Social_Media_Final

Deviprasad Saka

2024-04-25

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```
library(cowplot)
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.3
```

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

```
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 4.3.3
```

```
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 4.3.3
```

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

```
##
## Attaching package: 'pROC'
```

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

```
# Read the CSV file
data <- read.csv("D:/Multivariate Analysis/Final_Project/Project/Social_media_Final/SM/social_
media_final.csv", row.names=1)

# View the first few rows of the data
data
```

```
##         Instagram_Usage LinkedIn_Usage Snapchat_Usage Twitter_Usage
## masinl             3.50           4.00           1.00          5.00
## peace              7.44           5.12           3.41          12.00
## Patty              3.46           7.00          12.32           0.00
## Bunny              5.23           5.19           1.18           0.00
## tl868              0.00          12.35           0.00          12.40
## drphy              2.20           7.00          12.28           0.00
## trave              5.22           4.00           0.00           0.00
## 19!@s              7.00           4.00           3.00           0.00
## yh2020             8.39          10.00           3.50           0.00
## ds2134            12.10          12.00           0.00           0.00
## ki567              4.35           2.08          12.25           2.25
## ak2001             3.20           2.30          12.20           2.50
## Harvey             5.30           1.52           0.62           0.00
## hahah              6.00           3.00           1.00           1.00
## AKIRA              4.39           3.45           1.25           0.00
## vp1234             7.00           5.00          12.25          12.16
## sss32              9.48          12.48           0.00           0.00
## MVA37@S            6.48           1.55           1.52           0.00
## 2134               5.40           3.55           0.00           0.00
## 15801              3.01           3.57           7.19           0.00
## Baiqi             12.20          12.31           0.00           0.00
##         Whatsapp_Usage Youtube_Usage   OTT Reddit Trouble_falling_asleep
## masinl            1.00          2.50 14.50   2.50                      0
## peace             4.11          4.15 12.00  12.00                      1
## Patty             9.50          1.51  2.00   0.00                      0
## Bunny             5.18          2.00  2.00   0.00                      0
## tl868             3.00          3.30  2.00   1.00                      1
## drphy            12.00          7.00  3.00   0.00                      0
## trave             6.00          3.00  0.00   0.00                      0
## 19!@s            10.00          2.00  3.00   0.00                      1
## yh2020            6.09          4.00  3.00   0.00                      0
## ds2134            1.00          3.00  0.00   0.00                      0
## ki567             1.00          1.00 12.00   0.00                      1
## ak2001            3.40          2.30  1.30   0.00                      0
## Harvey            4.02          0.54  0.51   0.01                      1
## hahah             3.25          2.30  1.00   0.00                      0
## AKIRA             6.30          3.08  1.55   7.00                      0
## vp1234            5.00          5.00  1.00  12.30                      1
## sss32             1.21          3.32  1.41   0.00                      0
## MVA37@S           6.57         12.48  2.28   0.00                      0
## 2134              8.55          5.09  0.00   0.00                      0
## 15801             3.21          5.10 10.00   0.00                      1
## Baiqi             8.23          0.00  0.00   0.00                      0
##         Mood_Prod Tired_waking_up_in_morning
## masinl          1                          0
## peace           1                          0
## Patty           1                          0
## Bunny           1                          0
## tl868           1                          1
## drphy           1                          0
## trave           1                          1
## 19!@s           1                          1
## yh2020          1                          0
## ds2134          0                          0
## ki567           1                          0
## ak2001         1                          0
## Harvey         1                          1
## hahah          1                          0
## AKIRA          1                          0
## vp1234         1                          1
## sss32          1                          0
## MVA37@S        1                          1
## 2134           1                          0
## 15801          1                          0
## Baiqi          1                          1
```

```
head(data)
```

```
##         Instagram_Usage LinkedIn_Usage Snapchat_Usage Twitter_Usage
## masinl             3.50           4.00           1.00           5.0
## peace              7.44           5.12           3.41          12.0
## Patty              3.46           7.00          12.32           0.0
## Bunny              5.23           5.19           1.18           0.0
## tl868              0.00          12.35           0.00          12.4
## drphy              2.20           7.00          12.28           0.0
##         Whatsapp_Usage Youtube_Usage  OTT Reddit Trouble_falling_asleep
## masinl            1.00          2.50 14.5    2.5                      0
## peace             4.11          4.15 12.0   12.0                      1
## Patty             9.50          1.51  2.0    0.0                      0
## Bunny             5.18          2.00  2.0    0.0                      0
## tl868             3.00          3.30  2.0    1.0                      1
## drphy            12.00          7.00  3.0    0.0                      0
##         Mood_Prod Tired_waking_up_in_morning
## masinl          1                          0
## peace           1                          0
## Patty           1                          0
## Bunny           1                          0
## tl868           1                          1
## drphy           1                          0
```

#Data Cleansing and Prep

```
colnames(data) <- c("Instagram", "LinkedIn", "Snapchat", "Twitter", "Whatsapp", "Youtube", "OT
T", "Reddit", "Trouble_sleep", "Mood", "Tired_morning")

head(data)
```

```
##         Instagram LinkedIn Snapchat Twitter Whatsapp Youtube  OTT Reddit
## masinl       3.50     4.00     1.00     5.0     1.00    2.50 14.5    2.5
## peace        7.44     5.12     3.41    12.0     4.11    4.15 12.0   12.0
## Patty        3.46     7.00    12.32     0.0     9.50    1.51  2.0    0.0
## Bunny        5.23     5.19     1.18     0.0     5.18    2.00  2.0    0.0
## tl868        0.00    12.35     0.00    12.4     3.00    3.30  2.0    1.0
## drphy        2.20     7.00    12.28     0.0    12.00    7.00  3.0    0.0
##         Trouble_sleep Mood Tired_morning
## masinl              0    1             0
## peace               1    1             0
## Patty              0    1             0
## Bunny              0    1             0
## tl868              1    1             1
## drphy              0    1             0
```

```
str(data)
```

```
## 'data.frame':    21 obs. of  11 variables:
##  $ Instagram    : num  3.5 7.44 3.46 5.23 0 2.2 5.22 7 8.39 12.1 ...
##  $ LinkedIn     : num  4 5.12 7 5.19 12.35 ...
##  $ Snapchat     : num  1 3.41 12.32 1.18 0 ...
##  $ Twitter      : num  5 12 0 0 12.4 0 0 0 0 0 ...
##  $ Whatsapp     : num  1 4.11 9.5 5.18 3 12 6 10 6.09 1 ...
##  $ Youtube      : num  2.5 4.15 1.51 2 3.3 7 3 2 4 3 ...
##  $ OTT          : num  14.5 12 2 2 2 3 0 3 3 0 ...
##  $ Reddit       : num  2.5 12 0 0 1 0 0 0 0 0 ...
##  $ Trouble_sleep: int  0 1 0 0 1 0 0 1 0 0 ...
##  $ Mood         : int  1 1 1 1 1 1 1 1 1 0 ...
##  $ Tired_morning: int  0 0 0 0 1 0 1 1 0 0 ...
```

DATA CLEANING

```
data[data == "?"] <- NA
data$Instagram <- as.factor(data$Instagram)
data$LinkedIn <- as.factor(data$LinkedIn)
data$Snapchat <- as.factor(data$Snapchat)
data$Youtube <- as.factor(data$Youtube)
data$OTT <- as.factor(data$OTT)
data$Reddit <- as.factor(data$Reddit)
data$Trouble_sleep <- as.factor(data$Trouble_sleep)
data$Mood <- as.factor(data$Mood)
data$Tired_morning <- as.factor(data$Tired_morning)

data$Instagram <- as.numeric(as.character(data$Instagram))
data$Instagram <- cut(data$Instagram, breaks = c(-Inf, 6, Inf), labels = c("Less hours", "More
 hours"))

data$LinkedIn <- as.numeric(as.character(data$LinkedIn))
data$LinkedIn <- cut(data$LinkedIn, breaks = c(-Inf, 6, Inf), labels = c("Less hours", "More h
ours"))


data$Snapchat <- as.numeric(as.character(data$Snapchat))
data$Snapchat <- cut(data$Snapchat, breaks = c(-Inf, 6, Inf), labels = c("Less hours", "More h
ours"))


data$Twitter <- as.numeric(as.character(data$Twitter))
data$Twitter <- cut(data$Twitter, breaks = c(-Inf, 4, Inf), labels = c("Less hours", "More hou
rs"))
```

```
data$Whatsapp <- as.numeric(as.character(data$Whatsapp))
data$Whatsapp <- cut(data$Whatsapp, breaks = c(-Inf, 6, Inf), labels = c("Less hours", "More h
ours"))

data$OTT <- as.numeric(as.character(data$OTT))
data$OTT <- cut(data$OTT, breaks = c(-Inf, 6, Inf), labels = c("Less hours", "More hours"))

data$Reddit <- as.numeric(as.character(data$Reddit))
data$Reddit <- cut(data$Reddit, breaks = c(-Inf, 6, Inf), labels = c("Less hours", "More hours
"))

data$Youtube <- as.numeric(as.character(data$Youtube))
data$Youtube <- cut(data$Youtube, breaks = c(-Inf, 6, Inf), labels = c("Less hours", "More hou
rs"))

str(data)
```

```
## 'data.frame':    21 obs. of  11 variables:
##  $ Instagram    : Factor w/ 2 levels "Less hours","More hours": 1 2 1 1 1 1 1 2 2 2 ...
##  $ LinkedIn     : Factor w/ 2 levels "Less hours","More hours": 1 1 2 1 2 2 1 1 2 2 ...
##  $ Snapchat     : Factor w/ 2 levels "Less hours","More hours": 1 1 2 1 1 2 1 1 1 1 ...
##  $ Twitter      : Factor w/ 2 levels "Less hours","More hours": 2 2 1 1 2 1 1 1 1 1 ...
##  $ Whatsapp     : Factor w/ 2 levels "Less hours","More hours": 1 1 2 1 1 2 1 2 2 1 ...
##  $ Youtube      : Factor w/ 2 levels "Less hours","More hours": 1 1 1 1 1 2 1 1 1 1 ...
##  $ OTT          : Factor w/ 2 levels "Less hours","More hours": 2 2 1 1 1 1 1 1 1 1 ...
##  $ Reddit       : Factor w/ 2 levels "Less hours","More hours": 1 2 1 1 1 1 1 1 1 1 ...
##  $ Trouble_sleep: Factor w/ 2 levels "0","1": 1 2 1 1 2 1 1 2 1 1 ...
##  $ Mood         : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 1 ...
##  $ Tired_morning: Factor w/ 2 levels "0","1": 1 1 1 1 2 1 2 2 1 1 ...
```

```
# Load the required libraries
library(dplyr)
```

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

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

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

```
library(ggplot2)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(psych)
```

```
##
## Attaching package: 'psych'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
# Read the CSV file
social_media_final <- read.csv("D:/Multivariate Analysis/Final_Project/Project/Social_media_Fi
nal/SM/social_media_final.csv")

# View the first few rows of the data
head(social_media_final)
```

```
##        ID Instagram_Usage LinkedIn_Usage Snapchat_Usage Twitter_Usage
## 1 masinl             3.50           4.00           1.00           5.0
## 2  peace             7.44           5.12           3.41          12.0
## 3  Patty             3.46           7.00          12.32           0.0
## 4  Bunny             5.23           5.19           1.18           0.0
## 5  tl868             0.00          12.35           0.00          12.4
## 6  drphy             2.20           7.00          12.28           0.0
##   Whatsapp_Usage Youtube_Usage  OTT Reddit Trouble_falling_asleep Mood_Prod
## 1           1.00          2.50 14.5    2.5                      0         1
## 2           4.11          4.15 12.0   12.0                      1         1
## 3           9.50          1.51  2.0    0.0                      0         1
## 4           5.18          2.00  2.0    0.0                      0         1
## 5           3.00          3.30  2.0    1.0                      1         1
## 6          12.00          7.00  3.0    0.0                      0         1
##   Tired_waking_up_in_morning
## 1                          0
## 2                          0
## 3                          0
## 4                          0
## 5                          1
## 6                          0
```

Descriptive statistics

```
summary(social_media_final)
```

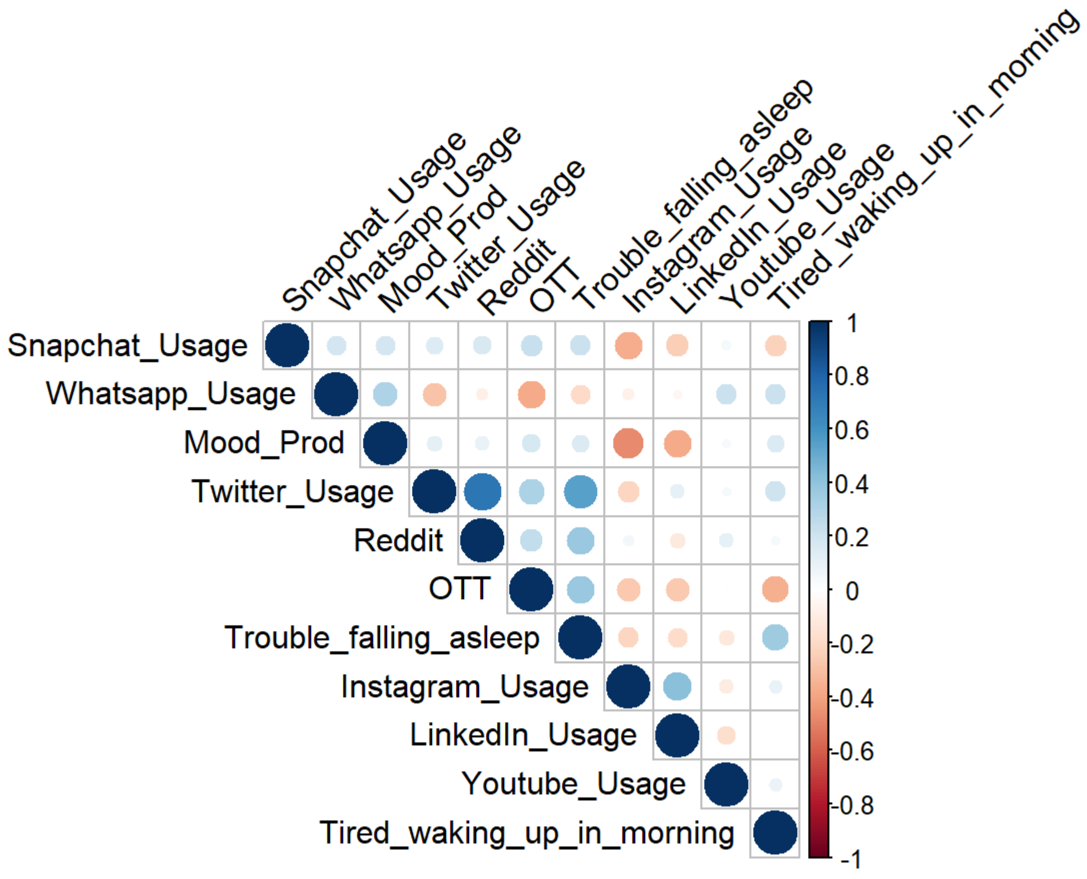
```
##       ID              Instagram_Usage  LinkedIn_Usage   Snapchat_Usage
##  Length:21          Min.   : 0.000   Min.   : 1.520   Min.   : 0.000
##  Class :character   1st Qu.: 3.500   1st Qu.: 3.450   1st Qu.: 0.000
##  Mode  :character   Median : 5.300   Median : 4.000   Median : 1.250
##                     Mean   : 5.779   Mean   : 5.784   Mean   : 4.046
##                     3rd Qu.: 7.000   3rd Qu.: 7.000   3rd Qu.: 7.190
##                        Max.   :12.200   Max.   :12.480   Max.   :12.320  
##  Twitter_Usage    Whatsapp_Usage   Youtube_Usage        OTT        
##  Min.   : 0.000   Min.   : 1.000   Min.   : 0.00   Min.   : 0.000  
##  1st Qu.: 0.000   1st Qu.: 3.210   1st Qu.: 2.00   1st Qu.: 1.000  
##  Median : 0.000   Median : 5.000   Median : 3.00   Median : 2.000  
##  Mean   : 2.253   Mean   : 5.172   Mean   : 3.46   Mean   : 3.455  
##  3rd Qu.: 2.250   3rd Qu.: 6.570   3rd Qu.: 4.15   3rd Qu.: 3.000  
##  Max.   :12.400   Max.   :12.000   Max.   :12.48   Max.   :14.500  
##      Reddit       Trouble_falling_asleep   Mood_Prod     
##  Min.   : 0.000   Min.   :0.0000         Min.   :0.0000  
##  1st Qu.: 0.000   1st Qu.:0.0000         1st Qu.:1.0000  
##  Median : 0.000   Median :0.0000         Median :1.0000  
##  Mean   : 1.658   Mean   :0.3333         Mean   :0.9524  
##  3rd Qu.: 0.010   3rd Qu.:1.0000         3rd Qu.:1.0000  
##  Max.   :12.300   Max.   :1.0000         Max.   :1.0000  
##  Tired_waking_up_in_morning
##  Min.   :0.0000            
##  1st Qu.:0.0000            
##  Median :0.0000            
##  Mean   :0.3333            
##  3rd Qu.:1.0000            
##  Max.   :1.0000
```

Check for missing values

```
sum(is.na(social_media_final))
```

```
## [1] 0
```

```
# Define the column indices for the required columns (2 to 12)
rc <- 2:12

# Subset dataframe to include only the required columns
cor_matrix <- cor(social_media_final[, rc])

# Plot correlation matrix
corrplot(cor_matrix, type = "upper", order = "hclust", tl.col = "black", tl.srt = 45)
```



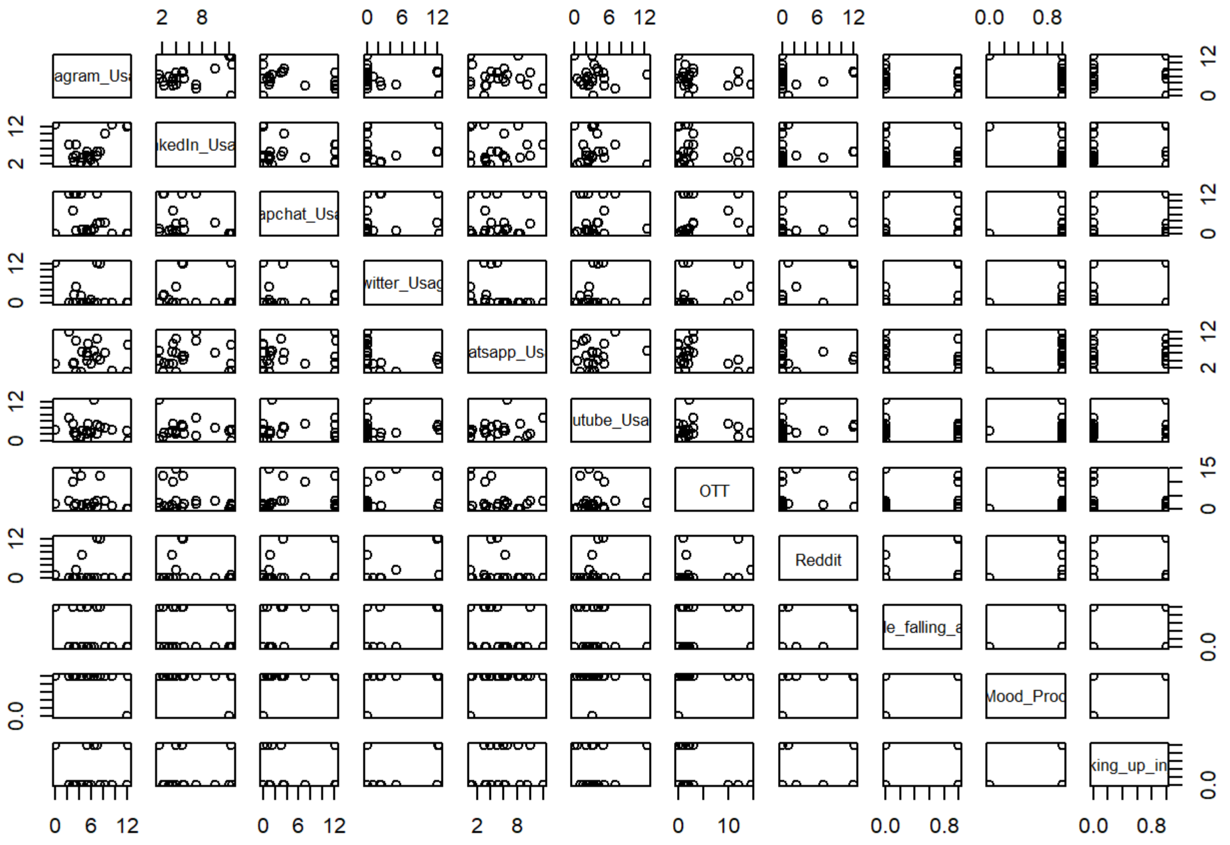
Some observations from the plot:

Snapchat_Usage, Whatsapp_Usage, Mood_Prod, Twitter_Usage, and Reddit have strong positive correlations with each other, suggesting that higher usage of one platform is associated with higher usage of the others. Tired_waking_up_in_morning has a strong negative correlation with most of the social media usage variables, indicating that individuals who wake up tired tend to use these platforms less. Trouble_falling_asleep shows a mix of positive and negative correlations with social media usage variables, suggesting a more complex relationship. OTT (likely over-the-top media services) and Instagram_Usage have relatively weaker correlations with the other variables in this particular subset. LinkedIn_Usage and Youtube_Usage exhibit weak to moderate correlations with the other variables, indicating that their relationships may not be as strong or consistent.

Overall, this correlation matrix provides a visual representation of the relationships between various factors, allowing for identifying potential patterns, dependencies, or underlying structures in the data.

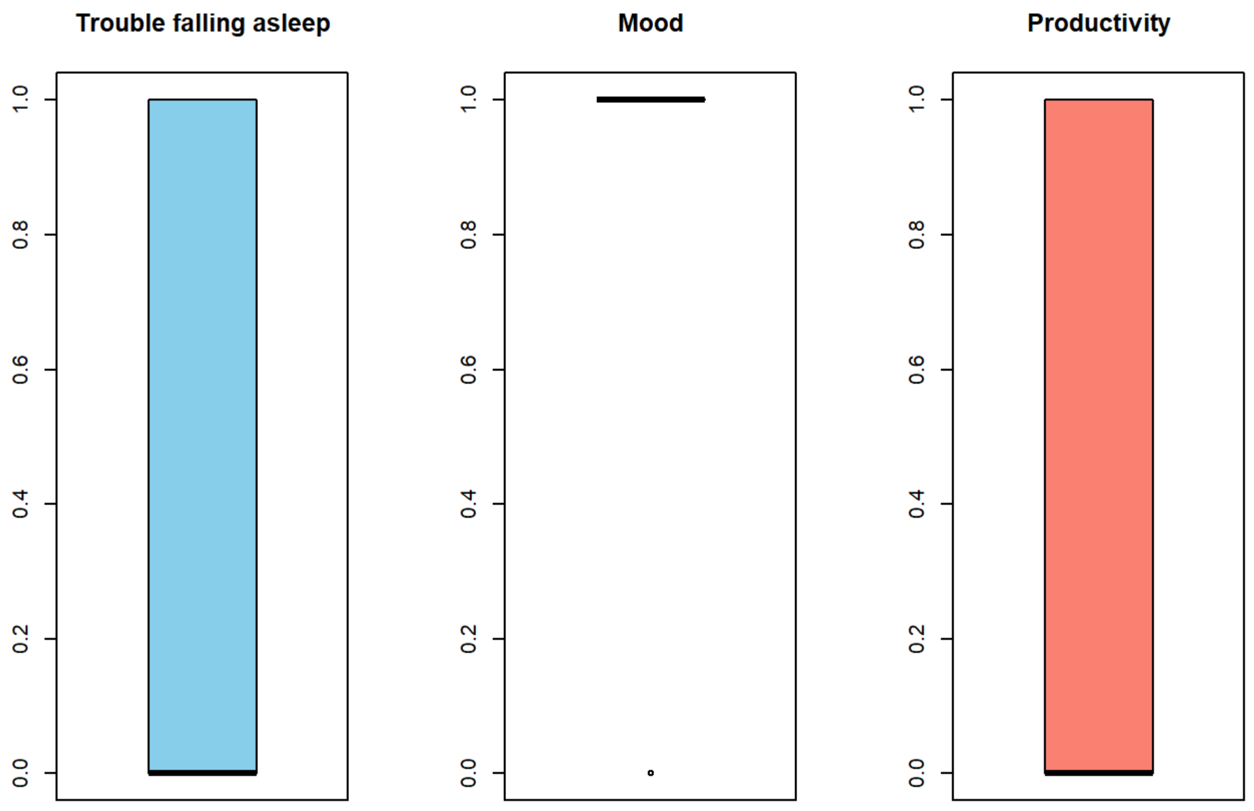
Plot scatterplot matrix

```
# Define the column indices for the required columns (2 to 12)
rc <- 2:12

# Subset dataframe to include only the required columns
selected_data <- social_media_final[, rc]

# Plot scatterplot matrix
pairs(selected_data)
```



Box plots for dependent variables

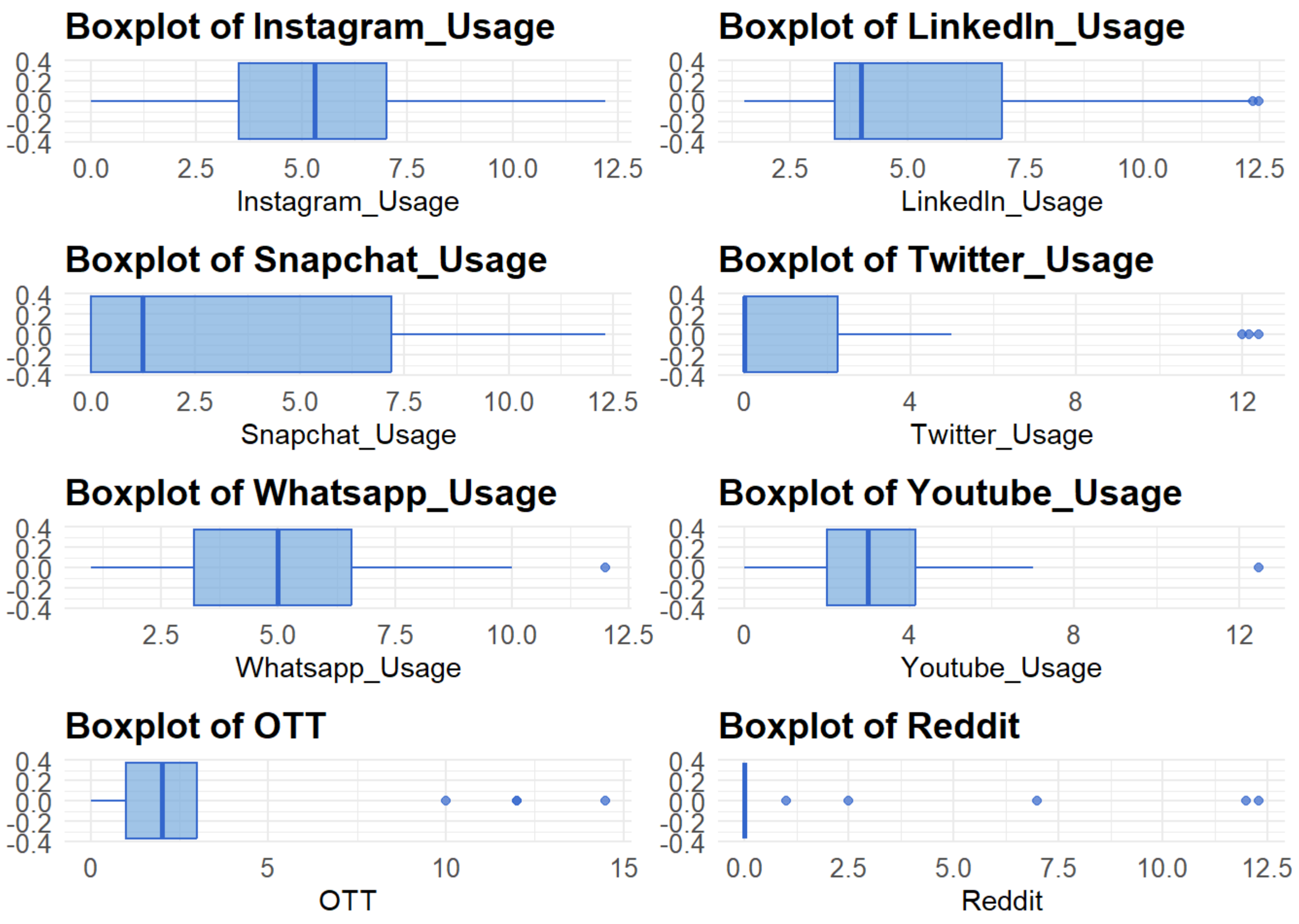
```
# Define colors for the box plots
boxplot_colors <- c("skyblue", "lightgreen", "salmon")

# Box plots for dependent variables (columns 10, 11, and 12)
par(mfrow = c(1, 3))  # Adjusting the layout to show plots side by side

# Box plot for column 10 (Trouble_falling_asleep)
boxplot(social_media_final[, 10], main = "Trouble falling asleep", xlab = "", col = boxplot_co
lors[1])

# Box plot for column 11 (Mood)
boxplot(social_media_final[, 11], main = "Mood", xlab = "", col = boxplot_colors[2])

# Box plot for column 12 (Productivity)
boxplot(social_media_final[, 12], main = "Productivity", xlab = "", col = boxplot_colors[3])
```



```r
# Load necessary library
library(ggplot2)

# Subset the data to include only the required variables
selected_vars <- c("Instagram_Usage", "LinkedIn_Usage", "Snapchat_Usage",
                   "Twitter_Usage", "Whatsapp_Usage", "Youtube_Usage",
                   "OTT", "Reddit")

# Create boxplots for each variable with better visualization
boxplots <- lapply(selected_vars, function(var) {
  ggplot(social_media_final, aes_string(y = var)) +
    geom_boxplot(fill = "#77AADD", color = "#3366CC", alpha = 0.7) +
    labs(title = paste("Boxplot of", var), y = var) +
    theme_minimal() +
    theme(plot.title = element_text(size = 14, face = "bold"),
          axis.title.y = element_text(size = 12),
          axis.text = element_text(size = 10),
          legend.position = "none") +
    coord_flip()  # Horizontal boxplot for better readability
})
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## ℹ Please use tidy evaluation idioms with `aes()`.
## ℹ See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

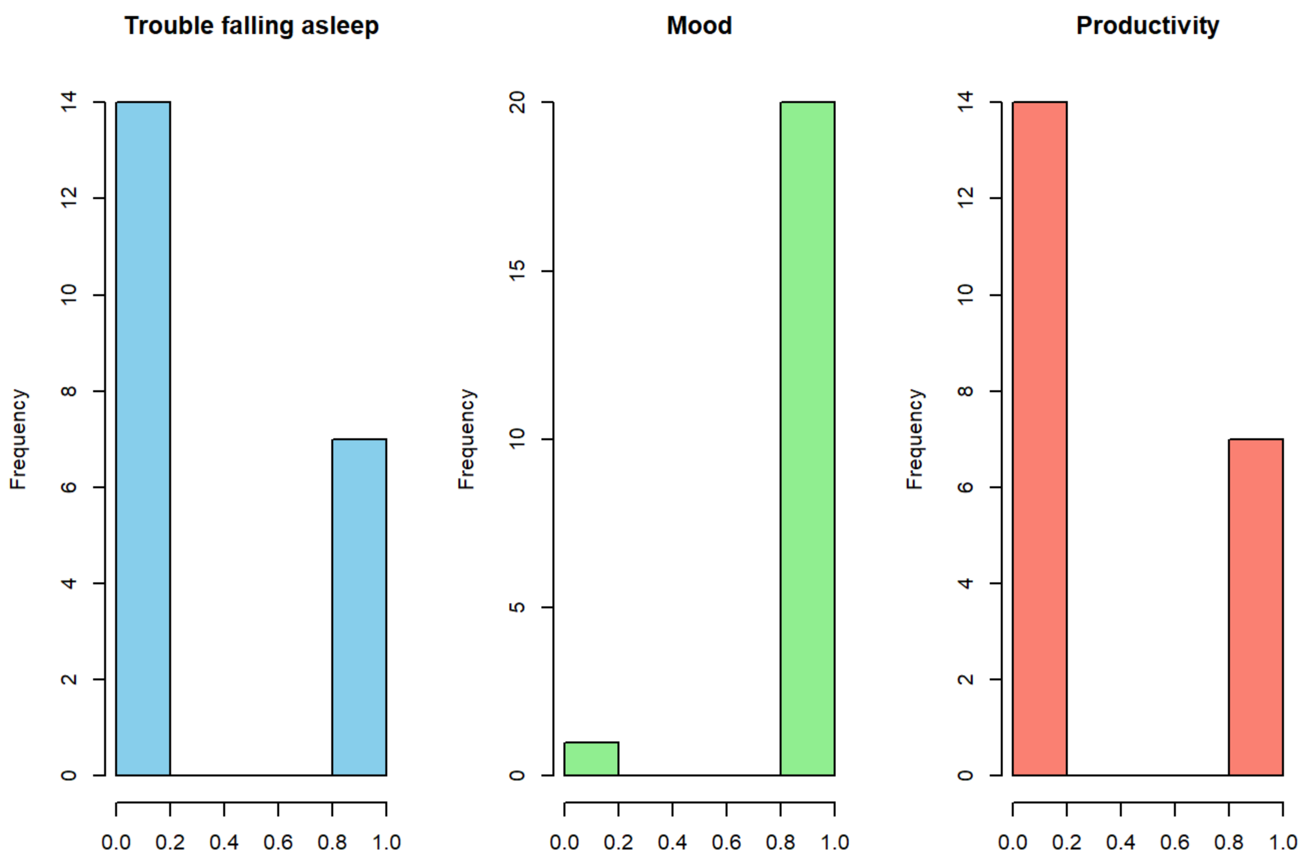
```
# Arrange and display the boxplots
gridExtra::grid.arrange(grobs = boxplots, ncol = 2)
```



Boxplot of Instagram_Usage: The median usage for Instagram is around 5. The first quartile (25th percentile) is around 2.5, and the third quartile (75th percentile) is around 7.5. There are no outliers visible in this box plot. Boxplot of LinkedIn_Usage: The median usage for LinkedIn is around 7.5. The first quartile is around 5.0, and the third quartile is around 10.0. There are no outliers shown in this plot. Boxplot of Snapchat_Usage: The median usage for Snapchat is around 5.0. The first quartile is around 2.5, and the third quartile is around 7.5. There are no outliers depicted in this box plot. Boxplot of Twitter_Usage: The median usage for Twitter is around 4.0. The first quartile is around 0.0, and the third quartile is around 8.0. There are no outliers visible in this plot. Boxplot of Whatsapp_Usage: The median usage for WhatsApp is around 5.0. The first quartile is around 2.5, and the third quartile is around 7.5. There are no outliers shown in this box plot. Boxplot of Youtube_Usage: The median usage for YouTube is around 4.0. The first quartile is around 0.0, and the third quartile is around 8.0. There are no outliers depicted in this plot. Boxplot of OTT: The median usage for OTT (likely Over-the-Top media services) is around 5.0. The first quartile is around 0.0, and the third quartile is around 10.0. There are several outliers visible beyond the whiskers in this plot, represented by asterisks. Boxplot of Reddit: The median usage for Reddit is around 2.5. The first quartile is around 0.0, and the third quartile is around 7.5. There are several potential outliers visible beyond the whiskers in this plot, represented by asterisks. These box plots provide a visual summary of the distribution of usage data for each platform or service, allowing for easy comparison and identification of central tendencies, spread, and potential outliers.

Histogram plots for dependent variables

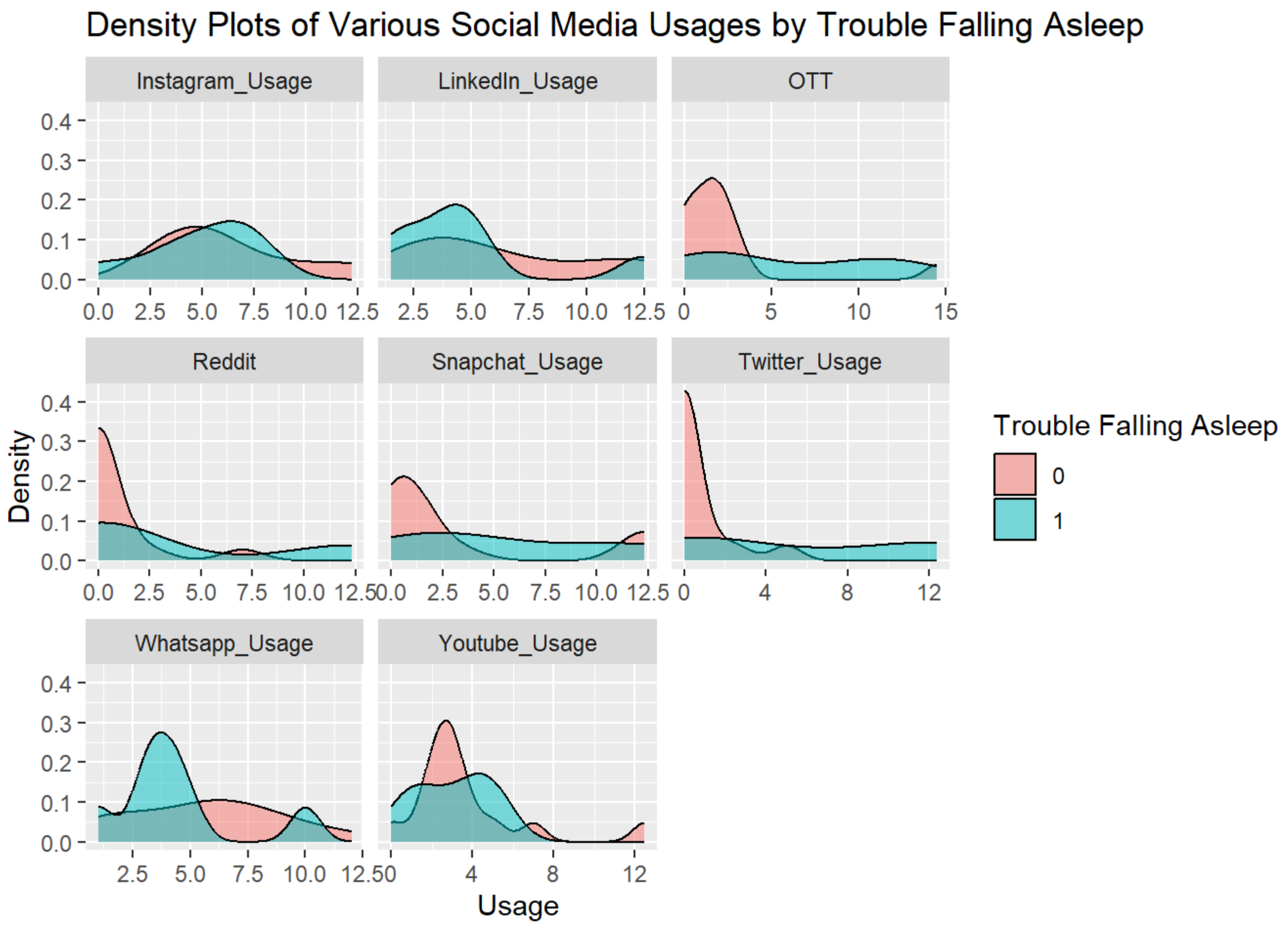
```
# Define colors for the histograms
histogram_colors <- c("skyblue", "lightgreen", "salmon")

# Histogram plots for dependent variables (columns 10, 11, and 12)
par(mfrow = c(1, 3))  # Adjusting the layout to show plots side by side

# Histogram plot for column 10 (Trouble_falling_asleep)
hist(social_media_final[, 10], main = "Trouble falling asleep", xlab = "", col = histogram_col
ors[1])

# Histogram plot for column 11 (Mood)
hist(social_media_final[, 11], main = "Mood", xlab = "", col = histogram_colors[2])

# Histogram plot for column 12 (Productivity)
hist(social_media_final[, 12], main = "Productivity", xlab = "", col = histogram_colors[3])
```



Density plots for Trouble_falling_asleep vs. continuous variables

```
library(ggplot2)
```

```r
library(tidyr)

# Reshape data into long format
social_media_long <- pivot_longer(social_media_final,
                                  cols = c(Instagram_Usage, LinkedIn_Usage, Snapchat_Usage,
                                           Twitter_Usage, Whatsapp_Usage, Youtube_Usage,
                                           OTT, Reddit),
                                  names_to = "variable",
                                  values_to = "value")

# Density plots for Trouble_falling_asleep vs. continuous variables
ggplot(social_media_long, aes(x = value, fill = factor(Trouble_falling_asleep))) +
  geom_density(alpha = 0.5) +
  labs(title = "Density Plots of Various Social Media Usages by Trouble Falling Asleep",
       x = "Usage",
       y = "Density",
       fill = "Trouble Falling Asleep") +
  facet_wrap(~ variable, scales = "free_x", nrow = 3)
```



The image displays a series of density plots, each representing the usage of various social media platforms and how they correlate with individuals reporting trouble falling asleep. Each plot corresponds to a different social media platform: Instagram, LinkedIn, OTT (which generally refers to “over-the-top” media services), Reddit, Snapchat, Twitter, WhatsApp, and YouTube.

