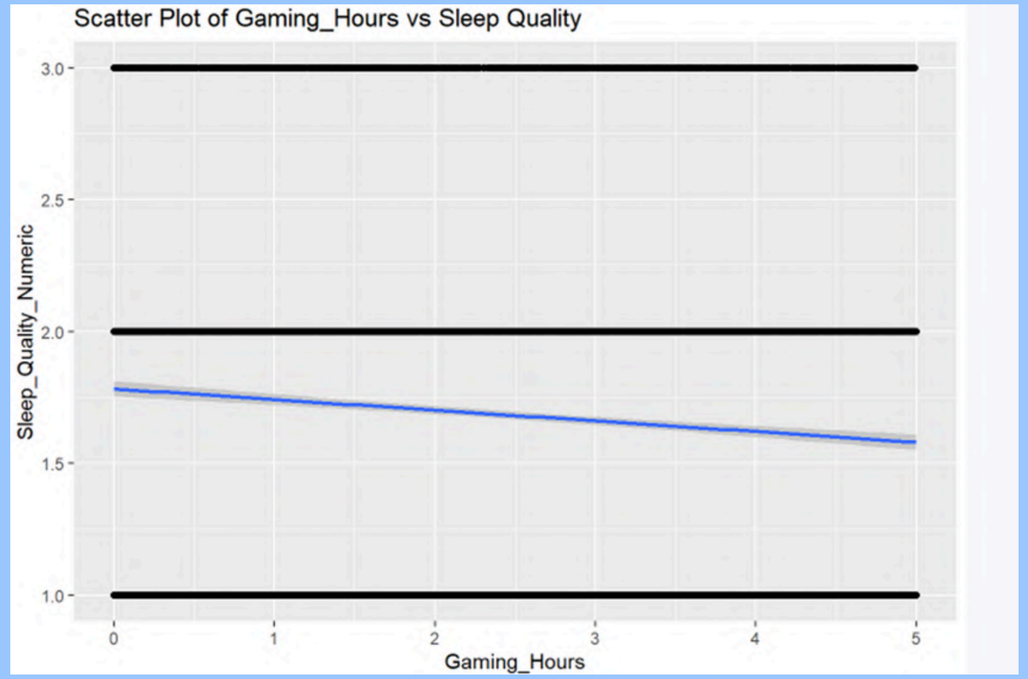




VISUAL INSPECTION

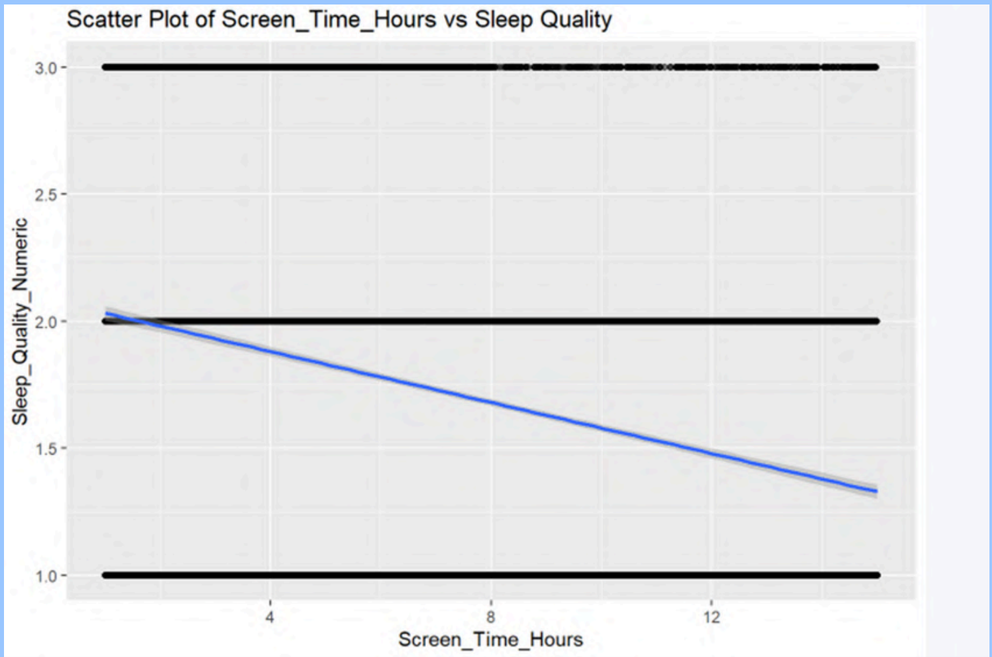
Categorical and Numerical Data Analysis

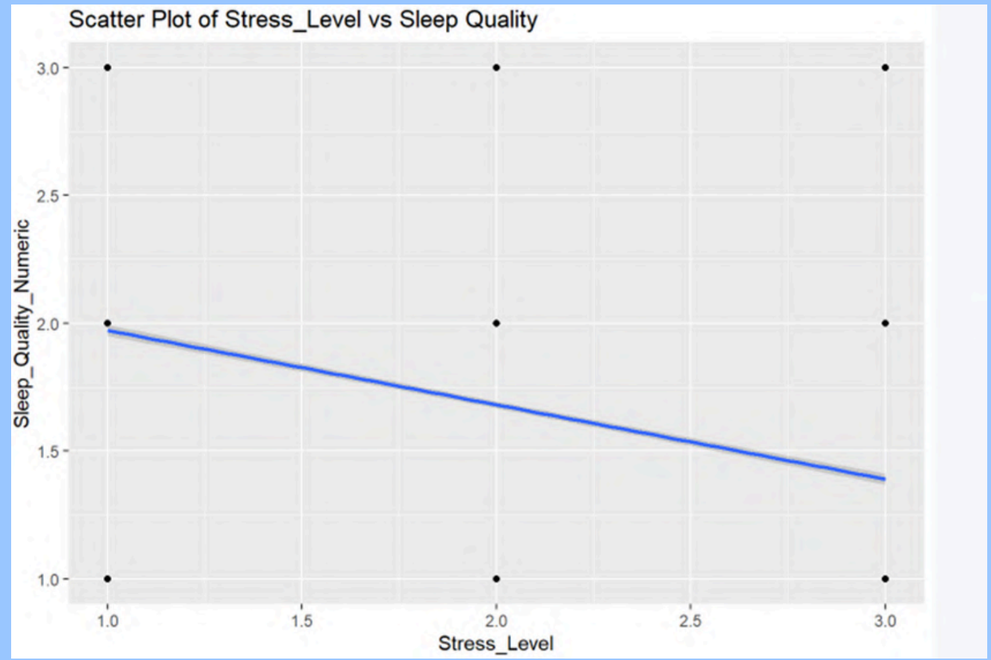




VISUAL INSPECTION

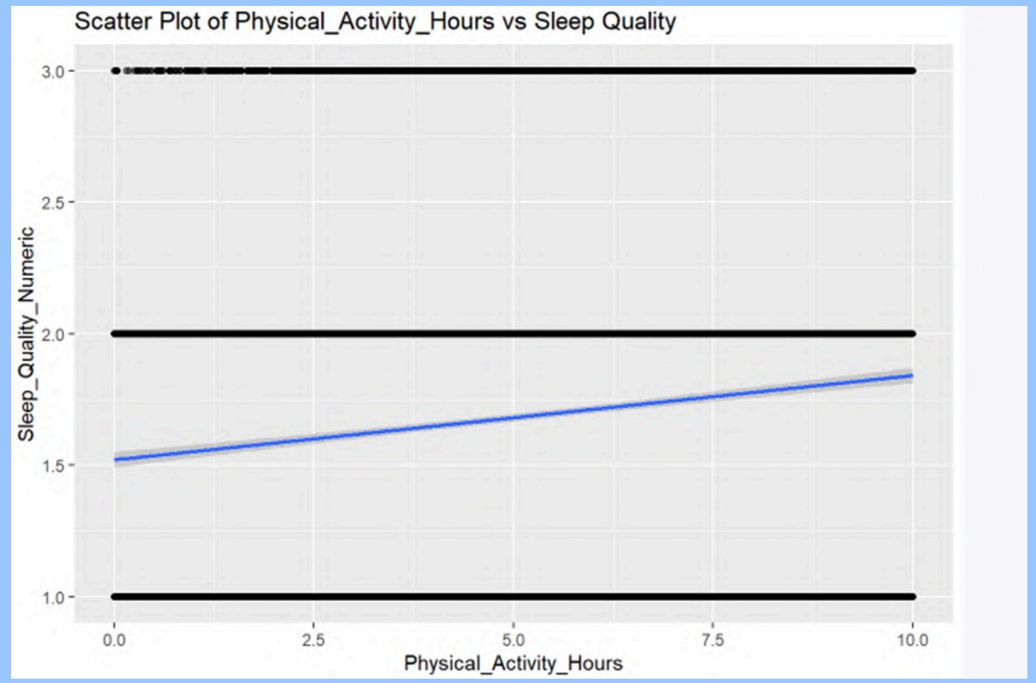
Categorical and Numerical Data Analysis





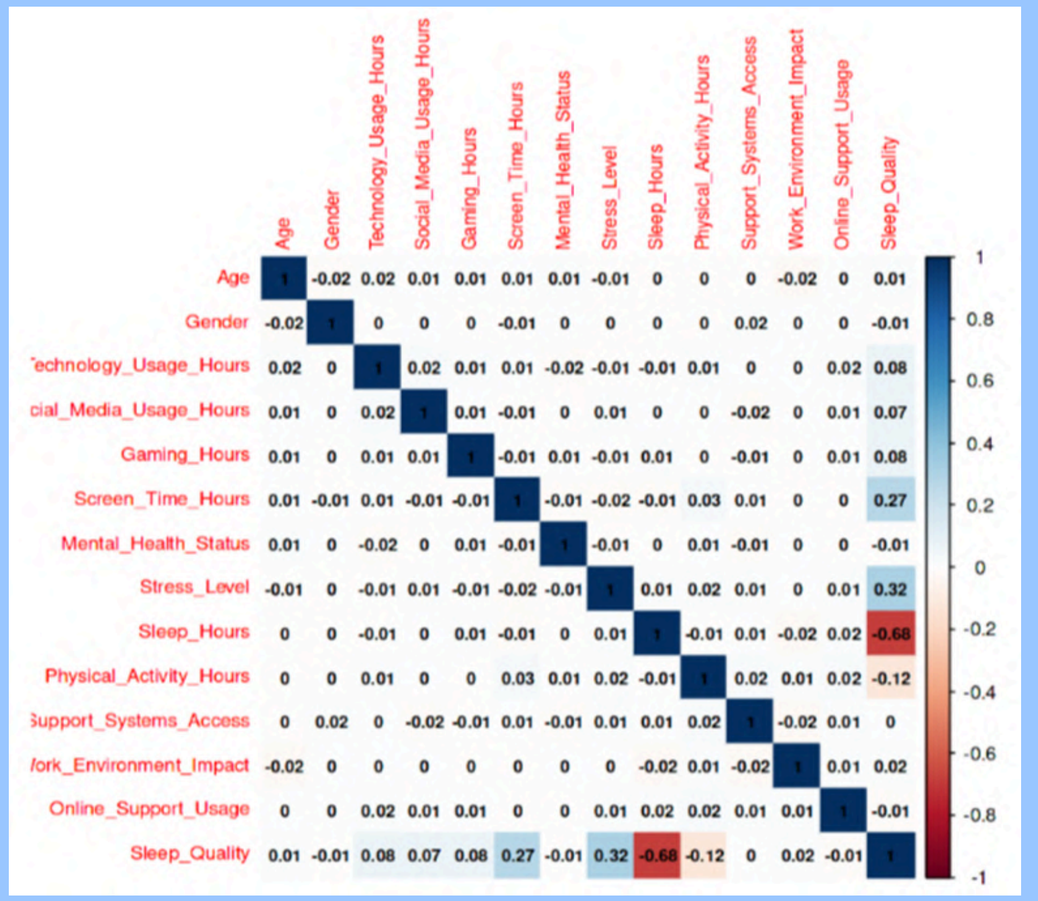
VISUAL INSPECTION

Categorical and Numerical Data Analysis



VISUAL INSPECTION

Pearson Correlation Matrix



VISUAL INSPECTION

Density Plot of Sleep Hours

```
density_plot <- ggplot(data, aes(x = Sleep_Hours)) +
  geom_density(fill = "blue", alpha = 0.5) +
  labs(title = "Density Plot of Sleep Hours", x = "Sleep Hours", y = "Density") +
  theme_minimal()

print(density_plot)
```

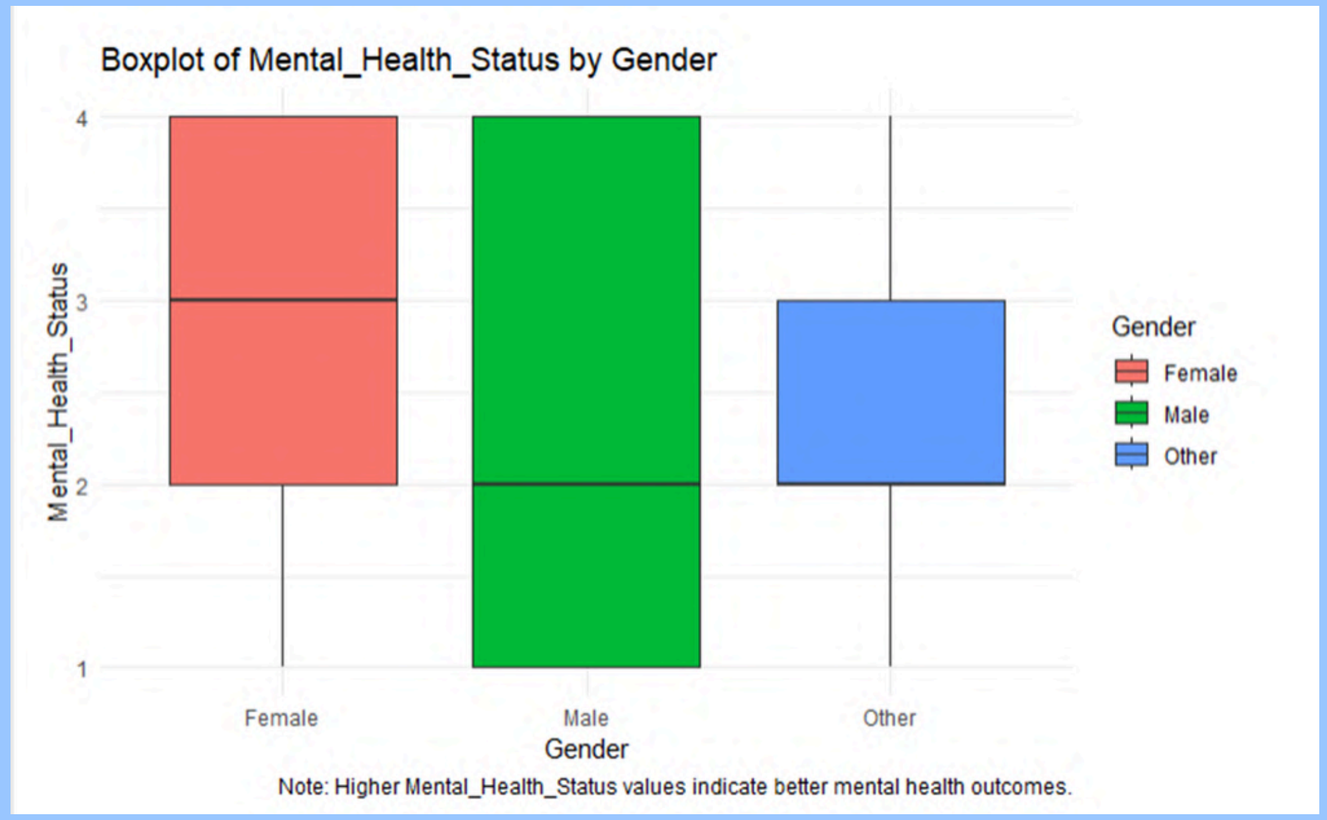
```
## The density plot visualizes the distribution of sleep hours, showing the most common sleep durations in the dataset.
```

VISUAL INSPECTION

Box Plot of Sleep Hours

```
boxplot <- ggplot(data, aes(x = Gender, y = Mental_Health_Status, fill = Gender)) +
  geom_boxplot() +
  labs(title = "Boxplot of Mental_Health_Status by Gender", x = "Gender", y = "Mental_Health_Status") +
  theme_minimal()

print(boxplot)
```

```
##
## Interpretation of the Boxplot of Mental_Health_Status by Gender:
##  - Female: Median Mental_Health_Status is around level 3, with moderate variability.
##  - Male: Median is also around level 3, but with a wider range, indicating greater variability.
##  - Other: Median is around level 2, with less variability, generally showing lower mental health status com
pared to other groups.
##  Overall, the distribution suggests differences in mental health status across genders, with 'Other' showin
g lower scores.
```



VISUAL INSPECTION

ECDF(Empirical Cumulative Distribution Function) Plot of Sleep Hours

```
ecdf_plot <- ggplot(data, aes(x = Sleep_Hours)) +
  stat_ecdf(geom = "step", color = "blue") +
  labs(title = "ECDF of Sleep Hours", x = "Sleep Hours", y = "Cumulative Probability") +
  theme_minimal()

print(ecdf_plot)
```

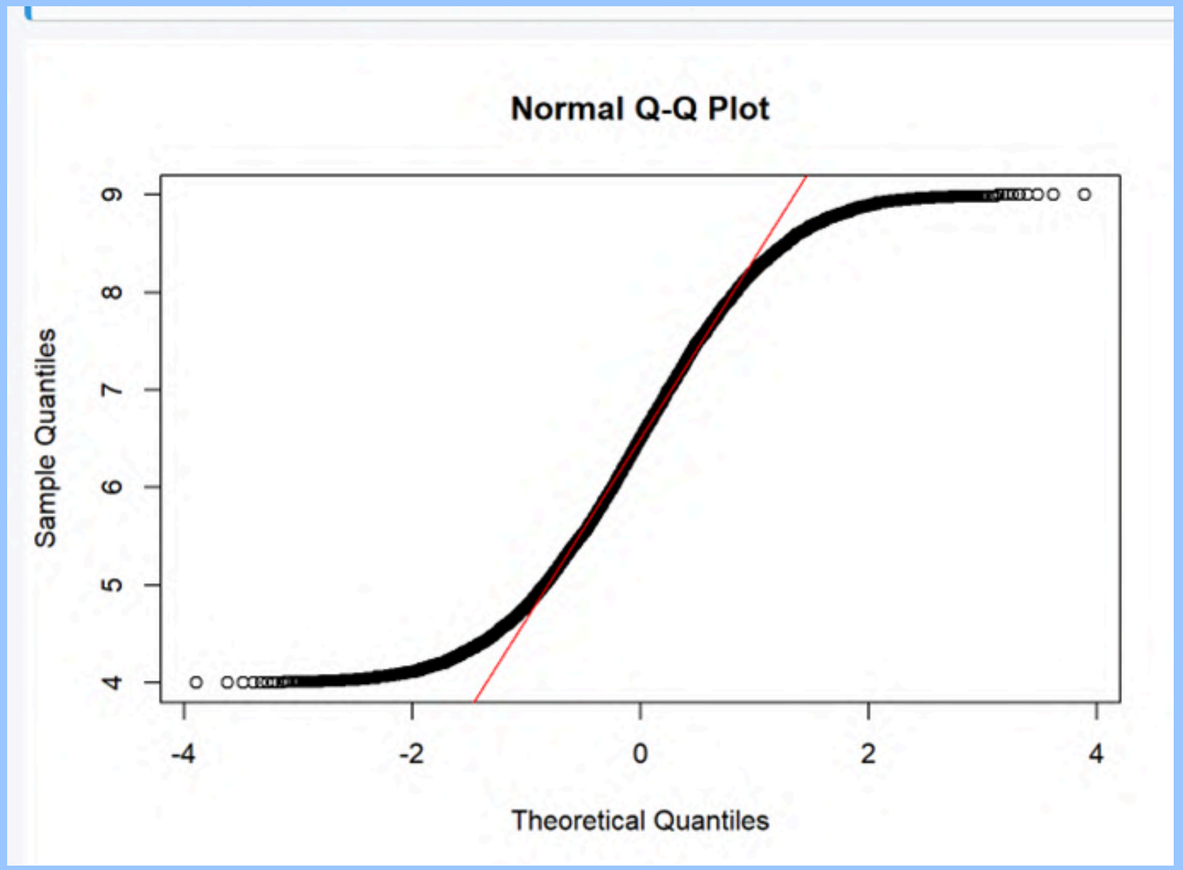
```
##
## The ECDF plot shows the cumulative distribution of sleep hours across the dataset. It indicates the proportion of ind
ividuals who have sleep hours less than or equal to a specific value. For example, if the curve reaches 0.5 at 7 hours,
this means that 50% of the participants sleep 7 hours or less. This plot is useful for understanding how sleep hours are
distributed and for identifying thresholds for sleep duration.
```

VISUAL INSPECTION

Q-Q Plot for Normality

```
qqnorm(data$Sleep_Hours)
qqline(data$Sleep_Hours, col = "red")  # Add a reference line
```

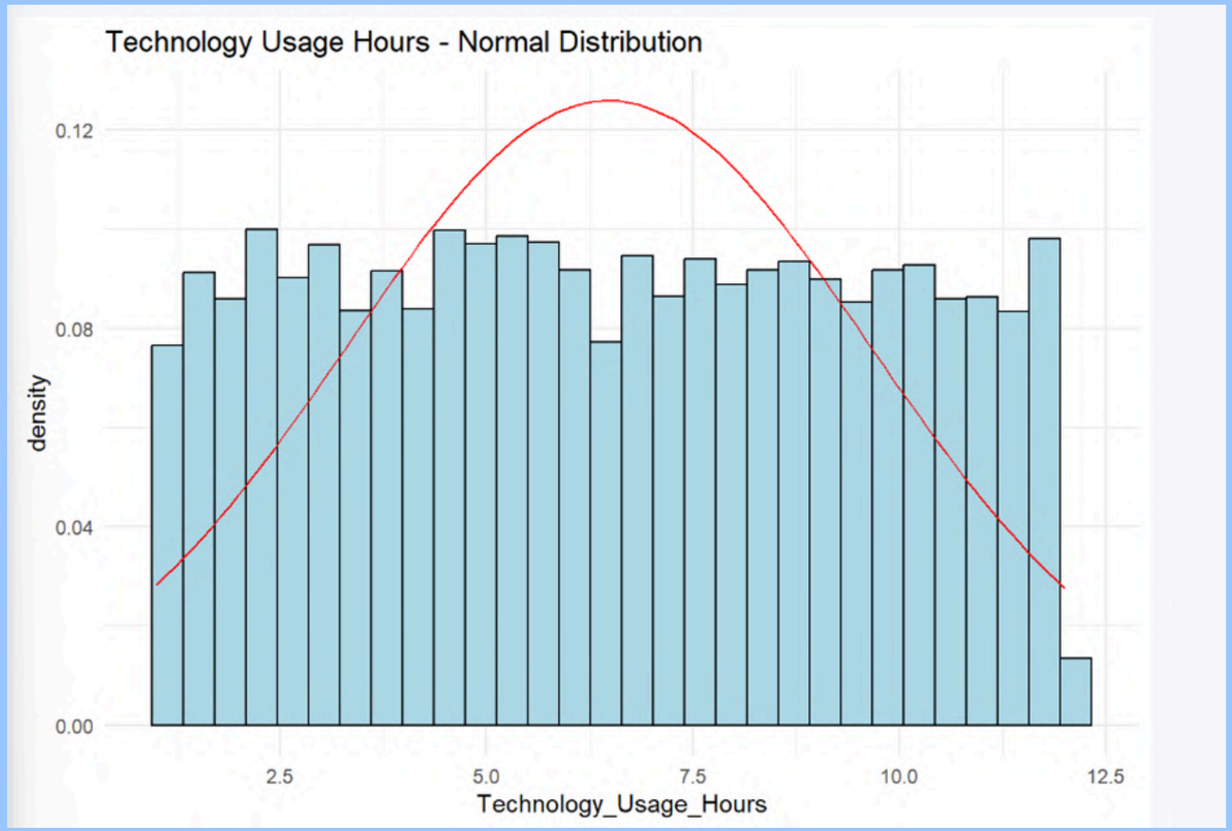
```
##
## The Q-Q plot is used to assess if the sleep hours data follows a normal distribution. Points that closely follow the red line indicate that the data is normally distributed.
```



DISTRIBUTION

Plot for Normal Distribution - Technology Usage Hours

```
# Plot for Normal Distribution - Technology Usage Hours
ggplot(data, aes(x = Technology_Usage_Hours)) +
  geom_histogram(aes(y = ..density..), bins = 30, fill = "lightblue", color = "black")
+
  stat_function(fun = dnorm, args = list(mean = mean(data$Technology_Usage_Hours, na.rm
= TRUE), sd = sd(data$Technology_Usage_Hours, na.rm = TRUE)), color = "red") +
  ggtitle("Technology Usage Hours - Normal Distribution") +
  theme_minimal()
```

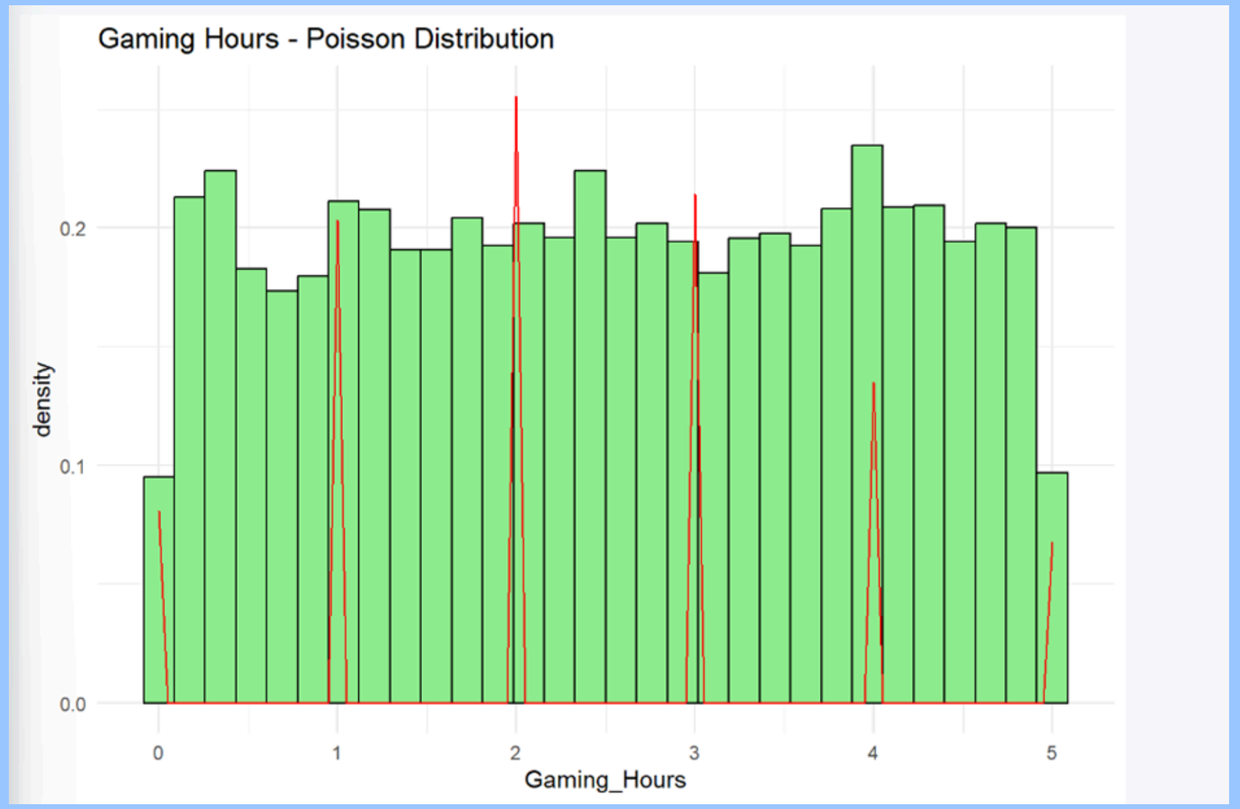


The histogram for Technology Usage Hours shows a significant deviation from the expected Normal distribution. The bars do not form the characteristic bell curve, and there is no smooth red line that closely follows the data. Instead, we observe uneven bar heights and an overall irregular distribution, indicating that the data does not follow a normal distribution. This could be due to outliers or the clustering of technology usage at certain levels, with some extreme values influencing the distribution. Therefore, it is likely that a different distribution model would be more appropriate for analyzing this data.