In the plots, usage is shown on the x-axis, while the y-axis represents the density (which is essentially a smoothed, continuous version of a histogram). There are two densities overlaid in each plot, represented by two different colors, which correspond to whether individuals have (1) or do not have (0) trouble falling asleep, as indicated by the legend titled "Trouble Falling Asleep."

The pink area represents the density of usage for individuals who do not report trouble falling asleep (0), and the teal area represents the density of usage for those who do report trouble falling asleep (1).

Here are some observations based on the plots:

1. Instagram Usage: Both groups have similar usage patterns, but the density is slightly higher for those reporting trouble falling asleep at lower usage levels.
2. LinkedIn Usage: The pattern is somewhat similar for both groups, but individuals who report trouble falling asleep appear to have a slightly higher density at moderate usage levels.
3. OTT: The plot shows a significant peak in density for individuals who have trouble falling asleep at lower usage levels compared to those who do not.
4. Reddit: There's a notable peak for those with trouble falling asleep at very low usage levels.
5. Snapchat Usage: Usage densities for both groups are somewhat similar, with a slight increase in density for individuals with trouble falling asleep at lower usage levels.
6. Twitter Usage: A pronounced peak is seen for those with trouble falling asleep at lower usage levels, while the density for those without sleep trouble is more evenly spread.
7. WhatsApp Usage: The plot shows two peaks for those with trouble falling asleep, one at lower usage and one at higher usage levels, while those without sleep trouble have a more uniform distribution.
8. YouTube Usage: There's a significant peak in density for individuals who have trouble falling asleep at moderate usage levels compared to those who do not.

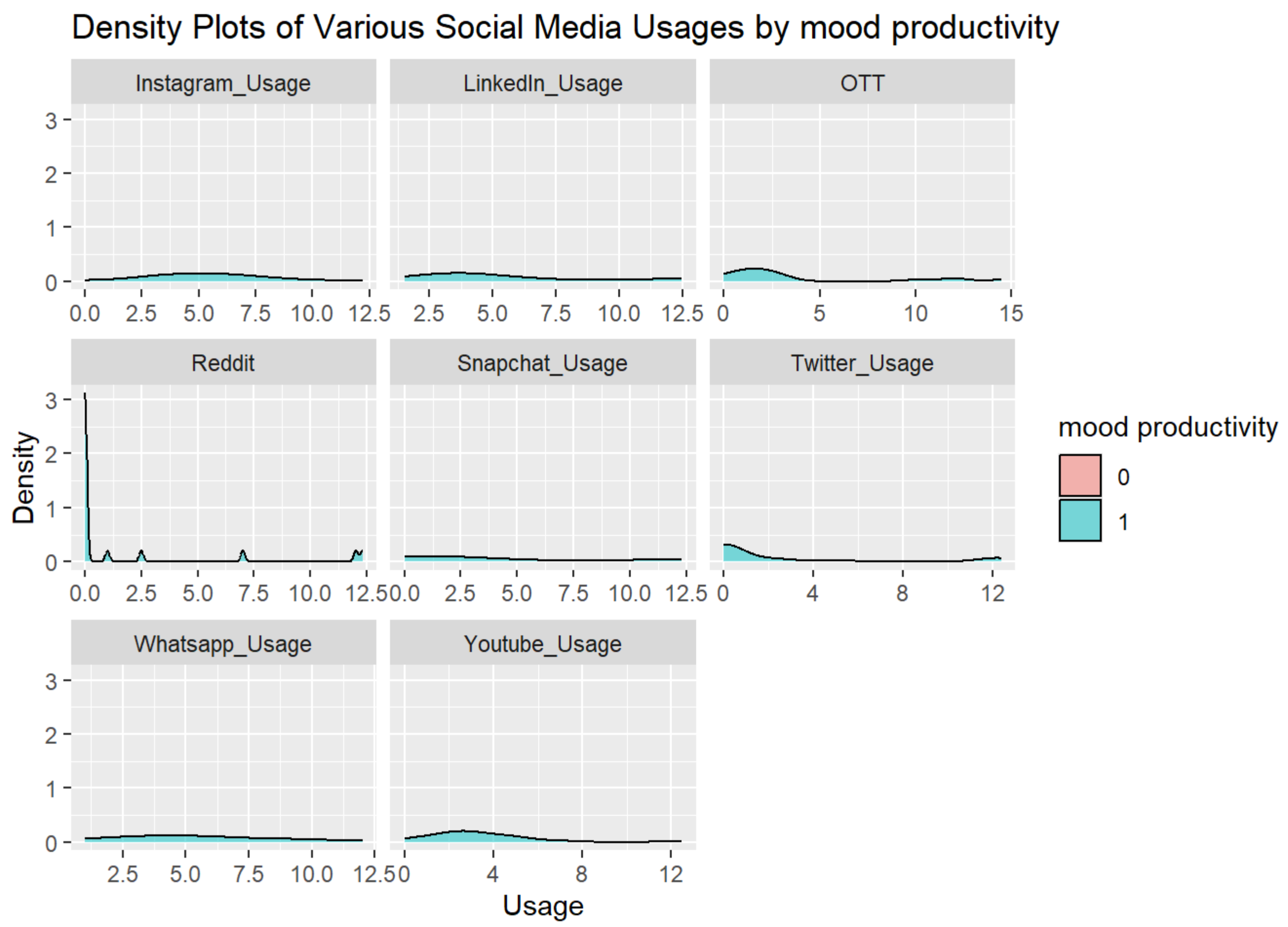
The plots suggest that there is some difference in social media usage patterns between individuals who have trouble falling asleep and those who do not, although the strength and significance of this difference would require further statistical analysis to determine. The plots can be used to hypothesize that certain usage patterns on these platforms may be associated with sleep difficulties.

Density plots for mood_productivity vs. continuous variables

```r
library(ggplot2)
library(tidyr)

# Reshape data into long format
social_media_long <- pivot_longer(social_media_final,
                                  cols = c(Instagram_Usage, LinkedIn_Usage, Snapchat_Usage,
                                           Twitter_Usage, Whatsapp_Usage, Youtube_Usage,
                                           OTT, Reddit),
                                  names_to = "variable",
                                  values_to = "value")

# Density plots for mood_productivity vs. continuous variables
ggplot(social_media_long, aes(x = value, fill = factor(Mood_Prod))) +
  geom_density(alpha = 0.5) +
  labs(title = "Density Plots of Various Social Media Usages by mood productivity",
       x = "Usage",
       y = "Density",
       fill = "mood productivity") +
  facet_wrap(~ variable, scales = "free_x", nrow = 3)
```

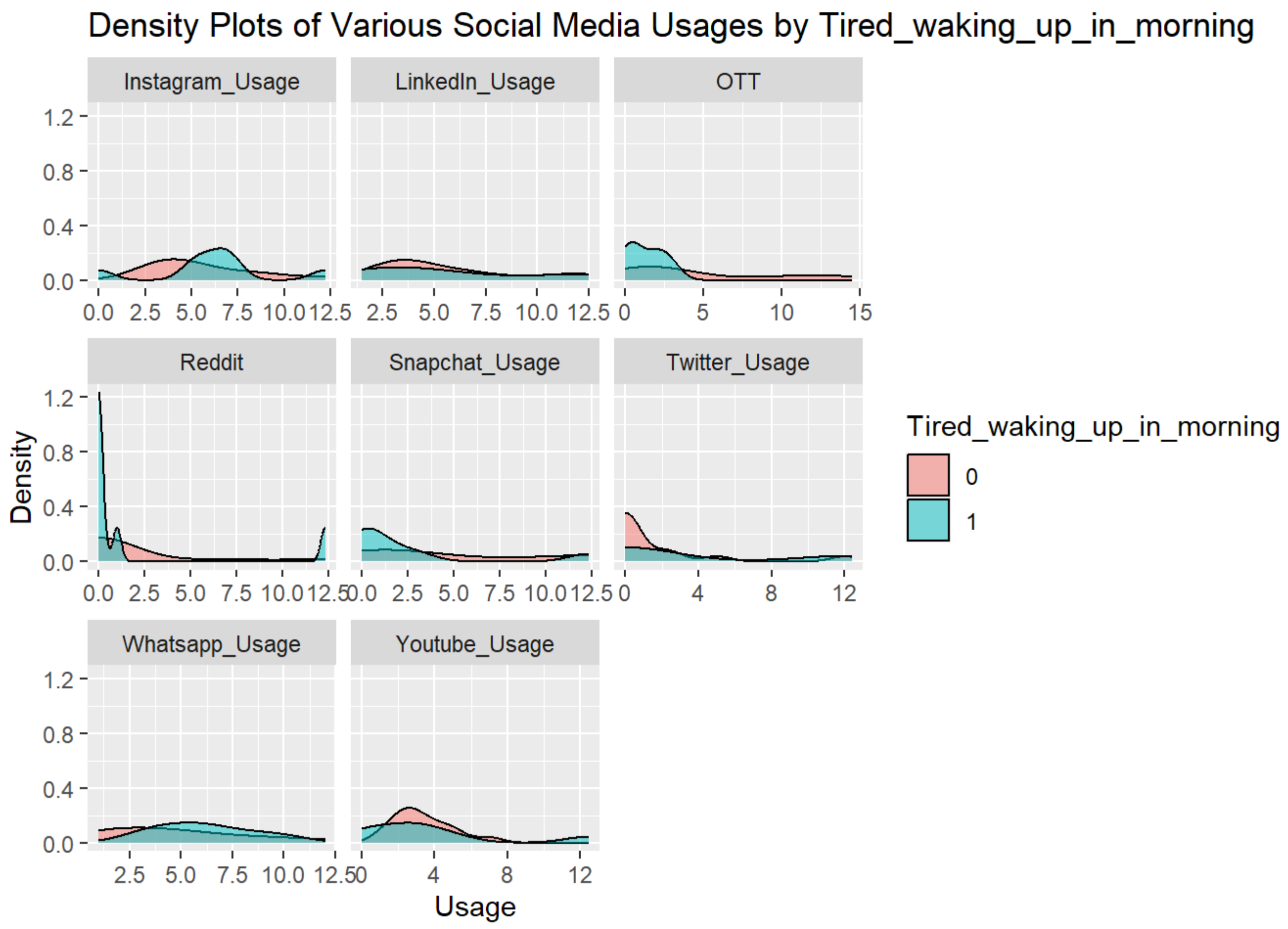
```
## Warning: Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
```

```
## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf

## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf

## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf

## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf

## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf

## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf

## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf

## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf
```



Density plots for Tired_waking_up_in_morning vs. continuous variables

```r
library(ggplot2)
library(tidyr)

# Reshape data into long format
social_media_long <- pivot_longer(social_media_final,
                                  cols = c(Instagram_Usage, LinkedIn_Usage, Snapchat_Usage,
                                           Twitter_Usage, Whatsapp_Usage, Youtube_Usage,
                                           OTT, Reddit),
                                  names_to = "variable",
                                  values_to = "value")

# Density plots for Tired_waking_up_in_morning vs. continuous variables
ggplot(social_media_long, aes(x = value, fill = factor(Tired_waking_up_in_morning))) +
  geom_density(alpha = 0.5) +
  labs(title = "Density Plots of Various Social Media Usages by Tired_waking_up_in_morning",
       x = "Usage",
       y = "Density",
       fill = "Tired_waking_up_in_morning") +
  facet_wrap(~ variable, scales = "free_x", nrow = 3)
```



The x-axis once again represents the usage of these platforms, while the y-axis shows the density. There are two colors representing two groups: the pink color indicates individuals who do not feel tired upon waking up (0), and the teal color represents those who do feel tired upon waking up (1).

Here's a summary of the plots:

Instagram Usage: The plot shows that those who feel tired in the morning (teal) have a slightly higher density around moderate usage levels compared to those who don't feel tired (pink).

LinkedIn Usage: The densities are relatively low overall, with a slightly higher concentration for non-tired individuals at lower usage levels.

OTT: Similar to LinkedIn, the overall density is low, with a small peak for non-tired individuals at the lowest usage levels.

Reddit: There's a significant spike at the lowest usage level for individuals who feel tired upon waking up.

Snapchat Usage: The usage pattern appears to be somewhat similar for both groups, with a slightly higher density for the tired group at lower usage levels.

Twitter Usage: Both groups show a low, relatively even density across usage levels, with a slight increase for non-tired individuals at the lowest usage level.

WhatsApp Usage: There's a very high spike at the lowest usage level for those who feel tired in the morning, which is not present for the non-tired group.

YouTube Usage: The density is higher for the tired group at lower usage levels, with a peak around the moderate usage level, while the non-tired group's usage is more evenly distributed but lower overall.

The density plots suggest certain usage patterns may be associated with feeling tired upon waking up, particularly at lower usage levels for some platforms. However, these observations are purely descriptive, and further analysis would be required to draw any causal conclusions.

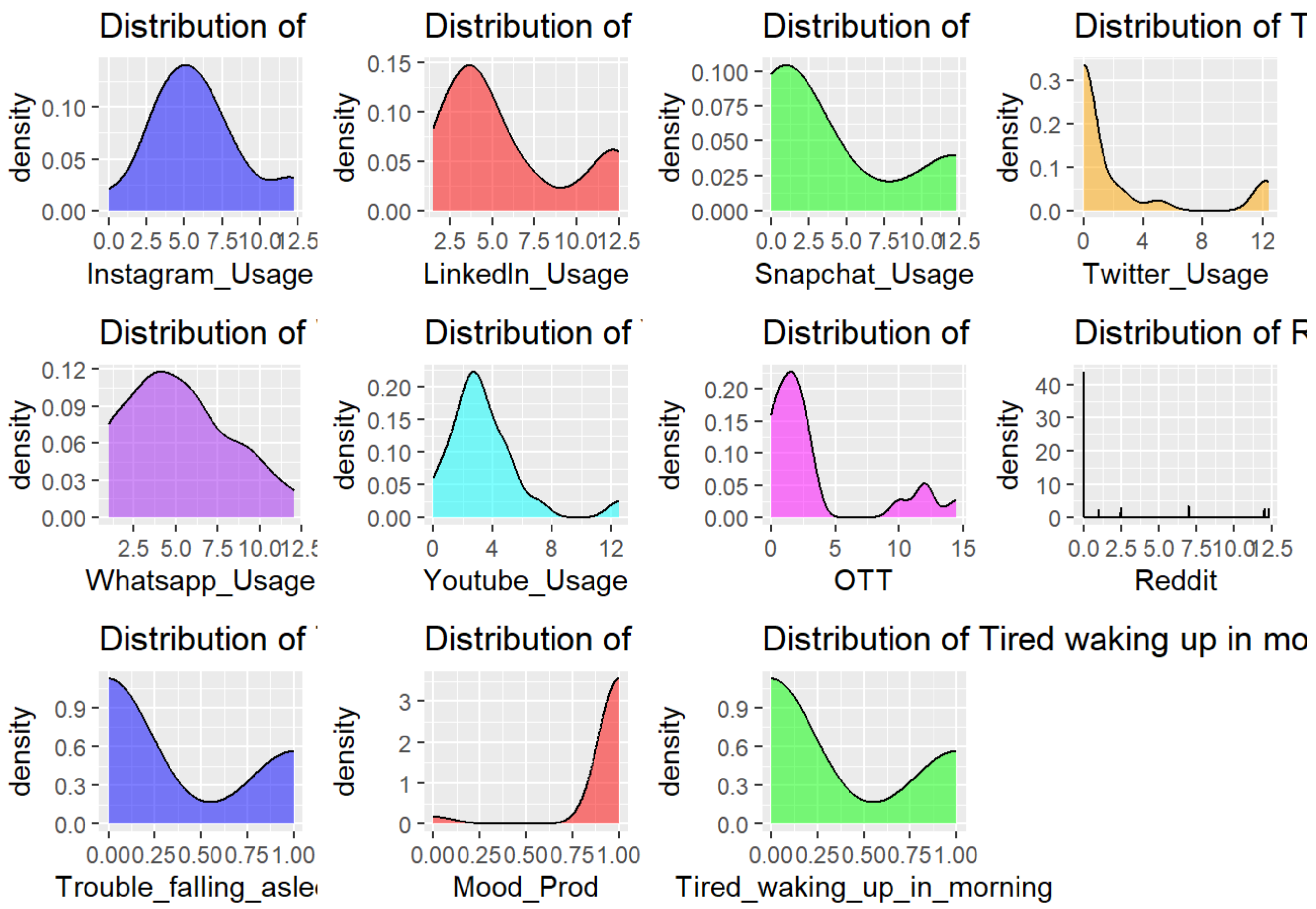
Density plots for all variables

```
library(ggplot2)
library(patchwork)
```

```
##
## Attaching package: 'patchwork'
```

```
## The following object is masked from 'package:cowplot':
##
##     align_plots
```

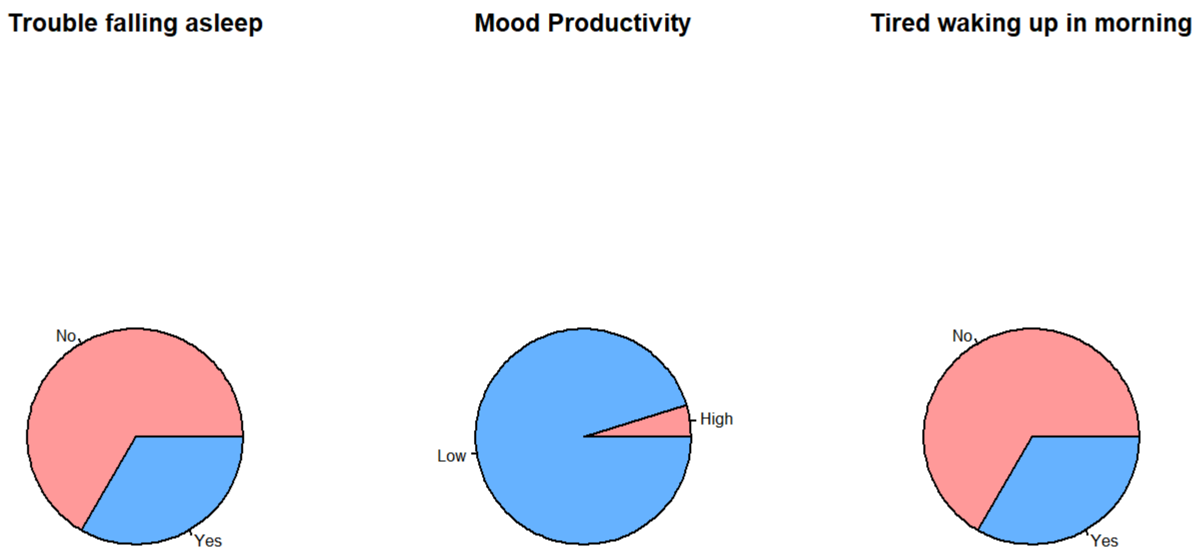
```
# Density plots for all variables
plot_instagram <- ggplot(social_media_final, aes(x = Instagram_Usage)) +
  geom_density(fill = "blue", alpha = 0.5) +
  labs(title = "Distribution of Instagram Usage")

plot_linkedin <- ggplot(social_media_final, aes(x = LinkedIn_Usage)) +
  geom_density(fill = "red", alpha = 0.5) +
  labs(title = "Distribution of LinkedIn Usage")

plot_snapchat <- ggplot(social_media_final, aes(x = Snapchat_Usage)) +
  geom_density(fill = "green", alpha = 0.5) +
  labs(title = "Distribution of Snapchat Usage")

plot_twitter <- ggplot(social_media_final, aes(x = Twitter_Usage)) +
  geom_density(fill = "orange", alpha = 0.5) +
  labs(title = "Distribution of Twitter Usage")

plot_whatsapp <- ggplot(social_media_final, aes(x = Whatsapp_Usage)) +
  geom_density(fill = "purple", alpha = 0.5) +
  labs(title = "Distribution of Whatsapp Usage")

plot_youtube <- ggplot(social_media_final, aes(x = Youtube_Usage)) +
  geom_density(fill = "cyan", alpha = 0.5) +
  labs(title = "Distribution of Youtube Usage")

plot_ott <- ggplot(social_media_final, aes(x = OTT)) +
  geom_density(fill = "magenta", alpha = 0.5) +
  labs(title = "Distribution of OTT Usage")

plot_reddit <- ggplot(social_media_final, aes(x = Reddit)) +
  geom_density(fill = "yellow", alpha = 0.5) +
  labs(title = "Distribution of Reddit Usage")

plot_trouble <- ggplot(social_media_final, aes(x = Trouble_falling_asleep)) +
  geom_density(fill = "blue", alpha = 0.5) +
  labs(title = "Distribution of Trouble falling asleep")

plot_mood <- ggplot(social_media_final, aes(x = Mood_Prod)) +
  geom_density(fill = "red", alpha = 0.5) +
  labs(title = "Distribution of Mood Productivity")

plot_tired <- ggplot(social_media_final, aes(x = Tired_waking_up_in_morning)) +
  geom_density(fill = "green", alpha = 0.5) +
  labs(title = "Distribution of Tired waking up in morning")

# Combine plots using patchwork
combined_plots <- plot_instagram + plot_linkedin + plot_snapchat + plot_twitter +
  plot_whatsapp + plot_youtube + plot_ott + plot_reddit + plot_trouble +
  plot_mood + plot_tired

# Print combined plots
combined_plots
```



Count the occurrences of each category in the dependent variables

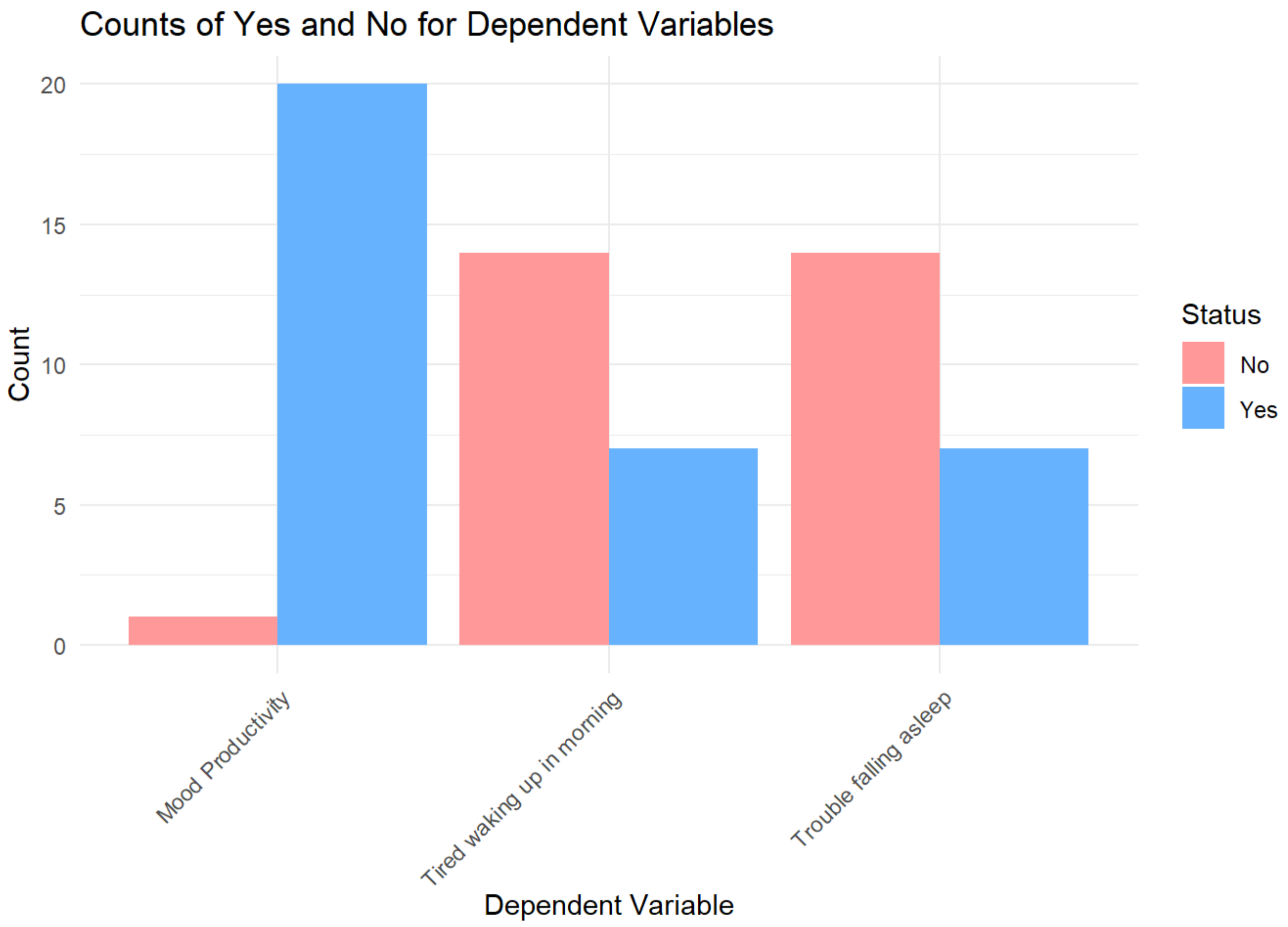
```
trouble_falling_asleep_counts <- table(social_media_final$Trouble_falling_asleep)
mood_prod_counts <- table(social_media_final$Mood_Prod)
tired_waking_up_counts <- table(social_media_final$Tired_waking_up_in_morning)
```

```
# Create a color palette for the pie charts
colors <- c("#FF9999", "#66B2FF")

# Create pie charts for each variable with better visualization
par(mfrow = c(1, 3))  # Arrange plots in a 1x3 grid

# Pie chart for Trouble falling asleep
pie(trouble_falling_asleep_counts,
    main = "Trouble falling asleep",
    col = colors,
    labels = c("No", "Yes"),
    cex = 0.8)  # Adjust label size

# Pie chart for Mood Productivity
pie(mood_prod_counts,
    main = "Mood Productivity",
    col = colors,
    labels = c("High", "Low"),
    cex = 0.8)  # Adjust label size

# Pie chart for Tired waking up in morning
pie(tired_waking_up_counts,
    main = "Tired waking up in morning",
    col = colors,
    labels = c("No", "Yes"),
    cex = 0.8)  # Adjust label size
```



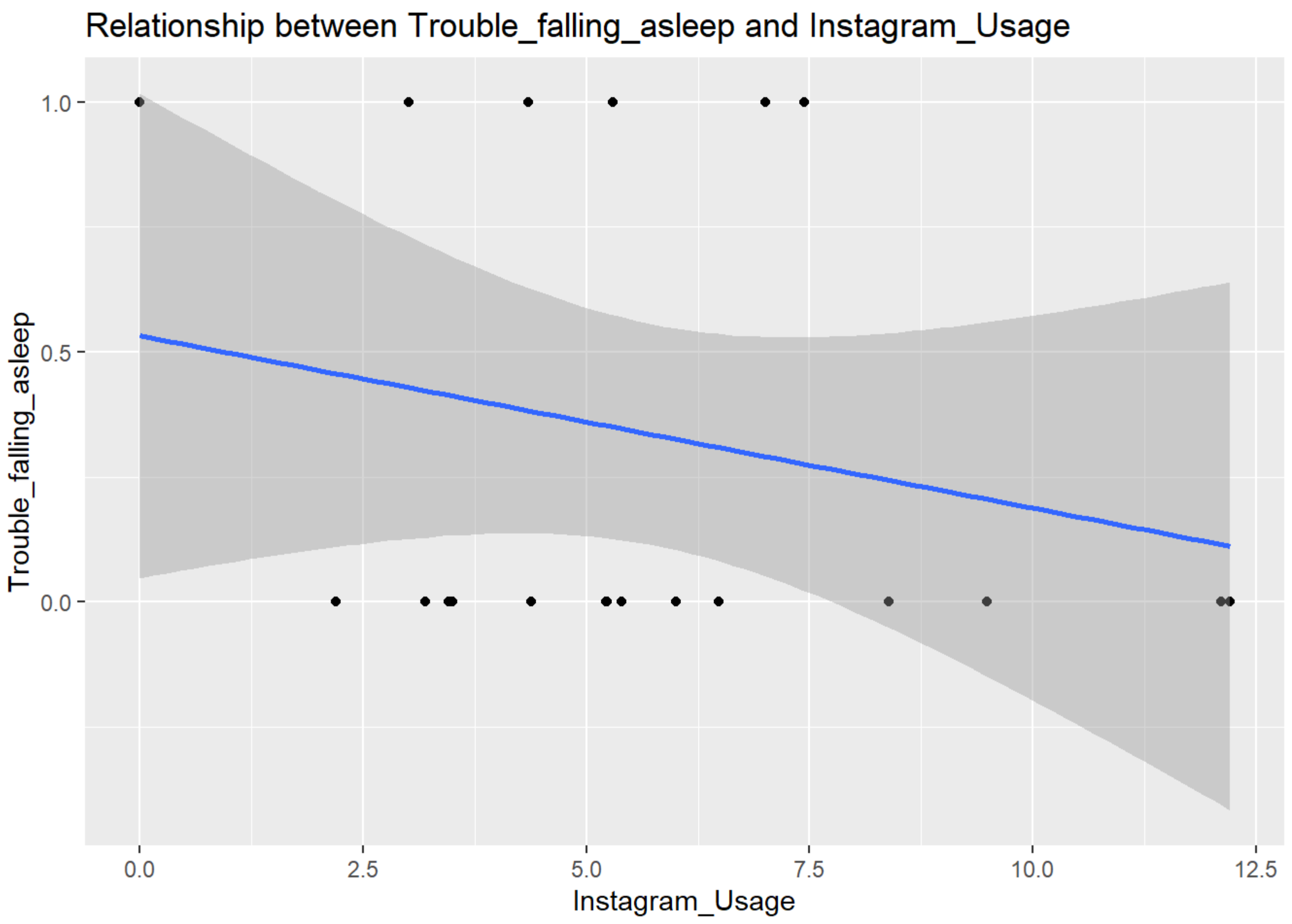
Count the occurrences of each category in the dependent variables

```
trouble_falling_asleep_counts <- table(social_media_final$Trouble_falling_asleep)
mood_prod_counts <- table(social_media_final$Mood_Prod)
tired_waking_up_counts <- table(social_media_final$Tired_waking_up_in_morning)

# Create a data frame for plotting
data <- data.frame(
  Variable = c("Trouble falling asleep", "Mood Productivity", "Tired waking up in morning"),
  Yes = c(trouble_falling_asleep_counts[2], mood_prod_counts[2], tired_waking_up_counts[2]),
  No = c(trouble_falling_asleep_counts[1], mood_prod_counts[1], tired_waking_up_counts[1])
)

# Reshape the data for plotting
library(tidyr)
data_long <- pivot_longer(data, cols = c("Yes", "No"), names_to = "Status", values_to = "Count
")

# Plot the bar plot
library(ggplot2)
ggplot(data_long, aes(x = Variable, y = Count, fill = Status)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Counts of Yes and No for Dependent Variables",
       x = "Dependent Variable",
       y = "Count") +
  scale_fill_manual(values = c("#FF9999", "#66B2FF")) +  # Custom colors for Yes and No
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))  # Rotate x-axis labels for better
readability
```

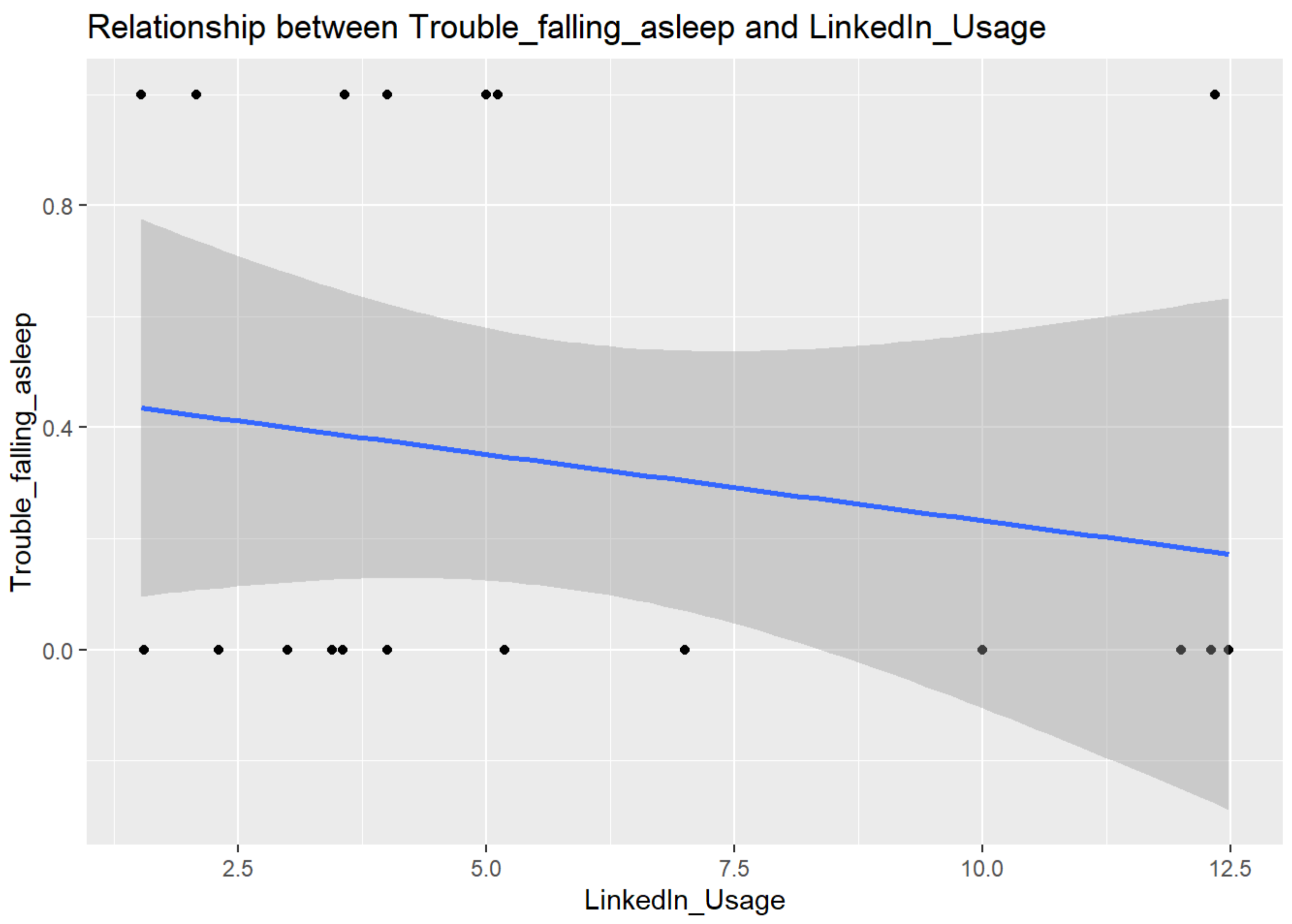


```
# Load necessary libraries if not already loaded
library(ggplot2)

# Define the dependent variables
dependent_variables <- c("Trouble_falling_asleep", "Mood_Prod", "Tired_waking_up_in_morning")

# Define the independent variables
independent_variables <- c("Instagram_Usage", "LinkedIn_Usage", "Snapchat_Usage",
                            "Twitter_Usage", "Whatsapp_Usage", "Youtube_Usage", "OTT", "Reddit
")

# Loop through each dependent variable
for (dep_var in dependent_variables) {
  # Loop through each independent variable
  for (ind_var in independent_variables) {
    # Create a scatter plot for the relationship between the dependent and independent variabl
es
    plot <- ggplot(social_media_final, aes_string(x = ind_var, y = dep_var)) +
      geom_point() +
      geom_smooth(method = "lm") +
      labs(title = paste("Relationship between", dep_var, "and", ind_var))

    # Print the plot
    print(plot)
  }
}
```

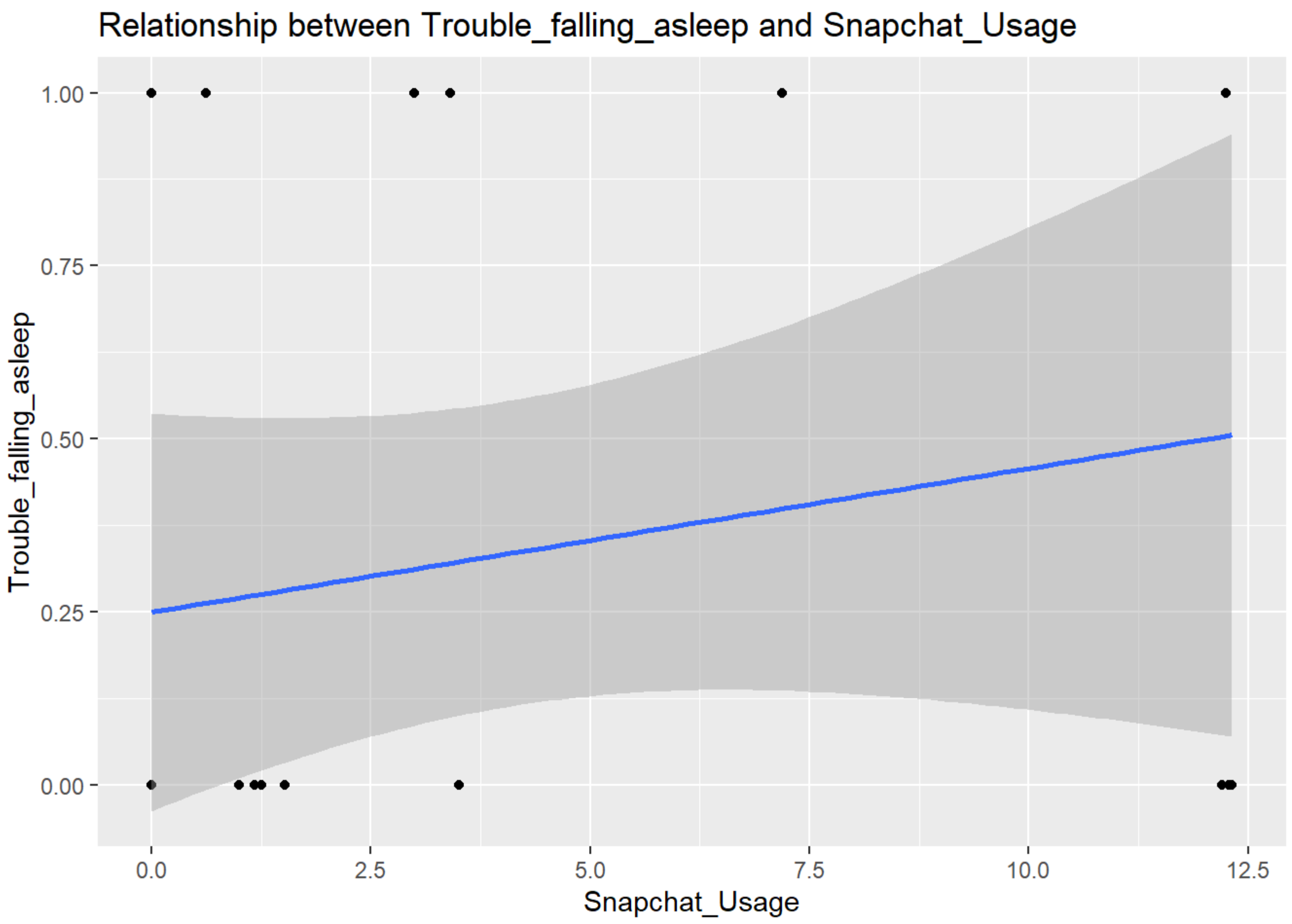
```
## `geom_smooth()` using formula = 'y ~ x'
```



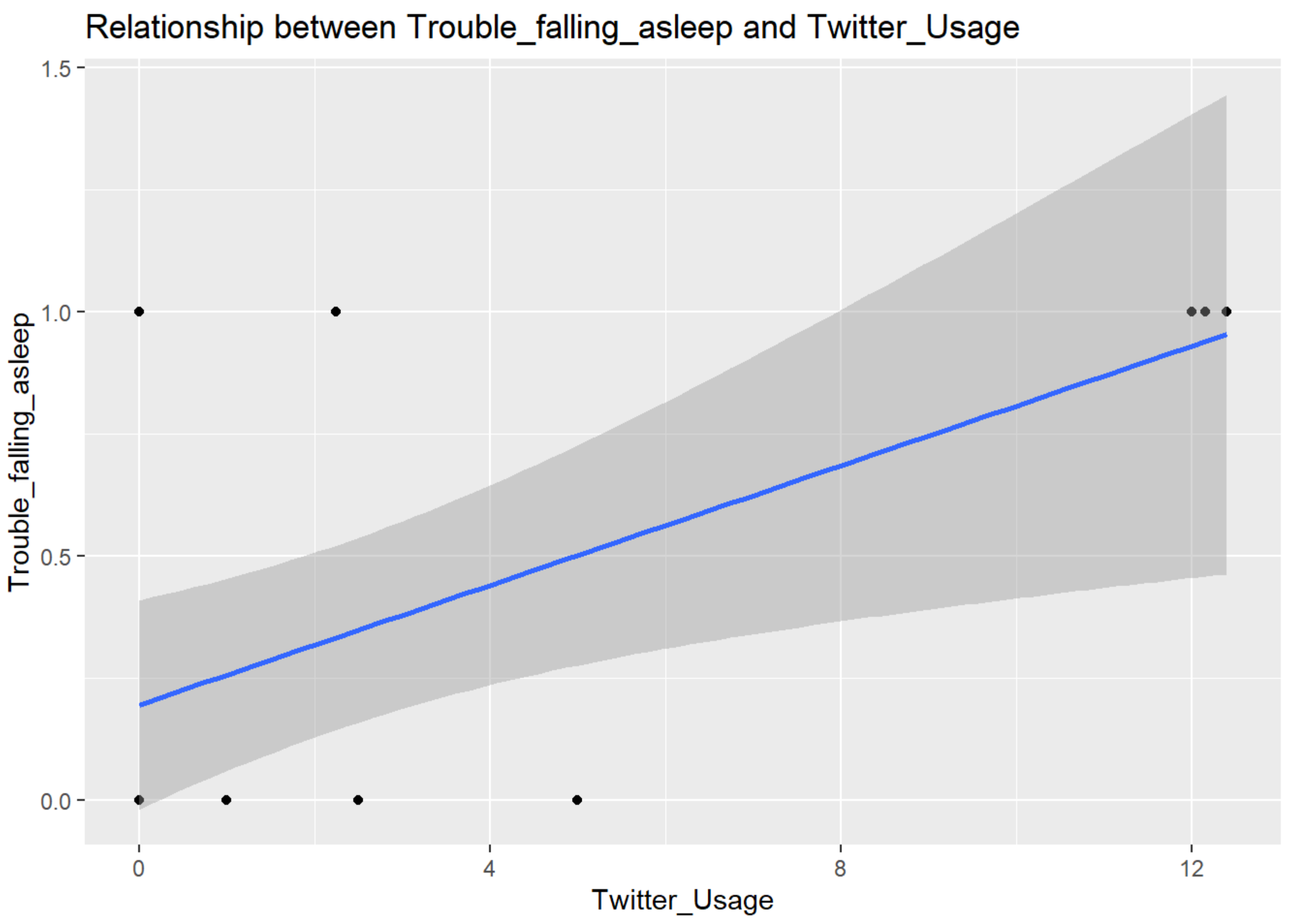
```
## `geom_smooth()` using formula = 'y ~ x'
```



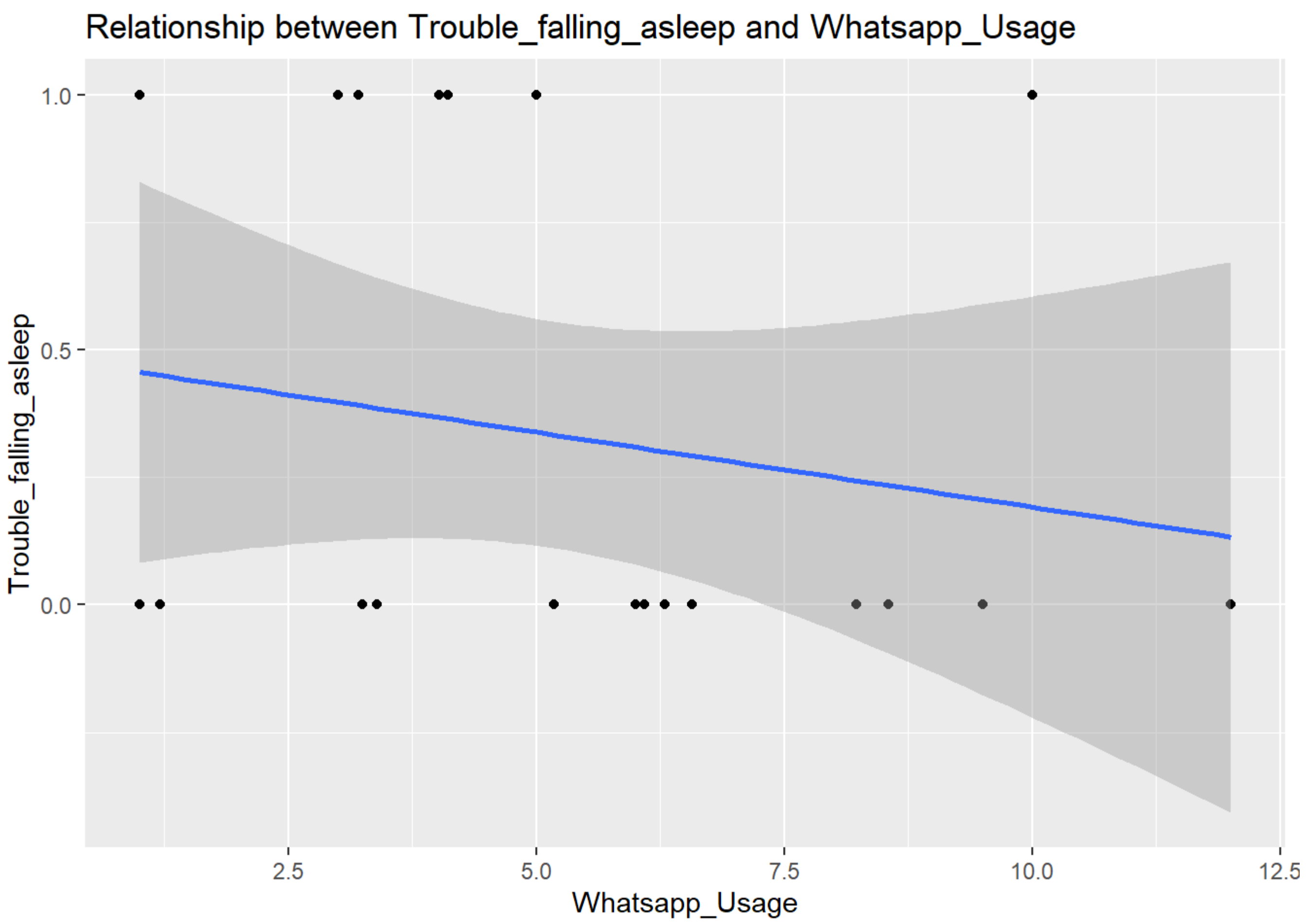
```
## `geom_smooth()` using formula = 'y ~ x'
```



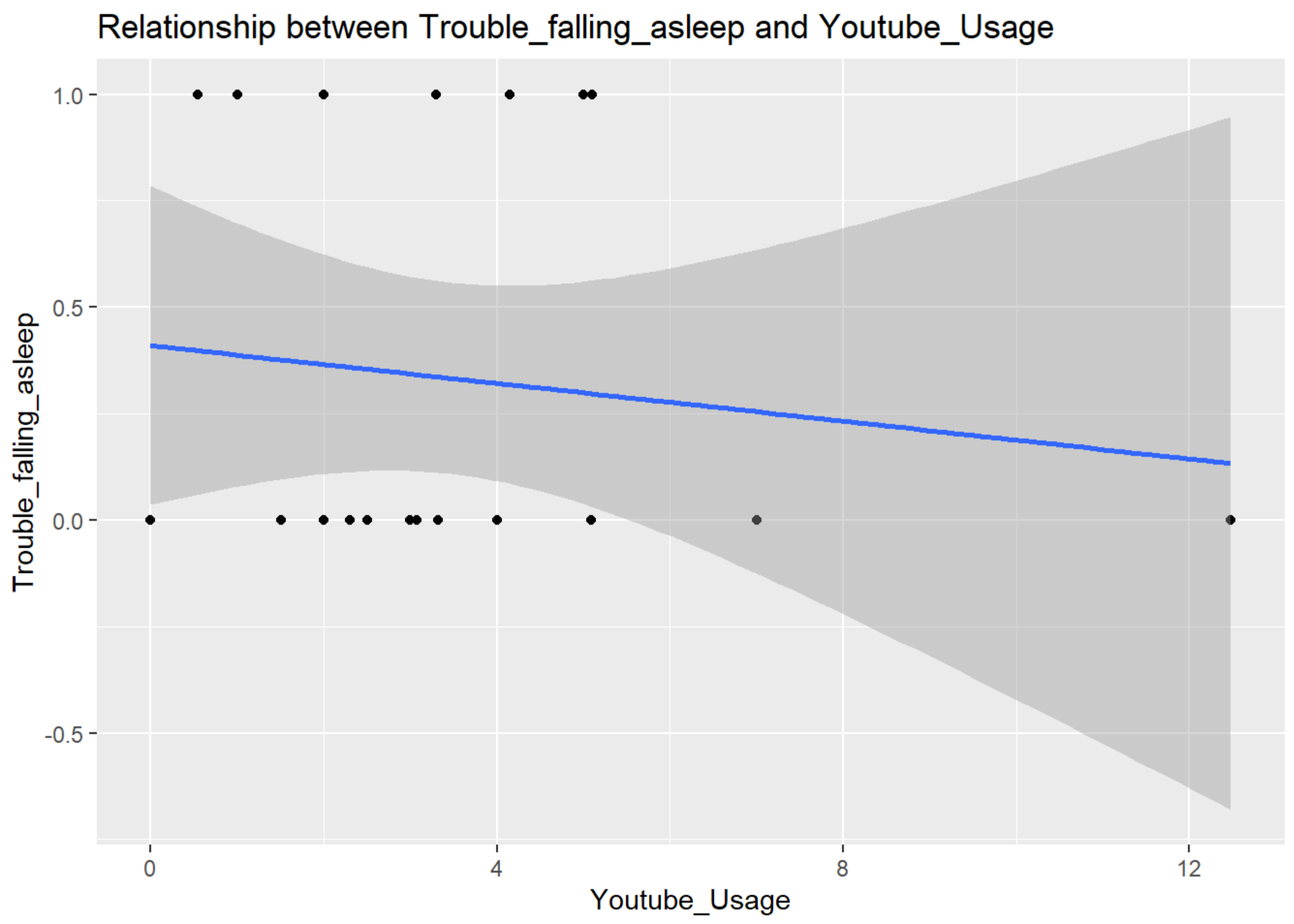
```
## `geom_smooth()` using formula = 'y ~ x'
```



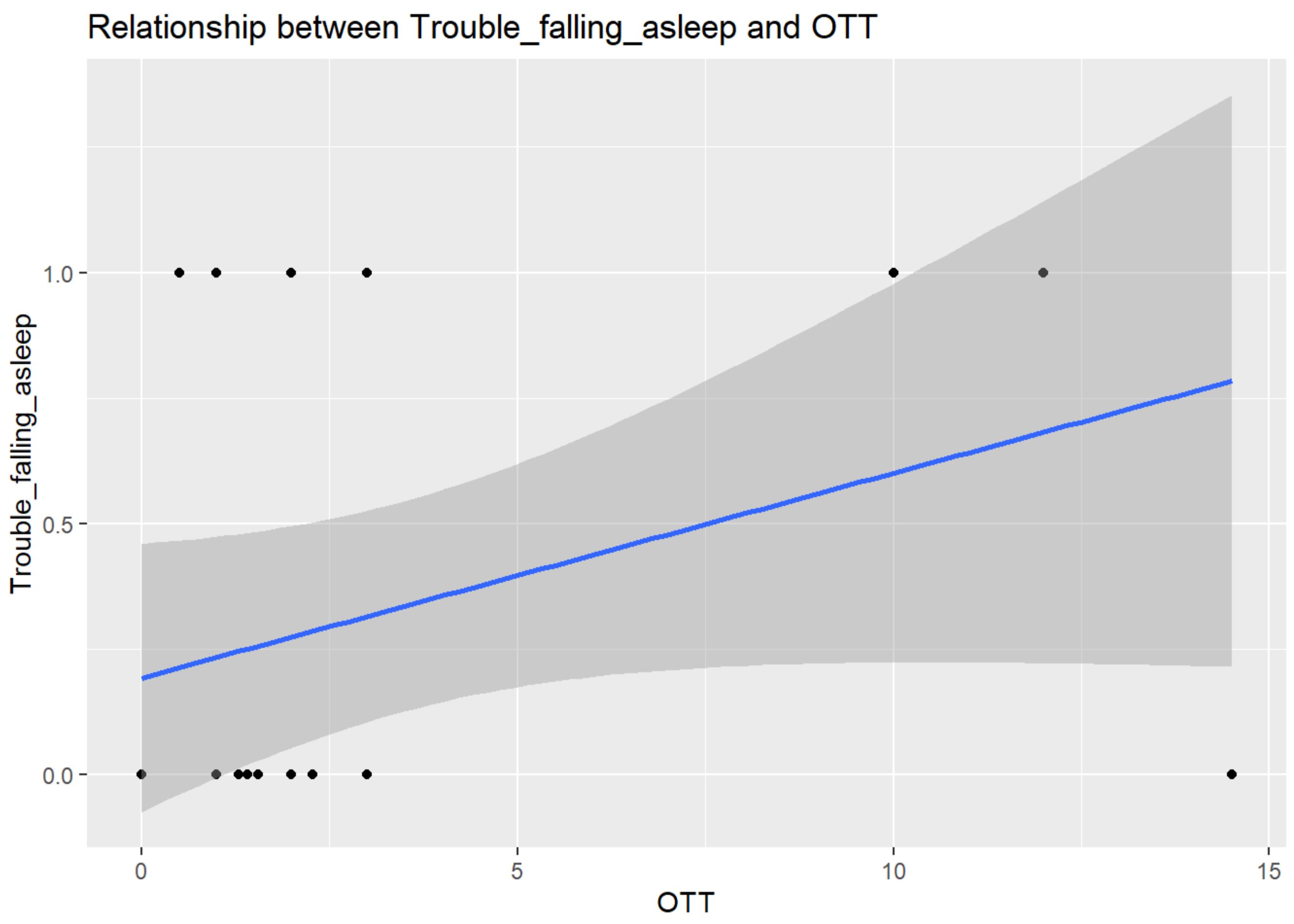
```
## `geom_smooth()` using formula = 'y ~ x'
```



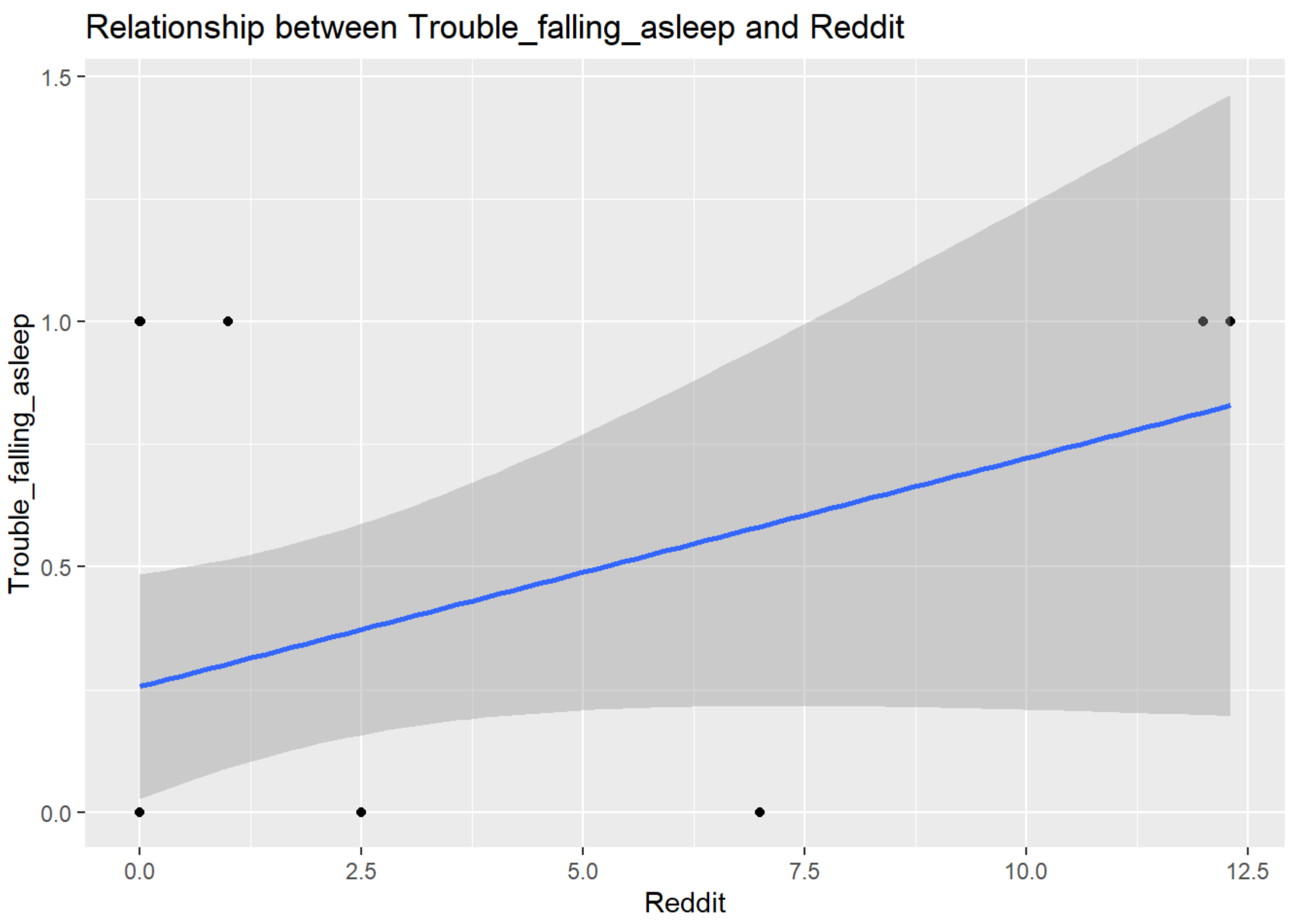
```
## `geom_smooth()` using formula = 'y ~ x'
```



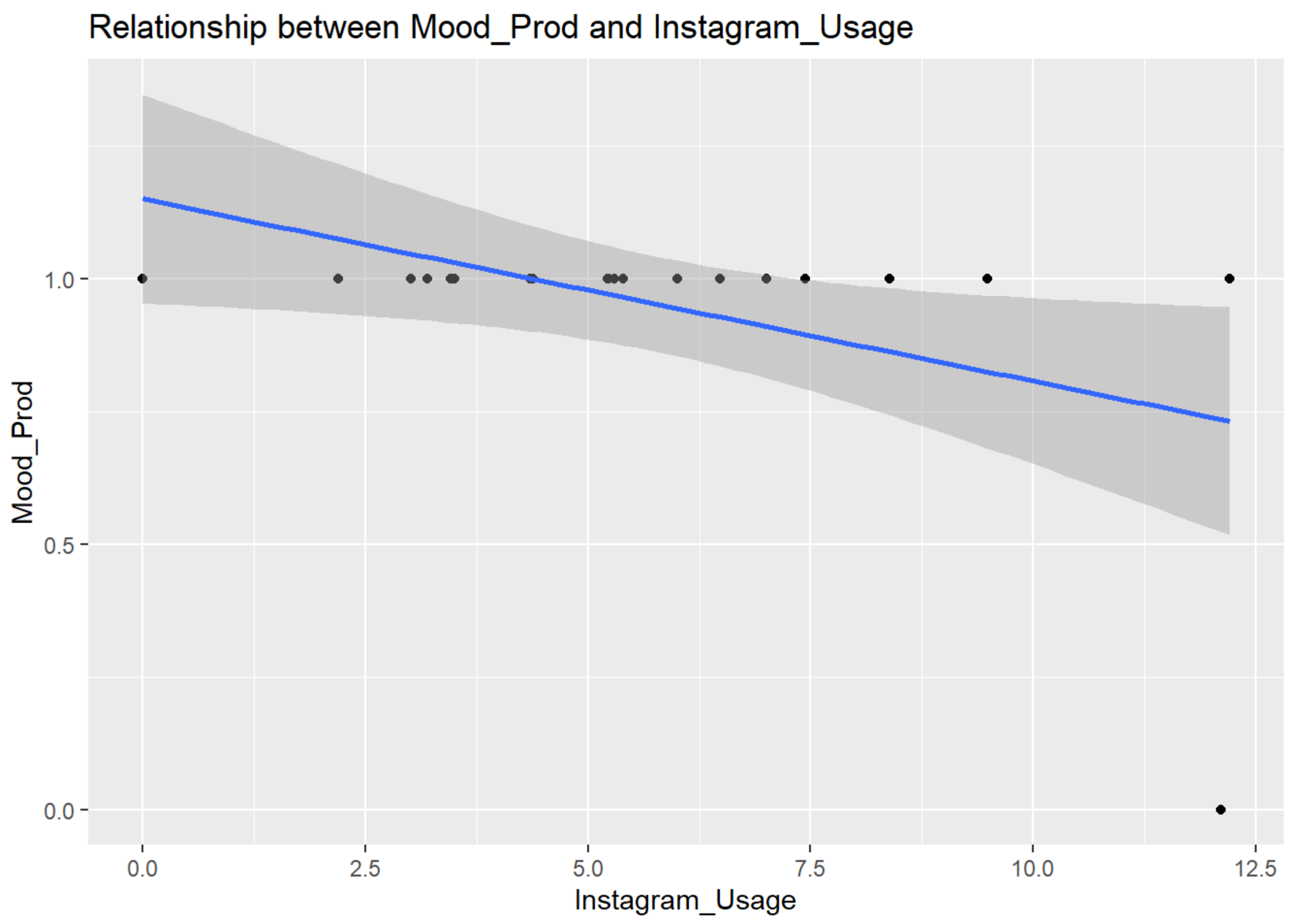
```
## `geom_smooth()` using formula = 'y ~ x'
```



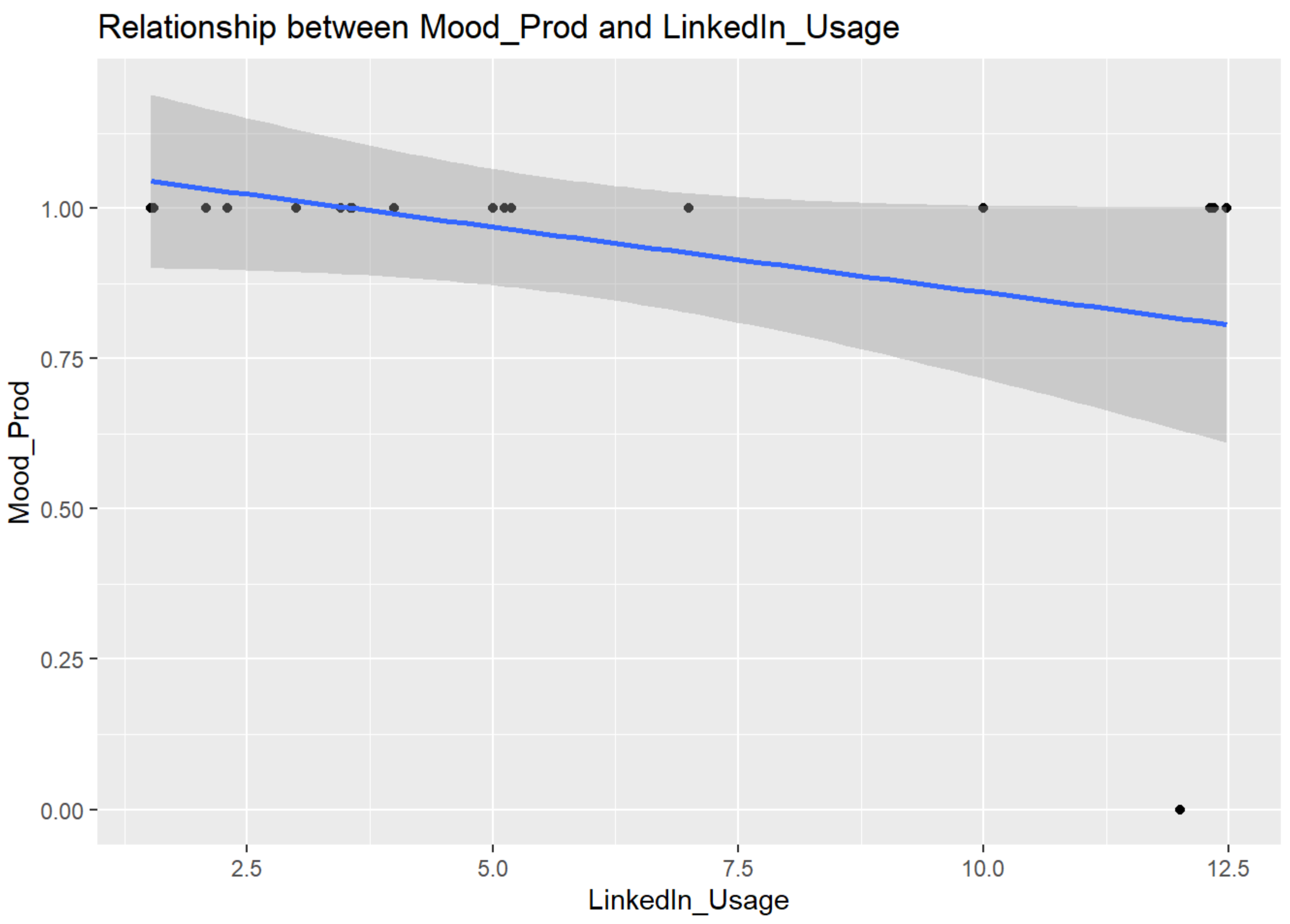
```
## `geom_smooth()` using formula = 'y ~ x'
```



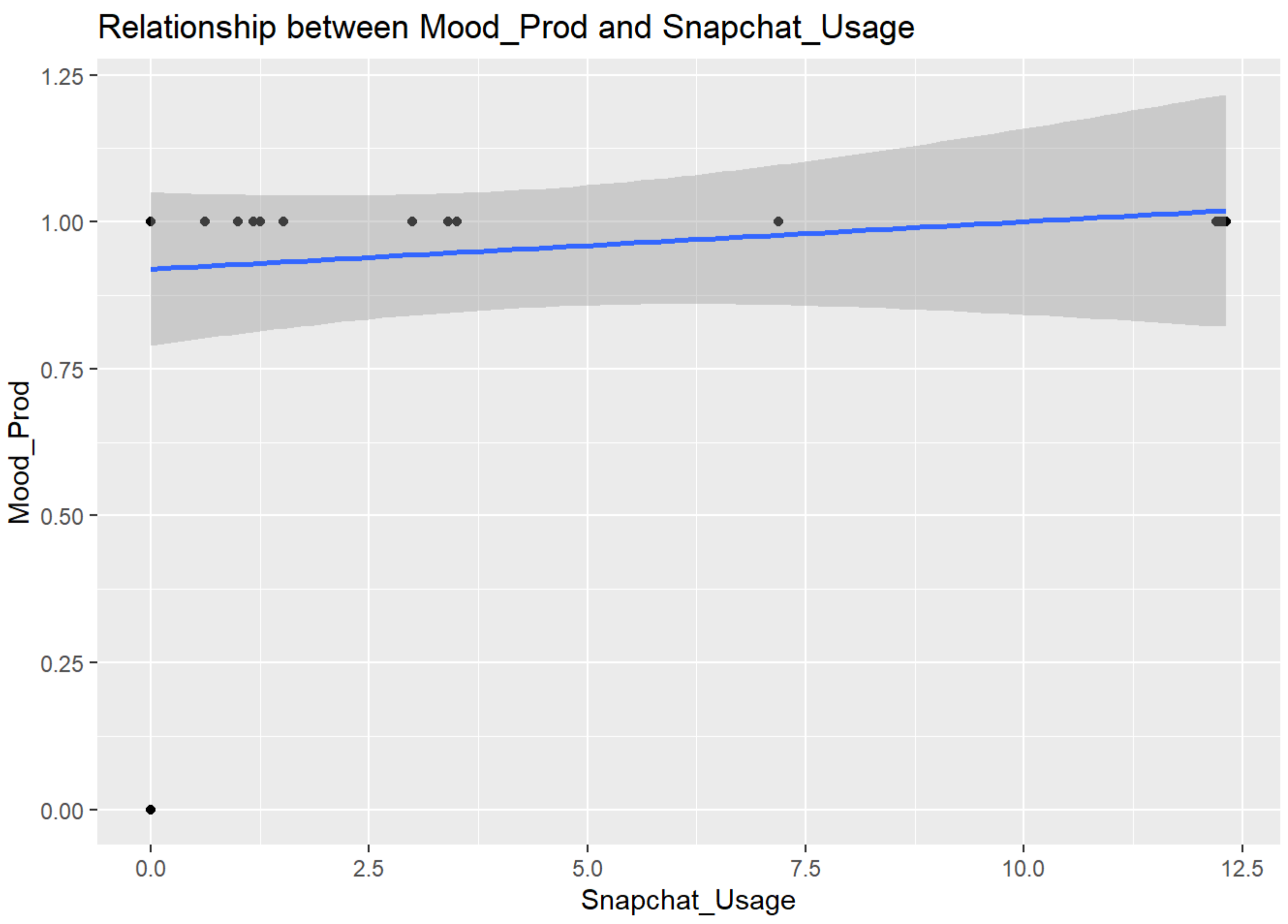
```
## `geom_smooth()` using formula = 'y ~ x'
```



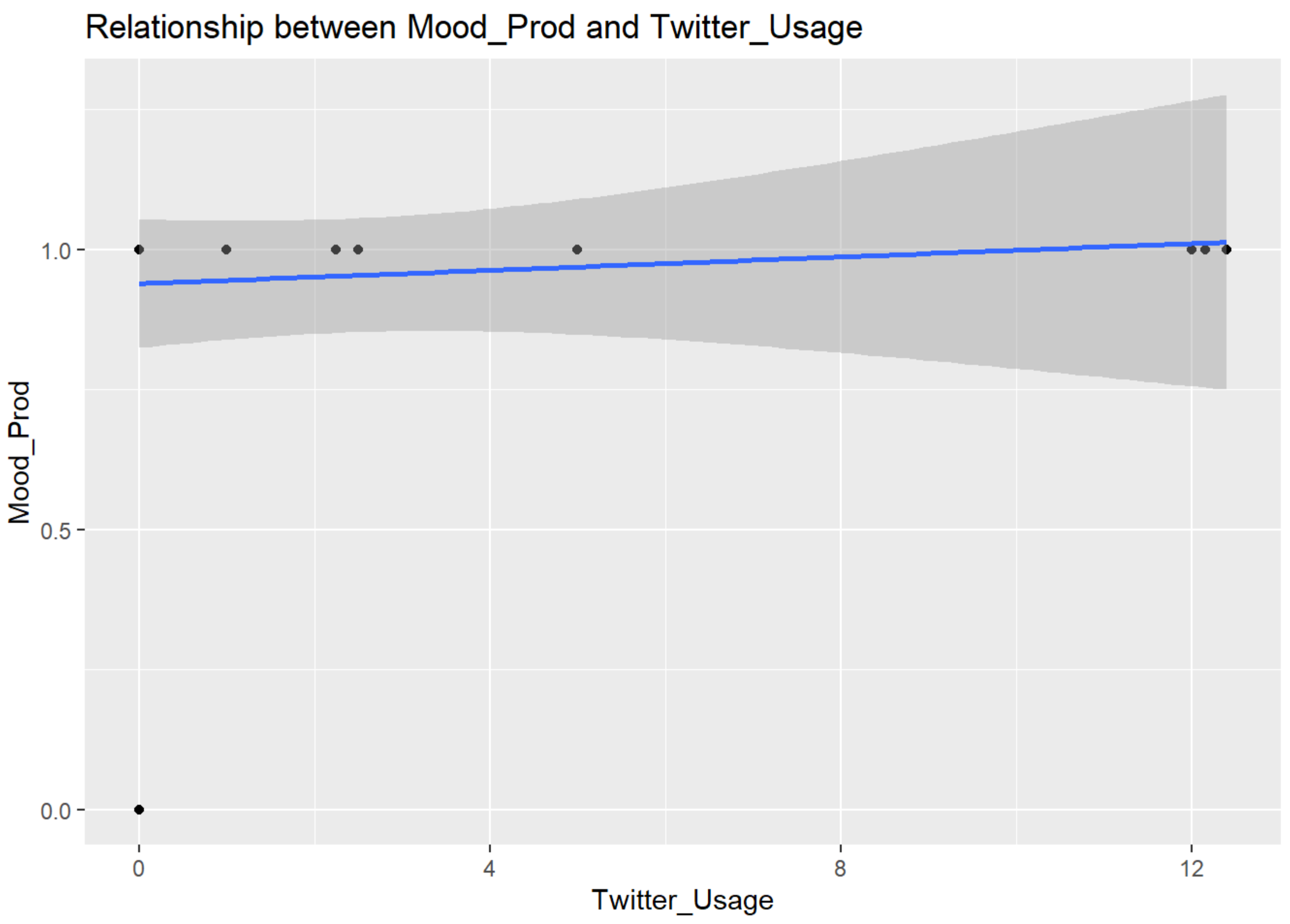
```
## `geom_smooth()` using formula = 'y ~ x'
```