DISTRIBUTION

Poisson Distribution - Gaming Hours

```
# Plot for Poisson Distribution - Gaming Hours
ggplot(data, aes(x = Gaming_Hours)) +
  geom_histogram(aes(y = ..density..), bins = 30, fill = "lightgreen", color = "black")
+
  stat_function(fun = dpois, args = list(lambda = mean(data$Gaming_Hours, na.rm = TRU
E)), color = "red") +
  ggtitle("Gaming Hours - Poisson Distribution") +
  theme_minimal()
```

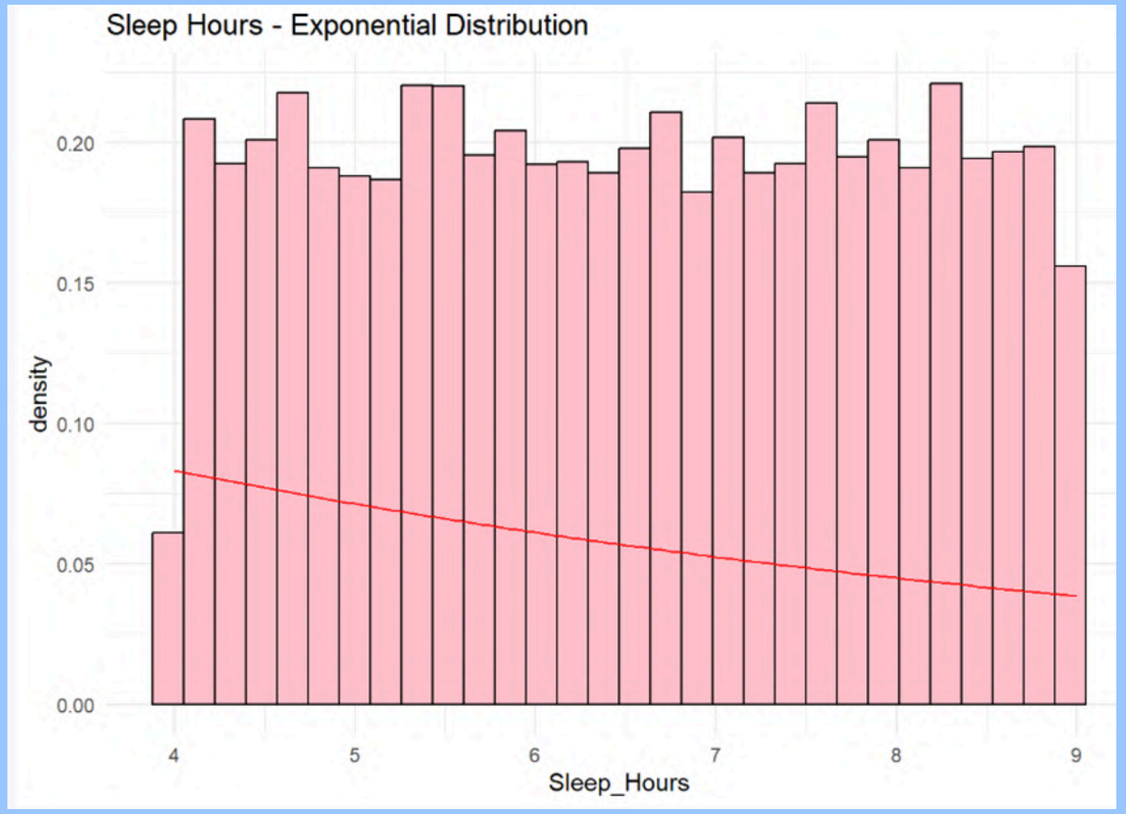


The plot for Gaming Hours exhibits a clear deviation from the expected Poisson distribution. The sharp, spiked red line contrasts with the irregular heights of the histogram bars, showing that the data does not match the characteristics of a Poisson process. The Poisson distribution assumes a steady rate of occurrence over time, which doesn't seem to apply to the gaming hours, likely because the data is influenced by various factors such as different gaming habits among users. As such, the Poisson model may not be the best fit, and alternative distributions, such as the Negative Binomial, may offer a better representation of the data.

DISTRIBUTION

Plot for Exponential Distribution - Sleep Hours

```
# Plot for Exponential Distribution - Sleep Hours
ggplot(data, aes(x = Sleep_Hours)) +
  geom_histogram(aes(y = ..density..), bins = 30, fill = "pink", color = "black") +
  stat_function(fun = dexp, args = list(rate = 1 / mean(data$Sleep_Hours, na.rm = TRU
E)), color = "red") +
  ggtitle("Sleep Hours - Exponential Distribution") +
  theme_minimal()
```

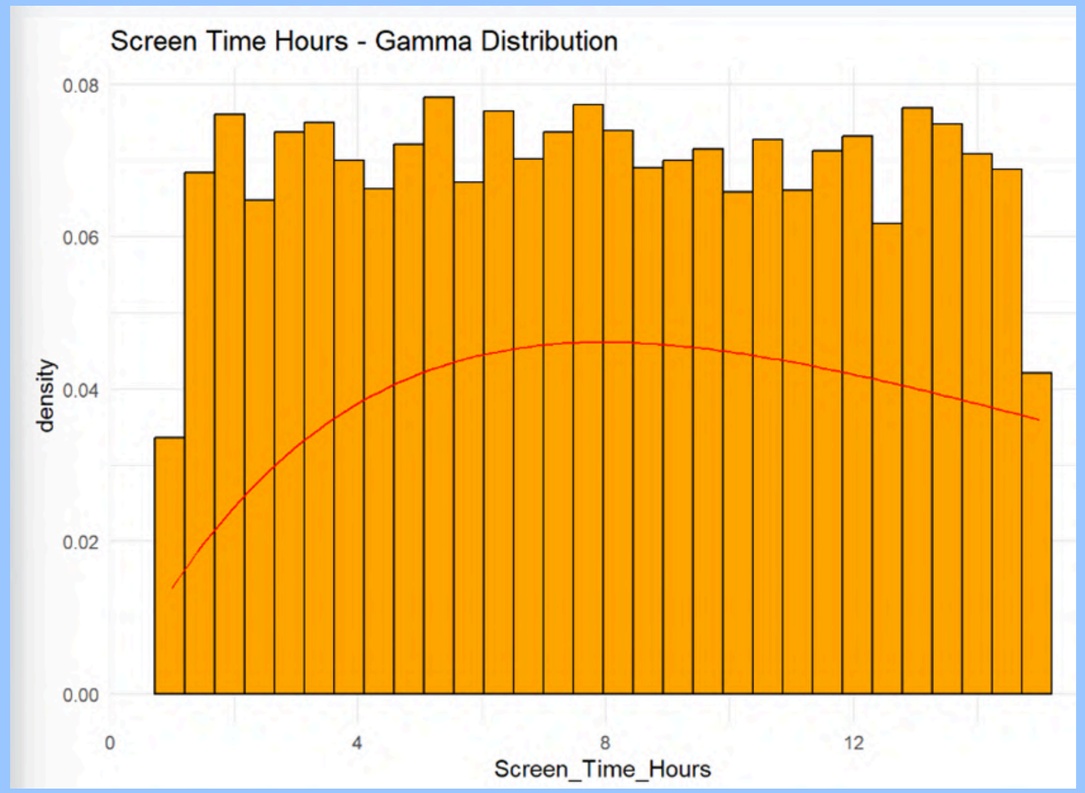


The histogram for Sleep Hours shows some alignment with the Exponential distribution, but the fit is not perfect. While the general shape of the data suggests a right-skewed distribution, similar to the Exponential model, there are noticeable deviations from the red line, indicating that the data may not entirely follow the expected pattern. This suggests that while the Exponential distribution might be a reasonable approximation, the data could potentially follow a different distribution or exhibit additional complexities not captured by the Exponential model. The fit appears plausible, but further analysis would be needed to confirm the best model for this dataset.

DISTRIBUTION

Plot for Gamma Distribution - Screen Time Hours

```
# Plot for Gamma Distribution - Screen Time Hours
ggplot(data, aes(x = Screen_Time_Hours)) +
  geom_histogram(aes(y = ..density..), bins = 30, fill = "orange", color = "black") +
  stat_function(fun = dgamma, args = list(shape = 2, rate = 1 / mean(data$Screen_Time_H
ours, na.rm = TRUE)), color = "red") +
  ggtitle("Screen Time Hours - Gamma Distribution") +
  theme_minimal()
```



The histogram for Screen Time Hours aligns well with the Gamma distribution, showing a right-skewed shape as expected. However, there are minor discrepancies in the heights of the bars compared to the red line, indicating slight variations in the data. The Gamma distribution is suitable for modeling continuous, positive data with a skewed distribution, and it appears to capture the overall trend in screen time behavior. The slight differences in bar heights may be attributed to sampling noise or variability in the actual data, but the overall structure supports the use of the Gamma distribution as a reasonable model for this variable.

TESTING

Chi-Squared Test only if there are valid entries

```r
if (length(unique_genders) > 0 && length(unique_stress_levels) > 0) {
  chi_squared_test <- chisq.test(table(data$Gender, data$Stress_Level))

  chi_squared_results <- list()
  chi_squared_results$p_value <- chi_squared_test$p.value
  chi_squared_results$inference <- if (chi_squared_test$p.value > 0.05) {
    "No significant association between Gender and Stress_Level."
  } else {
    "Significant association between Gender and Stress_Level."
  }
```

Chi-Squared Test for Independence:
P-value: 0.9422046
Inference: No significant association between Gender and Stress_Level

```r
  cat("\nChi-Squared Test for Independence:\n")
  cat("P-value:", chi_squared_results$p_value, "\n")
  cat("Inference:", chi_squared_results$inference, "\n")
} else {
  cat("\nCannot perform Chi-Squared Test: Insufficient valid entries in Gender or Stress_Level.\n")
}
```

TESTING

Kolmogorov-Smirnov Test for normality

```
ks_test <- ks.test(data$Sleep_Hours, "pnorm", mean = mean(data$Sleep_Hours, na.rm = TRUE), sd = sd(data$Sleep_Hours, na.rm = TRUE))
```

```
print(ks_test)
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  data$Sleep_Hours
## D = 0.061324, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

The test indicates that the distribution of Sleep Hours significantly deviates from a normal distribution (06.0613 2378, p-value 4.333326e-33).

This suggests that Sleep Hours is not normally distributed.

TESTING

Anderson-Darling Test

```
ad_test <- ad.test(data$Sleep_Hours)
print(ad_test)
```

```
##
##  Anderson-Darling normality test
##
## data:  data$Sleep_Hours
## A = 116.1, p-value < 2.2e-16
```

The test indicates that the distribution of Sleep Hours significantly deviates from a normal distribution (A = 116.1 p-value 3.7e-24).

This further supports that Sleep Hours is not normally distributed.

REGRESSION ANALYSIS FOR PREDICTING SLEEP HOURS

The primary objective of this project is to develop a robust regression model to predict Sleep Hours based on various behavioral, demographic, and lifestyle factors.

The dataset includes features such as Gaming Hours, Physical Activity Hours, Technology Usage Hours, Age, and others that may influence sleep patterns.

To achieve this, we employed several regression techniques, ranging from simple linear models to more complex tree-based models.

The goal was to compare the performance of different models and select the one that provides the best predictive accuracy while maintaining interpretability.

CATEGORICAL ENCODING

Categorical variables are those that represent distinct categories or labels (e.g., Gender, Mental_Health_Status). Machine learning models cannot directly work with these categorical variables, so they must be converted into numerical representations. There are several methods to encode categorical data:

- One-Hot Encoding: This method creates binary columns for each category. For example, the "Gender" variable with two categories ("Male" and "Female") would be transformed into two binary columns: Gender_Male and Gender_Female, where a value of 1 represents the presence of that category.

- Label Encoding: This method assigns a unique integer to each category. For example, "Male" could be encoded as 0 and "Female" as 1. This method is simpler but can introduce unintended ordinal relationships between categories.

TRAIN-TEST SPLITTING:

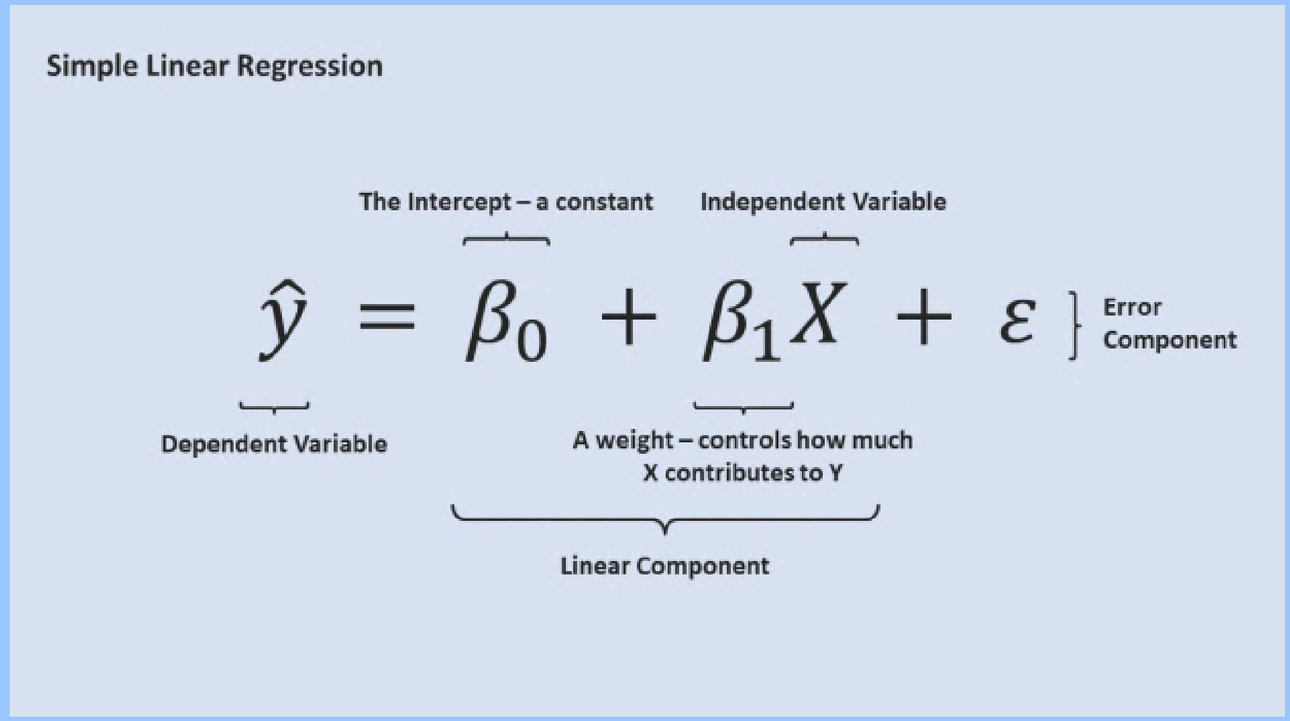
To evaluate the performance of our models, we split the dataset into two parts:

- Training Set: Used to train the model. Test Set: Used to evaluate the model's performance on unseen data.
- This split ratio is chosen for having a wider testing range.