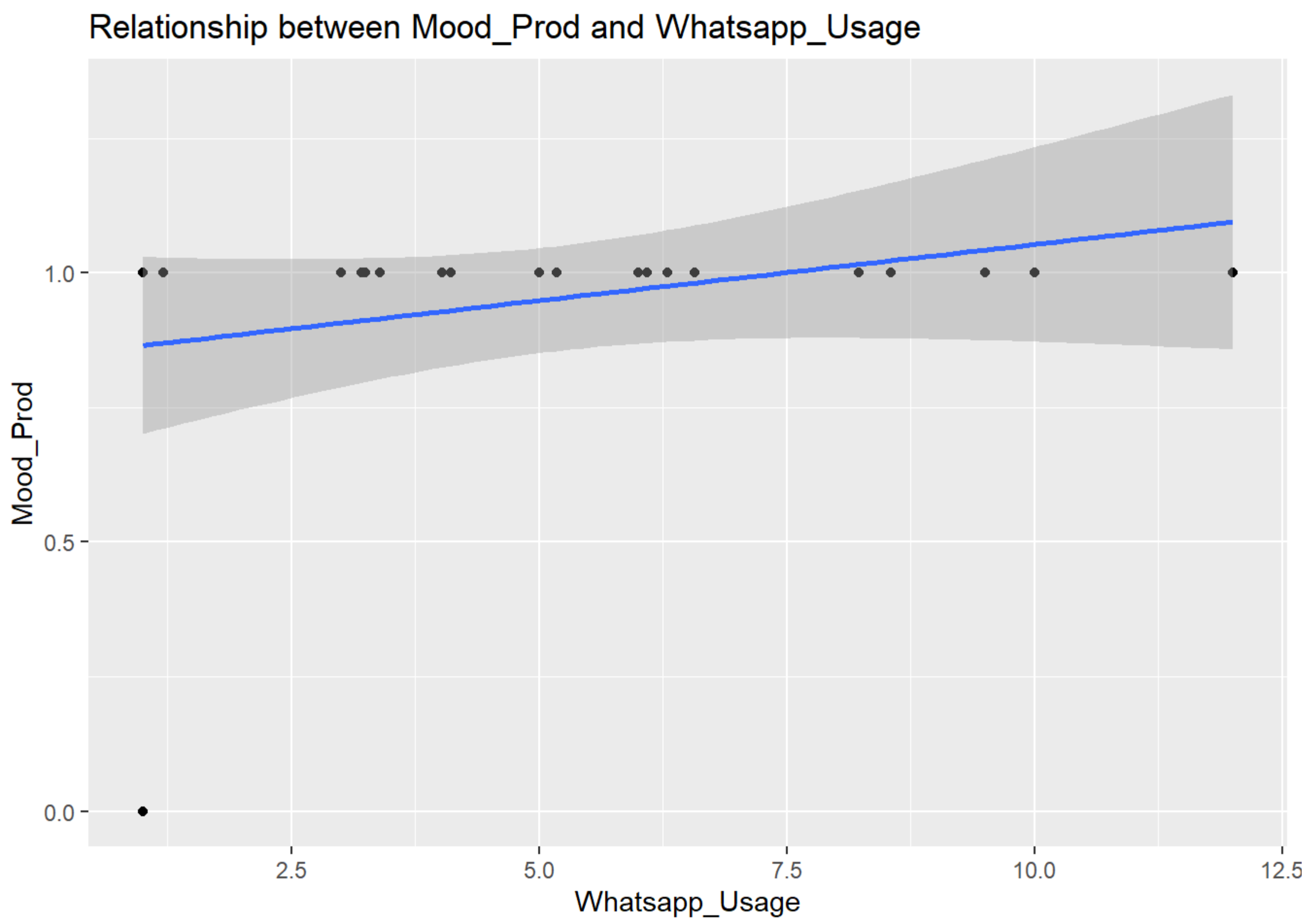
```
## `geom_smooth()` using formula = 'y ~ x'
```



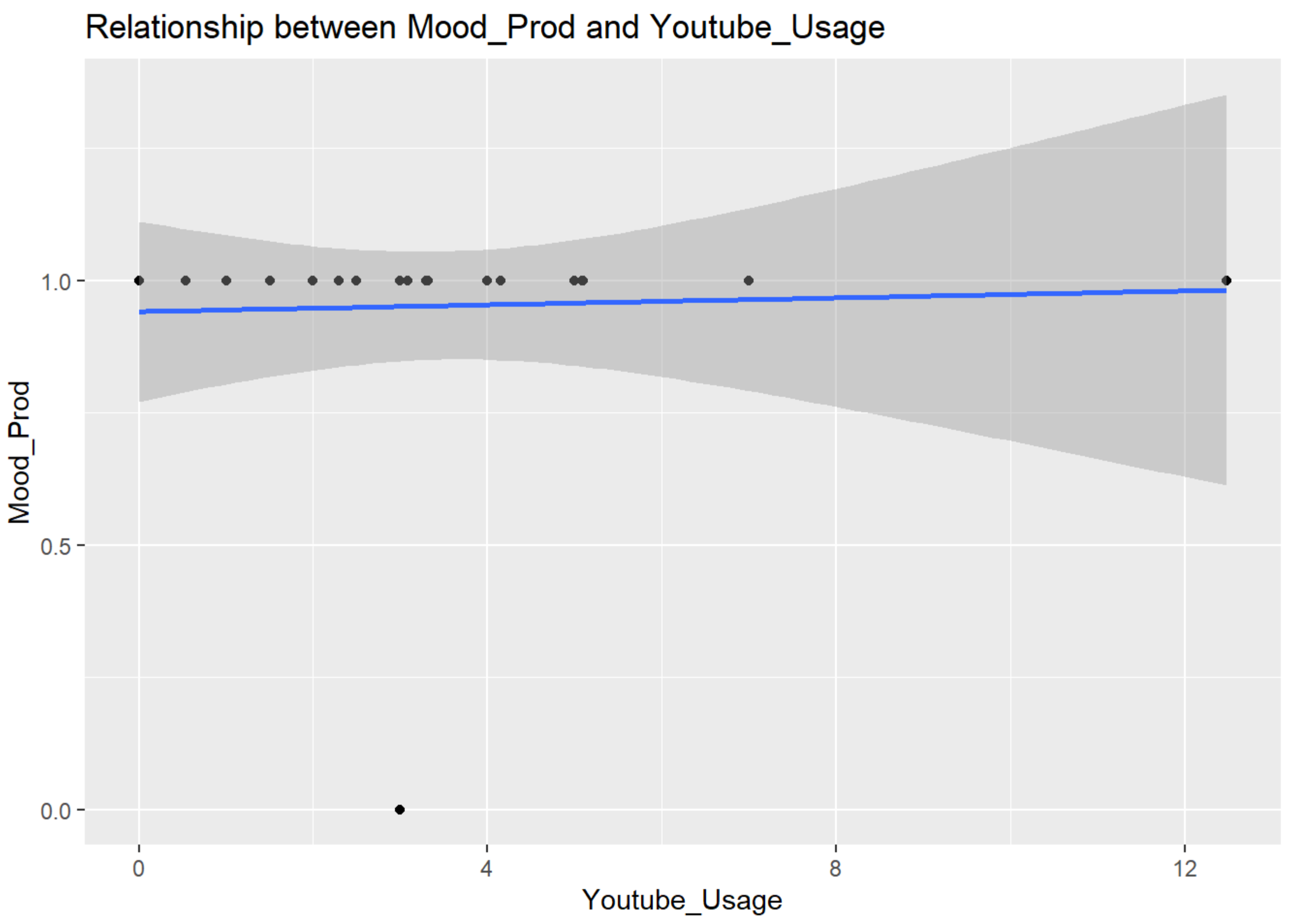
```
## `geom_smooth()` using formula = 'y ~ x'
```



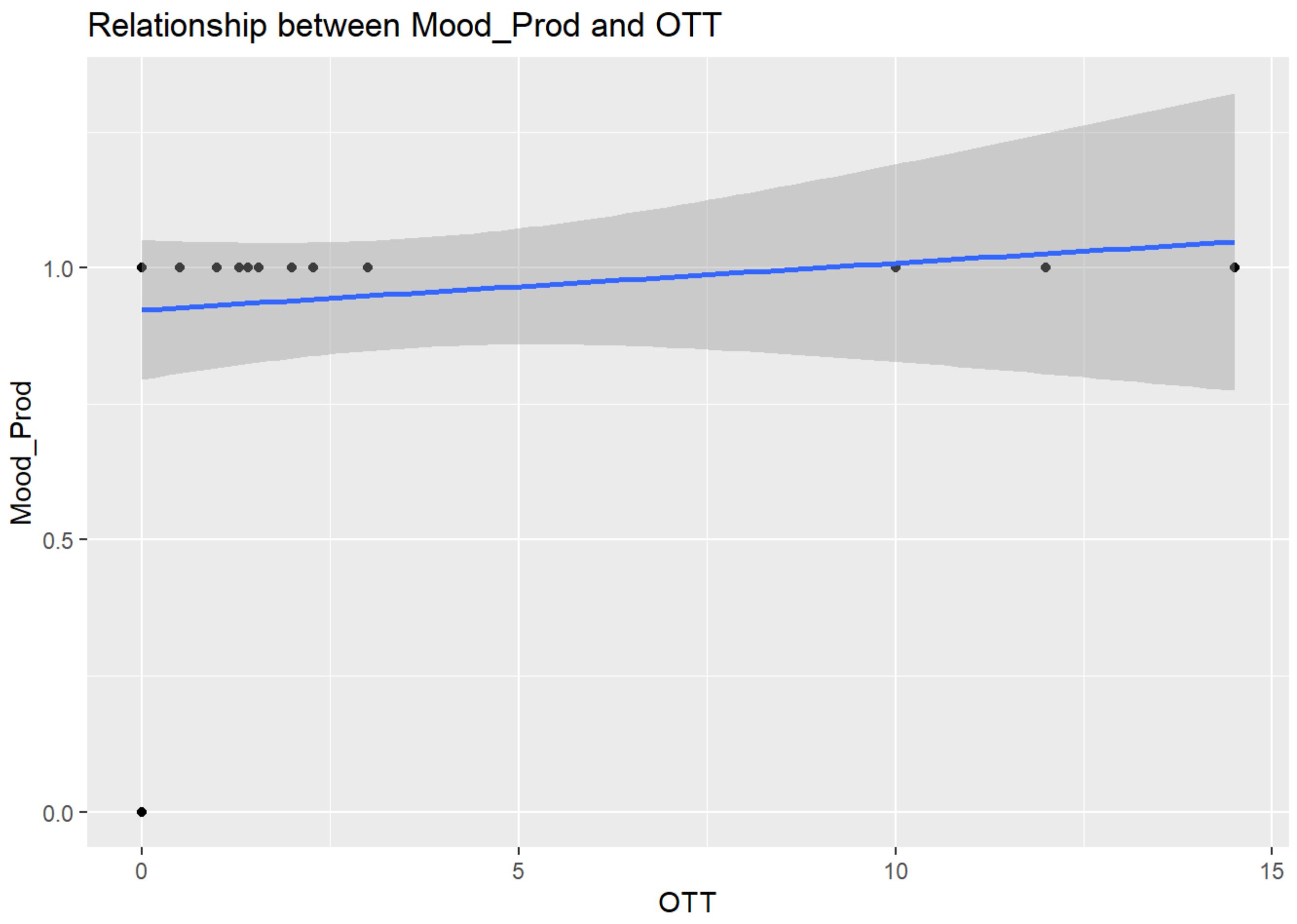
```
## `geom_smooth()` using formula = 'y ~ x'
```



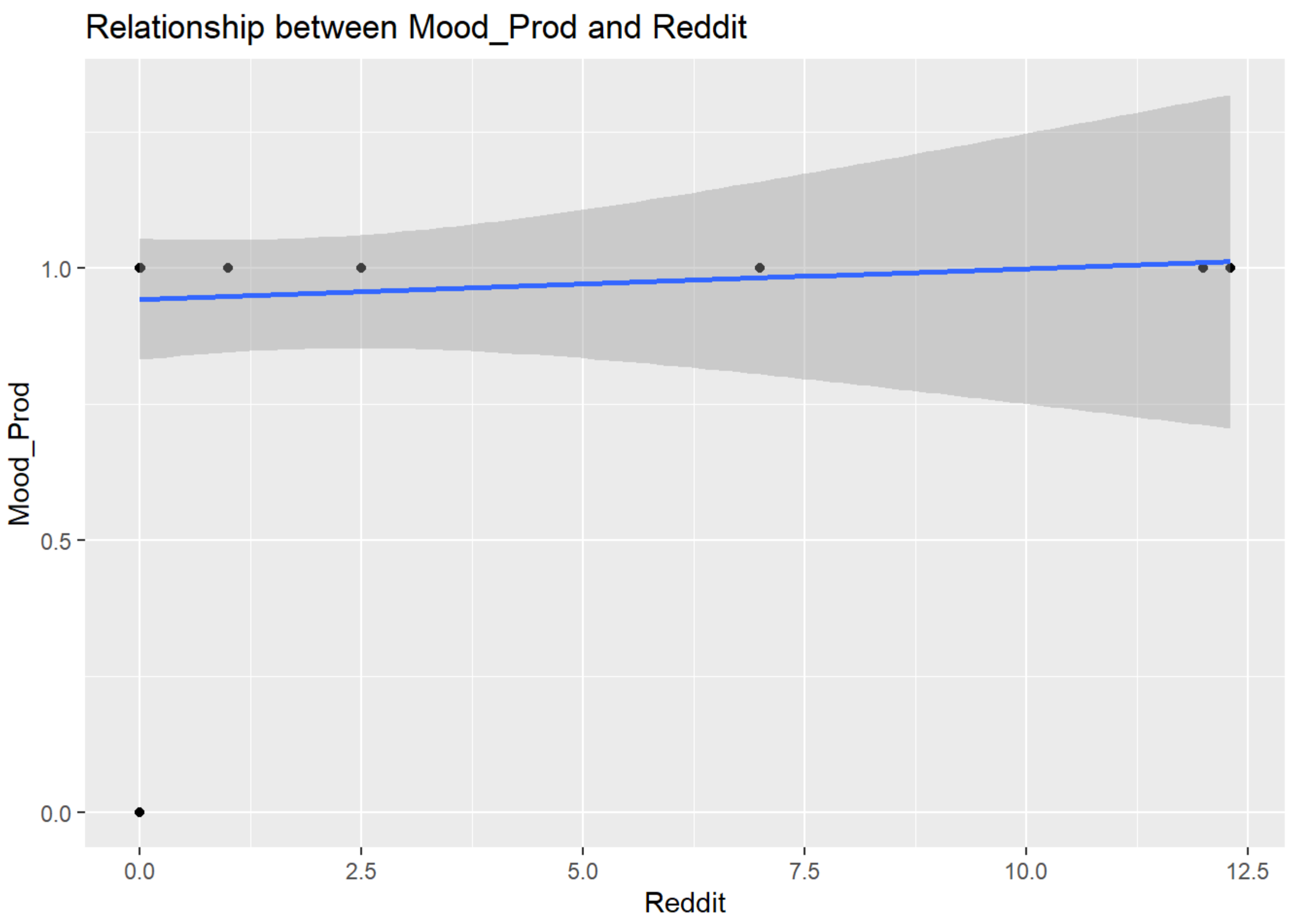
```
## `geom_smooth()` using formula = 'y ~ x'
```



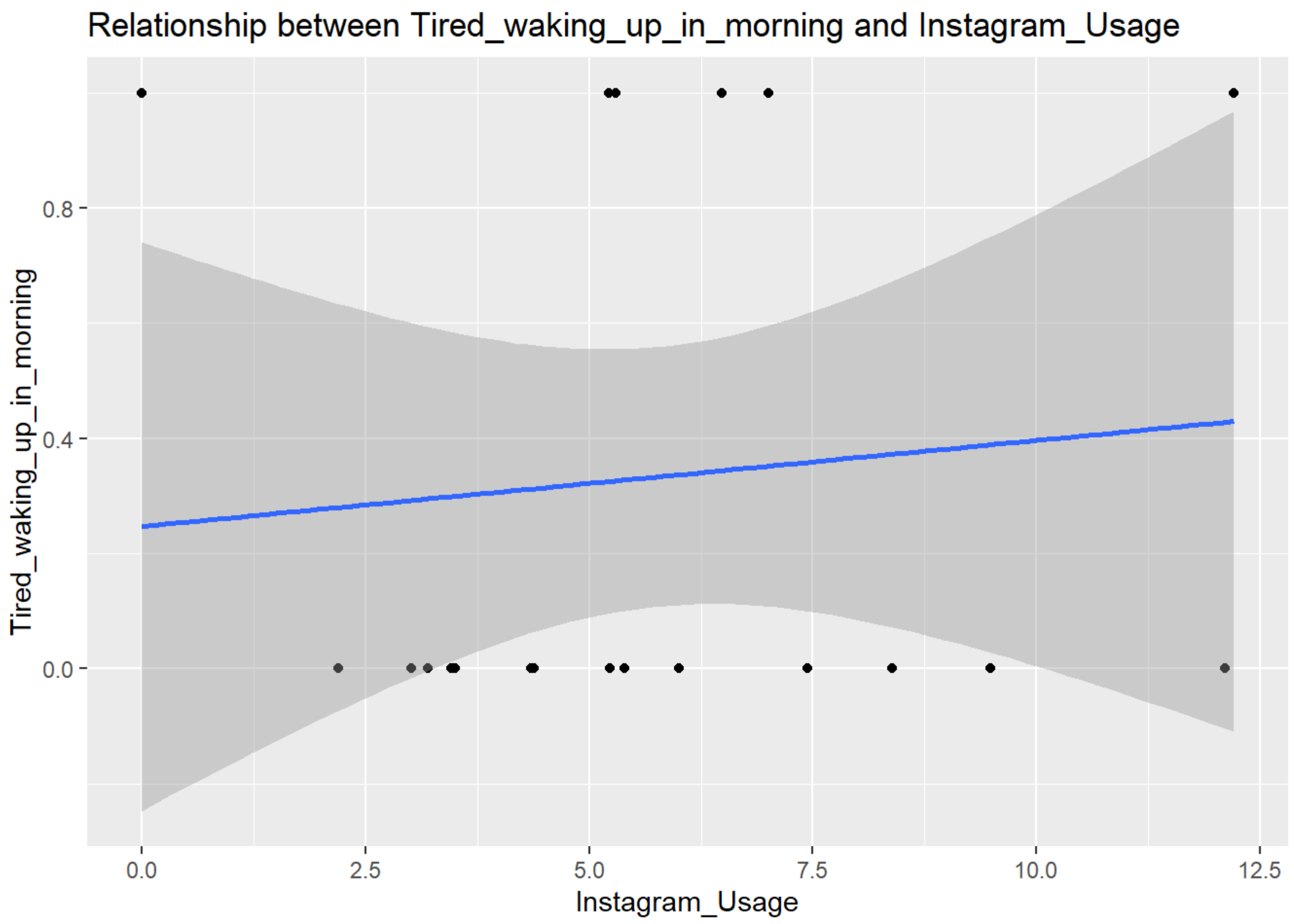
```
## `geom_smooth()` using formula = 'y ~ x'
```



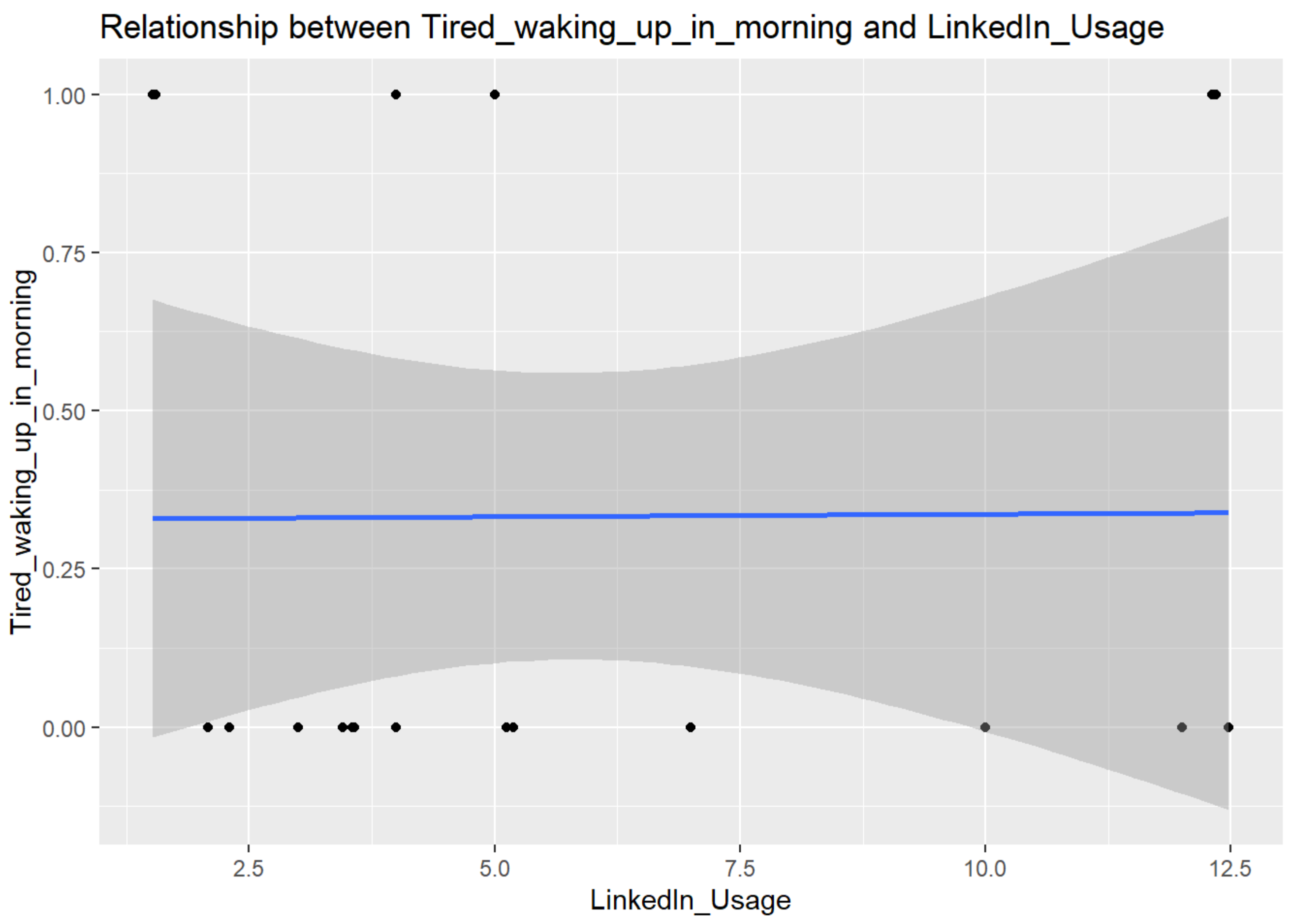
```
## `geom_smooth()` using formula = 'y ~ x'
```



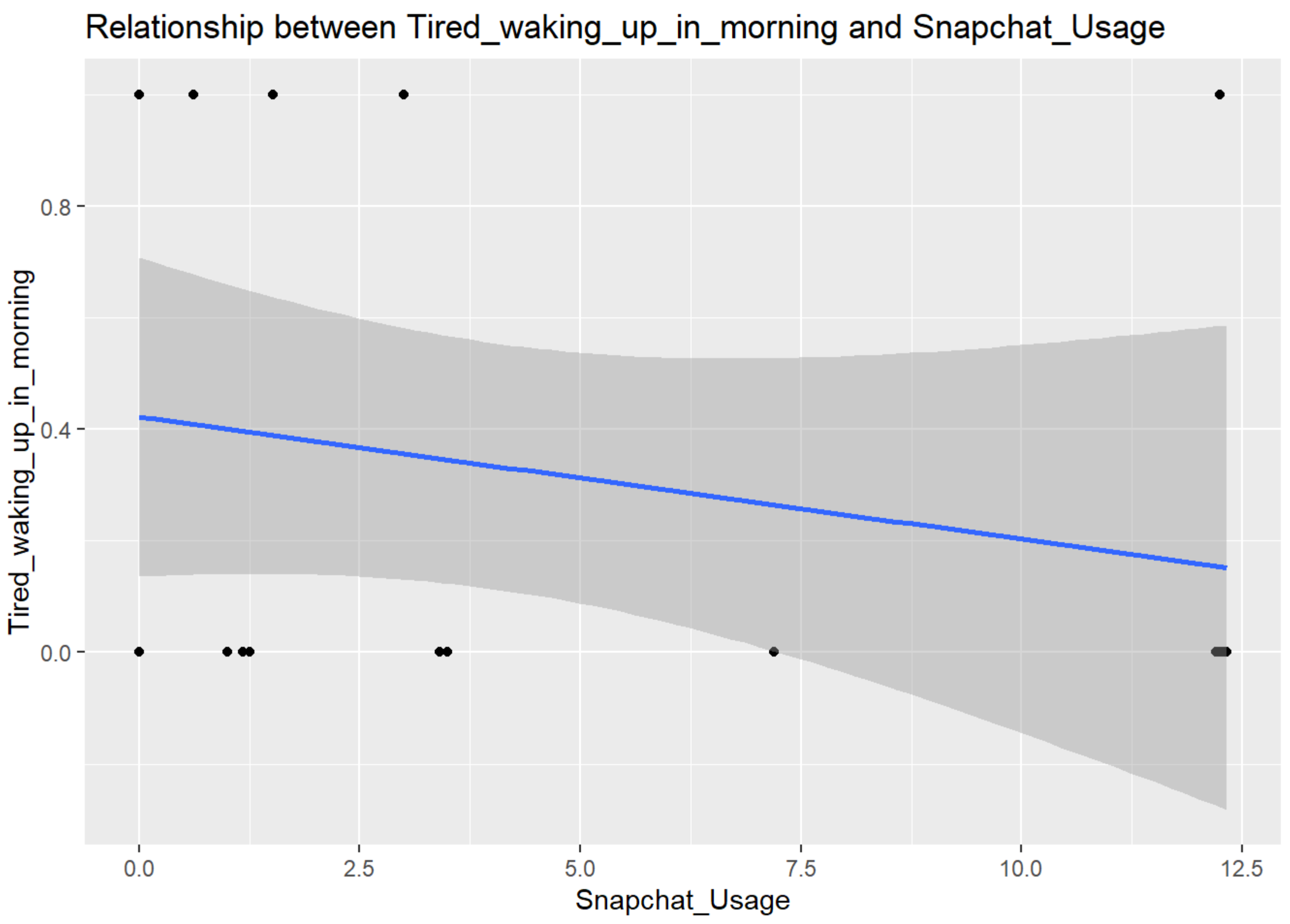
```
## `geom_smooth()` using formula = 'y ~ x'
```



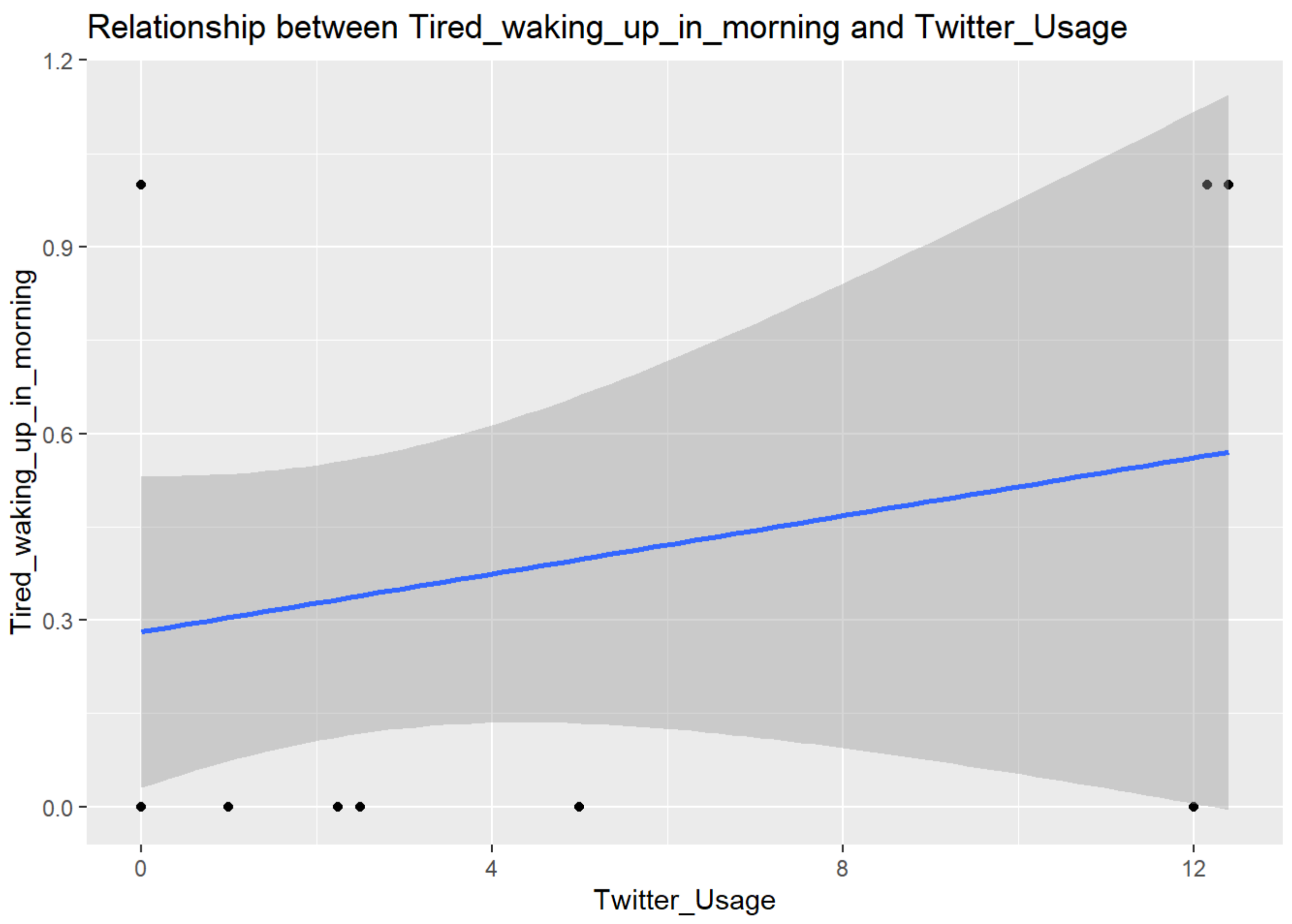
```
## `geom_smooth()` using formula = 'y ~ x'
```



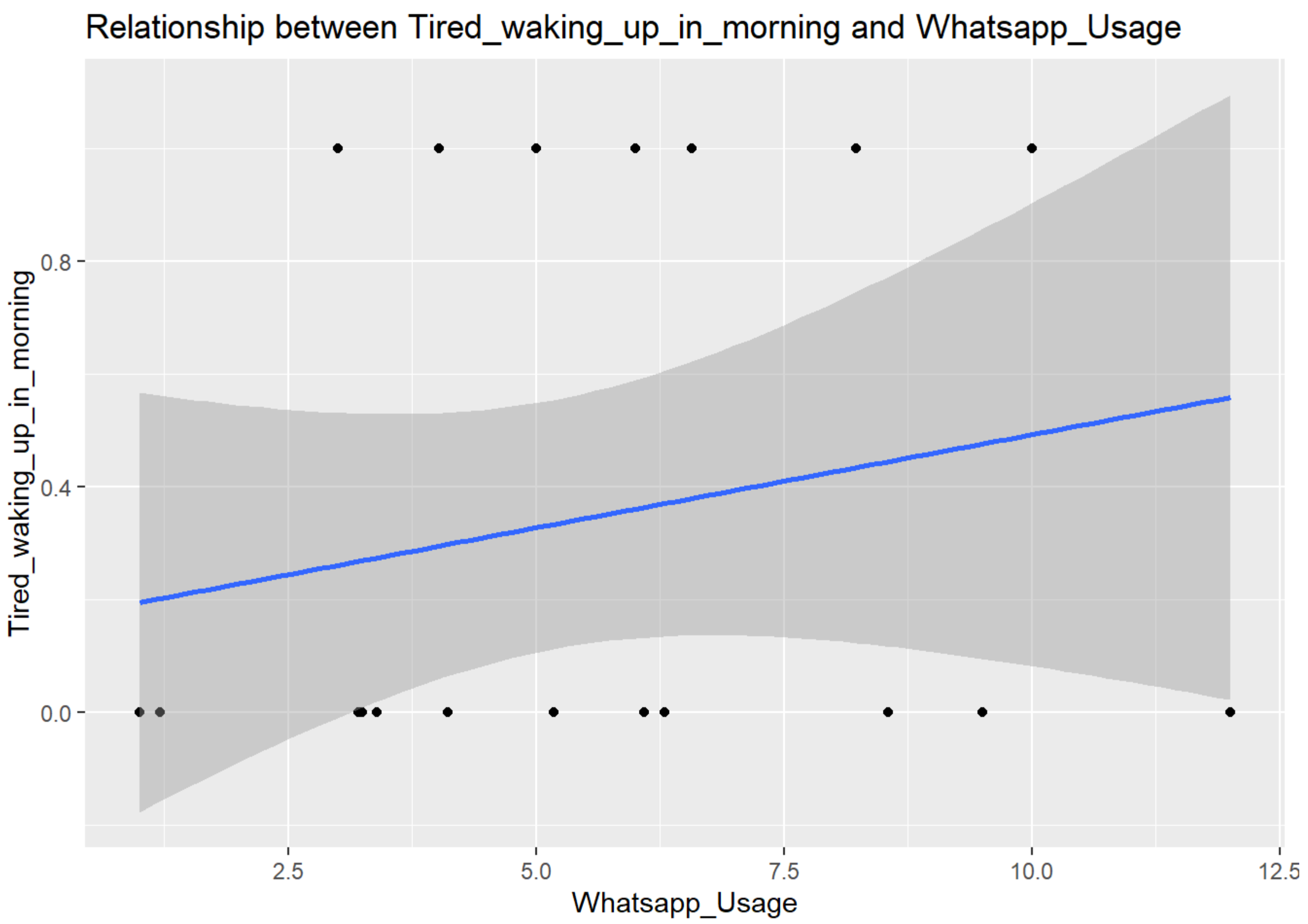
```
## `geom_smooth()` using formula = 'y ~ x'
```



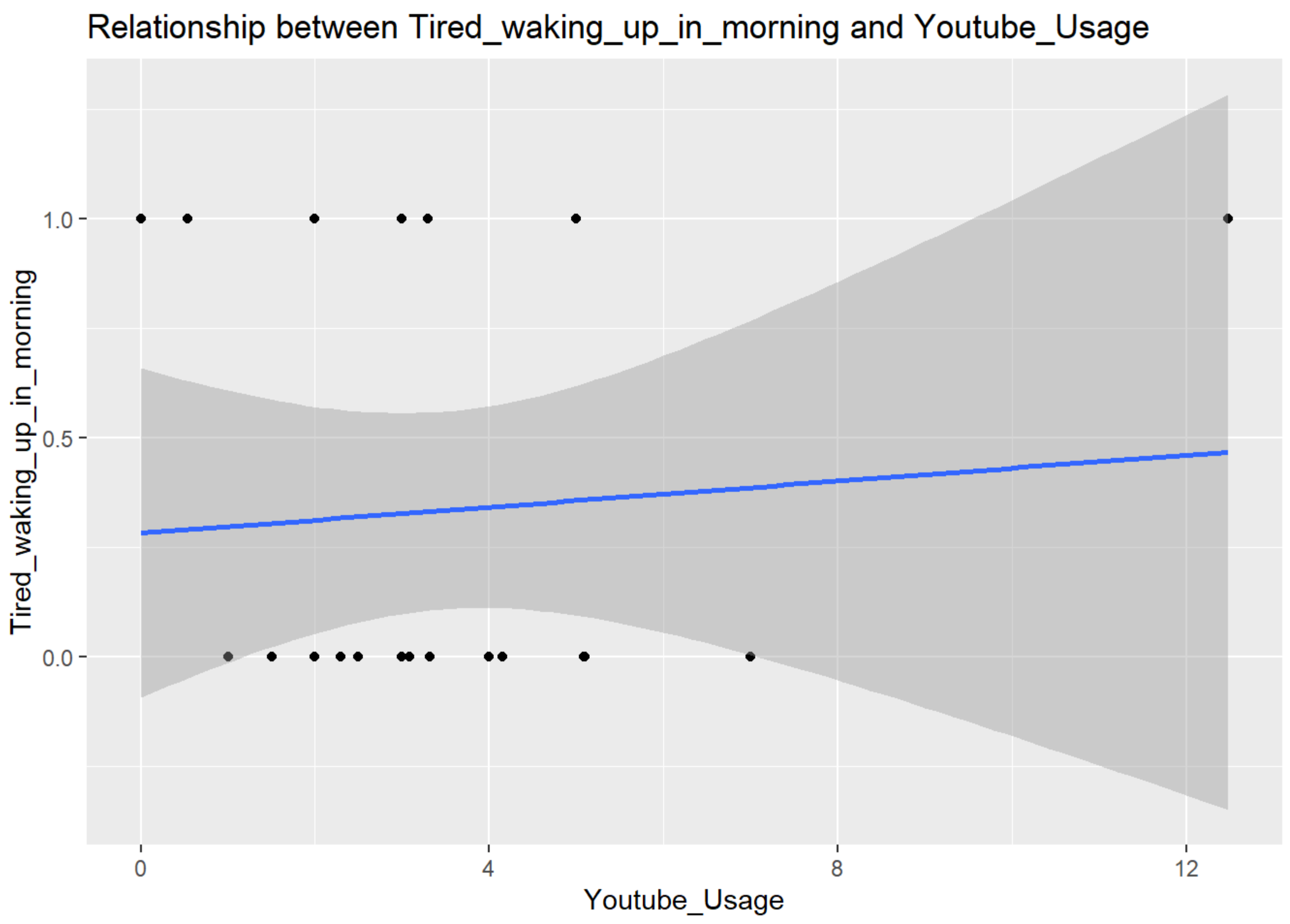
```
## `geom_smooth()` using formula = 'y ~ x'
```



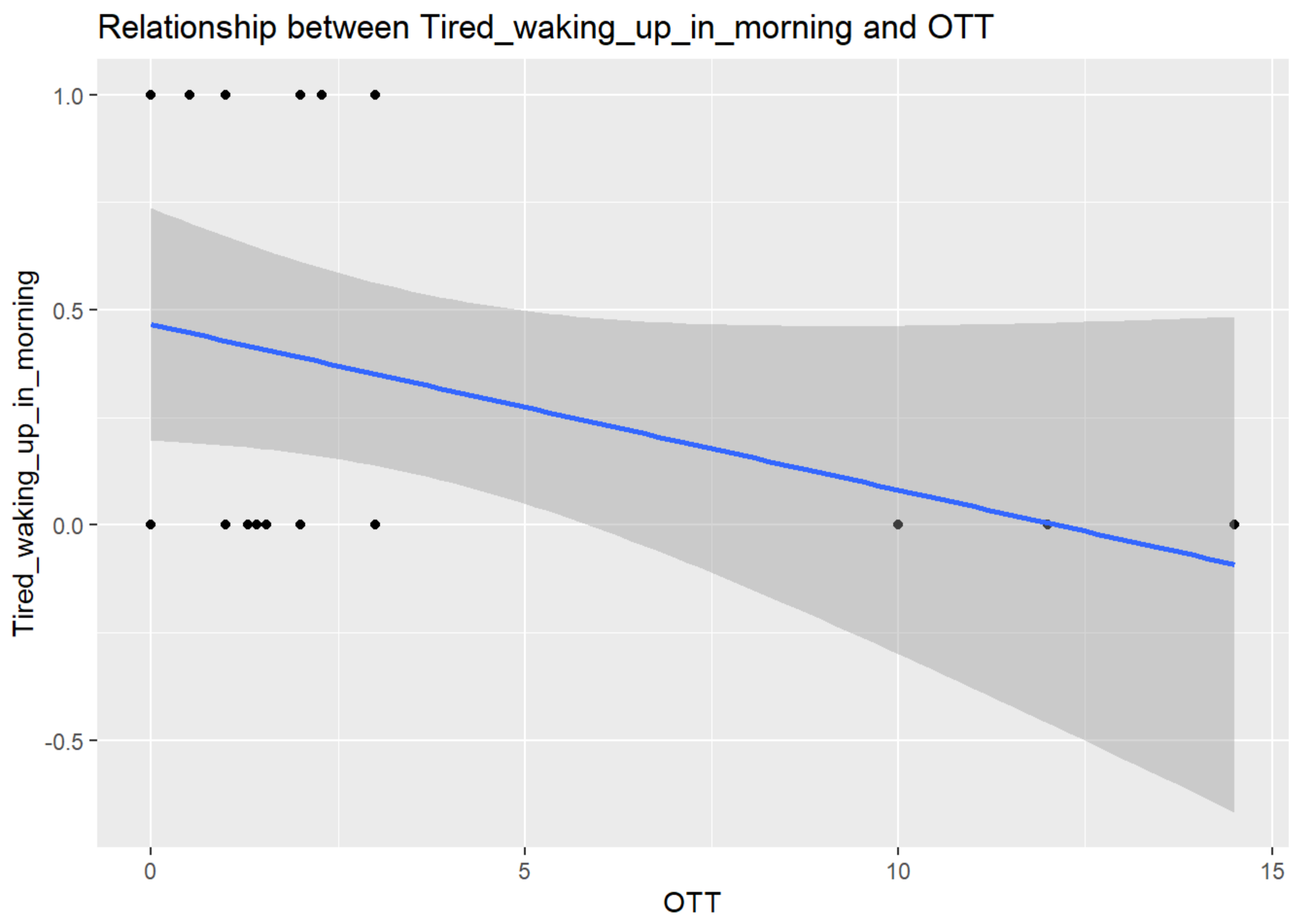
```
## `geom_smooth()` using formula = 'y ~ x'
```



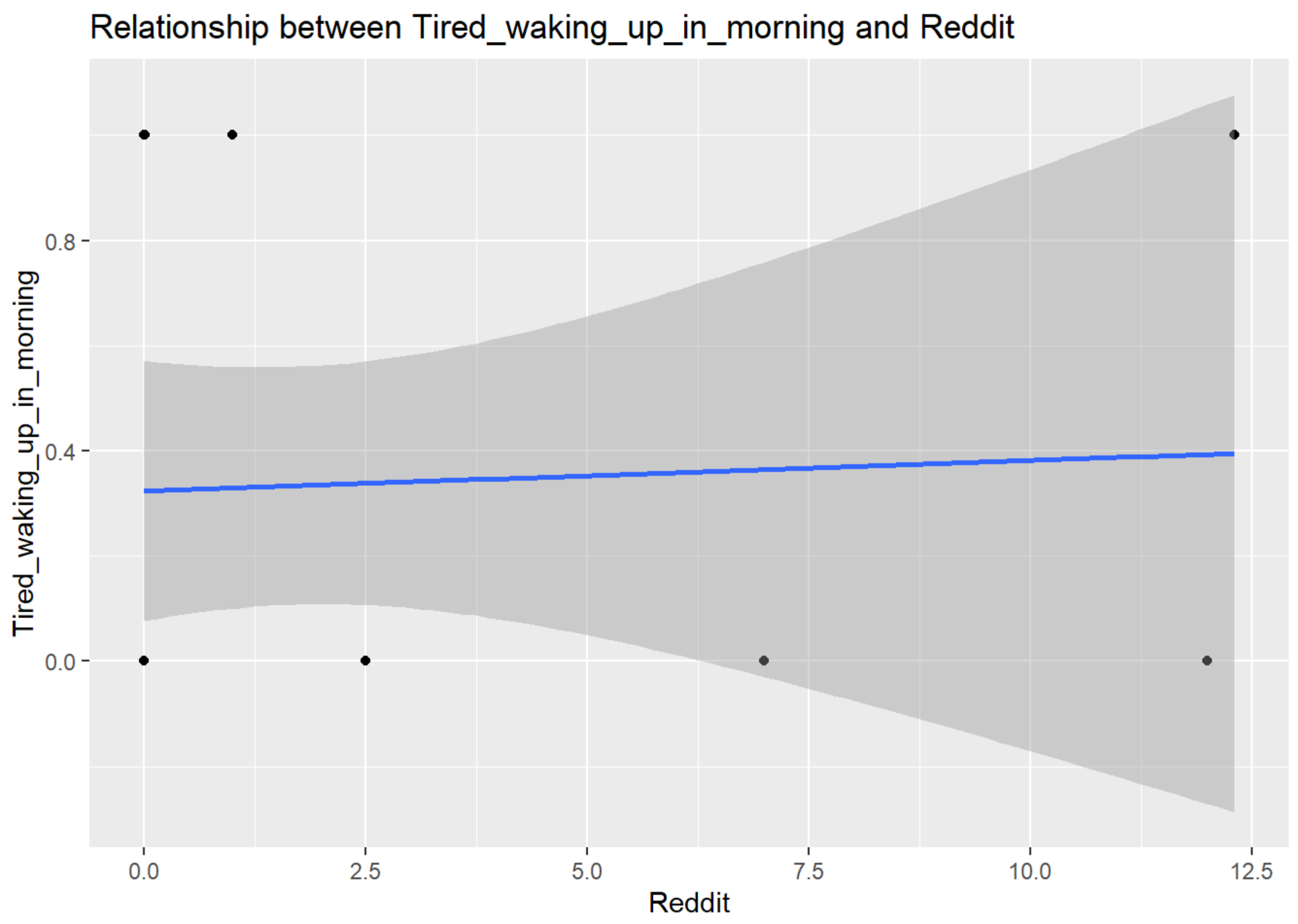
```
## `geom_smooth()` using formula = 'y ~ x'
```



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



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



Mean and Variance Analysis

```
mean_var_analysis <- function(data, variable_name) {
  mean_val <- mean(data[[variable_name]])
  var_val <- var(data[[variable_name]])

  cat("Variable:", variable_name, "\n")
  cat("Mean:", mean_val, "\n")
  cat("Variance:", var_val, "\n\n")
}

# Apply the function to each variable
mean_var_analysis(social_media_final, "Instagram_Usage")
```

```
## Variable: Instagram_Usage
## Mean: 5.778571
## Variance: 9.201723
```

```
mean_var_analysis(social_media_final, "LinkedIn_Usage")
```

```
## Variable: LinkedIn_Usage
## Mean: 5.784286
## Variance: 14.29608
```

```
mean_var_analysis(social_media_final, "Snapchat_Usage")
```

```
## Variable: Snapchat_Usage
## Mean: 4.04619
## Variance: 24.99824
```

```
mean_var_analysis(social_media_final, "Twitter_Usage")
```

```
## Variable: Twitter_Usage
## Mean: 2.252857
## Variance: 18.81777
```

```
mean_var_analysis(social_media_final, "Whatsapp_Usage")
```

```
## Variable: Whatsapp_Usage
## Mean: 5.172381
## Variance: 9.998999
```

```
mean_var_analysis(social_media_final, "Youtube_Usage")
```

```
## Variable: Youtube_Usage
## Mean: 3.460476
## Variance: 7.060935
```

```
mean_var_analysis(social_media_final, "OTT")
```

```
## Variable: OTT
## Mean: 3.454762
## Variance: 20.00731
```

```
mean_var_analysis(social_media_final, "Reddit")
```

```
## Variable: Reddit
## Mean: 1.657619
## Variance: 14.69192
```

```
mean_var_analysis(social_media_final, "Trouble_falling_asleep")
```

```
## Variable: Trouble_falling_asleep
## Mean: 0.3333333
## Variance: 0.2333333
```

```
mean_var_analysis(social_media_final, "Mood_Prod")
```

```
## Variable: Mood_Prod
## Mean: 0.952381
## Variance: 0.04761905
```

```
mean_var_analysis(social_media_final, "Tired_waking_up_in_morning")
```

```
## Variable: Tired_waking_up_in_morning
## Mean: 0.3333333
## Variance: 0.2333333
```

#Perform F-test and t-test

```
perform_tests <- function(variable_name, data) {
  # Perform F-test
  f_test <- var.test(data[[variable_name]] ~ data$Trouble_falling_asleep)

  # Perform t-test
  t_test <- t.test(data[[variable_name]] ~ data$Trouble_falling_asleep)

  # Print results
  cat("Variable:", variable_name, "\n")
  cat("F-test results:\n")
  cat("  F-statistic:", f_test$statistic, "\n")
  cat("  p-value:", f_test$p.value, "\n\n")
  cat("t-test results:\n")
  cat("  t-statistic:", t_test$statistic, "\n")
  cat("  p-value:", t_test$p.value, "\n\n")
}

# Perform tests for each variable
perform_tests("Instagram_Usage", social_media_final)
```

```
## Variable: Instagram_Usage
## F-test results:
##   F-statistic: 1.407548
##   p-value: 0.7048621
##
## t-test results:
##   t-statistic: 1.026571
##   p-value: 0.321821
```

```
perform_tests("LinkedIn_Usage", social_media_final)
```

```
## Variable: LinkedIn_Usage
## F-test results:
##   F-statistic: 1.182308
##   p-value: 0.8855713
##
```

```
## t-test results:
##   t-statistic: 0.8568985
##   p-value: 0.4069261
```

```
perform_tests("Snapchat_Usage", social_media_final)
```

```
## Variable: Snapchat_Usage
## F-test results:
##   F-statistic: 0.9254988
##   p-value: 0.8451404
##
## t-test results:
##   t-statistic: -0.9477588
##   p-value: 0.3624443
```

```
perform_tests("Twitter_Usage", social_media_final)
```

```
## Variable: Twitter_Usage
## F-test results:
##   F-statistic: 0.05308698
##   p-value: 2.046153e-05
##
## t-test results:
##   t-statistic: -2.057799
##   p-value: 0.08293642
```

```
perform_tests("Whatsapp_Usage", social_media_final)
```

```
## Variable: Whatsapp_Usage
## F-test results:
##   F-statistic: 1.435712
##   p-value: 0.6854257
##
## t-test results:
##   t-statistic: 0.9081149
##   p-value: 0.3788634
```

```
perform_tests("Youtube_Usage", social_media_final)
```

```
## Variable: Youtube_Usage
## F-test results:
##   F-statistic: 2.614543
##   p-value: 0.2452841
##
## t-test results:
##   t-statistic: 0.6270227
##   p-value: 0.5385795
```

```
perform_tests("OTT", social_media_final)
```

```
## Variable: OTT
## F-test results:
##   F-statistic: 0.4818352
##   p-value: 0.2541279
##
## t-test results:
##   t-statistic: -1.571152
##   p-value: 0.1506024
```

```
perform_tests("Reddit", social_media_final)
```

```
## Variable: Reddit
## F-test results:
##   F-statistic: 0.1099967
##   p-value: 0.0009873817
##
## t-test results:
##   t-statistic: -1.295042
##   p-value: 0.238333
```

```
sm <- read.csv("C:/Users/sakad/Downloads/social_media_cleaned.csv")
str(sm)
```

```
## 'data.frame':    21 obs. of  10 variables:
##  $ character                   : chr  "masinl" "peace" "Patty" "Bunny" ...
##  $ Instagram                   : num  3.5 7.73 3.77 5.38 12 2.33 5.37 7 8.65 0.17 ...
##  $ LinkedIn                    : num  4 5.2 7 5.32 0.58 7 4 4 10 0 ...
##  $ SnapChat                    : num  1 3.68 0.53 1.3 0 0.47 0 3 3.83 0 ...
##  $ Twitter                     : num  5 0 0 0 0.67 0 0 0 0 0 ...
##  $ Whatsapp.Wechat             : num  1 4.18 9.83 5.3 3 12 6 10 6.15 1 ...
##  $ youtube                     : num  2.5 4.25 1.85 2 3.5 7 3 2 4 3 ...
##  $ OTT                         : num  14.5 0 2 2 2 3 0 3 3 0 ...
##  $ Reddit                      : num  2.5 0 0 0 1 0 0 0 0 0 ...
##  $ How.you.felt.the.entire.week.: int  3 3 4 4 3 5 4 4 3 2 ...
```

```
sm1 <- sm[, 2:9]
sm1
```

```
##    Instagram LinkedIn SnapChat Twitter Whatsapp.Wechat youtube   OTT Reddit
## 1       3.50     4.00     1.00    5.00            1.00    2.50 14.50   2.50
## 2       7.73     5.20     3.68    0.00            4.18    4.25  0.00   0.00
## 3       3.77     7.00     0.53    0.00            9.83    1.85  2.00   0.00
## 4       5.38     5.32     1.30    0.00            5.30    2.00  2.00   0.00
## 5      12.00     0.58     0.00    0.67            3.00    3.50  2.00   1.00
## 6       2.33     7.00     0.47    0.00           12.00    7.00  3.00   0.00
## 7       5.37     4.00     0.00    0.00            6.00    3.00  0.00   0.00
## 8       7.00     4.00     3.00    0.00           10.00    2.00  3.00   0.00
## 9        8.65     10.00      3.83     0.00                6.15     4.00   3.00    0.00
## 10       0.17      0.00      0.00     0.00                1.00     3.00   0.00    0.00
## 11       4.58      2.13      0.42     2.42                1.00     1.00   0.00    0.00
## 12       3.33      2.50      0.33     2.83                3.67     2.50   1.50    0.00
## 13       5.30      1.52     14.87     0.00                4.02     0.54   0.51    0.01
## 14       6.00      3.00      1.00     1.00                3.42     2.50   1.00    0.00
## 15       4.65      3.75      1.42     0.00                6.50     3.13   1.92    7.00
## 16       7.00      5.00      0.42     0.00                5.00     5.00   1.00    0.50
## 17       9.80      0.80      0.00     0.27               13.35     3.53   1.68    0.00
## 18       6.80      1.92      1.87     0.00                6.95     0.80   2.47    0.00
## 19       5.67      3.92      0.00     0.00                8.92     5.15   0.00    0.00
## 20      15.02      3.95      7.32     0.00               15.35     5.17  10.00    0.00
## 21       0.33      0.52      0.00     0.00                8.38     0.00   0.00    0.00
```

PCA Q: How do relation between social media platforms inform our understanding of user behavior and interactions?

```
#Get the correlations between the variables
cor(sm1, use = "complete.obs")
```

```
##                   Instagram     LinkedIn    SnapChat    Twitter Whatsapp.Wechat
## Instagram        1.00000000  0.096836443  0.28970103 -0.1927972      0.37784893
## LinkedIn         0.09683644  1.000000000  0.02529643 -0.1306175      0.22899698
## SnapChat         0.28970103  0.025296428  1.00000000 -0.1801613      0.08104862
## Twitter         -0.19279717 -0.130617547 -0.18016133  1.0000000     -0.49574787
## Whatsapp.Wechat  0.37784893  0.228996980  0.08104862 -0.4957479      1.00000000
## youtube          0.33021732  0.452160243 -0.15977901 -0.1881819      0.37182165
## OTT              0.26738411  0.185463793  0.13165277  0.5569830      0.13356020
## Reddit          -0.07458765 -0.007079162 -0.08096904  0.1649095     -0.13445849
##                     youtube       OTT       Reddit
## Instagram        0.33021732 0.2673841 -0.074587647
## LinkedIn         0.45216024 0.1854638 -0.007079162
## SnapChat        -0.15977901 0.1316528 -0.080969036
## Twitter         -0.18818187 0.5569830  0.164909542
## Whatsapp.Wechat  0.37182165 0.1335602 -0.134458488
## youtube          1.00000000 0.1607195  0.026007762
## OTT              0.16071951 1.0000000  0.232981971
## Reddit           0.02600776 0.2329820  1.000000000
```

The result is a correlation matrix showing the relationships between different social media platforms based on their usage patterns or user engagement. Each cell in the matrix represents the correlation coefficient between pairs of platforms, ranging from -1 to 1. A value of 1 indicates a perfect positive correlation (when one platform’s usage increases, the other’s also tends to increase), while -1 indicates a perfect negative correlation (when one platform’s usage increases, the other’s tends to decrease).

For instance, the correlation between Instagram and LinkedIn is 0.097, suggesting a weak positive correlation, implying that users who engage more with Instagram may also have a slight tendency to engage more with LinkedIn. Conversely, the correlation between Twitter and Whatsapp/Wechat is -0.496, indicating a moderate negative correlation, implying that users who are highly active on Twitter may be less active on Whatsapp/Wechat, and vice versa.

The diagonal elements (where the platform is compared to itself) always have a correlation coefficient of 1 since a platform perfectly correlates with itself. This matrix helps understand how different social media platforms are related to each other in terms of user engagement or usage patterns, which can be valuable for marketers, researchers, and social media strategists in understanding user behavior and optimizing their strategies across multiple platforms.

Q: What are the implications of the rotated factor loadings on social media platforms in understanding their associations and contributions to different dimensions captured by principal components in factor analysis or principal component analysis?

```
#Computing Principal Components
social_pca <- prcomp(sm1,scale=TRUE)
social_pca
```

```
## Standard deviations (1, .., p=8):
## [1] 1.4937855 1.3062652 1.1302644 0.9384154 0.9007773 0.7693882 0.6079359
## [8] 0.3622309
##
## Rotation (n x k) = (8 x 8):
##                         PC1         PC2          PC3         PC4         PC5
## Instagram        0.43528749 -0.15383754  0.378064635  0.03091067  0.36691411
## LinkedIn         0.35654047 -0.20942720 -0.329973958 -0.19635277 -0.70307433
## SnapChat         0.15951942  0.03373703  0.717834288  0.10382304 -0.50998908
## Twitter         -0.39042544 -0.53811531  0.040061762 -0.33459600  0.09394086
## Whatsapp.Wechat  0.52692924  0.06069670  0.007260553  0.09076207  0.28302310
## youtube          0.45376608 -0.20713849 -0.406499207 -0.03646460  0.09495664
## OTT              0.08164899 -0.68438404  0.215954720 -0.12833890  0.07485857
## Reddit          -0.12369550 -0.35608430 -0.139149370  0.90095400 -0.08616338
##                         PC6         PC7         PC8
## Instagram       -0.51106683 -0.48946149  0.08707842
## LinkedIn         0.02588829 -0.42218323  0.09143040
## SnapChat        -0.04950250  0.39275988  0.17478671
## Twitter         -0.02565582  0.04598525  0.65790782
## Whatsapp.Wechat  0.68945551  0.02304825  0.39292343
## youtube         -0.39820221  0.64499580  0.03189625
## OTT              0.31729474  0.06579314 -0.59265177
## Reddit          -0.02102173 -0.07024430  0.11841633
```

The provided result represents the rotated factor loadings from a factor analysis or principal component analysis (PCA) of various social media platforms. In such analyses, multiple variables (in this case, social media platforms) are examined to identify underlying factors or components that explain the shared variance among them. The table shows the loadings of each social media platform on the identified principal components (PCs), denoted as PC1 through PC8. Each row corresponds to a social media platform, and each column represents a principal component.

For Example: - Instagram has high positive loadings on PC1, PC2, and PC3, indicating a strong association with these components. - LinkedIn loads highly on PC1 and PC5 but negatively on PC2 and PC7, suggesting a more complex relationship with these components. - Snapchat shows a substantial loading on PC3 but negligible loadings on other components, implying its strong association with PC3 only.

These loadings represent the correlation between each social media platform and the identified components. Higher absolute values (close to 1) suggest a stronger relationship, while values closer to 0 indicate a weaker association. The rotated factor loadings help in interpreting the underlying structure of the data, revealing which social media platforms are closely related and how they contribute to different dimensions captured by the principal components.

```
summary(social_pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5     PC6    PC7    PC8
## Standard deviation     1.4938 1.3063 1.1303 0.9384 0.9008 0.76939 0.6079 0.3622
## Proportion of Variance 0.2789 0.2133 0.1597 0.1101 0.1014 0.07399 0.0462 0.0164
## Cumulative Proportion  0.2789 0.4922 0.6519 0.7620 0.8634 0.93740 0.9836 1.0000
```

```
eigen_social<- social_pca$sdev^2
eigen_social
```

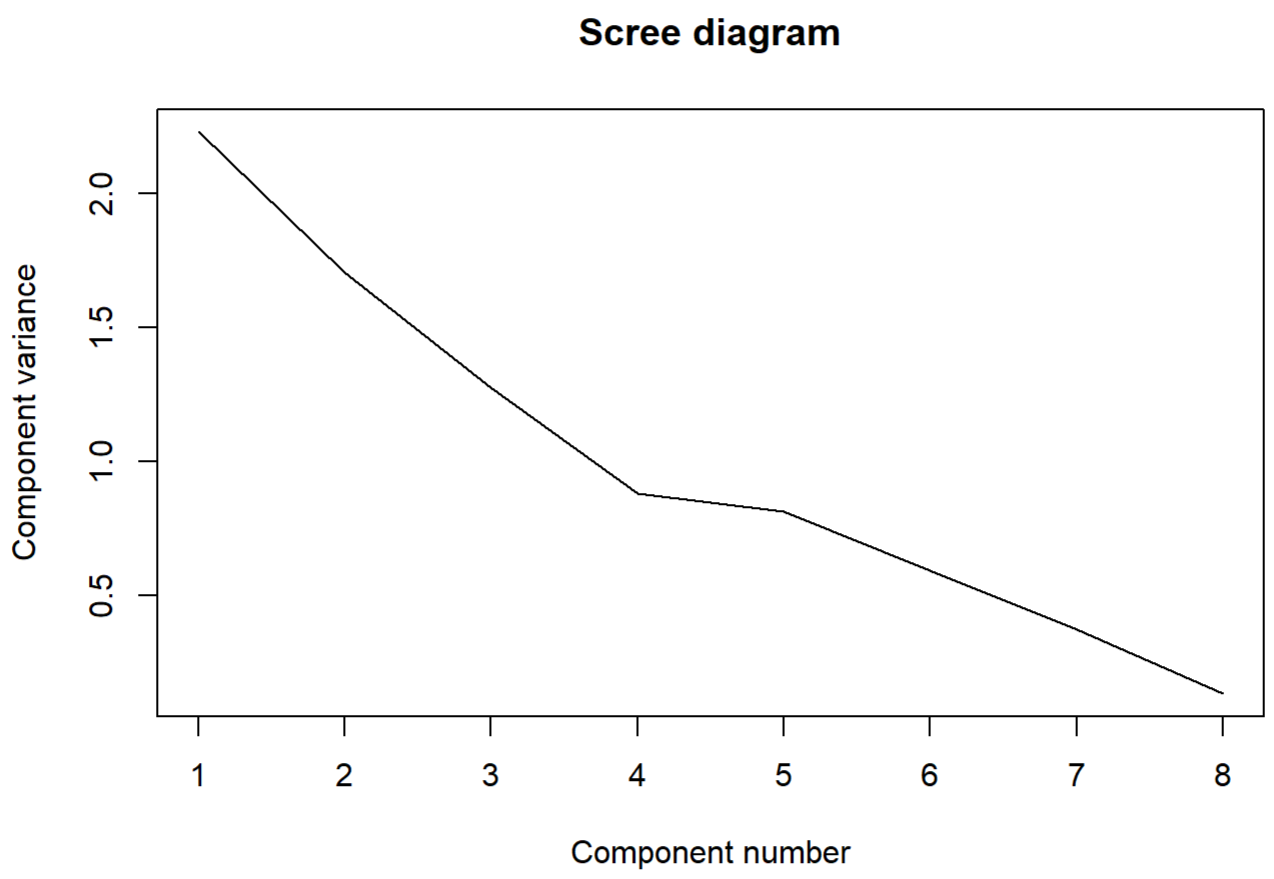
```
## [1] 2.2313950 1.7063288 1.2774976 0.8806234 0.8113997 0.5919581 0.3695861
## [8] 0.1312112
```

The PCA analysis reveals that PC1 and PC2 together contribute approximately 50% of the total variance.

Q: What the variance explained by each component, and how does it suggest the number of components to retain for analysis?

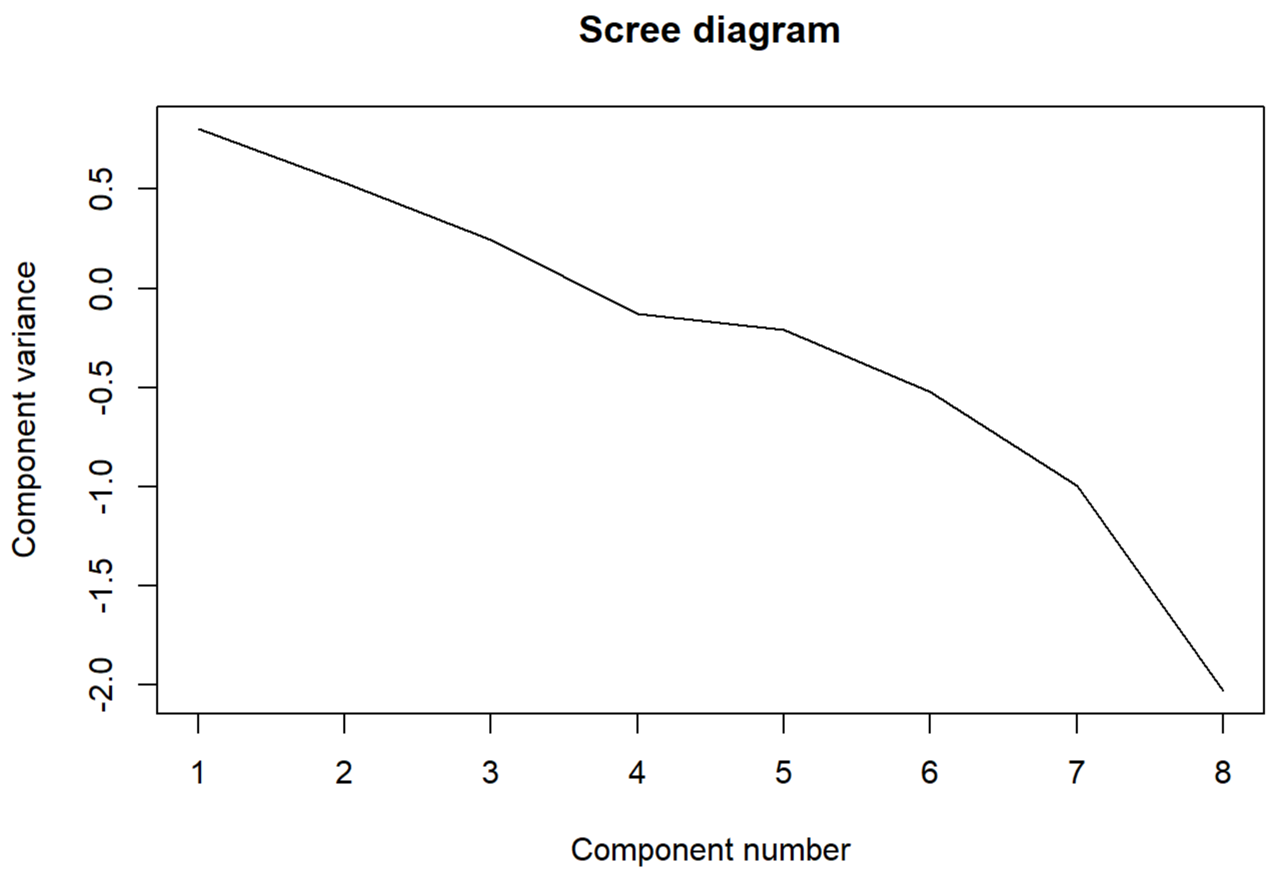
Screeplot

```
plot(eigen_social, xlab = "Component number", ylab = "Component variance", type = "l", main =
"Scree diagram")
```



From this Scree diagram, we can see that the first component has a variance just above 2.0, and the variance decreases with each additional component. The curve starts to flatten out after the third component, suggesting that retaining the first two or three components might be appropriate for this particular analysis. Components beyond this point are often considered to contribute less to explaining the variance and may be less meaningful.

```
plot(log(eigen_social), xlab = "Component number", ylab = "Component variance", type = "l", ma
in = "Scree diagram")
```



Based on the scree plot, it’s advantageous to include PC1 through PC5, as they collectively explain 84% of the total variance.

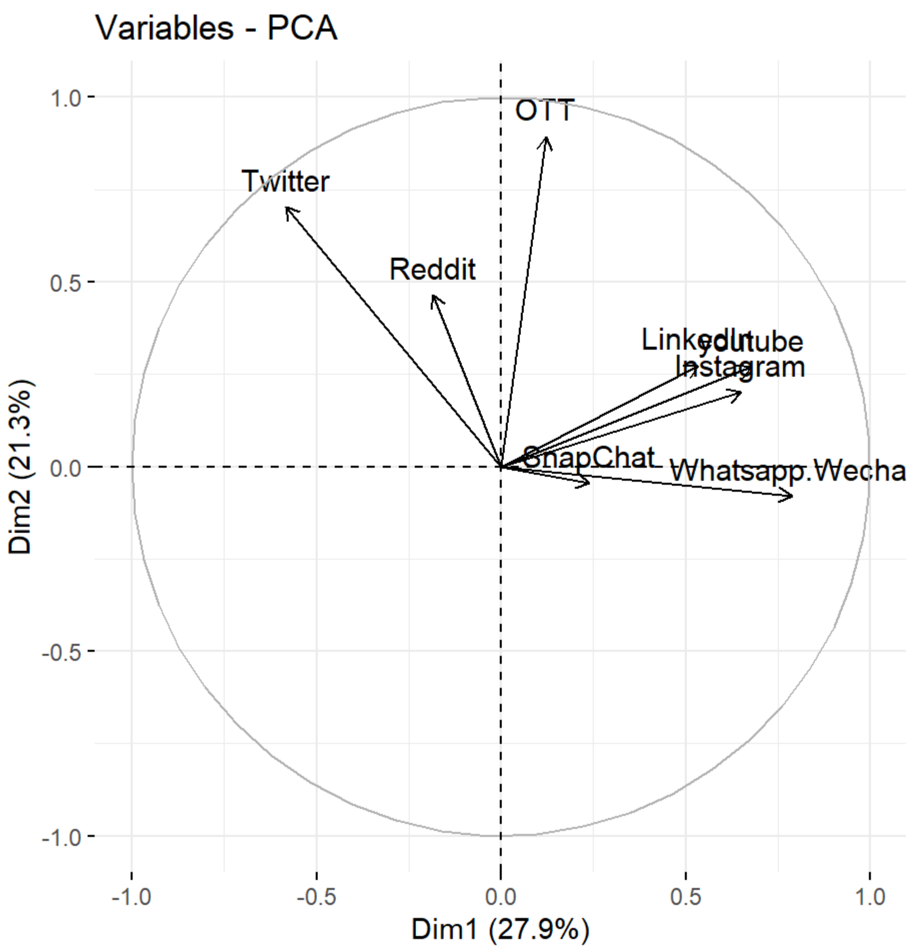
Visualization using Principal Components

```
library(FactoMineR)
library("factoextra")
```

```
## Warning: package 'factoextra' was built under R version 4.3.3
```

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

```
res.pca <- PCA(sm1, graph = FALSE)
fviz_pca_var(res.pca, col.var = "black")
```



Factor Analysis

```r
# load library for factor analysis
library(ggplot2)
library(psych)
```

Q: Determine the optimal number of factors for your dataset?

```r
fa.parallel(sm1)
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.

## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

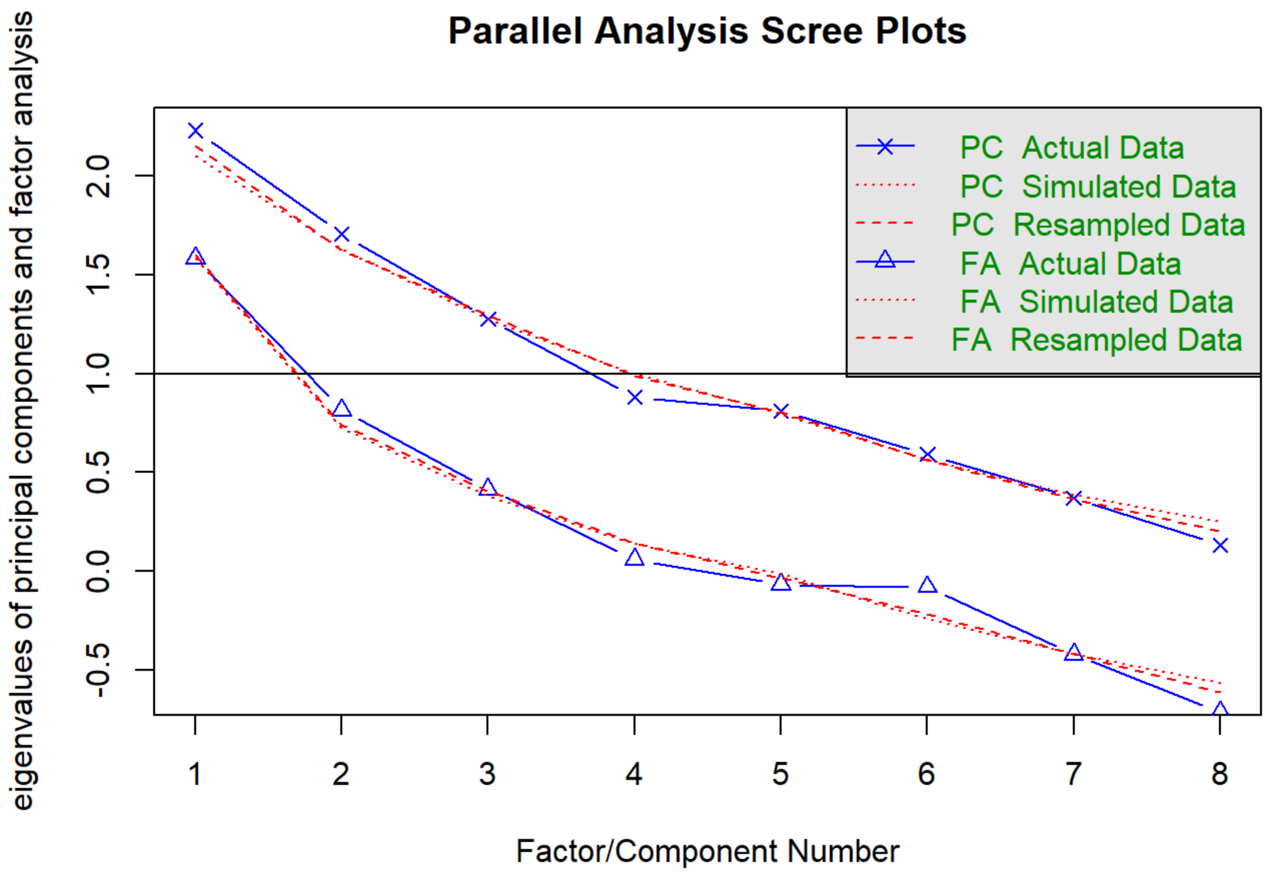
```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```



```
## Parallel analysis suggests that the number of factors =  0  and the number of components =
 0
```

Parallel analysis indicates that both the number of factors and the number of components are zero.

Q: Provide an explanation for the output of your factor model?

```
fit.pc <- principal(sm1, nfactors=2, rotate="varimax")
fit.pc
```

```
## Principal Components Analysis
## Call: principal(r = sm1, nfactors = 2, rotate = "varimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                   RC1   RC2    h2   u2 com
## Instagram         0.68  0.01 0.463 0.54 1.0
## LinkedIn          0.59  0.11 0.358 0.64 1.1
## SnapChat          0.22 -0.11 0.059 0.94 1.5
## Twitter          -0.36  0.84 0.834 0.17 1.4
## Whatsapp.Wechat   0.73 -0.30 0.626 0.37 1.3
## youtube           0.73  0.07 0.533 0.47 1.0
## OTT               0.37  0.82 0.814 0.19 1.4
## Reddit           -0.04  0.50 0.250 0.75 1.0
##
##                        RC1  RC2
## SS loadings           2.19 1.75
## Proportion Var        0.27 0.22
## Cumulative Var        0.27 0.49
## Proportion Explained  0.56 0.44
## Cumulative Proportion 0.56 1.00
##
## Mean item complexity =  1.2
## Test of the hypothesis that 2 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.13
##  with the empirical chi square  21.05  with prob <  0.072
##
## Fit based upon off diagonal values = 0.71
```

- Large absolute values (near 1) signify a robust association between the variable and the factor.
- h2 quantifies the extent to which factors account for variable variance.
- u2 denotes the portion of variance not captured by the factors.

Principal Components Analysis Call: principal(r = sm1, nfactors = 2, rotate = "varimax") Standardized loadings (pattern matrix) based upon correlation matrix

```
                   RC1  RC2
```

SS loadings 2.27 1.80 Proportion Var 0.25 0.20 Cumulative Var 0.25 0.45 Proportion Explained 0.56 0.44 Cumulative Proportion 0.56 1.00

Mean item complexity = 1.3 Test of the hypothesis that 2 components are sufficient.