```
# Splitting data into training and testing sets
split <- initial_split(finaldf, prop = 0.6)
trainData <- training(split)
testValData <- testing(split)
```

LINEAR REGRESSION

- We began with a simple linear regression model to establish a baseline. This method assumes a linear relationship between the predictors (e.g., Gaming_Hours, Age) and the target variable (Sleep_Hours).
- While linear regression is easy to interpret, it may not capture complex relationships in the data.



REGRESSION FOR PREDICTING SLEEP HOURS

Linear Model

```r
# Function to convert categorical columns to numeric and store the mapping
convert_to_numeric <- function(data) {
  mappings <- list()
  for (column in names(data)) {
    if (is.factor(data[[column]]) || is.character(data[[column]])) {
      levels <- unique(data[[column]])
      mappings[[column]] <- setNames(seq_along(levels), levels)
      data[[column]] <- as.numeric(factor(data[[column]], levels = levels))
    }
  }
  return(list(data = data, mappings = mappings))
}

# Convert the DataFrame and get mappings
datawithmap <- convert_to_numeric(data)
finaldf <- datawithmap$data
maps <- datawithmap$mappings

# Splitting data into training and testing sets
split <- initial_split(finaldf, prop = 0.6)
trainData <- training(split)
testValData <- testing(split)

# Fit linear model
linearmodel <- lm(Sleep_Hours ~ ., data = trainData)
summary(linearmodel)
```

```
## Call:
## lm(formula = Sleep_Hours ~ ., data = trainData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.86105 -0.63433 -0.04863  0.60126  3.13426
##
## Coefficients: (1 not defined because of singularities)
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 8.446e+00  1.017e-01  83.050  < 2e-16 ***
## User_ID                    -2.700e-07  4.120e-06  -0.066    0.948
## Age                        -1.164e-04  8.575e-04  -0.136    0.892
## Gender                      1.454e-02  1.461e-02   0.995    0.320
## Technology_Usage_Hours      2.249e-02  3.773e-03   5.960 2.66e-09 ***
## Social_Media_Usage_Hours    3.910e-02  5.156e-03   7.584 3.85e-14 ***
## Gaming_Hours                8.467e-02  8.281e-03  10.224  < 2e-16 ***
## Screen_Time_Hours           8.629e-02  3.097e-03  27.857  < 2e-16 ***
## Mental_Health_Status       -5.326e-03  1.063e-02  -0.501    0.616
## Stress_Level                5.105e-01  1.555e-02  32.833  < 2e-16 ***
## Physical_Activity_Hours    -6.727e-02  4.145e-03 -16.229  < 2e-16 ***
## Support_Systems_Access      3.283e-02  2.377e-02   1.381    0.167
## Work_Environment_Impact    -7.685e-03  1.441e-02  -0.533    0.594
## Online_Support_Usage        3.761e-02  2.377e-02   1.582    0.114
## Sleep_Quality              -1.694e+00  1.797e-02 -94.266  < 2e-16 ***
## Sleep_Quality_Numeric             NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9193 on 5985 degrees of freedom
## Multiple R-squared:  0.5989, Adjusted R-squared:  0.598
## F-statistic: 638.3 on 14 and 5985 DF,  p-value: < 2.2e-16
```

REGRESSION FOR PREDICTING SLEEP HOURS

Linear Model (Evaluation and Comparison)

```
cat("Mean Absolute Error (MAE):", mae, "\n")
```

```
## Mean Absolute Error (MAE): 0.7322782
```

```
cat("Mean Squared Error (MSE):", mse, "\n")
```

```
## Mean Squared Error (MSE): 0.8449346
```

```
cat("Root Mean Squared Error (RMSE):", rmse, "\n")
```

```
## Root Mean Squared Error (RMSE): 0.9192032
```

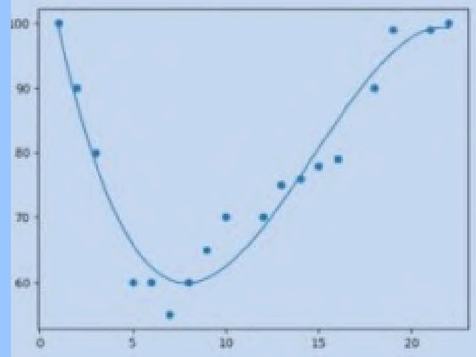
```
##   Actual.Sleep.Hours Predicted.Sleep.Hours
## 1               8.01                8.386794
## 2               8.04                8.418825
## 3               5.55                6.535490
## 4               8.61                7.389491
## 5               8.61                7.433276
## 6               7.11                7.255287
```

Mean squared error	$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2$
Root mean squared error	$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}$
Mean absolute error	$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n} \lvert e_t \rvert$

REGRESSION FOR PREDICTING SLEEP HOURS

Polynomial Regression

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_k x^k$$



Matrix Notation

$$\begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^k \\ 1 & x_2 & x_2^2 & \dots & x_2^k \\ \dots & \dots & \dots & \dots & \dots \\ 1 & x_n & x_n^2 & \dots & x_n^k \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \dots \\ \beta_k \end{bmatrix} \quad or\ \vec{y} = X.\vec{\beta} \Longrightarrow$$

$$\vec{\hat{\beta}} = (X^T X)^{-1} X^T \vec{y}$$

```r
# Initialize vectors to store metrics for each degree
degrees <- 1:4
mae_list <- c()
mse_list <- c()
rmse_list <- c()

# For loop to fit polynomial models from degree 1 to 12
for (degree in degrees) {

  # Fit polynomial regression model for the current degree using trainData
  poly_model <- lm(Sleep_Hours ~ poly(as.matrix(X_train), degree, raw = TRU
E), data = trainData)

  # Apply the same polynomial transformation to X_test before predicting
  X_test_poly <- as.data.frame(poly(as.matrix(X_test), degree, raw = TRUE))
```

```r
  # Predict using the polynomial model on test data
  poly_predictions <- predict(poly_model, newdata = X_test_poly)

  # Calculate metrics
  actuals <- testValData$Sleep_Hours
  mae <- mean(abs(poly_predictions - actuals))
  mse <- mean((poly_predictions - actuals)^2)
  rmse <- sqrt(mse)

  # Store the metrics in the lists
  mae_list[degree] <- mae
  mse_list[degree] <- mse
  rmse_list[degree] <- rmse
```

REGRESSION FOR PREDICTING SLEEP HOURS

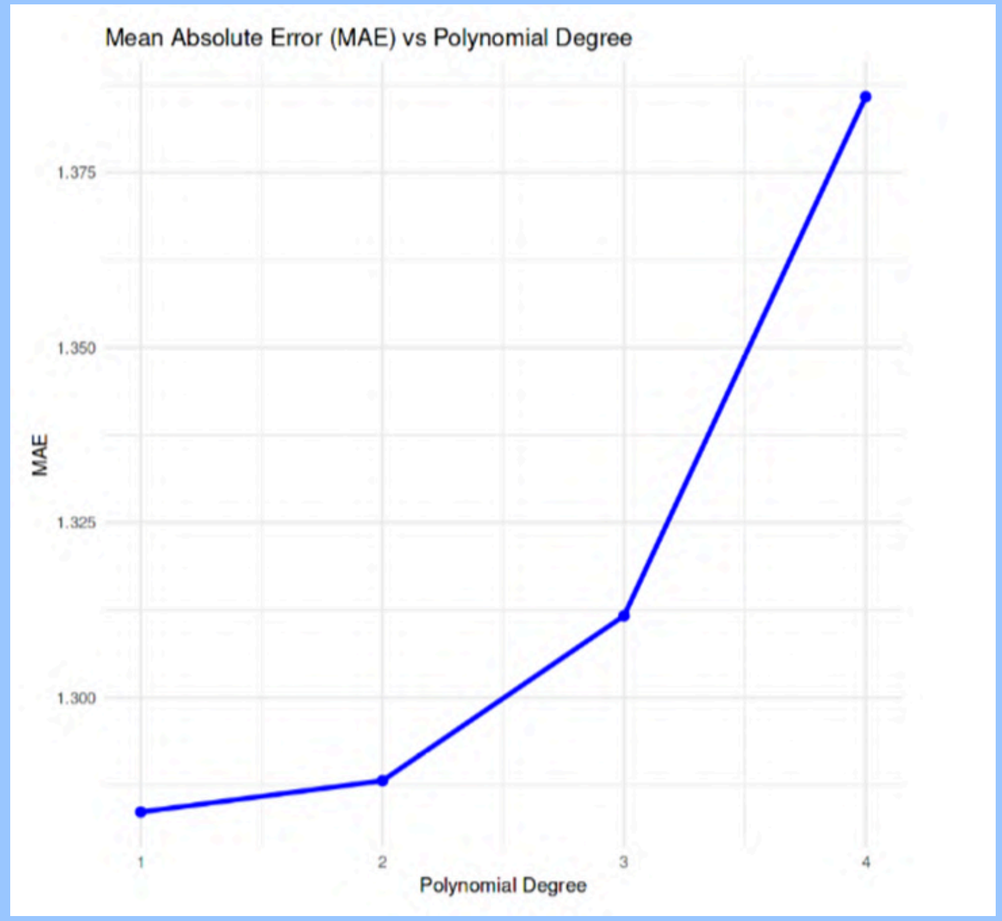
Polynomial Regression (Evaluation and plots for 4 degrees)

```
Degree: 1
Mean Absolute Error (MAE): 1.283664
Mean Squared Error (MSE): 2.160732
Root Mean Squared Error (RMSE): 1.469943
```

```
Degree: 2
Mean Absolute Error (MAE): 1.288129
Mean Squared Error (MSE): 2.181306
Root Mean Squared Error (RMSE): 1.476924
```

```
Degree: 3
Mean Absolute Error (MAE): 1.311683
Mean Squared Error (MSE): 2.306469
Root Mean Squared Error (RMSE): 1.518706
```