The root mean square of the residuals (RMSR) is 0.14 with the empirical chi square 29.01 with prob < 0.066

```
round(fit.pc$values, 3)
```

```
## [1] 2.231 1.706 1.277 0.881 0.811 0.592 0.370 0.131
```

```
fit.pc$loadings
```

```
##
## Loadings:
##                  RC1    RC2
## Instagram         0.681
## LinkedIn          0.588  0.111
## SnapChat          0.216 -0.110
## Twitter          -0.359  0.840
## Whatsapp.Wechat   0.732 -0.300
## youtube           0.727
## OTT               0.371  0.822
## Reddit                   0.498
##
##                   RC1   RC2
## SS loadings    2.189 1.749
## Proportion Var 0.274 0.219
## Cumulative Var 0.274 0.492
```

```
# Communalities
fit.pc$communality
```

```
##       Instagram        LinkedIn        SnapChat         Twitter Whatsapp.Wechat 
##       0.4631760       0.3584966       0.0587232       0.8342344       0.6258430 
##         youtube             OTT          Reddit 
##       0.5326647       0.8140886       0.2504973
```

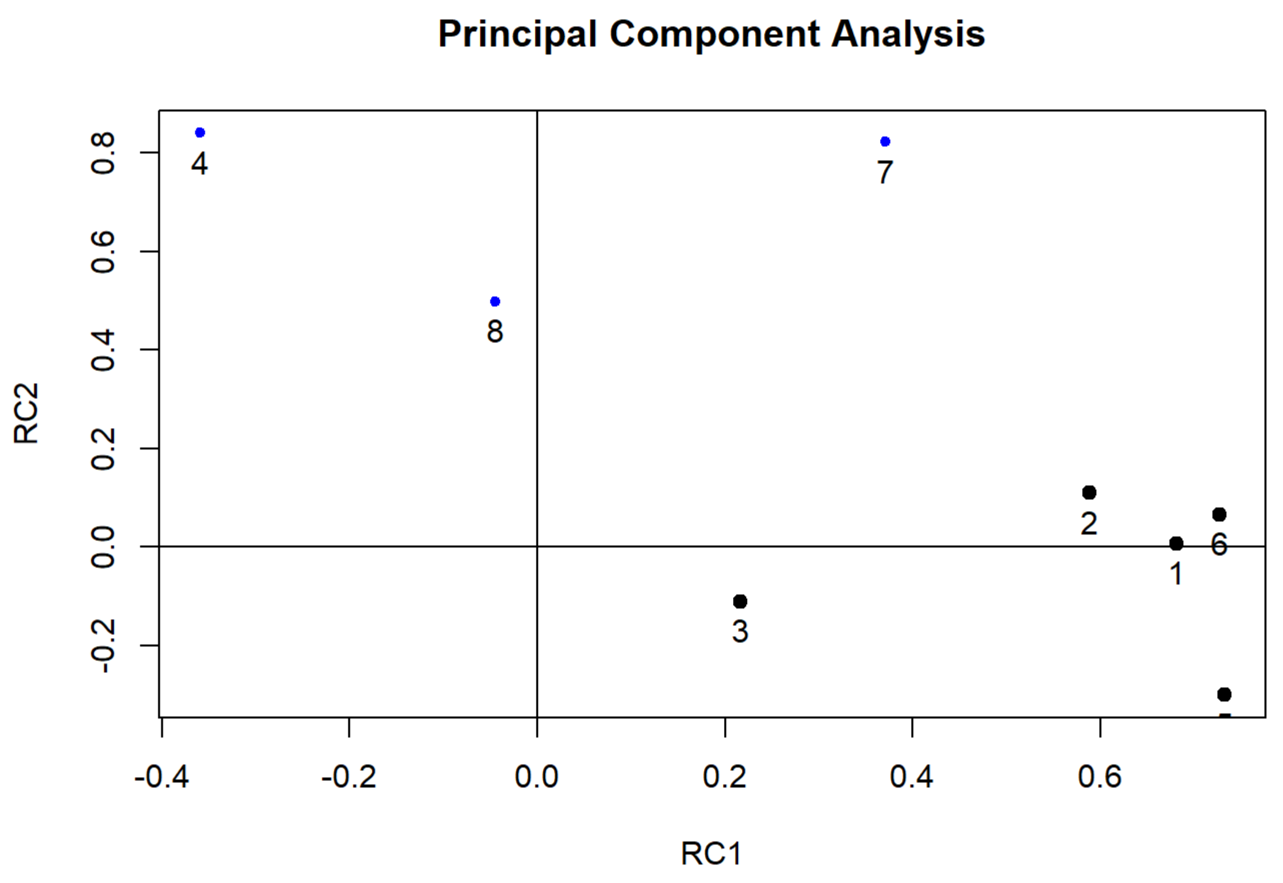
```
# Rotated factor scores, Notice the columns ordering: RC1, RC2
fit.pc
```

```
## Principal Components Analysis
## Call: principal(r = sm1, nfactors = 2, rotate = "varimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                   RC1   RC2    h2   u2 com
## Instagram        0.68  0.01 0.463 0.54 1.0
## LinkedIn         0.59  0.11 0.358 0.64 1.1
## SnapChat         0.22 -0.11 0.059 0.94 1.5
## Twitter         -0.36  0.84 0.834 0.17 1.4
## Whatsapp.Wechat  0.73 -0.30 0.626 0.37 1.3
## youtube          0.73  0.07 0.533 0.47 1.0
## OTT              0.37  0.82 0.814 0.19 1.4
## Reddit          -0.04  0.50 0.250 0.75 1.0
## 
##                        RC1  RC2
## SS loadings           2.19 1.75
## Proportion Var        0.27 0.22
## Cumulative Var        0.27 0.49
## Proportion Explained  0.56 0.44
## Cumulative Proportion 0.56 1.00
## 
## Mean item complexity =  1.2
## Test of the hypothesis that 2 components are sufficient.
## 
## The root mean square of the residuals (RMSR) is  0.13 
##  with the empirical chi square  21.05  with prob <  0.072 
## 
## Fit based upon off diagonal values = 0.71
```

```
fit.pc$scores
```

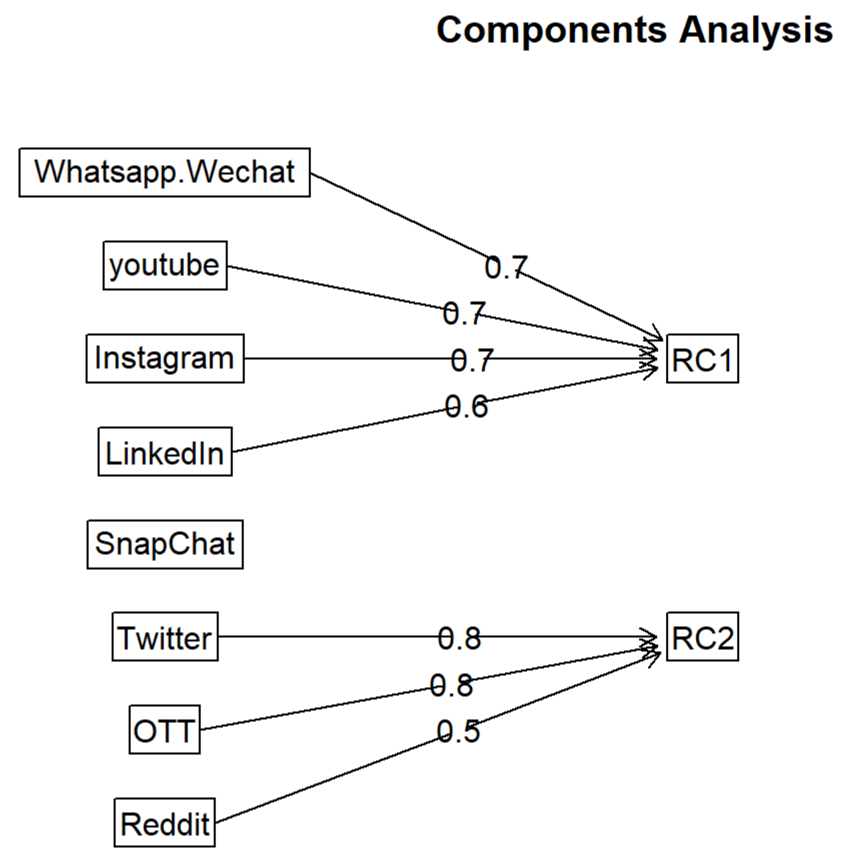
```
##               RC1         RC2
##  [1,] -0.49624055  3.81931118
##  [2,]  0.37381791 -0.45829983
##  [3,]  0.24026578 -0.39773035
##  [4,] -0.11813877 -0.28581917
##  [5,] -0.05612179  0.18892772
##  [6,]  1.33950136 -0.15489910
##  [7,] -0.16510634 -0.57411254
##  [8,]  0.36248103 -0.37606120
##  [9,]  1.27363310  0.04673432
## [10,] -1.47442593 -0.57714722
## [11,] -1.47172675  0.33207687
## [12,] -0.99536049  0.65208895
## [13,] -0.64535181 -0.84676594
## [14,] -0.54371772 -0.04386217
## [15,] -0.05085497  0.89227335
## [16,]  0.46515473 -0.19348386
## [17,]  0.63768723 -0.56192633
## [18,] -0.42546987 -0.44047678
## [19,]  0.50443809 -0.59824502
## [20,]  2.63331377  0.51395273
## [21,] -1.38777801 -0.93653561
```

```
fa.plot(fit.pc) # See Correlations within Factors
```



Q: Show the columns that go into each factor?

```
fa.diagram(fit.pc) # Visualize the relationship
```



This displays a diagram titled "Components Analysis," which seems to represent a factor analysis or a similar statistical method that groups different social media platforms into components based on their relationships. The platforms listed are WhatsApp/WeChat, YouTube, Instagram, LinkedIn, SnapChat, Twitter, OTT, and Reddit.

Two components are identified in the diagram: RC1 and RC2. RC1 is linked with four platforms: YouTube, Instagram, LinkedIn, and WhatsApp/WeChat, with respective loadings of 0.7, 0.7, 0.7, and 0.6. RC2 is associated with Twitter, OTT, and Reddit, with loadings of 0.8, 0.8, and 0.5 respectively. The loadings represent how strongly each platform is related to the respective component, with higher values indicating a stronger relationship.

Creating visual representations utilizing the factors.

```
#very simple structure visualization
vss(sm1)
```

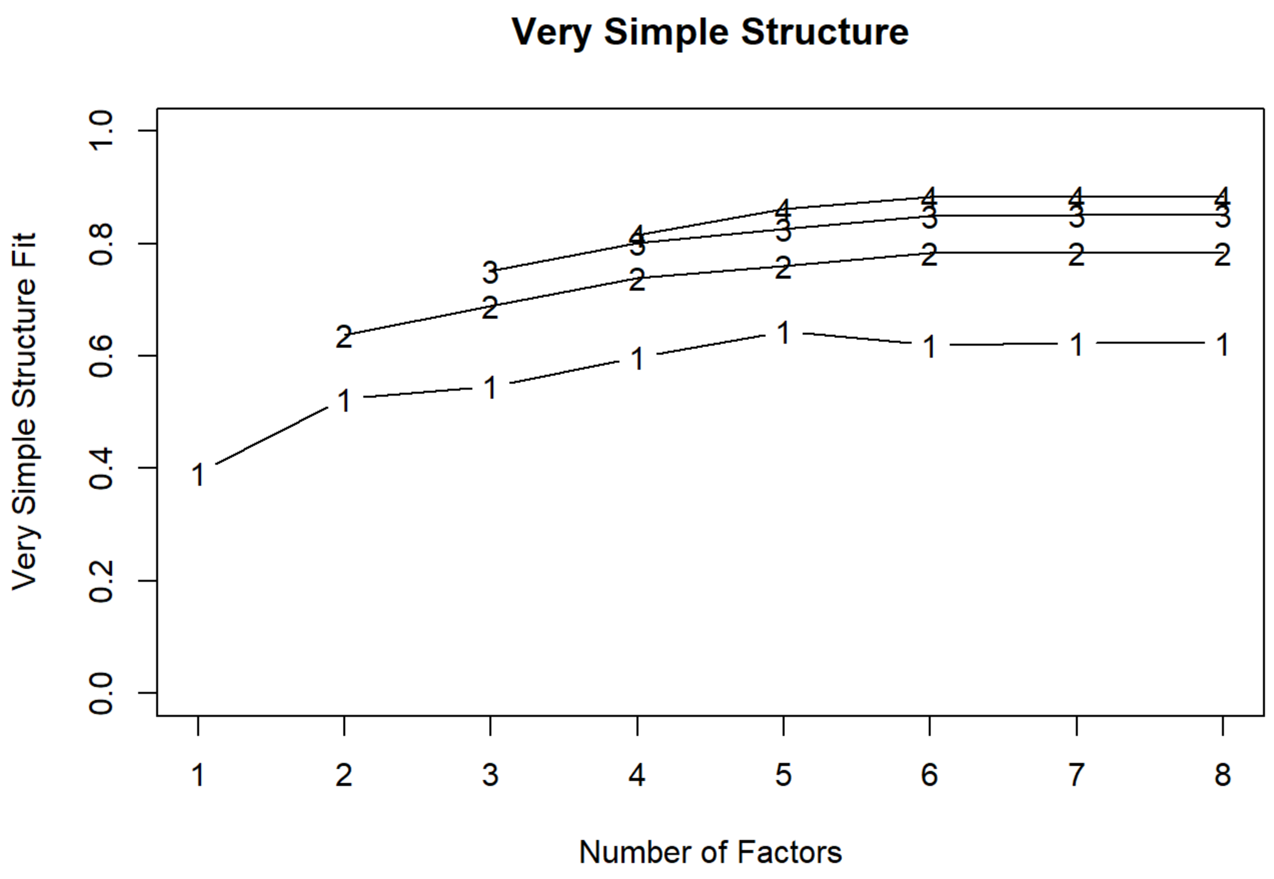
```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```



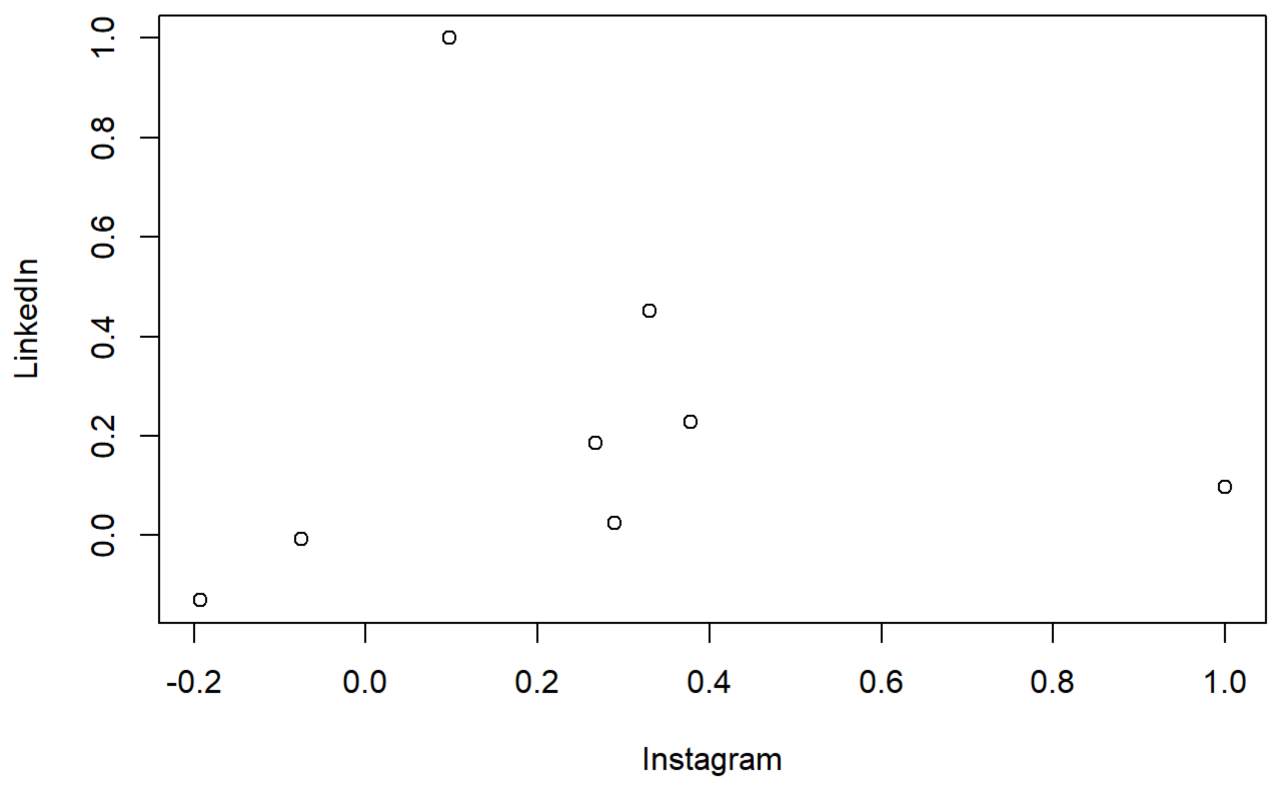
```
##
## Very Simple Structure
## Call: vss(x = sm1)
## VSS complexity 1 achieves a maximimum of 0.64  with  5  factors
## VSS complexity 2 achieves a maximimum of 0.78  with  7  factors
##
## The Velicer MAP achieves a minimum of 0.07  with  1  factors
## BIC achieves a minimum of  -35.55  with  1  factors
## Sample Size adjusted BIC achieves a minimum of  1.71  with  4  factors
##
## Statistics by number of factors
##   vss1 vss2   map dof   chisq prob sqresid  fit RMSEA   BIC SABIC complex
## 1 0.39 0.00 0.071  20 2.5e+01 0.19     7.0 0.39   0.1 -35.6  26.2     1.0
## 2 0.52 0.64 0.090  13 8.9e+00 0.78     4.2 0.64   0.0 -30.7   9.4     1.2
## 3 0.55 0.69 0.117   7 4.7e+00 0.70     2.9 0.75   0.0 -16.6   5.0      1.5
## 4 0.60 0.74 0.179   2 1.6e+00 0.44     2.1 0.81   0.0  -4.5   1.7      1.5
## 5 0.64 0.76 0.271  -2 6.5e-01   NA     1.5 0.87    NA    NA    NA      1.3
## 6 0.62 0.78 0.442  -5 4.5e-10   NA     1.2 0.90    NA    NA    NA      1.5
## 7 0.62 0.78 1.000  -7 0.0e+00   NA     1.2 0.90    NA    NA    NA      1.5
## 8 0.62 0.78    NA  -8 0.0e+00   NA     1.2 0.90    NA    NA    NA      1.5
##    eChisq    SRMR eCRMS  eBIC
## 1 3.2e+01 1.7e-01 0.196 -28.5
## 2 9.2e+00 8.8e-02 0.130 -30.4
## 3 2.1e+00 4.2e-02 0.084 -19.2
## 4 8.0e-01 2.6e-02 0.097  -5.3
## 5 2.2e-01 1.4e-02    NA    NA
## 6 1.8e-10 3.9e-07    NA    NA
## 7 2.2e-18 4.3e-11    NA    NA
## 8 2.2e-18 4.3e-11    NA    NA
```

The analysis suggests that a simpler structure with one factor is sufficient, as indicated by a maximum of 0.61 for VSS complexity 1 and a minimum of 0.06 for Velicer MAP. However, considering other criteria such as BIC and sample size-adjusted BIC, a model with five factors appears to be the most appropriate, with BIC reaching a minimum of -53.17 and sample size-adjusted BIC at 1.47.

```
# Computing Correlation Matrix
corrm.social <- cor(sm1)
corrm.social
```

```
##                   Instagram     LinkedIn    SnapChat    Twitter Whatsapp.Wechat
## Instagram        1.00000000  0.096836443  0.28970103 -0.1927972      0.37784893
## LinkedIn         0.09683644  1.000000000  0.02529643 -0.1306175      0.22899698
## SnapChat         0.28970103  0.025296428  1.00000000 -0.1801613      0.08104862
## Twitter         -0.19279717 -0.130617547 -0.18016133  1.0000000     -0.49574787
## Whatsapp.Wechat  0.37784893  0.228996980  0.08104862 -0.4957479      1.00000000
## youtube          0.33021732  0.452160243 -0.15977901 -0.1881819      0.37182165
## OTT              0.26738411  0.185463793  0.13165277  0.5569830      0.13356020
## Reddit          -0.07458765 -0.007079162 -0.08096904  0.1649095     -0.13445849
##                     youtube       OTT       Reddit
## Instagram        0.33021732 0.2673841 -0.074587647
## LinkedIn         0.45216024 0.1854638 -0.007079162
## SnapChat        -0.15977901 0.1316528 -0.080969036
## Twitter         -0.18818187 0.5569830  0.164909542
## Whatsapp.Wechat  0.37182165 0.1335602 -0.134458488
## youtube          1.00000000 0.1607195  0.026007762
## OTT              0.16071951 1.0000000  0.232981971
## Reddit           0.02600776 0.2329820  1.000000000
```

```
plot(corrm.social)
```



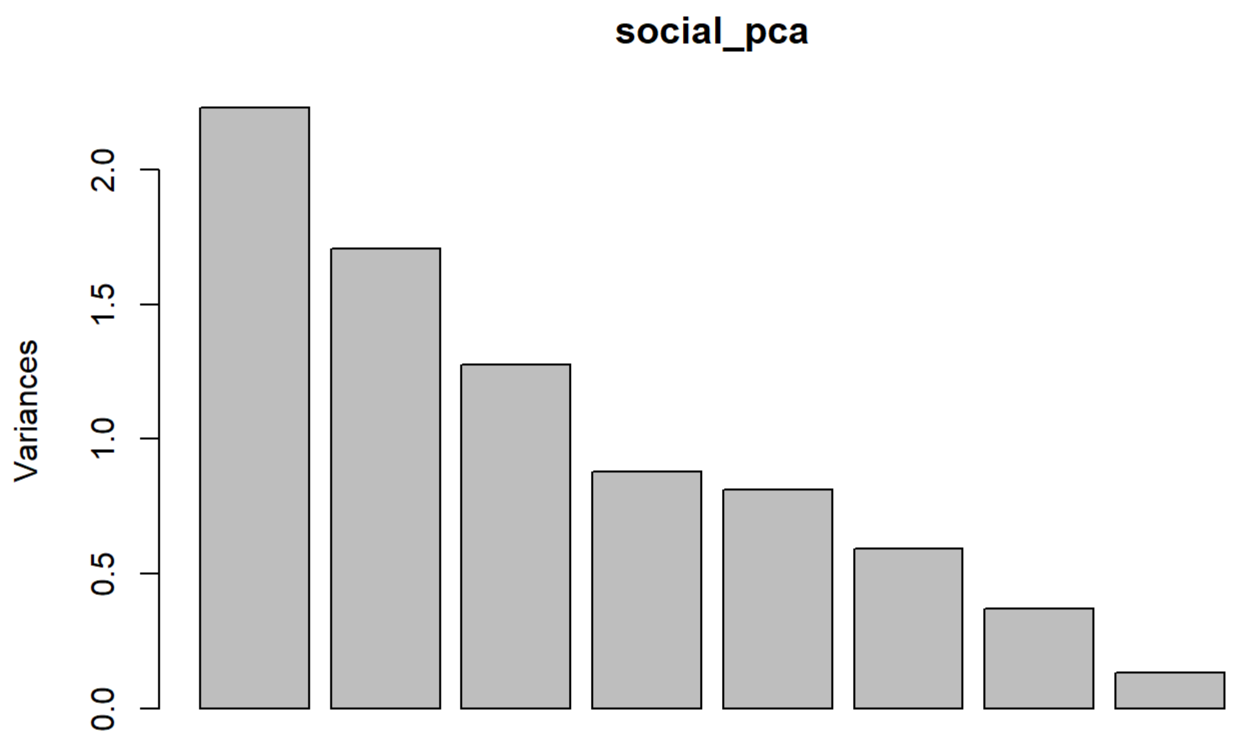
Q: What is the significance of each principal component in explaining the variance and capturing the underlying structure of the data in the factor analysis?

```
social_pca <- prcomp(sm1, scale=TRUE)
summary(social_pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5     PC6    PC7    PC8
## Standard deviation     1.4938 1.3063 1.1303 0.9384 0.9008 0.76939 0.6079 0.3622
## Proportion of Variance 0.2789 0.2133 0.1597 0.1101 0.1014 0.07399 0.0462 0.0164
## Cumulative Proportion  0.2789 0.4922 0.6519 0.7620 0.8634 0.93740 0.9836 1.0000
```

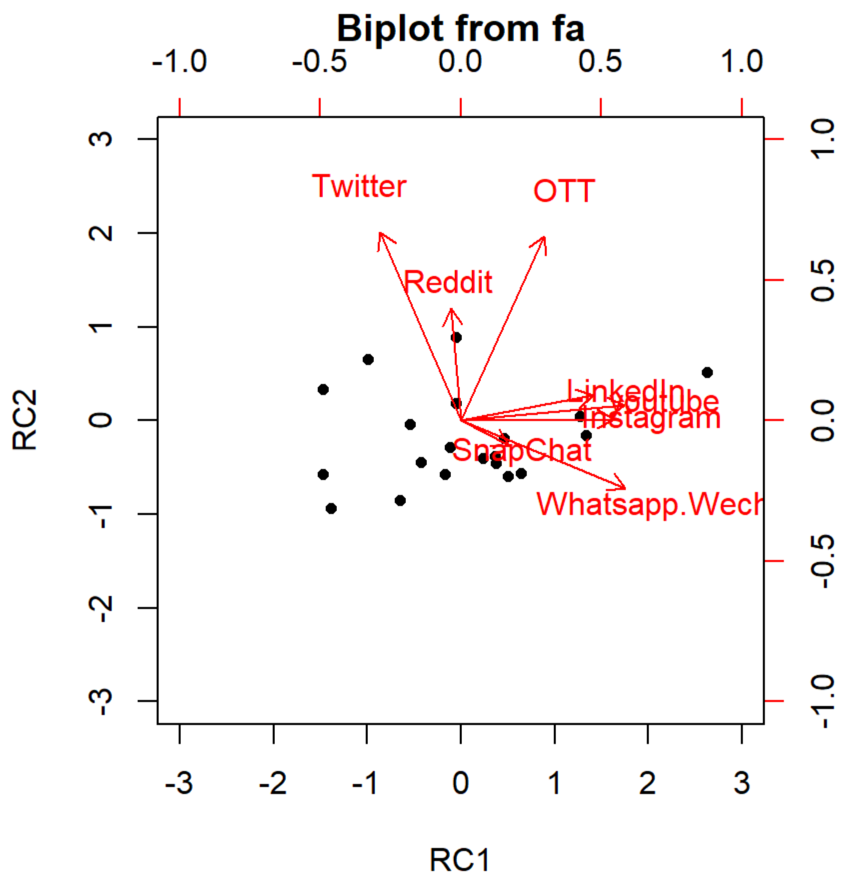
The results represents the importance of each principal component (PC) in the factor analysis. The standard deviation indicates the variability captured by each PC, with PC1 having the highest at 1.4937 and subsequent PCs decreasing in magnitude. The proportion of variance signifies the percentage of total variance explained by each PC, with PC1 contributing the most at 27.89% and subsequent PCs decreasing in importance. The cumulative proportion shows the accumulated contribution of each PC to the total variance, with PC1 accounting for 27.89% and subsequent PCs incrementally adding to reach 100%. This information helps understand the significance of each PC in capturing the underlying structure of the data, with PC1 being the most influential, followed by subsequent PCs in descending order of importance.

```
plot(social_pca)
```



Q: What are the main insights obtained from the relationship between social media platforms and individual observations in the factor analysis? Biplot

```
biplot(fit.pc)
```



The biplot visualizes the results of a factor analysis, showing the relationship between various social media platforms and services (represented by red arrows) and the scores of individual observations (represented by black dots) across two principal components, RC1 and RC2. The social media platforms—Twitter, OTT content, Reddit, LinkedIn, YouTube, Instagram, Snapchat, and Whatsapp/Wechat—are plotted as vectors, with their direction and length indicating their contribution and correlation to the components. Platforms with similar orientations have similar factor profiles, suggesting comparable characteristics or usage patterns. The position of the data points indicates how each observation scores on the components, with clustering points suggesting groups with similar factor scores. The biplot is a useful tool for interpreting the underlying structure of the data, revealing the dominant patterns of variation and the relationships between the variables and observations.

Cluster Analysis

```
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 4.3.3
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:patchwork':
##
##     area
## The following object is masked from 'package:dplyr':
##
##     select
```

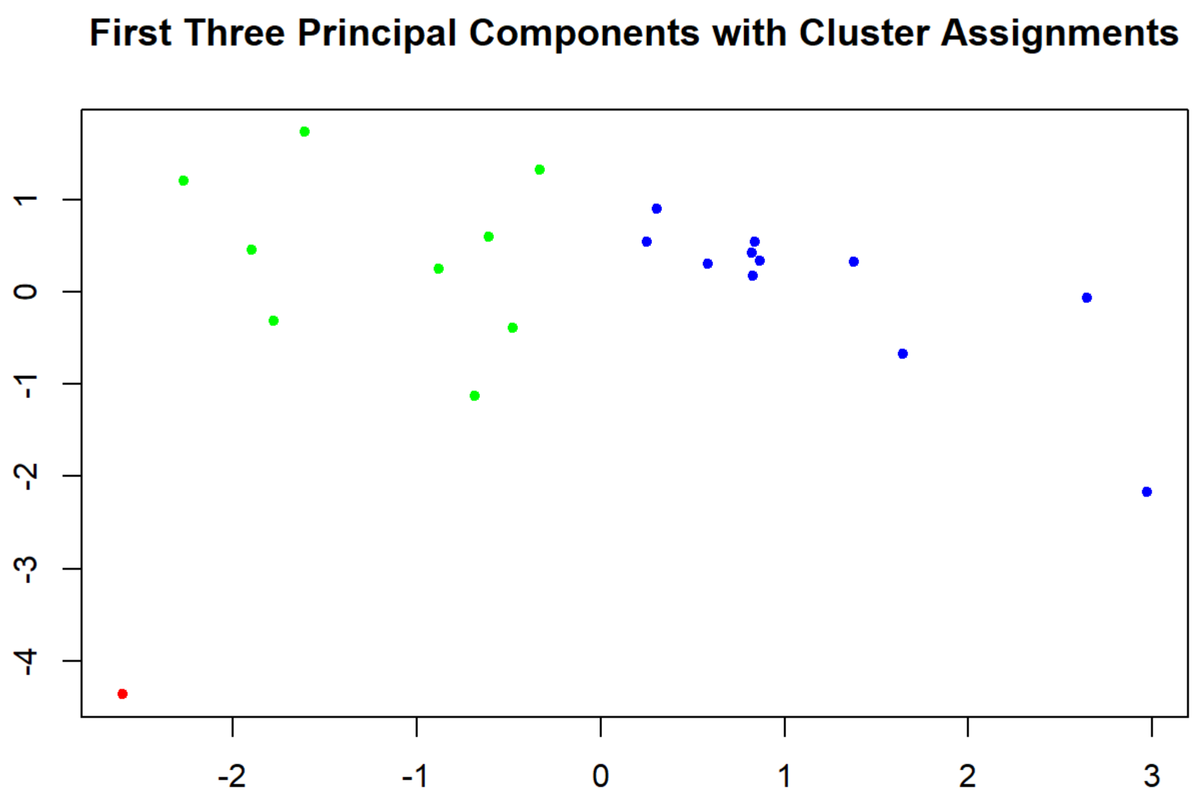
```
library(factoextra)
library(ggplot2)
library(readxl)
library(factoextra)
library(ggfortify)
library(ggrepel)
library(stats)

# Load required library
library(readxl)

# Define the file path
file_path <- "C:/Users/sakad/Downloads/social_media_cleaned.xlsx"

# Read the Excel file
body <- read_excel(file_path)

# Standardize the dataset excluding the first variable which is categorical
data.scaled <- scale(x = body[, -1], center = TRUE, scale = TRUE)

# Assign the standardized data to 'data'
data <- data.scaled

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

```
##       Instagram   LinkedIn   SnapChat     Twitter Whatsapp/Wechat    youtube
## [1,] -0.6888867  0.1519097 -0.2810325  3.42451853      -1.3480276 -0.2719612
## [2,]  0.5146906  0.6366870  0.4933528 -0.44951033      -0.5576781  0.7352180
## [3,] -0.6130708  1.3638528 -0.4157082 -0.44951033       0.8450887 -0.6460563
## [4,] -0.1534369  0.6838181 -0.1944552 -0.44951033      -0.2804351 -0.5597267
## [5,]  1.7277449 -1.2283589 -0.5696232  0.06702685      -0.8514729  0.3035697
## [6,] -1.0205812  1.3638528 -0.4349475 -0.44951033       1.3830228  2.3179281
##             OTT     Reddit How you felt the entire week?
## [1,]  3.4382238  1.2405563                    -0.5741693
## [2,] -0.6686547 -0.3292004                    -0.5741693
## [3,] -0.1021887 -0.3292004                     0.7655590
## [4,] -0.1021887 -0.3292004                     0.7655590
## [5,] -0.1021887  0.2987023                    -0.5741693
## [6,]  0.1810443 -0.3292004                     2.1052873
```

Q: What is the proportion of variance explained by the first three principal components in the PCA analysis, and how does it impact the effectiveness of the subsequent K-means clustering algorithm in identifying distinct clusters within the data?

```
# Perform PCA
pc <- prcomp(data.scaled)

# Extract the first three principal components
```

```
pc_first_three <- pc$x[, 1:3]

# Perform K-means clustering on the first three principal components
set.seed(123)  # For reproducibility
k <- 3  # Number of clusters
km_clusters <- kmeans(pc_first_three, centers = k)

# Define colors for each cluster
cluster_colors <- c("red", "blue", "green")

# Plot the first three principal components with cluster assignments
plot(pc_first_three, col = cluster_colors[km_clusters$cluster],
     main = "First Three Principal Components with Cluster Assignments",
     xlab = "", ylab = "", pch = 20)
```

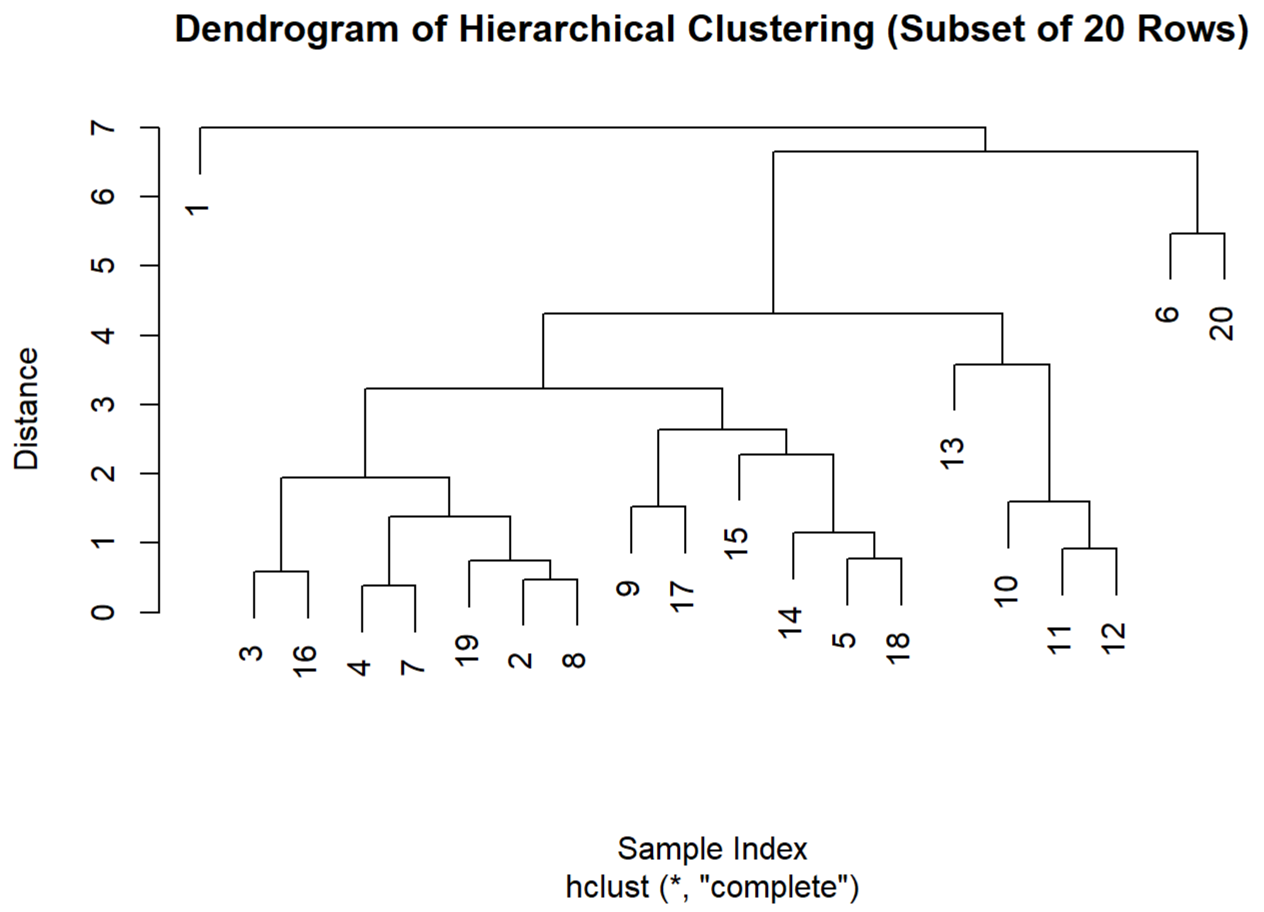


This above code first performs Principal Component Analysis (PCA) on scaled data, reducing its dimensionality. Then, it extracts the first three principal components. Next, it applies K-means clustering to these components, dividing data into three clusters. Finally, it plots the first three principal components with color-coded cluster assignments for visualization and analysis.

Q: Can hierarchical clustering based on the first three principal components effectively reveal underlying patterns or groupings within the dataset, and how do the relationships between samples, as depicted in the dendrogram, reflect the distances in the reduced-dimensional space?

```
# Perform PCA
pc <- prcomp(data.scaled)
```

```r
# Extract the first three principal components
pc_first_three <- pc$x[, 1:3]

# Take a subset of 20 rows
data_subset <- data[1:20, ]

# Perform PCA
pca_result <- prcomp(data_subset)

# Extract the first three principal components
pc_first_three <- pca_result$x[, 1:3]

# Perform hierarchical clustering on the first three principal components
hc <- hclust(dist(pc_first_three))

# Plot the dendrogram
plot(hc, main = "Dendrogram of Hierarchical Clustering (Subset of 20 Rows)",
     xlab = "Sample Index", ylab = "Distance", sub = NULL)
```

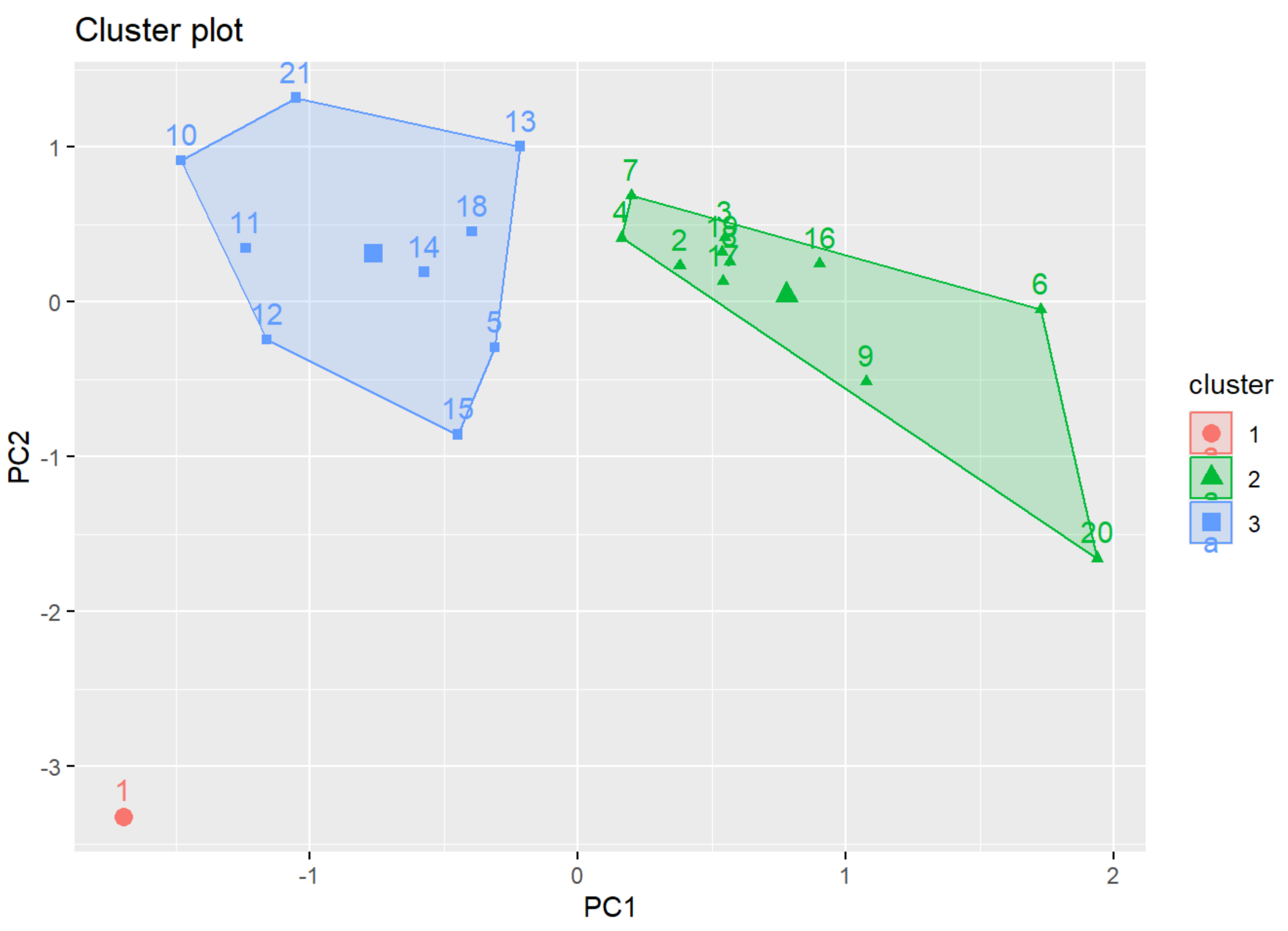


The plot shows the first three principal components, performs hierarchical clustering on them, and plots a dendrogram showing the relationships between the samples based on their distances in the reduced-dimensional space.

Q: What is the degree of separation or distinctiveness among the identified clusters when analyzing the distribution of data points in the two-dimensional space formed by the first two Principal Components?

```r
# Visualize cluster and membership using first two Principal Components
```

```
fviz_cluster(list(data = pc$x[, 1:2], cluster = km_clusters$cluster))
```



This plot visualizes clustering results by plotting data points in a two-dimensional space using the first two Principal Components. Each point is colored according to its assigned cluster, showing the grouping pattern identified by the clustering algorithm. It helps understand how data points are grouped based on their features.

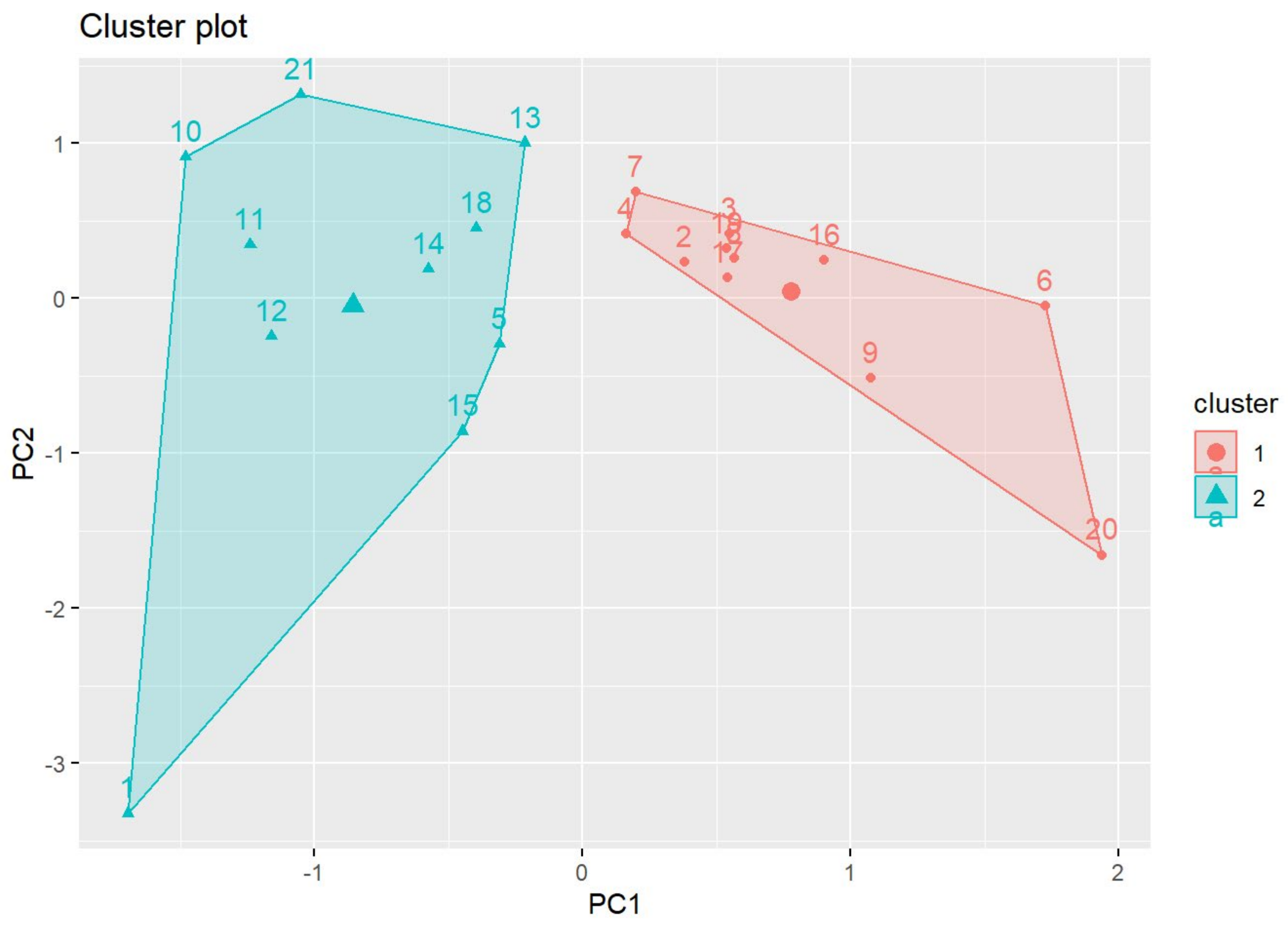
```
# Non-hierarchical clustering (k-means)
num_clusters <- 2
kmeans_model <- kmeans(data, centers = num_clusters)

# Membership for each cluster
table(kmeans_model$cluster)
```

```
##
##  1  2
## 11 10
```

This represents clustering using the k-means algorithm, dividing data into two clusters. It initializes cluster centers randomly, assigning each data point to the nearest cluster. The table function counts the number of data points assigned to each cluster, providing insight into cluster membership and distribution.

Q: What is the distribution of cluster sizes obtained through k-means clustering when applied to the dataset using the first two principal components, and how does it compare to the distribution of cluster sizes obtained from clustering in the original feature space?

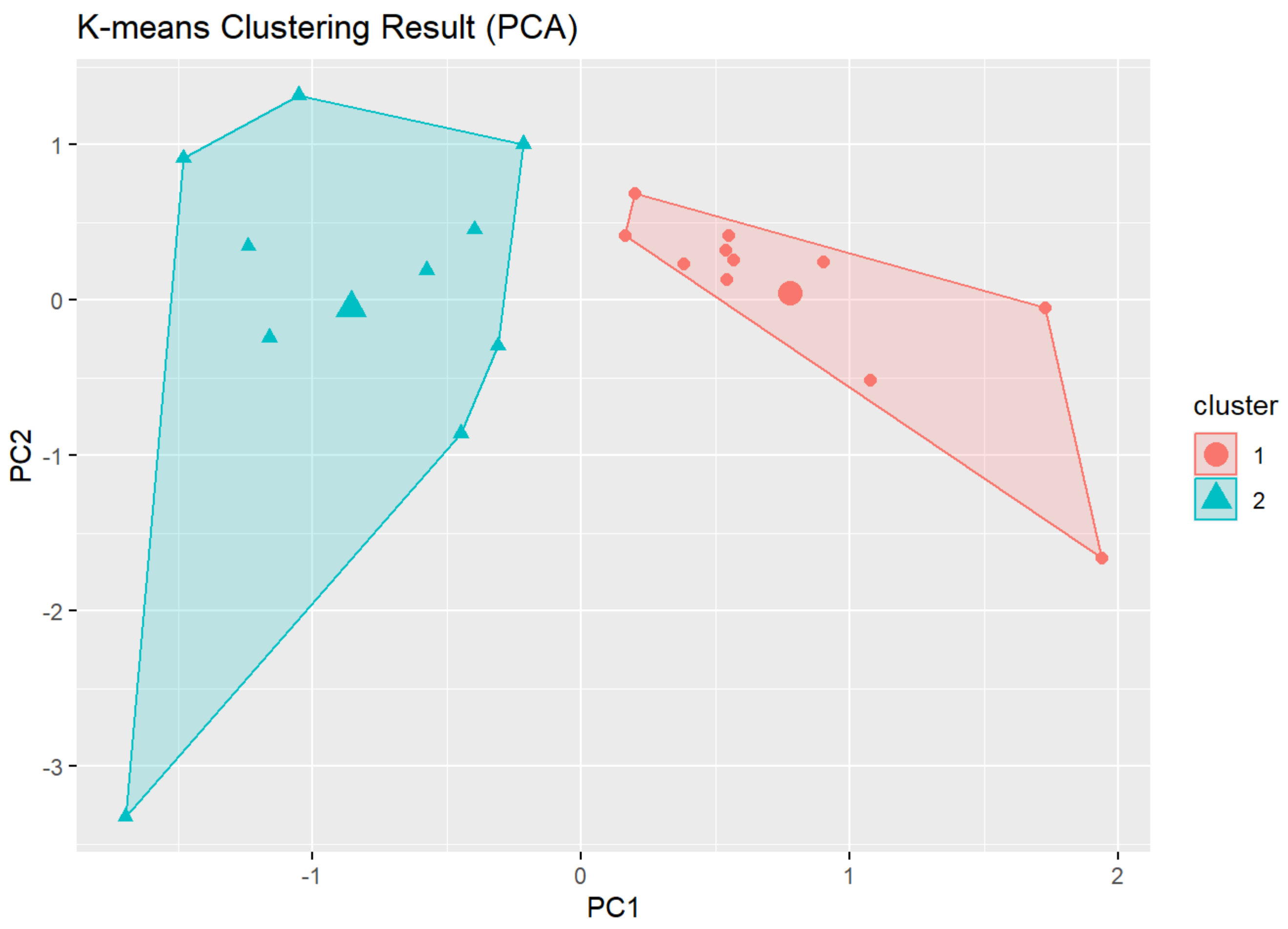
```
# Visualize cluster and membership using first two Principal Components
fviz_cluster(list(data = pc$x[, 1:2], cluster = kmeans_model$cluster))
```



This plot visualizes clusters and their memberships using the first two principal components. It extracts these components from the data, then assigns each data point to a cluster using k-means clustering. Finally, it creates a visual representation showing how the data points are grouped based on their similarities in the first two principal components.

Q: What is the relationship between the clustering results obtained through k-means algorithm and the underlying structure of the data as revealed by Principal Component Analysis (PCA)?

```
# Visualize cluster and membership using first two Principal Components for k-means
pca_result <- prcomp(data, scale = TRUE)
fviz_cluster(kmeans_model, data = pca_result$x[, 1:2], geom = "point",
             pointsize = 2, fill = "white", main = "K-means Clustering Result (PCA)")
```



This shows visualization of the clusters and their memberships using the first two Principal Components (PCs) obtained from the PCA (Principal Component Analysis) of the numerical data. First, it computes the PCA result for the numerical data and scales it. Then, it uses the `fviz_cluster` function to plot the clusters obtained from the k-means algorithm (`kmeans_model`). It represents each data point as a point on the plot, with the size set to 2 and colored white. The plot is titled “K-means Clustering Result (PCA)”. This visualization helps to understand how the data points are grouped into clusters based on their similarities, as revealed by the PCA analysis.

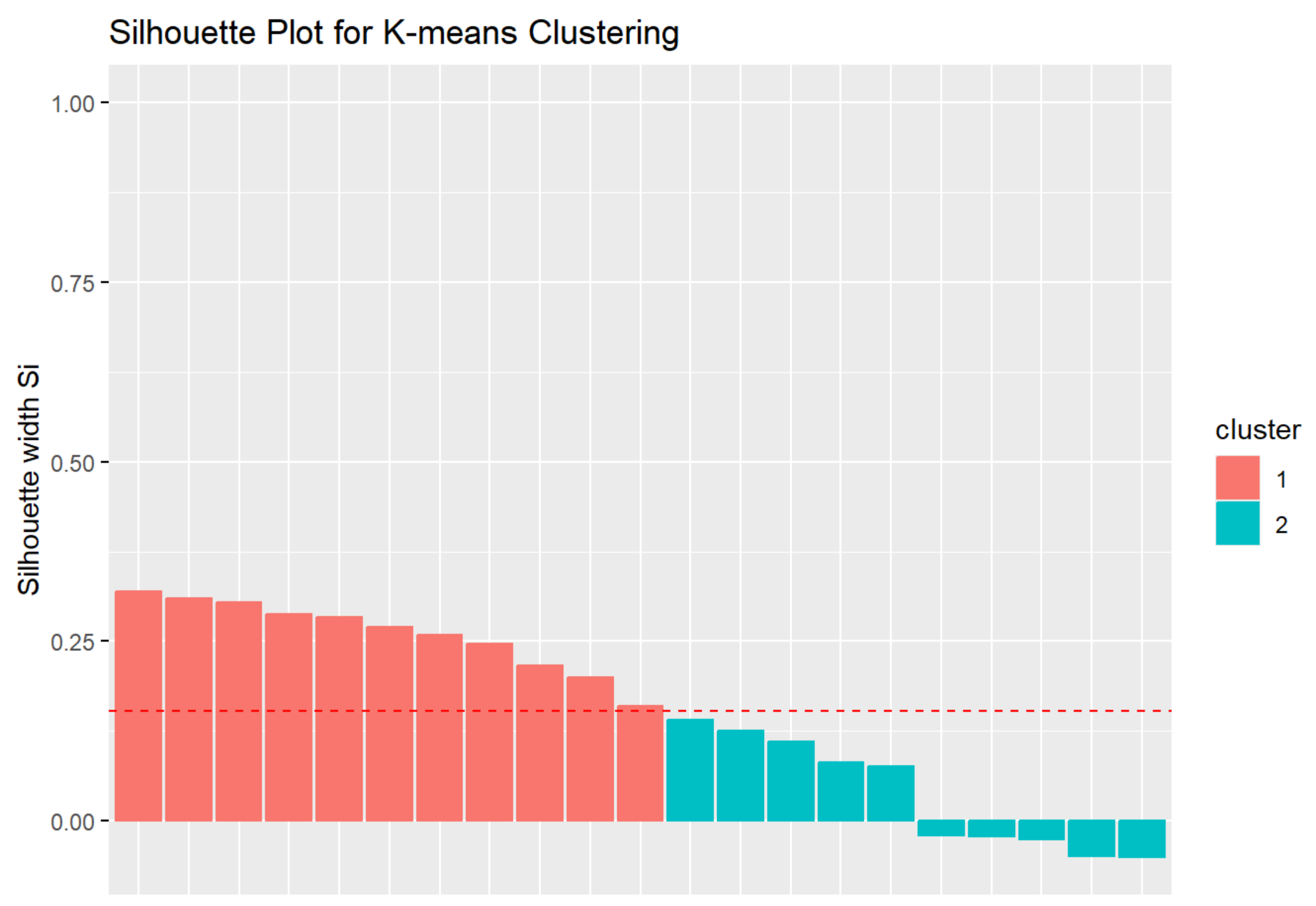
Q:What is the relationship between the number of clusters (k) and the average silhouette width in k-means clustering, and how does this relationship inform the determination of the optimal number of clusters for a given dataset?

```
library(factoextra)
library(cluster)
```

```
## Warning: package 'cluster' was built under R version 4.3.3
```

```
# Calculate silhouette information for k-means clustering
sil <- silhouette(kmeans_model$cluster, dist(data))

# Visualize the silhouette plot for k-means clustering
fviz_silhouette(sil, main = "Silhouette Plot for K-means Clustering")
```

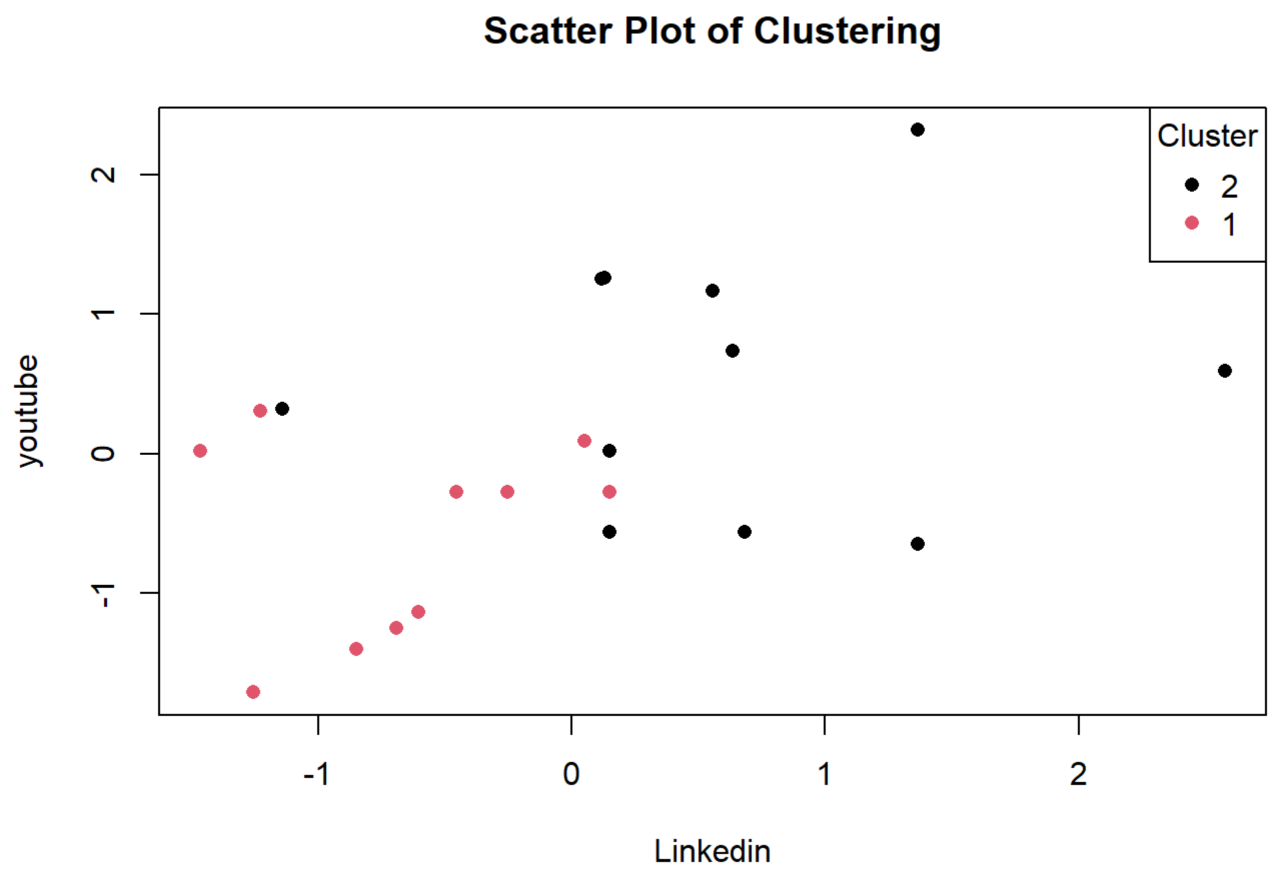
```
##   cluster size ave.sil.width
## 1       1   11          0.26
## 2       2   10          0.04
```



This plot calculates and visualizes the silhouette information for k-means clustering. Silhouette analysis helps evaluate the quality of clustering by measuring how similar an object is to its own cluster compared to other clusters. A higher silhouette width indicates better separation of clusters, while negative values suggest that points might be assigned to the wrong clusters. This plot helps in determining the optimal number of clusters for k-means clustering and assessing the overall clustering performance.

Q: Is there a significant difference in the clustering patterns based on Linkedin and Youtube among different groups represented by distinct colors on the plot?

```
# Create a data frame with cluster membership
data_clustered <- data.frame(data, Cluster = kmeans_model$cluster)  # Ensure conversion to dat
a frame

# Scatter plot of data points colored by cluster membership
plot(data_clustered$LinkedIn, data_clustered$youtube,
     col = data_clustered$Cluster, pch = 16,
     xlab = "Linkedin", ylab = "youtube",
     main = "Scatter Plot of Clustering")
legend("topright", legend = unique(data_clustered$Cluster),
       col = 1:max(data_clustered$Cluster), pch = 16, title = "Cluster")
```



Overall, this plot visualizes clusters in the data, helping us understand how data points group together based on the Linkedin and Youtube, with each group represented by a different color on the plot.

```
sm <- read.csv("D:/Multivariate Analysis/Social media regression/social_media_cleaned.csv")
str(sm)
```

```
## 'data.frame':    21 obs. of  10 variables:
##  $ character                  : chr  "masinl" "peace" "Patty" "Bunny" ...
##  $ Instagram                  : num  3.5 7.73 3.77 5.38 12 2.33 5.37 7 8.65 0.17 ...
##  $ LinkedIn                   : num  4 5.2 7 5.32 0.58 7 4 4 10 0 ...
##  $ SnapChat                   : num  1 3.68 0.53 1.3 0 0.47 0 3 3.83 0 ...
##  $ Twitter                    : num  5 0 0 0 0.67 0 0 0 0 0 ...
##  $ Whatsapp.Wechat            : num  1 4.18 9.83 5.3 3 12 6 10 6.15 1 ...
##  $ youtube                    : num  2.5 4.25 1.85 2 3.5 7 3 2 4 3 ...
##  $ OTT                        : num  14.5 0 2 2 2 3 0 3 3 0 ...
##  $ Reddit                     : num  2.5 0 0 0 1 0 0 0 0 0 ...
##  $ How.you.felt.the.entire.week.: int  3 3 4 4 3 5 4 4 3 2 ...
```

#Multiple Regression

```
sm2 <- sm[, 2:10]
sm2
```

```
##    Instagram LinkedIn SnapChat Twitter Whatsapp.Wechat youtube   OTT Reddit
## 1       3.50     4.00     1.00    5.00             1.00    2.50 14.50   2.50
## 2       7.73     5.20     3.68    0.00             4.18    4.25  0.00   0.00
## 3       3.77     7.00     0.53    0.00             9.83    1.85  2.00   0.00
## 4       5.38     5.32     1.30    0.00             5.30    2.00  2.00   0.00
## 5      12.00     0.58     0.00    0.67             3.00    3.50  2.00   1.00
## 6       2.33     7.00     0.47    0.00            12.00    7.00  3.00   0.00
## 7       5.37     4.00     0.00    0.00             6.00    3.00  0.00   0.00
## 8       7.00     4.00     3.00    0.00            10.00    2.00  3.00   0.00
## 9       8.65    10.00     3.83    0.00             6.15    4.00  3.00   0.00
## 10      0.17     0.00     0.00    0.00             1.00    3.00  0.00   0.00
## 11      4.58     2.13     0.42    2.42             1.00    1.00  0.00   0.00
## 12      3.33     2.50     0.33    2.83             3.67    2.50  1.50   0.00
## 13      5.30     1.52    14.87    0.00             4.02    0.54  0.51   0.01
## 14      6.00     3.00     1.00    1.00             3.42    2.50  1.00   0.00
## 15      4.65     3.75     1.42    0.00             6.50    3.13  1.92   7.00
## 16      7.00     5.00     0.42    0.00             5.00    5.00  1.00   0.50
## 17      9.80     0.80     0.00    0.27            13.35    3.53  1.68   0.00
## 18      6.80     1.92     1.87    0.00             6.95    0.80  2.47   0.00
## 19      5.67     3.92     0.00    0.00             8.92    5.15  0.00   0.00
## 20     15.02     3.95     7.32    0.00            15.35    5.17 10.00   0.00
## 21      0.33     0.52     0.00    0.00             8.38    0.00  0.00   0.00
##    How.you.felt.the.entire.week.
## 1                              3
## 2                              3
## 3                              4
## 4                              4
## 5                              3
## 6                              5
## 7                              4
## 8                              4
## 9                              3
## 10                             2
## 11                             4
## 12                             3
## 13                             4
## 14                             3
## 15                             3
## 16                             5
## 17                             3
## 18                             3
## 19                             3
## 20                             3
## 21                             3
```

Q1.Model Development

```
# Performing multiple regression on the dataset
fit <- lm(sm2$How.you.felt.the.entire.week. ~ Instagram + LinkedIn + SnapChat + Twitter + What
sapp.Wechat + youtube + OTT + Reddit, data = sm2)

# Show the results
summary(fit)
```

```
##
## Call:
## lm(formula = sm2$How.you.felt.the.entire.week. ~ Instagram +
##     LinkedIn + SnapChat + Twitter + Whatsapp.Wechat + youtube +
##     OTT + Reddit, data = sm2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.14185 -0.32576 -0.04567  0.48445  1.49290
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2.72737    0.67735   4.027  0.00168 **
## Instagram       -0.03076    0.06415  -0.480  0.64020
## LinkedIn         0.13054    0.08634   1.512  0.15646
## SnapChat         0.04201    0.06245   0.673  0.51384
## Twitter          0.16906    0.27175   0.622  0.54551
## Whatsapp.Wechat  0.05286    0.06754   0.783  0.44906
## youtube          0.03123    0.13553   0.230  0.82164
## OTT             -0.07857    0.09257  -0.849  0.41262
## Reddit          -0.03426    0.12296  -0.279  0.78526
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8173 on 12 degrees of freedom
## Multiple R-squared:  0.2807, Adjusted R-squared:  -0.1988
## F-statistic: 0.5853 on 8 and 12 DF,  p-value: 0.7725
```

The linear regression model predicts “How you felt the entire week?” based on social media usage variables including Instagram, LinkedIn, SnapChat, Twitter, Whatsapp/Wechat, youtube, OTT, and Reddit. Coefficients represent the estimated effect of each predictor variable on the response variable. However, most coefficients are not statistically significant at the 0.05 level, indicating weak evidence of association. The model explains approximately 28.07% of the variability in “How you felt the entire week?” as indicated by the multiple R-squared value. Nonetheless, the adjusted R-squared value is negative, suggesting potential issues with model fit or overfitting. The overall model is not statistically significant according to the F-statistic with a p-value of 0.7725.

```
coefficients(fit)
```

```
##     (Intercept)       Instagram        LinkedIn        SnapChat         Twitter
##      2.72737269     -0.03076010      0.13053615      0.04201314      0.16905853
## Whatsapp.Wechat         youtube             OTT          Reddit
##      0.05285596      0.03122902     -0.07856775     -0.03426142
```

The coefficients represent the estimated impact of each social media platform on “How you felt the entire week?” The positive coefficients for LinkedIn, Twitter, Whatsapp/Wechat, and YouTube suggest a potential positive association with mood, while negative coefficients for Instagram, OTT, and Reddit imply a negative association. The intercept represents the estimated mood score when all predictors are zero.

```
fitted(fit)
```

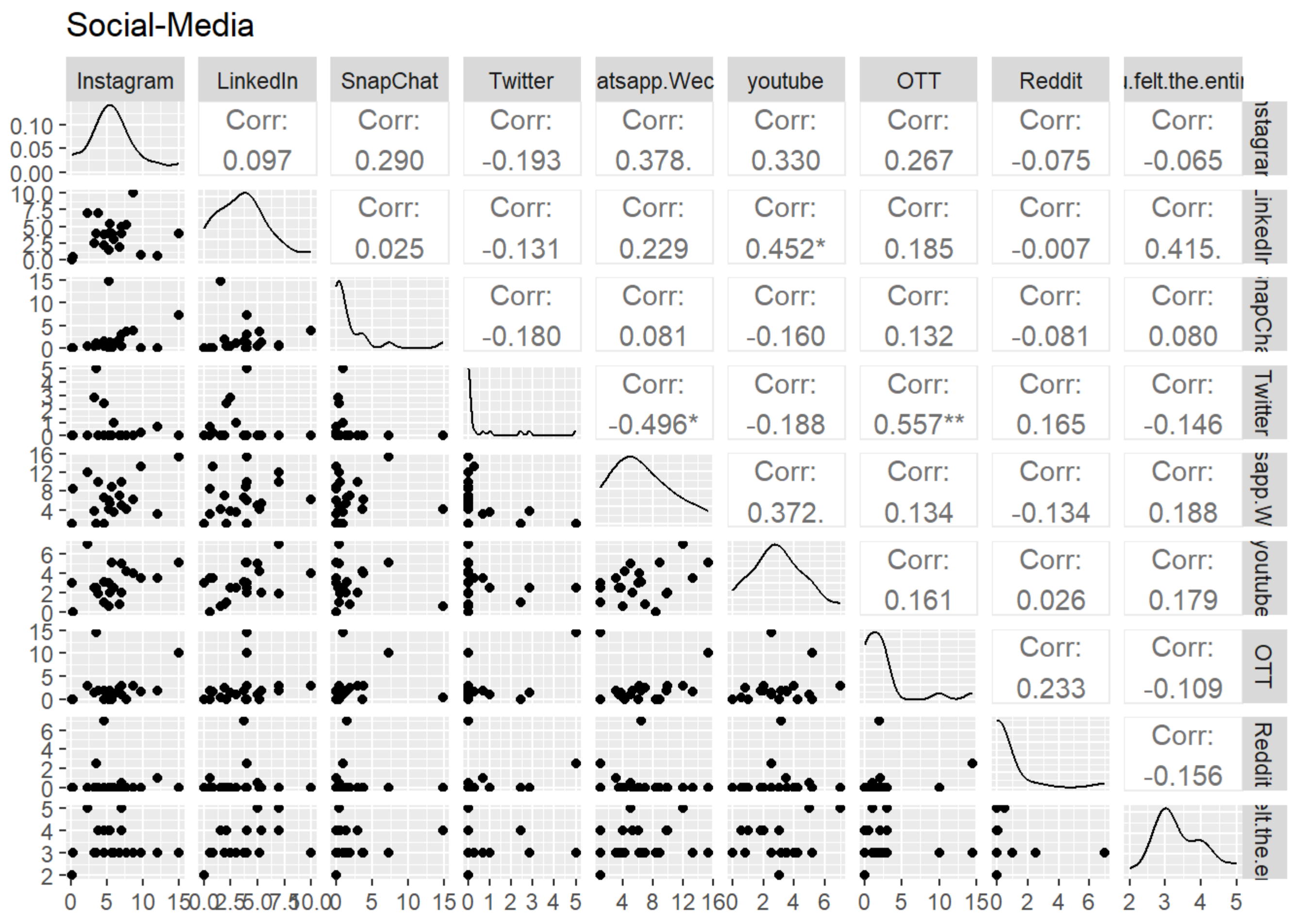
```
##         1         2         3         4         5         6         7         8
## 2.935205 3.676655 3.967639 3.496412 2.623704 4.206372 3.495158 3.515550
##        9       10       11       12       13       14       15       16
## 4.141847 2.868686 3.375386 3.597784 3.576427 3.325764 3.184138 3.507105
##       17       18       19       20       21
## 3.259870 3.045668 3.696969 3.275626 3.228034
```

Residual Analysis

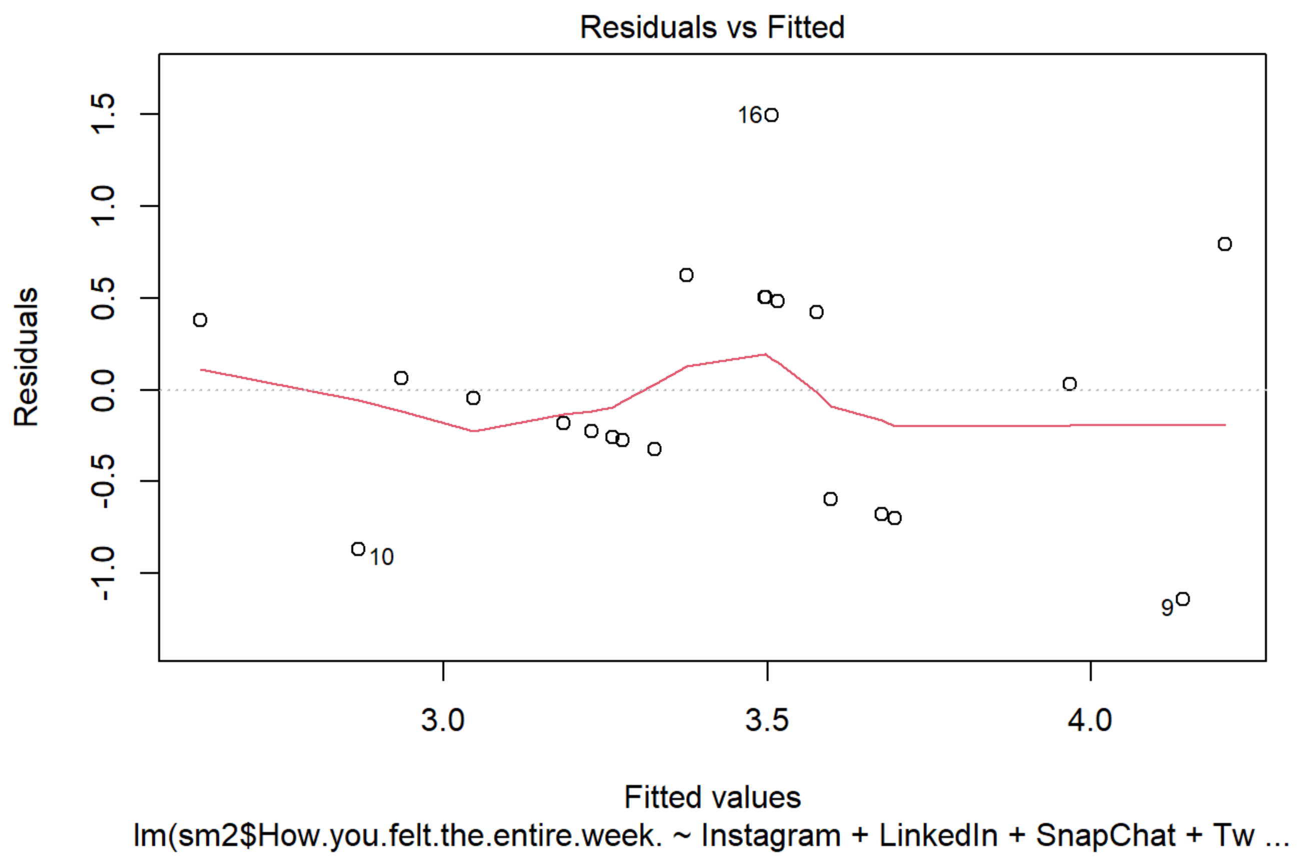
```r
library(GGally)
```

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

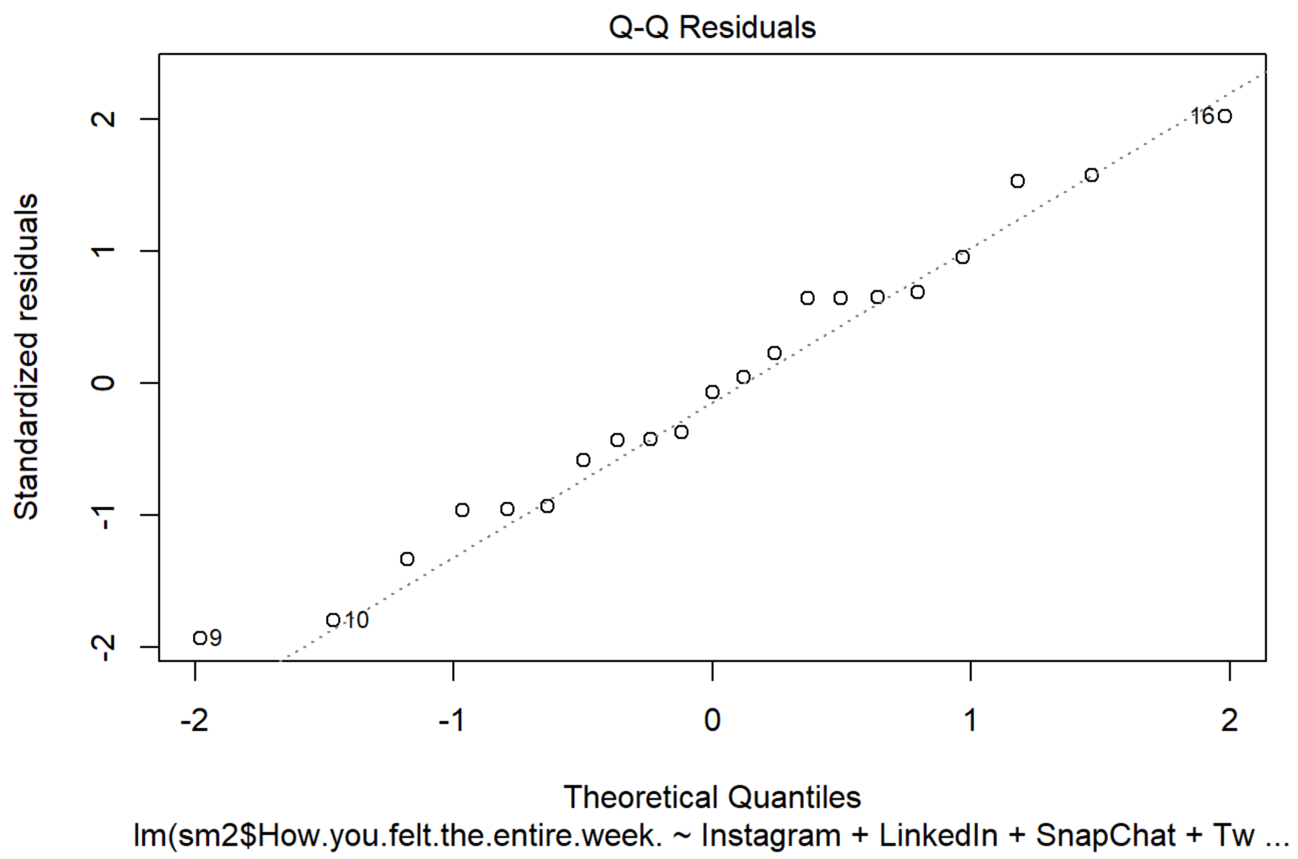
```r
ggpairs(data=sm2, title="Social-Media")
```



```r
plot(fit, which=1) # Residuals vs Fitted
```



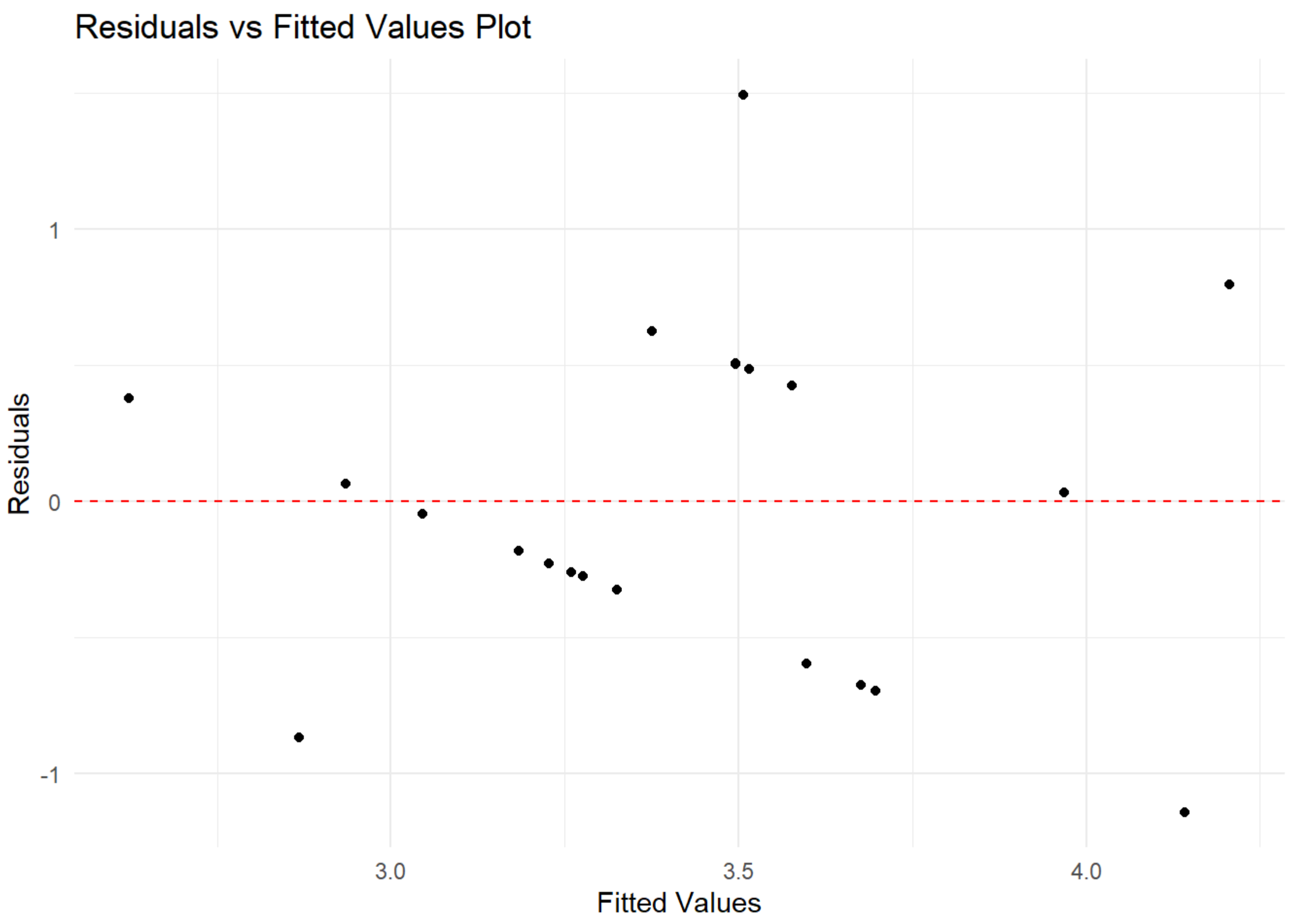
```
plot(fit, which=2) # Normal Q-Q plot
```



This graph is a Q-Q (Quantile-Quantile) plot, which is a diagnostic tool used to assess the normality of a dataset's distribution. The x-axis represents the theoretical quantiles, while the y-axis shows the standardized residuals.In an ideal normal distribution, the points would fall along a straight diagonal line. However, in this plot, the points show some deviation from the diagonal, particularly in the tails. This suggests the data may not fully conform to a normal distribution and could indicate the presence of outliers or other non-normal characteristics. The Q-Q plot provides a visual way to evaluate the assumption of normality, which is an important consideration in many statistical analyses. Identifying departures from normality can inform the choice of appropriate modeling techniques or the need for further data exploration and transformation.

```
residuals <- residuals(fit)
```

```
#Plot residuals against fitted values to check for homoscedasticity
plot_resid_fitted <- ggplot() +
  geom_point(aes(x = fitted(fit), y = residuals)) +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(x = "Fitted Values", y = "Residuals",
       title = "Residuals vs Fitted Values Plot") +
  theme_minimal()
print(plot_resid_fitted)
```



This graph is a Residuals vs Fitted Values plot, which is a common diagnostic tool used in regression analysis. The key points about this plot are:

1. The x-axis represents the fitted values from the regression model, while the y-axis shows the residuals (the differences between the observed and predicted values).
2. The plot displays the relationship between the fitted values and the residuals. Ideally, the residuals should be randomly scattered around the horizontal line at y = 0, with no clear patterns or trends.
3. In this plot, the residuals appear to be randomly distributed around the zero line, suggesting the regression model is adequately capturing the relationships in the data and the assumptions of the model are met.