```
Degree: 4
Mean Absolute Error (MAE): 1.385807
Mean Squared Error (MSE): 2.70696
Root Mean Squared Error (RMSE): 1.645284
```



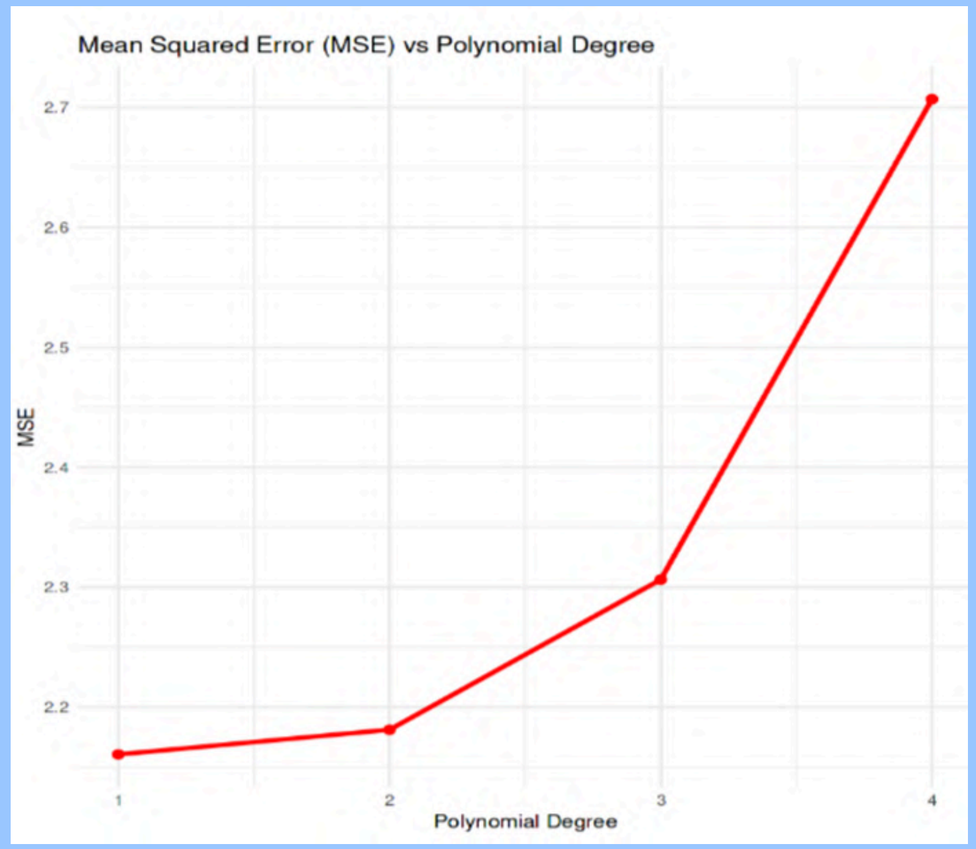
Degrees Tested: We tested polynomial degrees from 1 to 4.

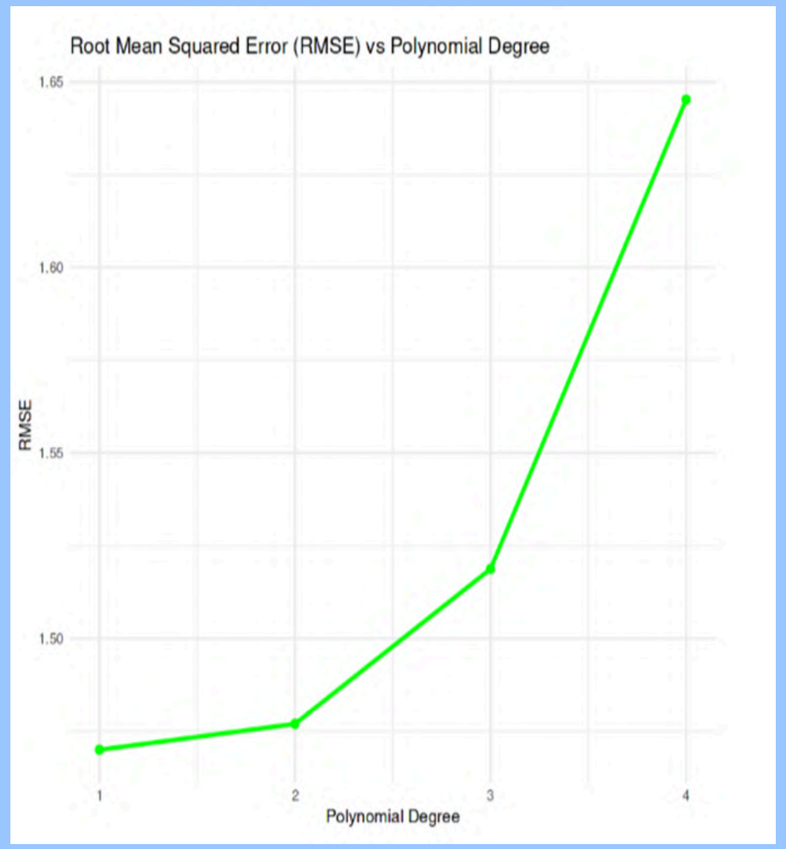
Evaluation: Metrics such as RMSE (Root Mean Squared Error) and MAE (Mean Absolute Error) were used to evaluate model performance.

Results: As seen in the attached plots, increasing the polynomial degree significantly reduced RMSE and MAE, especially at degree 4, where both metrics showed substantial improvement

REGRESSION FOR PREDICTING SLEEP HOURS

Polynomial Regression (plots for 4 degrees)



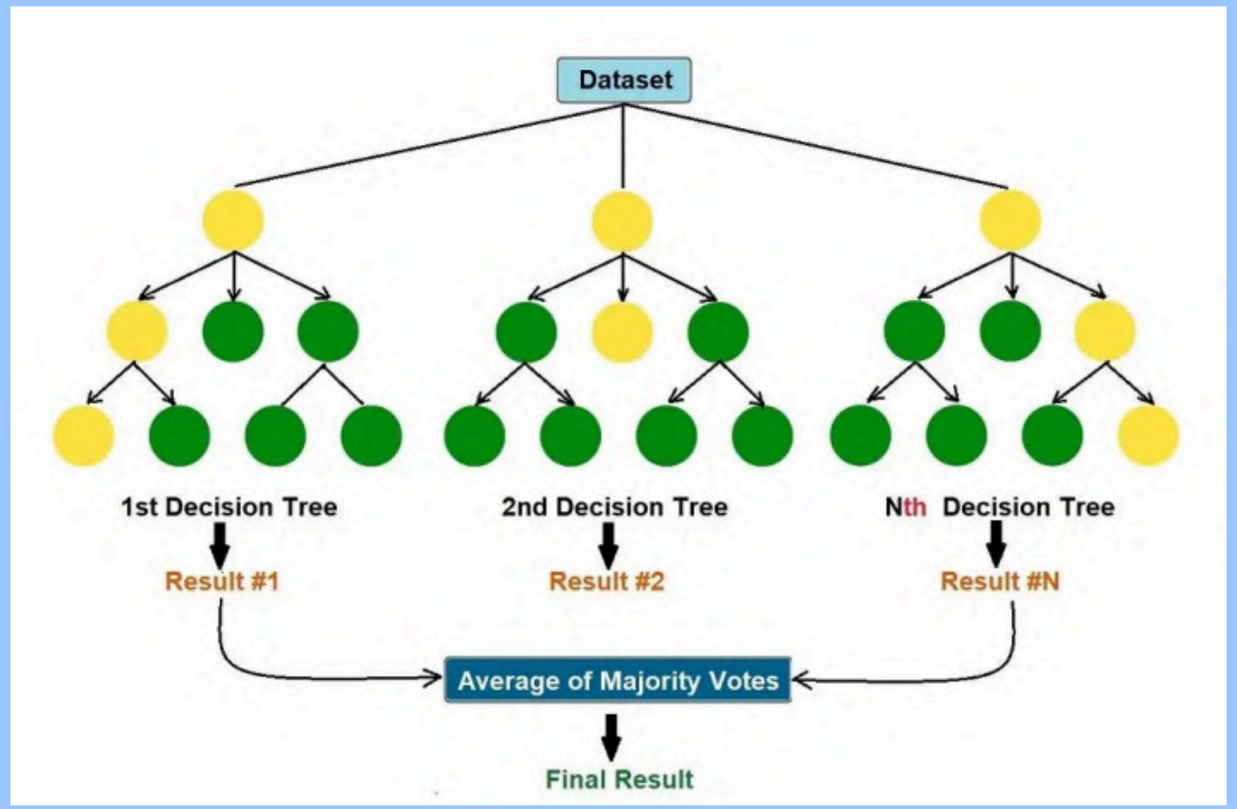


DECISION TREES

3.1 Random Forest (RF)

We explored decision tree-based models to capture complex interactions between features. Decision trees split the data into subsets based on feature values, allowing for non-linear relationships between predictors and sleep hours.

Random Forest (RF) Random Forest is an ensemble method that builds multiple decision trees and averages their predictions to improve accuracy and reduce overfitting. Each tree is trained on a random subset of the data, and the final prediction is the average of all tree outputs. This technique helps reduce variance and improve generalization.

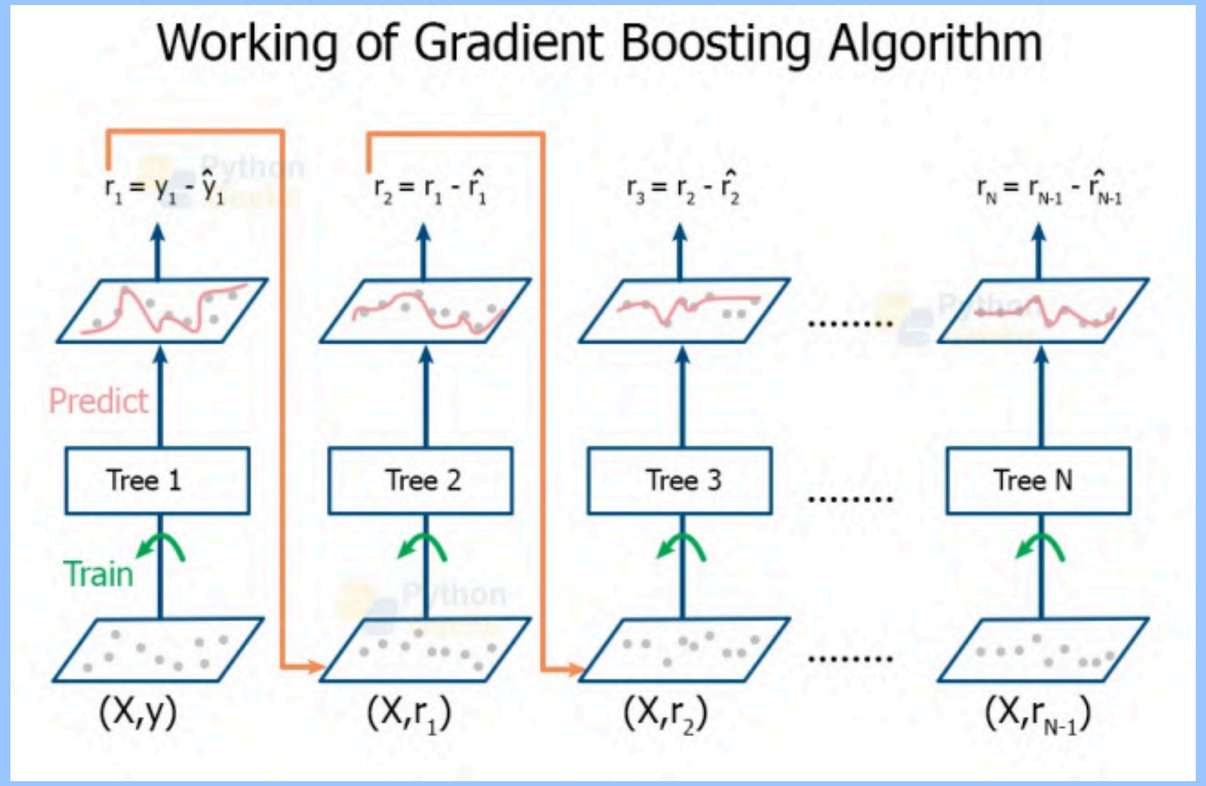


DECISION TREES

Gradient Boosting Machine (GBM)

Gradient Boosting Machine (GBM) builds trees sequentially, where each tree attempts to correct errors made by previous trees.

It focuses on minimizing the residual errors in a step-by-step process, making it more powerful than Random Forest for certain datasets.



DECISION TREES

Finding best MTRY value with Random Forest Model

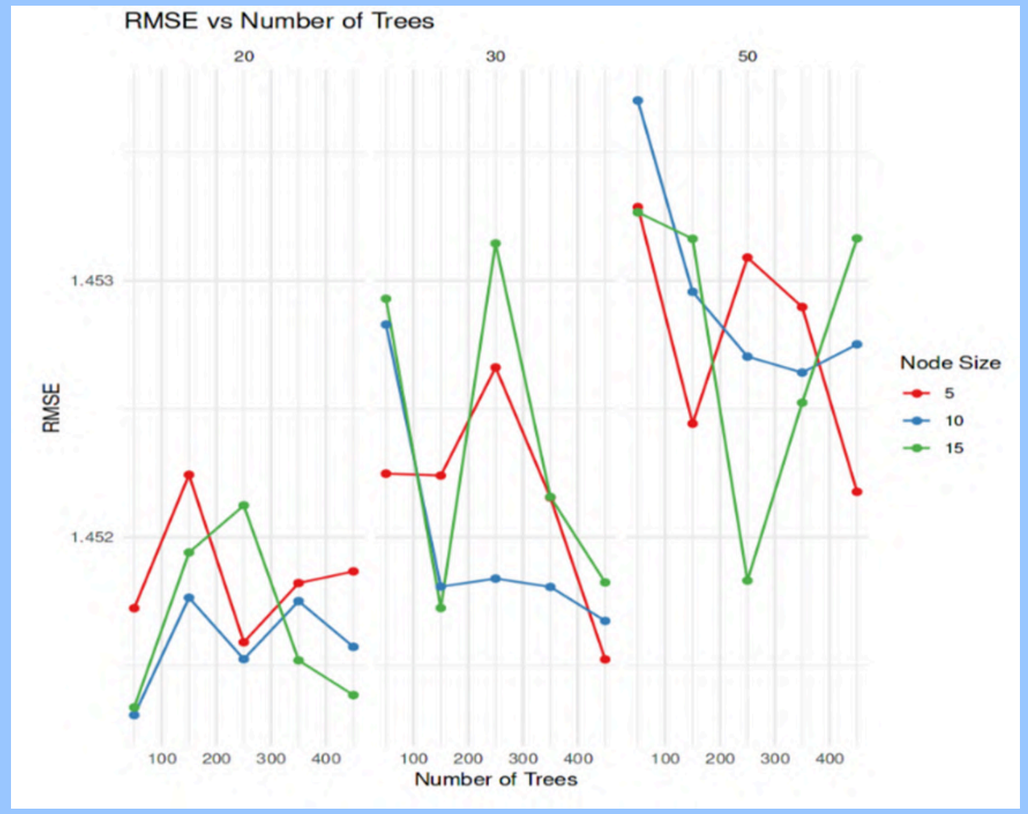
```r
# Train the Random Forest model with K-fold cross-validation using x and y i
nterface
rf_model <- train(
  x = features,
  y = target,
  method = "rf",
  trControl = train_control,
  tuneLength = 3
)

# Display the results
print(rf_model)
```

```
Random Forest

10000 samples
   12 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 8000, 7999, 8001, 8000, 8000
Resampling results across tuning parameters:

  mtry  RMSE      Rsquared      MAE
   2    1.463439  0.0019676966  1.266121
   7    1.470097  0.0010253140  1.271171
  12    1.471346  0.0009208063  1.271556

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was mtry = 2.
```

DECISION TREES

Finding best MTRY value with Random Forest Model

```
Current parameters: ntree=450, nodesize=15, maxnodes=50

Best parameter combination:
  ntree nodesize maxnodes     RMSE      MAE     Rsquared
4    50       10       20 1.451307 1.259001 0.0001024091

Top 10 parameter combinations:
   ntree nodesize maxnodes     RMSE      MAE     Rsquared
4     50       10       20 1.451307 1.259001 0.0001024091
7     50       15       20 1.451337 1.259424 0.0004923137
43   450       15       20 1.451385 1.259375 0.0002028729
34   350       15       20 1.451520 1.259394 0.0001579377
38   450        5       30 1.451524 1.259499 0.0011173030
22   250       10       20 1.451525 1.259283 0.0005232192
40   450       10       20 1.451573 1.259521 0.0001494893
19   250        5       20 1.451591 1.259510 0.0002374021
41   450       10       30 1.451674 1.259245 0.0001000566
1     50        5       20 1.451723 1.259676 0.0003407868
```



DECISION TREES

Ensemble approach (stacking multiple models)

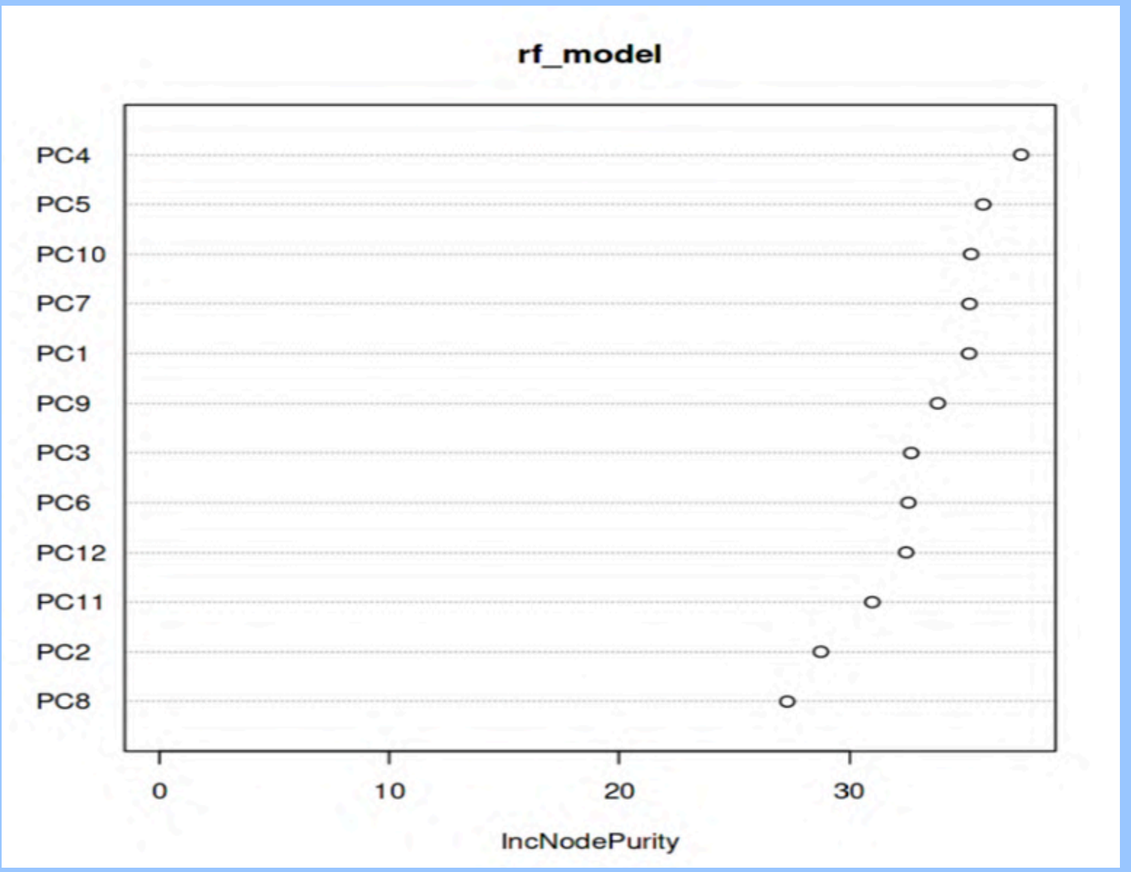
```r
# Simple average ensemble
ensemble_predictions <- (predictions_rf + predictions_xgb + predictions_gb
m) / 3

# Calculate performance metrics
rf_rmse <- sqrt(mean((predictions_rf - target)^2))
xgb_rmse <- sqrt(mean((predictions_xgb - target)^2))
gbm_rmse <- sqrt(mean((predictions_gbm - target)^2))
ensemble_rmse <- sqrt(mean((ensemble_predictions - target)^2))

# Print results
cat("\nModel Performance (RMSE):\n")
cat("Random Forest:", rf_rmse, "\n")
cat("XGBoost:", xgb_rmse, "\n")
cat("GBM:", gbm_rmse, "\n")
cat("Ensemble:", ensemble_rmse, "\n")

# Stop parallel processing
stopCluster(cl)
```

```
Loading required package: foreach

Loading required package: iterators

Loading required package: parallel


Training Models:

Training Random Forest...

Training XGBoost...

Training GBM...

Model Performance (RMSE):
Random Forest: 1.440298
XGBoost: 1.412972
GBM: 1.430844
Ensemble: 1.426808
```

DECISION TREES

RF Model with PCA features

```r
# Train a Random Forest model with tuned hyperparameters after PCA (no scali
ng needed for RF)
rf_model <- randomForest(
    x = train_data %>% select(-Sleep_Hours),
    y = train_data$Sleep_Hours,
    ntree = 350,
    mtry = 2,
    nodesize = 15,
    maxnodes = 20
)

# Predict on test data using Random Forest model
rf_preds <- predict(rf_model, newdata = test_data %>% select(-Sleep_Hour
s))

# Calculate RMSE for Random Forest model on test set
rf_rmse <- sqrt(mean((test_data$Sleep_Hours - rf_preds)^2))
print(paste("Random Forest RMSE after PCA:", rf_rmse))

# View feature importance in Random Forest model (after PCA)
importance(rf_model)
varImpPlot(rf_model)
```

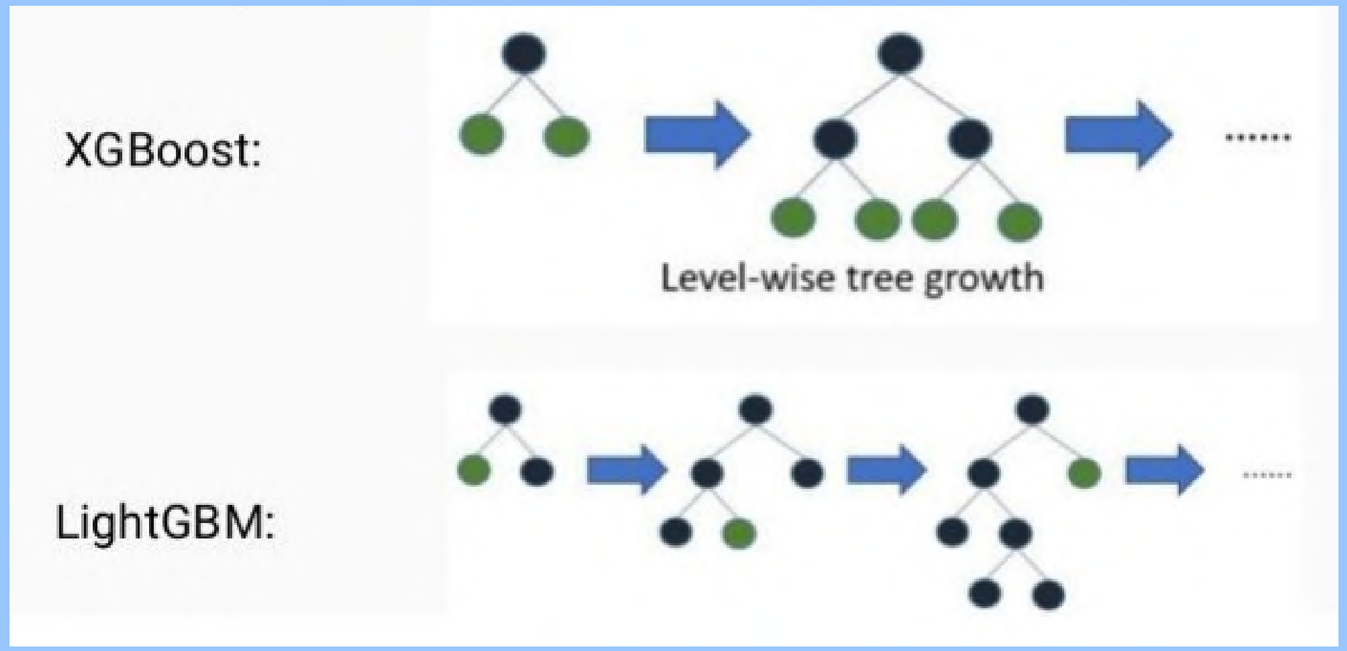


DECISION TREES

XGBoost

XGBoost is an optimized version of GBM that uses advanced regularization techniques (L1 and L2) to prevent overfitting while maintaining high accuracy. XGBoost is highly efficient and scalable, making it a popular choice for large datasets.

Feature Importance: XGBoost feature importance rankings, confirming that Gaming_Hours, Physical_Activity_Hours, Technology_Usage_Hours, and Age were among the most important predictors.



DECISION TREES

XGBoost with PC features

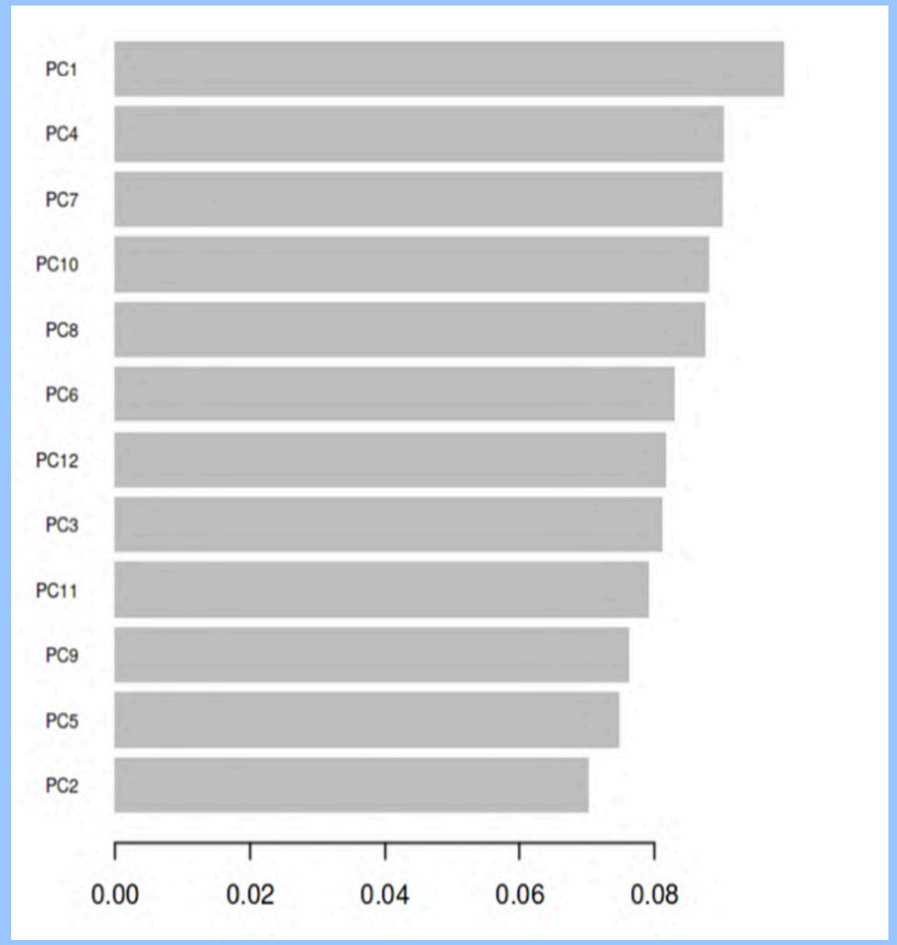
```r
# Predict on test data using XGBoost model
xgb_preds <- predict(xgb_model, newdata = dtest)

# Calculate RMSE for XGBoost model on test set
xgb_rmse <- sqrt(mean((test_label - xgb_preds)^2))
print(paste("XGBoost RMSE after PCA:", xgb_rmse))

# View feature importance in XGBoost model (after PCA)
importance_matrix <- xgb.importance(model=xgb_model)
print(importance_matrix)
xgb.plot.importance(importance_matrix)
```

```
[1]     train-rmse:5.883031+0.003464    test-rmse:5.883006+0.014203
Multiple eval metrics are present. Will use test_rmse for early stoppin
g.
Will train until test_rmse hasn't improved in 50 rounds.

[51]    train-rmse:1.318635+0.007327    test-rmse:1.538964+0.022395
[101]   train-rmse:1.023208+0.015553    test-rmse:1.488615+0.023449
Stopping. Best iteration:
[80]    train-rmse:1.110771+0.015206    test-rmse:1.484245+0.022518
```

DECISION TREES

XGBoost with PC features

```
[1] "XGBoost RMSE after PCA: 1.47191494792884"
    Feature       Gain      Cover  Frequency
     <char>      <num>      <num>      <num>
 1:     PC1 0.09909443 0.07550111 0.10717004
 2:     PC4 0.09016690 0.11571541 0.09346933
 3:     PC7 0.09000852 0.11626830 0.08890242
 4:    PC10 0.08790530 0.07311808 0.08311767
 5:     PC8 0.08737000 0.08390742 0.08022530
 6:     PC6 0.08286829 0.07890278 0.08007307
 7:    PC12 0.08167177 0.07841632 0.07946415
 8:     PC3 0.08095993 0.07791235 0.08235652
 9:    PC11 0.07903179 0.07692049 0.07413609
10:     PC9 0.07605811 0.06718051 0.07291825
11:     PC5 0.07469770 0.09240874 0.08022530
12:     PC2 0.07016726 0.06374848 0.07794185
```



CONCLUSION FOR REGRESSION

```
print(paste("Linear Regression RMSE after PCA with scaling:", linear_rms
e))
print(paste("Random Forest RMSE after PCA:", rf_rmse))
print(paste("XGBoost RMSE after PCA:", xgb_rmse))

best_model_name <- switch(which.min(c(linear_rmse, rf_rmse, xgb_rmse)),
                          "Linear Regression",
                          "Random Forest",
                          "XGBoost")

print(paste("Best Model After PCA is:", best_model_name))
```

```
[1] "Linear Regression RMSE after PCA with scaling: 1.45154120479356"
[1] "Random Forest RMSE after PCA: 1.45051301428582"
[1] "XGBoost RMSE after PCA: 1.47191494792884"
[1] "Best Model After PCA is: Random Forest"
```

CLASSIFICATION OF MENTAL HEALTH:

Introduction to Classification Using Trees

Classification trees, such as Decision Trees and Support Vector Machines (SVM), are powerful tools in machine learning used to predict categorical outcomes. Unlike regression trees that predict continuous values, classification trees split the data into branches to classify the outcome variable into discrete categories. Decision Trees work by recursively partitioning the data space and fitting a simple prediction model within each partition. This method is intuitive and easy to visualize, offering a straightforward interpretation of how decisions are made.

Data Preparation and Feature Engineering:

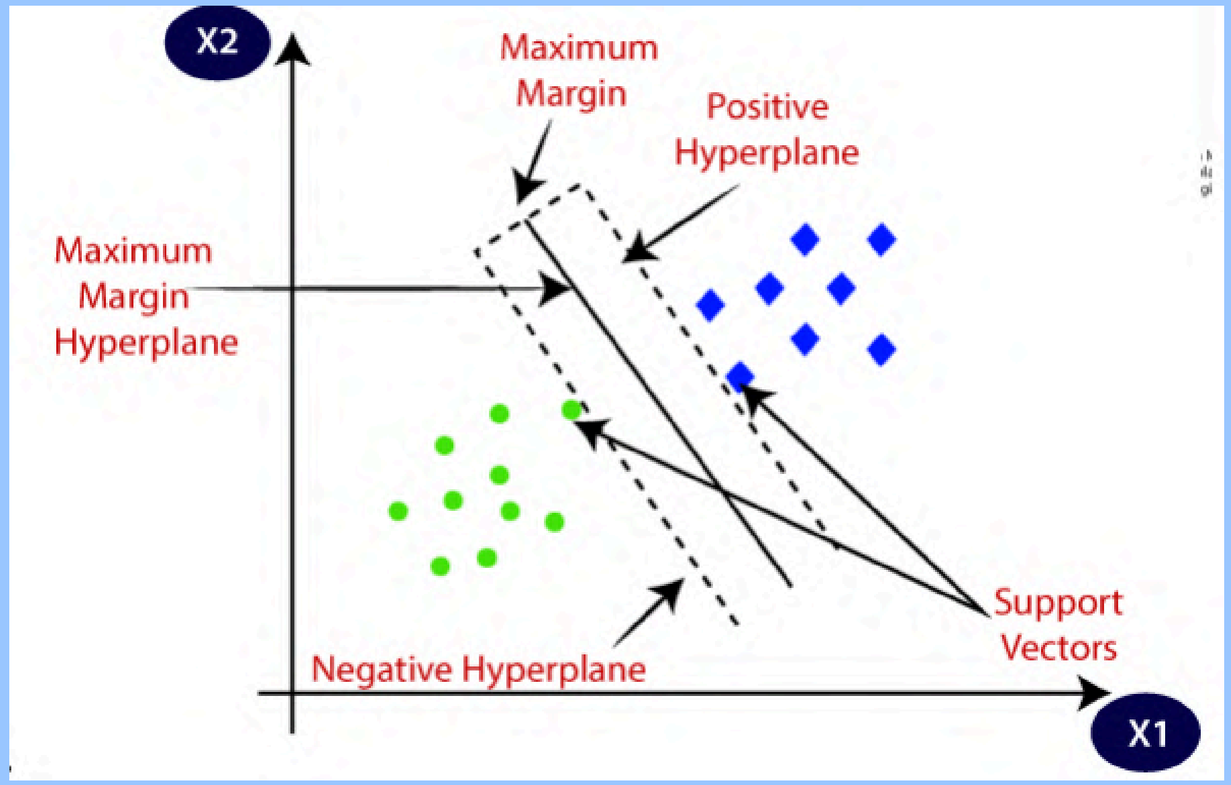
- Utilize a comprehensive dataset containing information on mental health, technology usage, and sleep patterns.
- Create a new target variable, "Sleep_Quality," derived from existing features using a custom function that considers age, gender, screen time, sleep hours, stress levels, physical activity, and various forms of technology usage.
- Convert categorical variables to appropriate numeric representations for model compatibility

CLASSIFICATION OF MENTAL HEALTH:

Model Development and Evaluation:

Implement multiple classification algorithms, including:
- a) Random Forest
- b) XGBoost
- c) Support Vector Machines (SVM)

The SVM model aims to find the hyperplane that best separates the classes of Sleep_Quality in a high-dimensional space. This approach is particularly effective for complex classification tasks where the classes are not linearly separable.

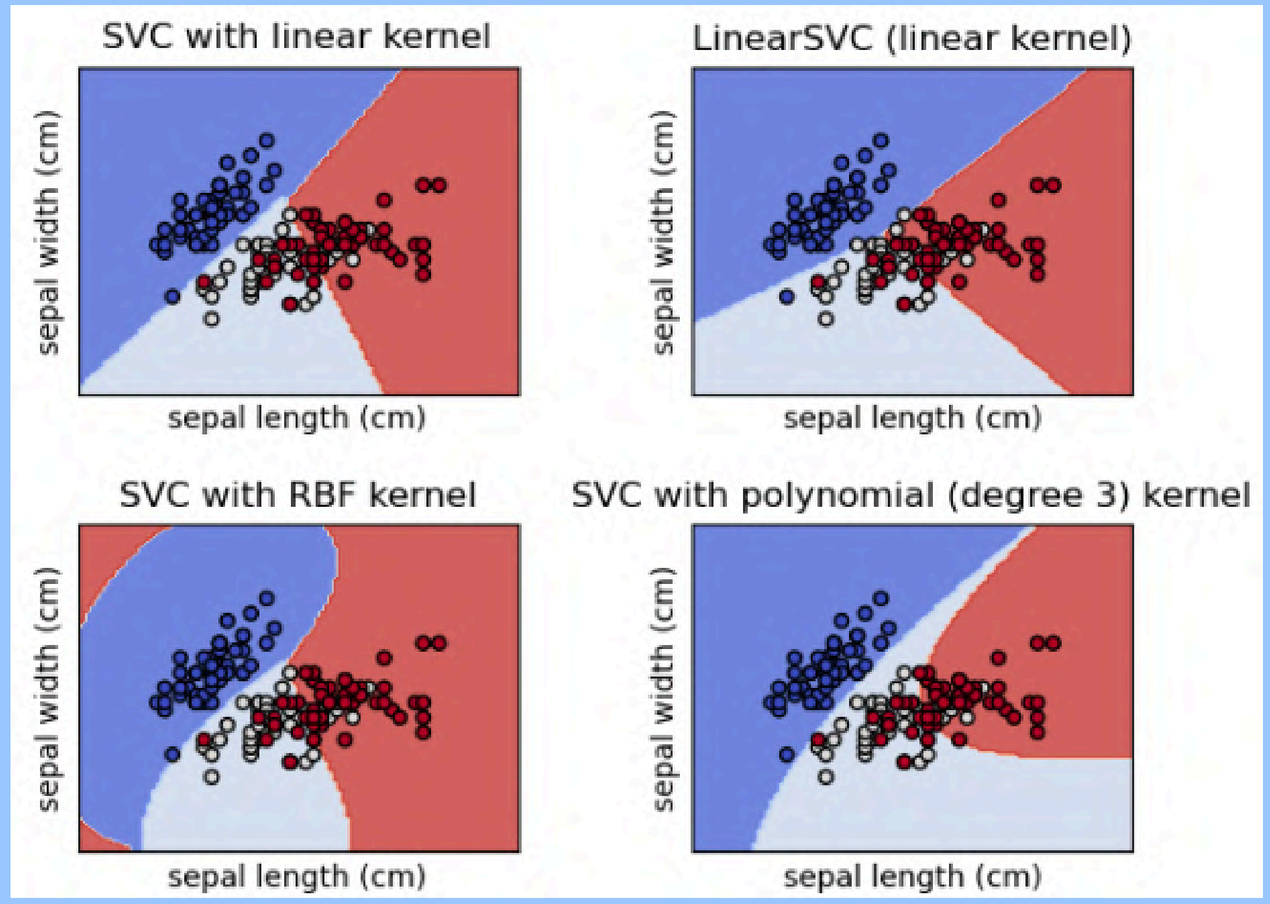


CLASSIFICATION OF MENTAL HEALTH:

Model Development and Evaluation:

VMs work by finding the optimal hyperplane that maximizes the margin between different classes. The kernel trick is employed to transform data into higher dimensions, making it easier to find a separating hyperplane in non-linear datasets.

The effectiveness of the SVM model is evaluated using confusion matrices, which provide a clear depiction of true positive, false positive, true negative, and false negative rates. Additionally, visualization of the decision boundary in a reduced feature space can offer insights into how the model separates different classes.



CLASSIFICATION OF MENTAL HEALTH:

Multinomial Logistic Regression

Multinomial Logistic Regression (MLR) is used in this project to model the relationship between multiple explanatory variables and a categorical response variable, Sleep_Quality, which has more than two levels (e.g., "Good," "Fair," "Poor"). The goal is to estimate the probabilities of different outcomes based on the input features.

MLR is an extension of binary logistic regression and is particularly useful when the response variable is categorical with more than two levels. It uses the log-odds of the probabilities of the outcomes as a linear combination of the predictors.

$$\Pr(Y = k|X = x) = \frac{e^{\beta_{k0}+\beta_{k1}x_1+\cdots+\beta_{kp}x_p}}{1+\sum_{l=1}^{K-1} e^{\beta_{l0}+\beta_{l1}x_1+\cdots+\beta_{lp}x_p}} \quad (4.10)$$

for $k = 1, \ldots, K-1$, and

$$\Pr(Y = K|X = x) = \frac{1}{1+\sum_{l=1}^{K-1} e^{\beta_{l0}+\beta_{l1}x_1+\cdots+\beta_{lp}x_p}}. \quad (4.11)$$

It is not hard to show that for $k = 1, \ldots, K-1$,

$$\log\left(\frac{\Pr(Y = k|X = x)}{\Pr(Y = K|X = x)}\right) = \beta_{k0} + \beta_{k1}x_1 + \cdots + \beta_{kp}x_p. \quad (4.12)$$

CLASSIFICATION OF MENTAL HEALTH:

Data Loading and Preprocessing

```
# Read data
data <- read.csv("/Users/aravind1803/Downloads/R/mental_health_with_sleep_qualit
y.csv")

# Remove unnecessary columns
data <- data %>%
  select(-c(User_ID, Sleep_Hours))

# Convert categorical variables to factors
categorical_cols <- c("Gender", "Mental_Health_Status", "Work_Environment_Impac
t",
                      "Support_Systems_Access", "Online_Support_Usage", "Sleep_Qu
ality")
data[categorical_cols] <- lapply(data[categorical_cols], factor)

# Feature Engineering
data <- data %>%
  mutate(
    Total_Screen_Time = Technology_Usage_Hours + Social_Media_Usage_Hours + Gami
ng_Hours,
    Screen_Activity_Ratio = Screen_Time_Hours / (Physical_Activity_Hours + 0.1),
    Tech_Social_Ratio = Technology_Usage_Hours / (Social_Media_Usage_Hours + 0.
1)
  )
```

CLASSIFICATION OF MENTAL HEALTH:

Model 1 - Random Forest

```
rf_model <- train(
  Sleep_Quality ~ .,
  data = train_data,
  method = "rf",
  trControl = trainControl(
    method = "cv",
    number = 5,  # Reduced from 10 to 5 for faster execution
    verboseIter = TRUE
  ),
  tuneGrid = expand.grid(mtry = sqrt(ncol(train_data)-1)), # Simplified tuning
  ntree = 100  # Reduced number of trees
)
```

```
## + Fold1: mtry=3.873
## - Fold1: mtry=3.873
## + Fold2: mtry=3.873
## - Fold2: mtry=3.873
## + Fold3: mtry=3.873
## - Fold3: mtry=3.873
## + Fold4: mtry=3.873
## - Fold4: mtry=3.873
## + Fold5: mtry=3.873
## - Fold5: mtry=3.873
## Aggregating results
## Fitting final model on full training set
```

CLASSIFICATION OF MENTAL HEALTH:

Model 2 – XGBoost

```
xgb_model <- train(
  Sleep_Quality ~ .,
  data = train_data,
  method = "xgbTree",
  trControl = ctrl,
  metric = "Accuracy",
  tuneGrid = expand.grid(
    nrounds = 100,
    max_depth = 6,
    eta = 0.3,
    gamma = 0,
    colsample_bytree = 1,
    min_child_weight = 1,
    subsample = 1
  ),
  verbose = FALSE
)
```

Model 3 - Multinomial Logistic Regression

```
multinom_model <- train(
  Sleep_Quality ~ .,
  data = train_data,
  method = "multinom",
  trControl = ctrl,
  metric = "Accuracy",
  trace = FALSE
)
```

CLASSIFICATION OF MENTAL HEALTH:

Model 4 - Support Vector Machine

```
# First, create proper scaling and preprocessing
preProcess_range <- preProcess(train_data[,-which(names(train_data) == "Sleep_Qu
ality")],
                               method = c("center", "scale"))

# Apply preprocessing to both training and test data
train_data_preprocessed <- predict(preProcess_range, train_data)
test_data_preprocessed <- predict(preProcess_range, test_data)

# Create a more robust control parameter
ctrl_svm <- trainControl(
  method = "cv",
  number = 5,
  classProbs = TRUE,
  verboseIter = TRUE,
  allowParallel = TRUE
)
```

```
# Define a specific tuning grid for SVM
svm_grid <- expand.grid(
  sigma = c(0.01, 0.02, 0.025, 0.03, 0.035, 0.04, 0.05),
  C = c(0.5, 1, 2, 4, 8, 16)
)

# Train SVM with modified parameters
set.seed(123)
svm_model <- train(
  Sleep_Quality ~ .,
  data = train_data_preprocessed,
  method = "svmRadial",
  trControl = ctrl_svm,
  tuneGrid = svm_grid,
  preProcess = NULL,  # Already preprocessed
  verbose = FALSE
)
```

CLASSIFICATION OF MENTAL HEALTH:

Model 4 - Support Vector Machine

```
## - Fold5: sigma=0.040, C=16.0
## + Fold5: sigma=0.050, C=16.0
## - Fold5: sigma=0.050, C=16.0
```

```
## Aggregating results
## Selecting tuning parameters
## Fitting sigma = 0.02, C = 0.5 on full training set
```

CLASSIFICATION OF MENTAL HEALTH:

Model 5 - Decision Tree

```
dt_model <- train(
  Sleep_Quality ~ .,
  data = train_data,
  method = "rpart",
  trControl = ctrl,
  metric = "Accuracy",
  tuneLength = 10
)
```

Model 6 - Gradient Boosting

```
gbm_model <- train(
  Sleep_Quality ~ .,
  data = train_data,
  method = "gbm",
  trControl = ctrl,
  metric = "Accuracy",
  tuneLength = 10,
  verbose = FALSE
)
```

CLASSIFICATION OF MENTAL HEALTH:

Model Evaluation

```
print(all_results[order(-all_results$Accuracy),])
```

```
##                 Model  Accuracy     Kappa
## Accuracy2     Multinom 0.5258419 0.1759037
## Accuracy4 DecisionTree 0.5255085 0.1681161
## Accuracy5          GBM 0.5245082 0.2077390
## Accuracy1      XGBoost 0.5225075 0.2144040
## Accuracy            RF 0.5195065 0.2050571
## Accuracy3          SVM 0.4924975 0.0000000
```

Ensemble Model

```
# Create ensemble predictions
ensemble_predictions <- data.frame(
  RF = predict(rf_model, test_data),
  XGB = predict(xgb_model, test_data),
  Multinom = predict(multinom_model, test_data),
  SVM = predict(svm_model, test_data),
  DT = predict(dt_model, test_data),
  GBM = predict(gbm_model, test_data)
)

# Majority voting
ensemble_final <- apply(ensemble_predictions, 1, function(x) {
  names(which.max(table(x)))
})
ensemble_final <- factor(ensemble_final, levels = levels(test_data$Sleep_Qualit
y))
```

CLASSIFICATION OF MENTAL HEALTH:

Ensemble Model

```
# Evaluate ensemble
ensemble_cm <- confusionMatrix(ensemble_final, test_data$Sleep_Quality)
print("Ensemble Model Results:")
```

```
## Statistics by Class:
##
##                      Class: Fair Class: Good Class: Poor
## Sensitivity              0.14172      0.48462      0.8003
## Specificity              0.90986      0.88987      0.3627
## Pos Pred Value           0.44099      0.48000      0.5493
## Neg Pred Value           0.67874      0.89167      0.6517
## Prevalence               0.33411      0.17339      0.4925
## Detection Rate           0.04735      0.08403      0.3941
## Detection Prevalence     0.10737      0.17506      0.7176
## Balanced Accuracy        0.52579      0.68725      0.5815
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Fair Good Poor
##       Fair  142   40  140
##       Good  118  252  155
##       Poor  742  228 1182
##
## Overall Statistics
##
##                Accuracy : 0.5255
##                  95% CI : (0.5075, 0.5435)
##     No Information Rate : 0.4925
##     P-Value [Acc > NIR] : 0.0001604
##
##                   Kappa : 0.1824
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
```

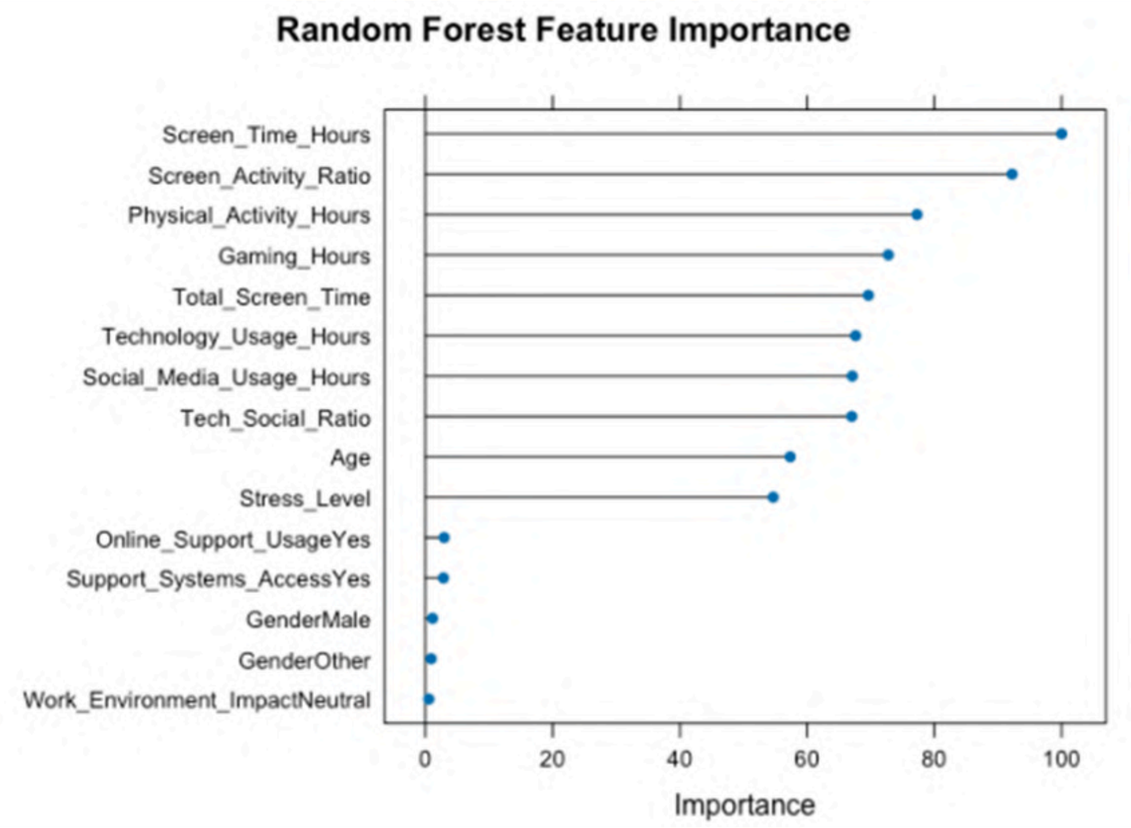
CLASSIFICATION OF MENTAL HEALTH:

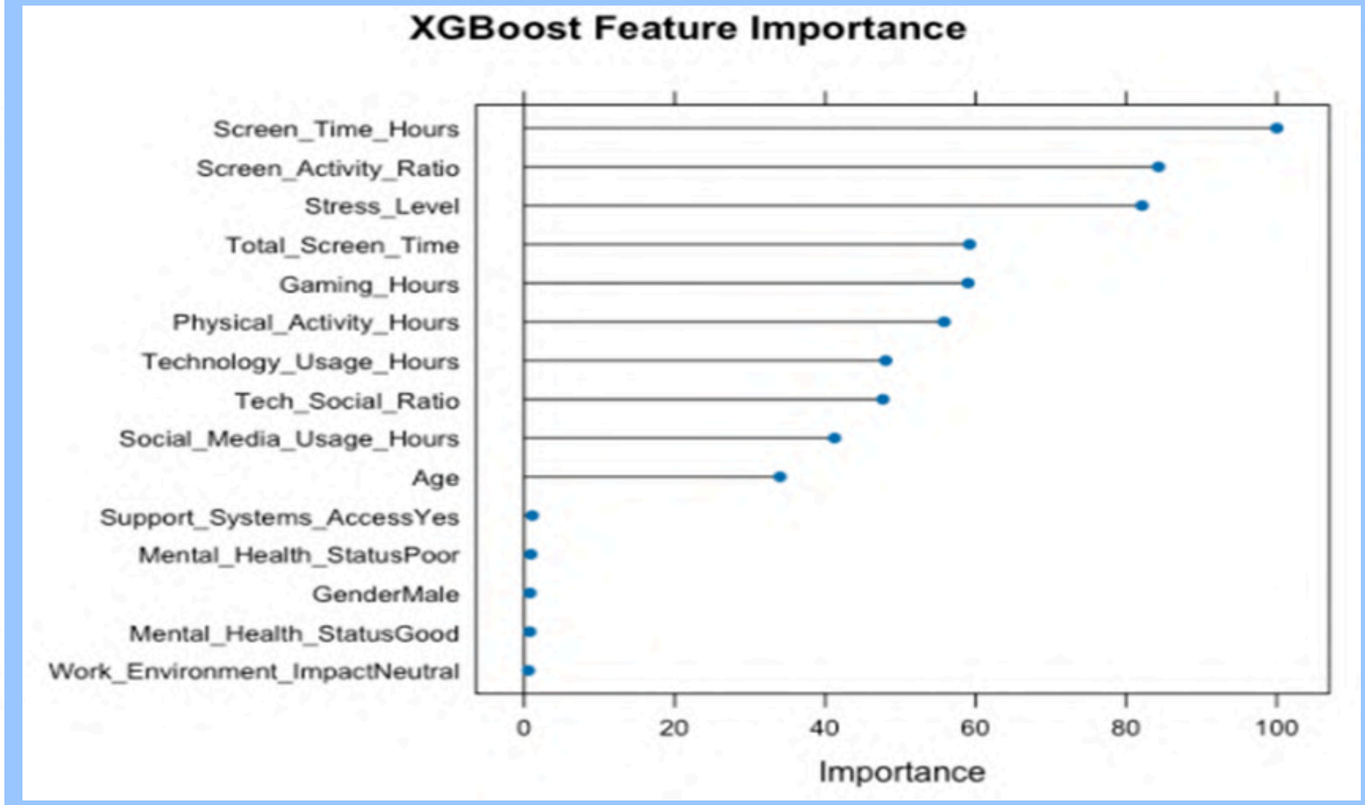
Feature Importance Analysis

```
# Random Forest importance
rf_importance <- varImp(rf_model)
# XGBoost importance
xgb_importance <- varImp(xgb_model)

# Plot feature importance
plot(rf_importance, top = 15, main = "Random Forest Feature Importance")
```

CLASSIFICATION OF MENTAL HEALTH:

Feature Importance Analysis

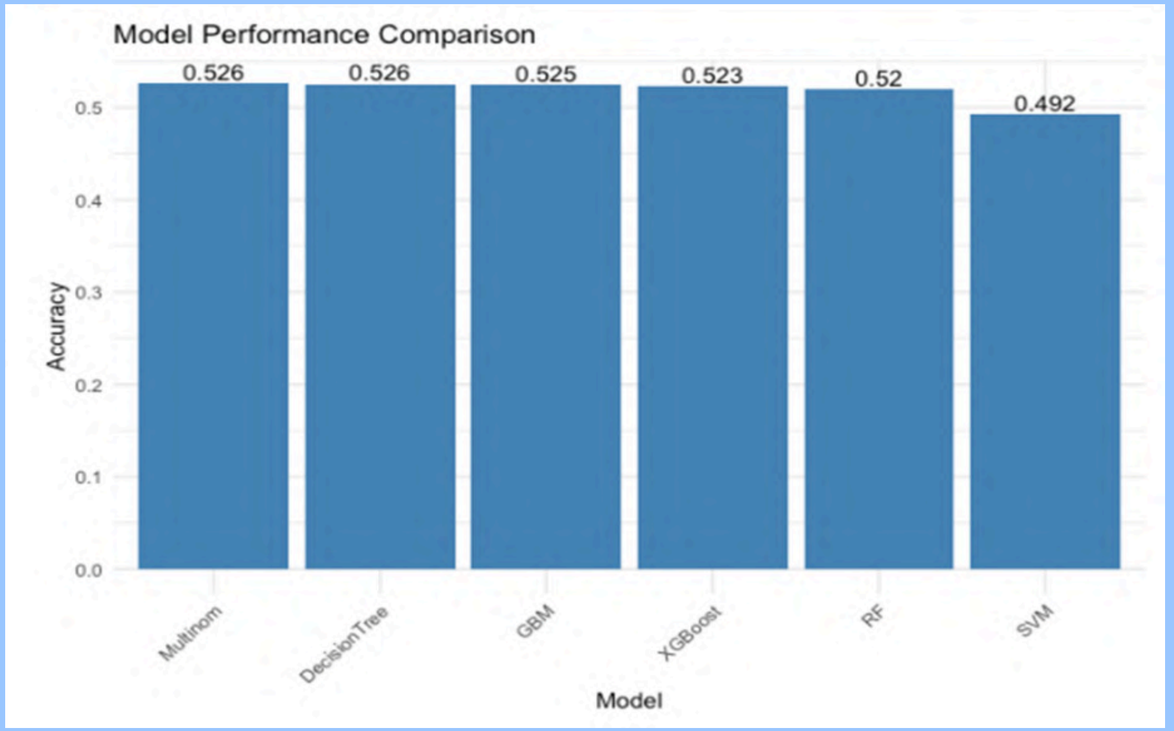




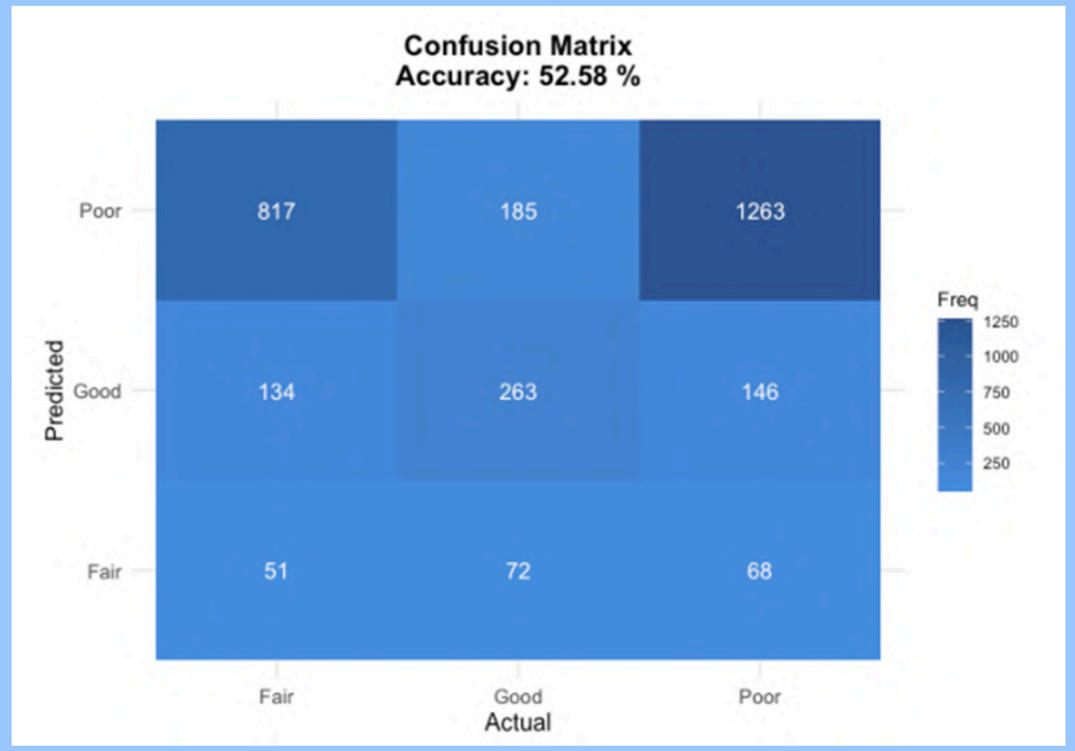
CLASSIFICATION OF MENTAL HEALTH:

Model Performance Visualization

```
# Create performance plot
performance_plot <- ggplot(all_results, aes(x = reorder(Model, -Accuracy), y = A
ccuracy)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  geom_text(aes(label = round(Accuracy, 3)), vjust = -0.3) +
  theme_minimal() +
  labs(title = "Model Performance Comparison",
       x = "Model",
       y = "Accuracy") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

print(performance_plot)
```



CONFUSION MATRIX




```
## Accuracy: 0.5258
```

```
## Kappa: 0.1759
```

```
##              Sensitivity Specificity Pos Pred Value Neg Pred Value Precision
## Class: Fair    0.0508982   0.9298948      0.2670157      0.6613248 0.2670157
## Class: Good    0.5057692   0.8870512      0.4843462      0.8953583 0.4843462
## Class: Poor    0.8551117   0.3416557      0.5576159      0.7084469 0.5576159
##                 Recall         F1 Prevalence Detection Rate Detection Prevalen
ce
## Class: Fair 0.0508982 0.08549874  0.3341114     0.01700567             0.06368
79
## Class: Good 0.5057692 0.49482596  0.1733911     0.08769590             0.18106
04
## Class: Poor 0.8551117 0.67504009  0.4924975     0.42114038             0.75525
18
##              Balanced Accuracy
## Class: Fair          0.4903965
## Class: Good          0.6964102
## Class: Poor          0.5983837
```

CONCLUSION

This project provides a comprehensive analysis of the relationship between technology usage and mental health through various statistical methods, visualizations, and predictive modeling techniques.

The dataset includes key variables such as daily screen time, stress levels, sleep quality, and self-reported mental health scores, allowing for a multifaceted exploration of how these factors interact.

- Dataset Analysis: The initial phase involved cleaning the dataset and performing exploratory data analysis (EDA). Key insights were derived from categorical and numerical data analysis, including correlation matrices that highlighted relationships among variables. For instance, a strong correlation was observed between screen time and stress levels, indicating that increased technology usage may contribute to higher stress.

CONCLUSION

- Visualizations: Various visualizations were employed to depict distributions and relationships within the data.

- Normal Distribution: Technology usage hours followed a normal distribution, suggesting most users engage with technology for a moderate amount of time.
- Poisson Distribution: Gaming hours were modeled using a Poisson distribution, indicating that gaming is infrequent among users.
- Exponential Distribution: Sleep hours displayed an exponential distribution, reflecting varied sleeping patterns among individuals.
- Gamma Distribution: Screen time hours showed a gamma distribution, highlighting right-skewed usage patterns where most users have moderate screen time but some exhibit very high usage.

CONCLUSION

Statistical Testing: Several tests were conducted to validate assumptions about the data

- The Kolmogorov-Smirnov test and Anderson-Darling test were employed to assess normality.
- Chi-squared tests were used to evaluate categorical variables where applicable.

Regression Analysis: Multiple regression models were developed to predict mental health scores based on the independent variables.

- A linear regression model provided baseline predictions, while polynomial regression explored non-linear relationships.
- Decision trees and ensemble methods like Random Forest and XGBoost were utilized for their robustness in handling complex interactions between features.

CONCLUSION

Classification Models: Various classification algorithms were tested to classify mental health scores:

- Models included Random Forest, XGBoost, Multinomial Logistic Regression, Support Vector Machines (SVM), Decision Trees, and Gradient Boosting.
- Performance metrics such as confusion matrices and feature importance analyses indicated that ensemble models generally outperformed single classifiers in accuracy.

CONCLUSION

The analysis demonstrates significant correlations between technology usage patterns and mental health indicators. The findings suggest that prolonged screen time may negatively impact mental well-being, particularly through increased stress levels.

The predictive models developed provide valuable insights into how daily habits can be quantified to inform strategies for improving mental health outcomes.Overall, this project underscores the importance of data-driven approaches in understanding behavioral patterns related to technology use and mental health.

Future work could expand on these findings by incorporating longitudinal data or exploring interventions aimed at mitigating negative impacts associated with high technology usage.

LINKS

GOOGLE DRIVE

THANK YOU

Presented by :

22BRS1021 S Mukunth

22BRS1095 Kishore K G

22BRS1099 ARAVIND N

22BRS1357 Abhinav Balakrishnan