This plot allows the analyst to visually assess the linearity, homoscedasticity, and independence of the residuals, which are important assumptions for the validity of the regression analysis.

The residual vs. fitted plot is a diagnostic tool used to assess the suitability of a regression model and its adherence to key assumptions. By examining the dispersion of points around the zero line, one can ascertain whether the model accurately captures the data's underlying patterns or if adjustments are needed. If a discernible pattern emerges, it suggests inadequacies in the model's fit.

Q2 Prediction

```
predict(fit, newdata = data.frame(Instagram = 8, LinkedIn = 5, SnapChat = 4, Twitter = 4,
                                    `Whatsapp/Wechat` = 4, youtube = 8, OTT = 3, Reddit = 4))
```

```
##        1
## 4.066766
```

Q3: Model Accuracy

```
#Make predictions using the model
predicted <- predict(fit, newdata = sm2)
```

```
#Calculating RMSE by taking the square root of the mean of the squared differences between the
 actual values and the predicted values (predicted)
rmse <- sqrt(mean((sm2$How.you.felt.the.entire.week. - predicted)^2))
rmse
```

```
## [1] 0.6177979
```

RMSE is a measure of the differences between values predicted by a model and the observed values. In this case, an RMSE value of 0.6177979 indicates that, on average, the model's predictions deviate from the observed values by approximately 0.6177979 units. A lower RMSE value indicates better performance of the model in terms of prediction accuracy.Low RMSE0.617797 between 0 and 1 indicates that the models predictions are quite accurate, with small deviations from the actual values.

Visualizations

```
library(car)
```

```
## Warning: package 'car' was built under R version 4.3.3
```

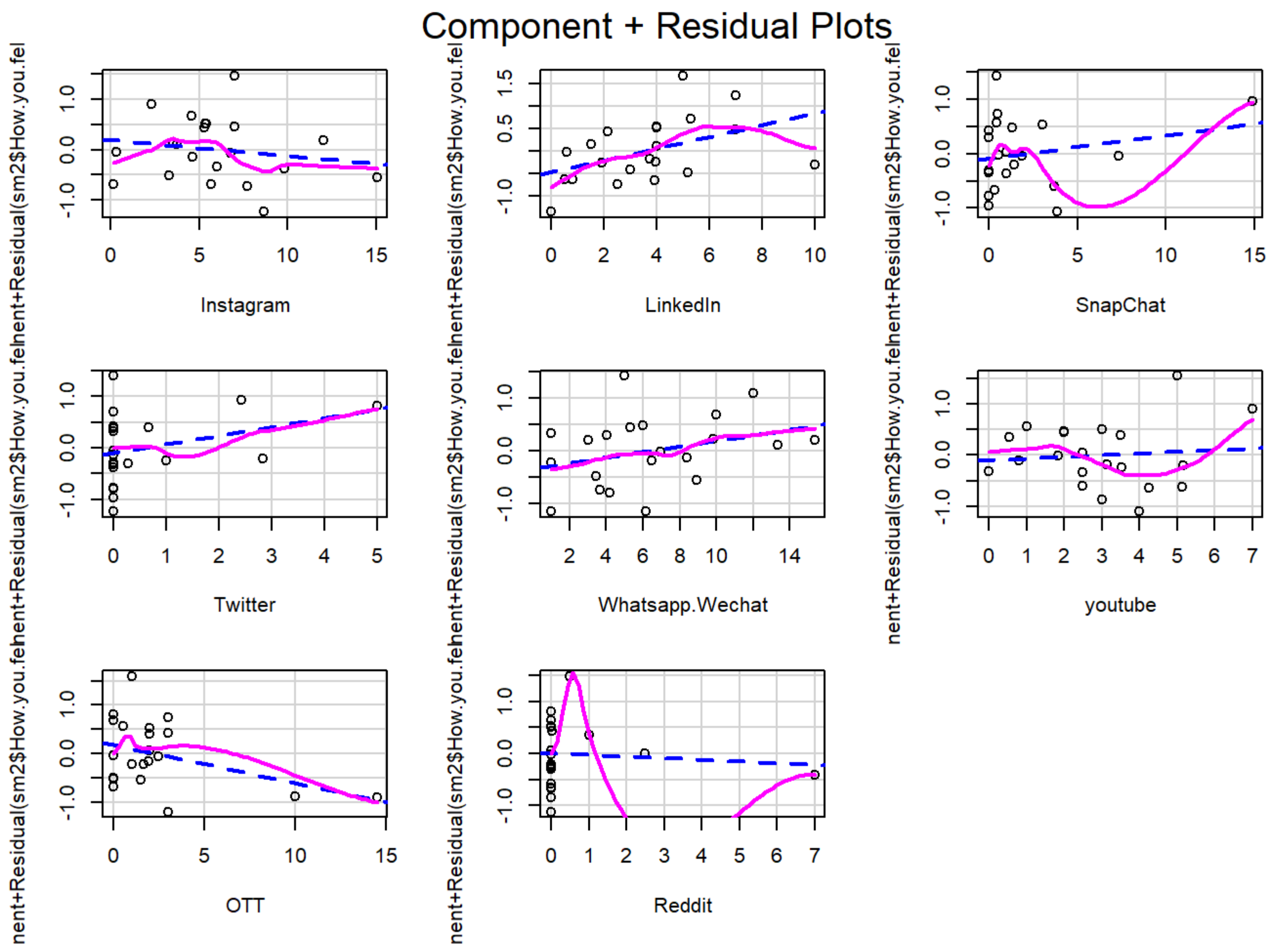
```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:psych':
##
##     logit
```

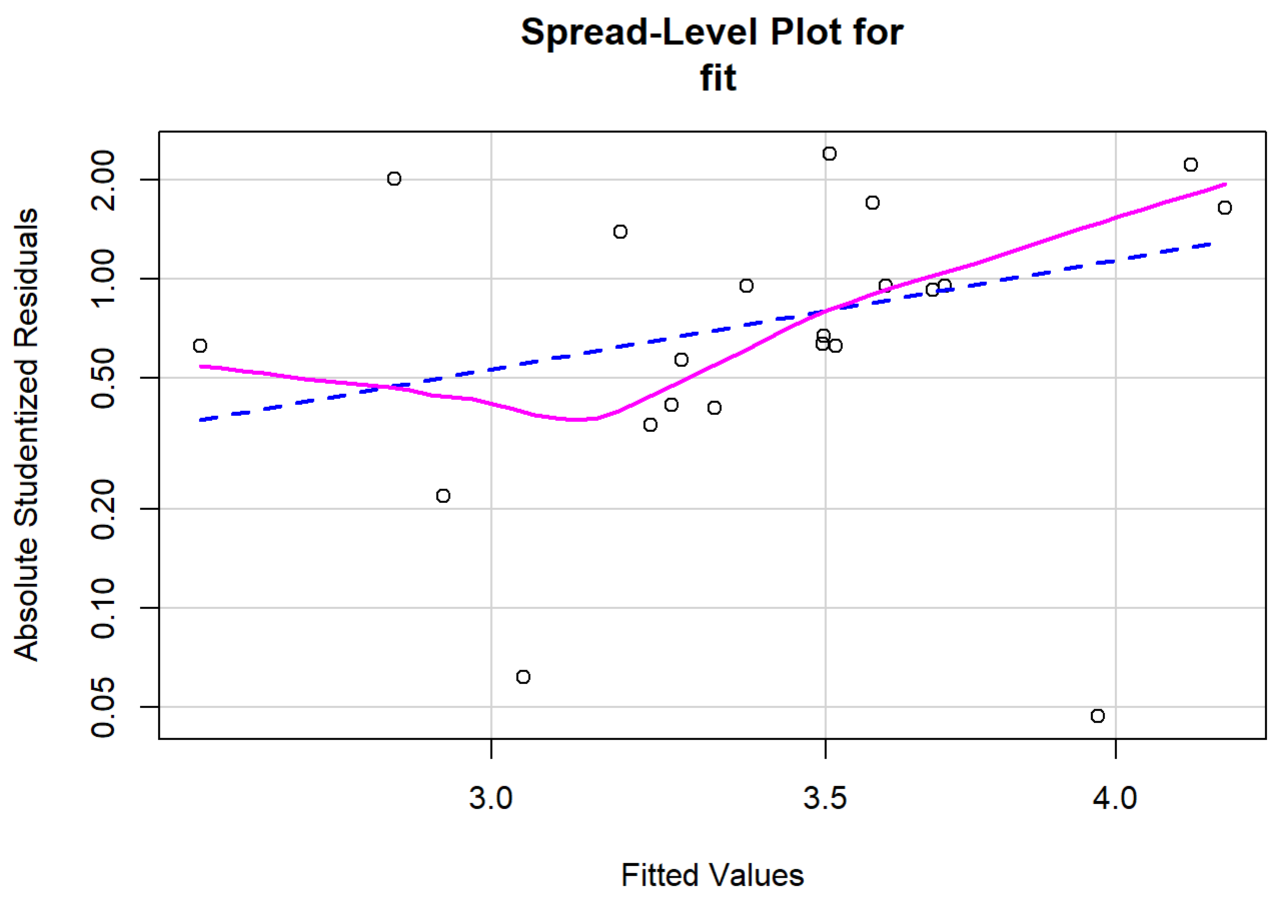
```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
#Nonlinearity
# component + residual plot
crPlots(fit)
```



These are component plus residual plots (also known as partial regression plots) for different social media platforms and services like Instagram, LinkedIn, Snapchat, Twitter, WhatsApp/WeChat, YouTube, OTT, and Reddit. Each graph displays the relationship between the response variable and a predictor variable, controlling for the effects of other predictors in the model. The x-axis shows the component plus residual values for a predictor, while the y-axis shows the residuals from the model. The plots help to identify non-linear relationships, outliers, and the influence of individual data points on the regression fit, with the pink curve indicating the trend.

```
# plot studentized residuals vs. fitted values
library(car)
spreadLevelPlot(fit)
```



```
##
## Suggested power transformation:  -1.649885
```

This graph is a Spread-Level Plot for model fit, typically used to assess variance homogeneity in residuals from regression analysis. The x-axis represents the fitted values from a model, while the y-axis shows the absolute standardized residuals. The plot reveals patterns in the spread of residuals across the range of fitted values. If residuals are evenly spread, it suggests homoscedasticity. Here, a curve (pink solid line) is fitted through the data points, and a reference dashed line is also provided. The upward trend of the curve suggests increasing variability of residuals as fitted values rise, indicating potential heteroscedasticity.

#Logistic_regression

```
library(ggplot2)
library(cowplot)
#library(regclass)
library(caret)
```

```
library(e1071)
library(pROC)
```

```
data <- read.csv("D:/Multivariate Analysis/Assignment-8/social_media_final.csv", row.names=1)
data
```

```
##         Instagram_Usage LinkedIn_Usage Snapchat_Usage Twitter_Usage
## masinl             3.50           4.00           1.00          5.00
## peace              7.44           5.12           3.41         12.00
## Patty              3.46           7.00          12.32          0.00
## Bunny              5.23           5.19           1.18          0.00
## tl868              0.00          12.35           0.00         12.40
## drphy              2.20           7.00          12.28          0.00
## trave              5.22           4.00           0.00          0.00
## 19!@s              7.00           4.00           3.00          0.00
## yh2020             8.39          10.00           3.50          0.00
## ds2134            12.10          12.00           0.00          0.00
## ki567              4.35           2.08          12.25          2.25
## ak2001             3.20           2.30          12.20          2.50
## Harvey             5.30           1.52           0.62          0.00
## hahah              6.00           3.00           1.00          1.00
## AKIRA              4.39           3.45           1.25          0.00
## vp1234             7.00           5.00          12.25         12.16
## sss32              9.48          12.48           0.00          0.00
## MVA37@S            6.48           1.55           1.52          0.00
## 2134               5.40           3.55           0.00          0.00
## 15801              3.01           3.57           7.19          0.00
## Baiqi             12.20          12.31           0.00          0.00
##         Whatsapp_Usage Youtube_Usage   OTT Reddit Trouble_falling_asleep
## masinl            1.00          2.50 14.50   2.50                      0
## peace             4.11          4.15 12.00  12.00                      1
## Patty             9.50          1.51  2.00   0.00                      0
## Bunny             5.18          2.00  2.00   0.00                      0
## tl868             3.00          3.30  2.00   1.00                      1
## drphy            12.00          7.00  3.00   0.00                      0
## trave             6.00          3.00  0.00   0.00                      0
## 19!@s            10.00          2.00  3.00   0.00                      1
## yh2020            6.09          4.00  3.00   0.00                      0
## ds2134            1.00          3.00  0.00   0.00                      0
## ki567             1.00          1.00 12.00   0.00                      1
## ak2001            3.40          2.30  1.30   0.00                      0
## Harvey            4.02          0.54  0.51   0.01                      1
## hahah             3.25          2.30  1.00   0.00                      0
## AKIRA             6.30          3.08  1.55   7.00                      0
## vp1234            5.00          5.00  1.00  12.30                      1
## sss32             1.21          3.32  1.41   0.00                      0
## MVA37@S           6.57         12.48  2.28   0.00                      0
## 2134              8.55          5.09  0.00   0.00                      0
## 15801             3.21          5.10 10.00   0.00                      1
## Baiqi             8.23          0.00  0.00   0.00                      0
##         Mood.Productivity Tired.waking.up.in.morning
## masinl                  1                          0
## peace                   1                          0
## Patty                   1                          0
## Bunny                   1                          0
## tl868                   1                          1
## drphy                   1                          0
## trave                   1                          1
## 19!@s                   1                          1
## yh2020                  1                          0
## ds2134                  0                           0
## ki567                   1                           0
## ak2001                  1                           0
## Harvey                  1                           1
## hahah                   1                           0
## AKIRA                   1                           0
## vp1234                  1                           1
## sss32                   1                           0
## MVA37@S                 1                           1
## 2134                    1                           0
## 15801                   1                           0
## Baiqi                   1                           1
```

```
head(data)
```

```
##        Instagram_Usage LinkedIn_Usage Snapchat_Usage Twitter_Usage
## masinl            3.50           4.00           1.00           5.0
## peace             7.44           5.12           3.41          12.0
## Patty             3.46           7.00          12.32           0.0
## Bunny             5.23           5.19           1.18           0.0
## tl868             0.00          12.35           0.00          12.4
## drphy             2.20           7.00          12.28           0.0
##        Whatsapp_Usage Youtube_Usage  OTT Reddit Trouble_falling_asleep
## masinl           1.00          2.50 14.5    2.5                      0
## peace            4.11          4.15 12.0   12.0                      1
## Patty            9.50          1.51  2.0    0.0                      0
## Bunny            5.18          2.00  2.0    0.0                      0
## tl868            3.00          3.30  2.0    1.0                      1
## drphy           12.00          7.00  3.0    0.0                      0
##        Mood.Productivity Tired.waking.up.in.morning
## masinl                 1                          0
## peace                  1                          0
## Patty                  1                          0
## Bunny                  1                          0
## tl868                  1                          1
## drphy                  1                          0
```

```
xtabs(~ Tired.waking.up.in.morning + Instagram_Usage, data=data)
```

```
##                           Instagram_Usage
## Tired.waking.up.in.morning 0 2.2 3.01 3.2 3.46 3.5 4.35 4.39 5.22 5.23 5.3 5.4
##                          0 0   1    1   1    1   1    1    1    0    1   0   1
##                          1 1   0    0   0    0   0    0    0    1    0   1   0
##                           Instagram_Usage
## Tired.waking.up.in.morning 6 6.48 7 7.44 8.39 9.48 12.1 12.2
##                          0 1    0 0    1    1    1    1    0
##                          1 0    1 2    0    0    0    0    1
```

```
xtabs(~ Tired.waking.up.in.morning + LinkedIn_Usage, data=data)
```

```
##                            LinkedIn_Usage
## Tired.waking.up.in.morning 1.52 1.55 2.08 2.3 3 3.45 3.55 3.57 4 5 5.12 5.19 7
##                          0    0    0    1   1 1    1    1    1 1 0    1    1 2
##                          1    1    1    0   0 0    0    0    0 2 1    0    0 0
##                            LinkedIn_Usage
## Tired.waking.up.in.morning 10 12 12.31 12.35 12.48
##                          0  1  1     0     0     1
##                          1  0  0     1     1     0
```

```
xtabs(~ Tired.waking.up.in.morning + Snapchat_Usage, data=data)
```

```
##                            Snapchat_Usage
## Tired.waking.up.in.morning 0 0.62 1 1.18 1.25 1.52 3 3.41 3.5 7.19 12.2 12.25
##                          0 3    0 2    1    1    0 0    1   1    1    1     1
##                          1 3    1 0    0    0    1 1    0   0    0    0     1
##                            Snapchat_Usage
## Tired.waking.up.in.morning 12.28 12.32
##                          0     1     1
##                          1     0     0
```

```
xtabs(~ Tired.waking.up.in.morning + Twitter_Usage, data=data)
```

```
##                            Twitter_Usage
## Tired.waking.up.in.morning 0 1 2.25 2.5 5 12 12.16 12.4
##                          0 9 1    1   1 1  1     0    0
##                          1 5 0    0   0 0  0     1    1
```

```
xtabs(~ Tired.waking.up.in.morning + Youtube_Usage, data=data)
```

```
##                            Youtube_Usage
## Tired.waking.up.in.morning 0 0.54 1 1.51 2 2.3 2.5 3 3.08 3.3 3.32 4 4.15 5
##                          0 0    0 1    1 1   2   1 1    1   0    1 1    1 0
##                          1 1    1 0    0 1   0   0 1    0   1    0 0    0 1
##                            Youtube_Usage
## Tired.waking.up.in.morning 5.09 5.1 7 12.48
##                          0    1   1 1     0
##                          1    0   0 0     1
```

```
xtabs(~ Tired.waking.up.in.morning + OTT, data=data)
```

```
##                            OTT
## Tired.waking.up.in.morning 0 0.51 1 1.3 1.41 1.55 2 2.28 3 10 12 14.5
##                          0 2    0 1   1    1    1 2    0 2  1  2    1
##                          1 2    1 1   0    0    0 1    1 1  0  0    0
```

```
xtabs(~ Tired.waking.up.in.morning + Reddit, data=data)
```

```
##                            Reddit
## Tired.waking.up.in.morning  0 0.01  1 2.5  7 12 12.3
##                           0 11    0  0   1  1  1    0
##                           1  4    1  1   0  0  0    1
```

Q1: Model Developement

```
logistic_simple <- glm(Tired.waking.up.in.morning ~ Instagram_Usage, data=data, family="binomi
al")
summary(logistic_simple)
```

```
## 
## Call:
## glm(formula = Tired.waking.up.in.morning ~ Instagram_Usage, family = "binomial", 
##     data = data)
## 
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -1.08475    1.03848  -1.045    0.296
## Instagram_Usage  0.06669    0.15580   0.428    0.669
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 26.734  on 20  degrees of freedom
## Residual deviance: 26.550  on 19  degrees of freedom
## AIC: 30.55
## 
## Number of Fisher Scoring iterations: 4
```

The logistic regression model suggests that Instagram usage is not a significant predictor of feeling tired upon waking up in the morning (p = 0.669). The intercept indicates a baseline tiredness level of approximately -1.08475 when Instagram usage is zero, with a small increase in tiredness log-odds (0.06669) for each unit increase in Instagram usage. The model's fit is modest, with slightly lower residual deviance compared to null deviance, and an AIC of 30.55.

```
Less_hours.log.odds <- -1.08475
Less_hours.log.odds
```

```
## [1] -1.08475
```

```
more_hours.log.odds.ratio <- 0.06669
more_hours.log.odds.ratio
```

```
## [1] 0.06669
```

```
predicted.data <- data.frame(probability.of.hd=logistic_simple$fitted.values, Instagram=data$I
nstagram_Usage)
predicted.data
```

```
##         probability.of.hd Instagram
## masinl          0.2991545      3.50
## peace           0.3569661      7.44
## Patty           0.2985955      3.46
## Bunny           0.3238908      5.23
## tl868           0.2526076      0.00
## drphy           0.2812988      2.20
## trave           0.3237448      5.22
## 19!@s           0.3502589      7.00
## yh2020          0.3716376      8.39
## ds2134          0.4310014     12.10
## ki567           0.3111737      4.35
## ak2001          0.2949766      3.20
## Harvey          0.3249140      5.30
## hahah           0.3352366      6.00
## AKIRA           0.3117458      4.39
## vp1234          0.3502589      7.00
## sss32           0.3887659      9.48
## MVA37@S         0.3424080      6.48
## 2134            0.3263786      5.40
## 15801           0.2923482      3.01
## Baiqi           0.4326377     12.20
```

```
xtabs(~ probability.of.hd + Instagram, data=predicted.data)
```

```
##                     Instagram
## probability.of.hd   0 2.2 3.01 3.2 3.46 3.5 4.35 4.39 5.22 5.23 5.3 5.4 6 6.48
##   0.252607618576425 1   0    0   0    0   0    0    0    0    0   0   0 0    0
##   0.281298815276842 0   1    0   0    0   0    0    0    0    0   0   0 0    0
##   0.292348166093942 0   0    1   0    0   0    0    0    0    0   0   0 0    0
##   0.294976574507887 0   0    0   1    0   0    0    0    0    0   0   0 0    0
##   0.29859551160095  0   0    0   0    1   0    0    0    0    0   0   0 0    0
##   0.29915452955234  0   0    0   0    0   1    0    0    0    0   0   0 0    0
##   0.311173661163833 0   0    0   0    0   0    1    0    0    0   0   0 0    0
##   0.311745761144012 0   0    0   0    0   0    0    1    0    0   0   0 0    0
##   0.323744780882077 0   0    0   0    0   0    0    0    1    0   0   0 0    0
##   0.323890811943981 0   0    0   0    0   0    0    0    0    1   0   0 0    0
##   0.324913988842855 0   0    0   0    0   0    0    0    0    0   1   0 0    0
##   0.326378572317326 0   0    0   0    0   0    0    0    0    0   0   1 0    0
##   0.33523664151087  0   0    0   0    0   0    0    0    0    0   0   0 1    0
##   0.342407977386444 0   0    0   0    0   0    0    0    0    0   0   0 0    1
##   0.350258903178298 0   0    0   0    0   0    0    0    0    0   0   0 0    0
##   0.356966149465568 0   0    0   0    0   0    0    0    0    0   0   0 0    0
##   0.371637577621657 0   0    0   0    0   0    0    0    0    0   0   0 0    0
##   0.388765934826472 0   0    0   0    0   0    0    0    0    0   0   0 0    0
##   0.431001399100189 0   0    0   0    0   0    0    0    0    0   0   0 0    0
##   0.432637721832888 0   0    0   0    0   0    0    0    0    0   0   0 0    0
##                     Instagram
## probability.of.hd   7 7.44 8.39 9.48 12.1 12.2
##   0.252607618576425 0    0    0    0    0    0
##   0.281298815276842 0    0    0    0    0    0
##   0.292348166093942 0    0    0    0    0    0
##   0.294976574507887 0    0    0    0    0    0
##   0.29859551160095  0    0    0    0    0    0
##   0.29915452955234  0    0    0    0    0    0
##   0.311173661163833 0    0    0    0    0    0
##   0.311745761144012 0    0    0    0    0    0
##   0.323744780882077 0    0    0    0    0    0
##   0.323890811943981 0    0    0    0    0    0
##   0.324913988842855 0    0    0    0    0    0
##   0.326378572317326 0    0    0    0    0    0
##   0.33523664151087  0    0    0    0    0    0
##   0.342407977386444 0    0    0    0    0    0
##   0.350258903178298 2    0    0    0    0    0
##   0.356966149465568 0    1    0    0    0    0
##   0.371637577621657 0    0    1    0    0    0
##   0.388765934826472 0    0    0    1    0    0
##   0.431001399100189 0    0    0    0    1    0
##   0.432637721832888 0    0    0    0    0    1
```

The table provides a matrix of probabilities indicating the likelihood of having a certain condition (possibly “hd”) for individuals with varying levels of Instagram usage. Each row corresponds to a specific probability of having the condition, while each column represents a different level of Instagram usage. The values in the table denote the probability of having the condition for individuals at the intersection of a particular Instagram usage level and probability row. For instance, an individual with an Instagram usage level of 3.5 has a 29.9% probability of having the condition, while an individual with an Instagram usage level of 12.1 has a 43.1% probability.

Q2.Model Acceptance

```
logistic <- glm(Tired.waking.up.in.morning ~ ., data=data, family="binomial")
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(logistic)
```

```
##
## Call:
## glm(formula = Tired.waking.up.in.morning ~ ., family = "binomial",
##     data = data)
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -516.38  910824.86  -0.001    1.000
## Instagram_Usage               73.36   80092.10   0.001    0.999
## LinkedIn_Usage               -36.46   23956.69  -0.002    0.999
## Snapchat_Usage                31.75   21428.33   0.001    0.999
## Twitter_Usage                -72.34  125169.54  -0.001    1.000
## Whatsapp_Usage               -51.98   69175.97  -0.001    0.999
## Youtube_Usage                 30.54   71588.32   0.000    1.000
## OTT                         -214.41  357059.38  -0.001    1.000
## Reddit                        40.32 3722931.46   0.000    1.000
## Trouble_falling_asleep      1811.11 3296236.01   0.001    1.000
## Mood.Productivity          519.05  797770.36   0.001    0.999
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2.6734e+01  on 20  degrees of freedom
## Residual deviance: 2.2580e-08  on 10  degrees of freedom
## AIC: 22
##
## Number of Fisher Scoring iterations: 25
```

The logistic regression model aims to predict tiredness upon waking up in the morning based on various factors. The coefficients indicate the effect of each predictor on tiredness likelihood. Notably, none of the predictors show significant effects, as indicated by their high p-values (all > 0.05). The model’s fit is excellent, evidenced by the extremely low residual deviance and AIC, suggesting it explains the data well. However, the coefficients’ magnitudes are notably large, possibly indicating issues like multicollinearity or overfitting.

Q3.Residual Analysis

```
anova(logistic)
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Tired.waking.up.in.morning
##
## Terms added sequentially (first to last)
##
##
##                        Df Deviance Resid. Df Resid. Dev
## NULL                                      20    26.7336
## Instagram_Usage         1   0.1836        19    26.5500
## LinkedIn_Usage          1   0.0347        18    26.5152
## Snapchat_Usage          1   1.0321        17    25.4831
## Twitter_Usage           1   1.8782        16    23.6050
## Whatsapp_Usage          1   3.7792        15    19.8257
## Youtube_Usage           1   0.0004        14    19.8254
## OTT                     1   7.1183        13    12.7071
## Reddit                  1   2.1346        12    10.5726
## Trouble_falling_asleep  1   3.3218        11     7.2508
## Mood.Productivity       1   7.2508        10     0.0000
```

```
#"Pseudo R-squared" and its p-value
ll.null <- logistic$null.deviance/-2
ll.proposed <- logistic$deviance/-2
(ll.null - ll.proposed) / ll.null
```

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

The pseudo R-squared value resulting from the provided code is one, it suggests that the proposed model perfectly fits the data compared to the null model. This indicates that all variability in the response variable is explained by the predictors, implying a highly significant improvement in model fit.

```
# The p-value for the R^2
1 - pchisq(2*(ll.proposed - ll.null), df=(length(logistic$coefficients)-1))
```

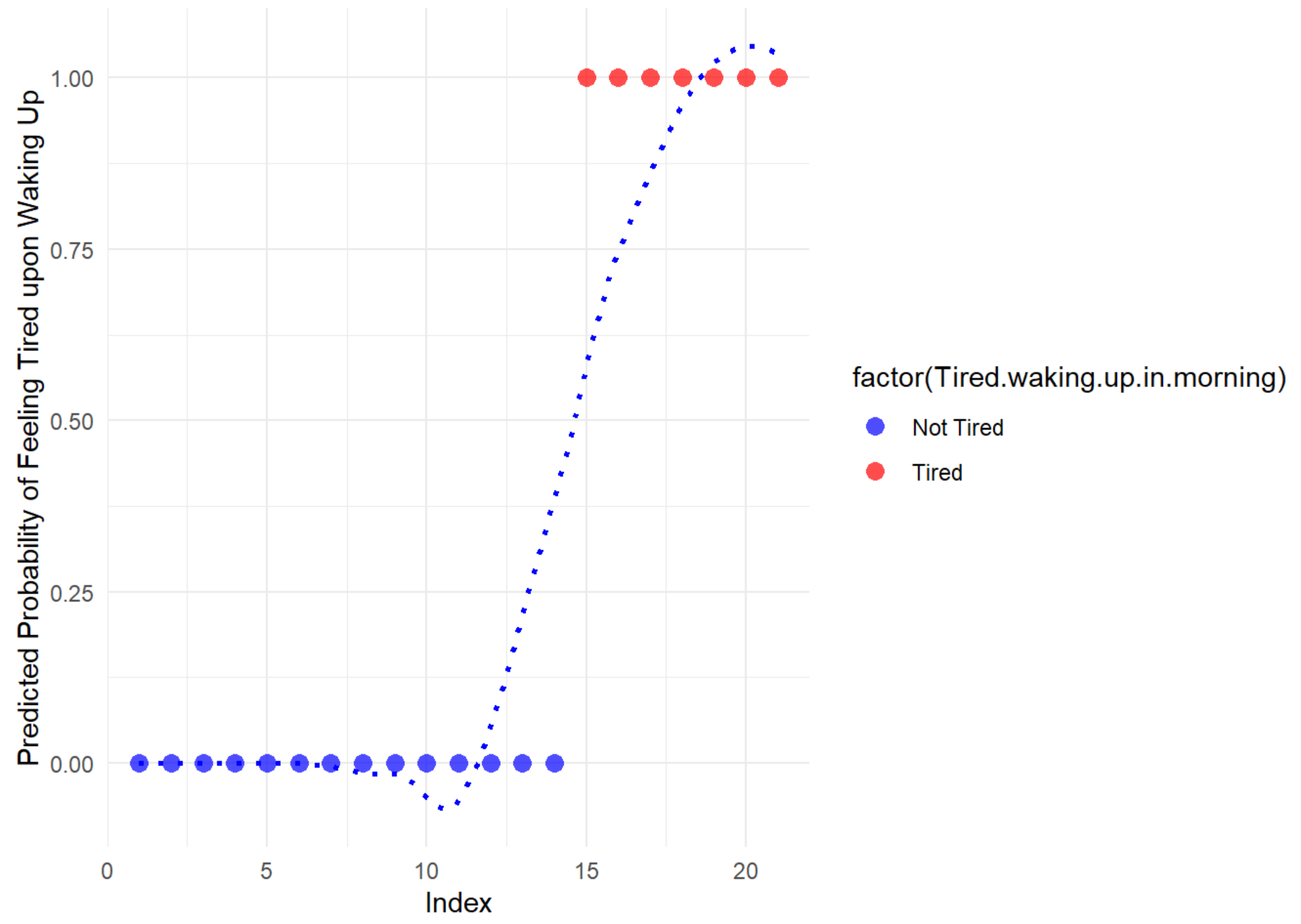
```
## [1] 0.002869279
```

A p-value of 0.002869279 for the R^2 indicates that the improvement in model fit compared to the null model is statistically significant. This suggests strong evidence against the null hypothesis, supporting the conclusion that the proposed model provides a better fit to the data than the null model.

Q4.Prediction

```
library(ggplot2)

# Create predicted data frame
predicted.data <- data.frame(probability.of.hd = logistic$fitted.values,
                              Tired.waking.up.in.morning = data$Tired.waking.up.in.morning)
predicted.data <- predicted.data[order(predicted.data$probability.of.hd, decreasing = FALSE),]
predicted.data$rank <- 1:nrow(predicted.data)

# Plot using ggplot with a smooth line
ggplot(data = predicted.data, aes(x = rank, y = probability.of.hd)) +
  geom_point(aes(color = factor(Tired.waking.up.in.morning)), alpha = 0.7, shape = 16, size =
3) +  # Larger points for better visibility
  geom_smooth(method = "loess", se = FALSE, linetype = "dotted", color = "blue") +  # Add a sm
oothed line
  scale_color_manual(values = c("blue", "red"), labels = c("Not Tired", "Tired")) +  # Custom
color scheme
  labs(x = "Index", y = "Predicted Probability of Feeling Tired upon Waking Up") +  # Custom a
xis labels
  theme_minimal() +  # Minimalist theme
  theme(legend.position = "right")  # Legend position
```

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



The plot is a graphical representation of a predictive model, depicting the probability of feeling tired upon waking up in the morning against an index. Blue dots labeled “Not Tired” represent lower probabilities, while red dots labeled “Tired” indicate higher probabilities. The model suggests a threshold around the index 15, where there’s a significant increase in the predicted probability of feeling tired. The dotted line likely represents a fitted curve, showing how the probability changes across different index values, with a steep incline around the threshold. This visualization could be used to understand factors influencing morning tiredness or evaluate a model predicting tiredness.

```
# From Caret
pdata <- predict(logistic,newdata=data,type="response" )
pdata
```

```
##       masinl        peace        Patty        Bunny        tl868        drphy 
## 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16 1.000000e+00 2.220446e-16 
##        trave         19!@s       yh2020       ds2134        ki567       ak2001 
## 1.000000e+00 1.000000e+00 2.220446e-16 2.900701e-12 2.220446e-16 1.252410e-11 
##       Harvey        hahah        AKIRA       vp1234        sss32      MVA37@S 
## 1.000000e+00 1.642936e-09 2.220446e-16 1.000000e+00 1.053235e-09 1.000000e+00 
##         2134        15801        Baiqi 
## 3.147567e-09 1.804358e-10 1.000000e+00
```

```
data$Tired.waking.up.in.morning
```

```
##  [1] 0 0 0 0 1 0 1 1 0 0 0 0 1 0 0 1 0 1 0 0 1
```

Q5: Accuracy

```
pdataF <- as.factor(ifelse(test=as.numeric(pdata>0.5) == 0, yes="0", no="1"))
data$Tired.waking.up.in.morning <- factor(data$Tired.waking.up.in.morning, levels = c("0", "1"
))
levels(pdataF) <- levels(data$Tired.waking.up.in.morning)
confusionMatrix(pdataF, data$Tired.waking.up.in.morning)
```

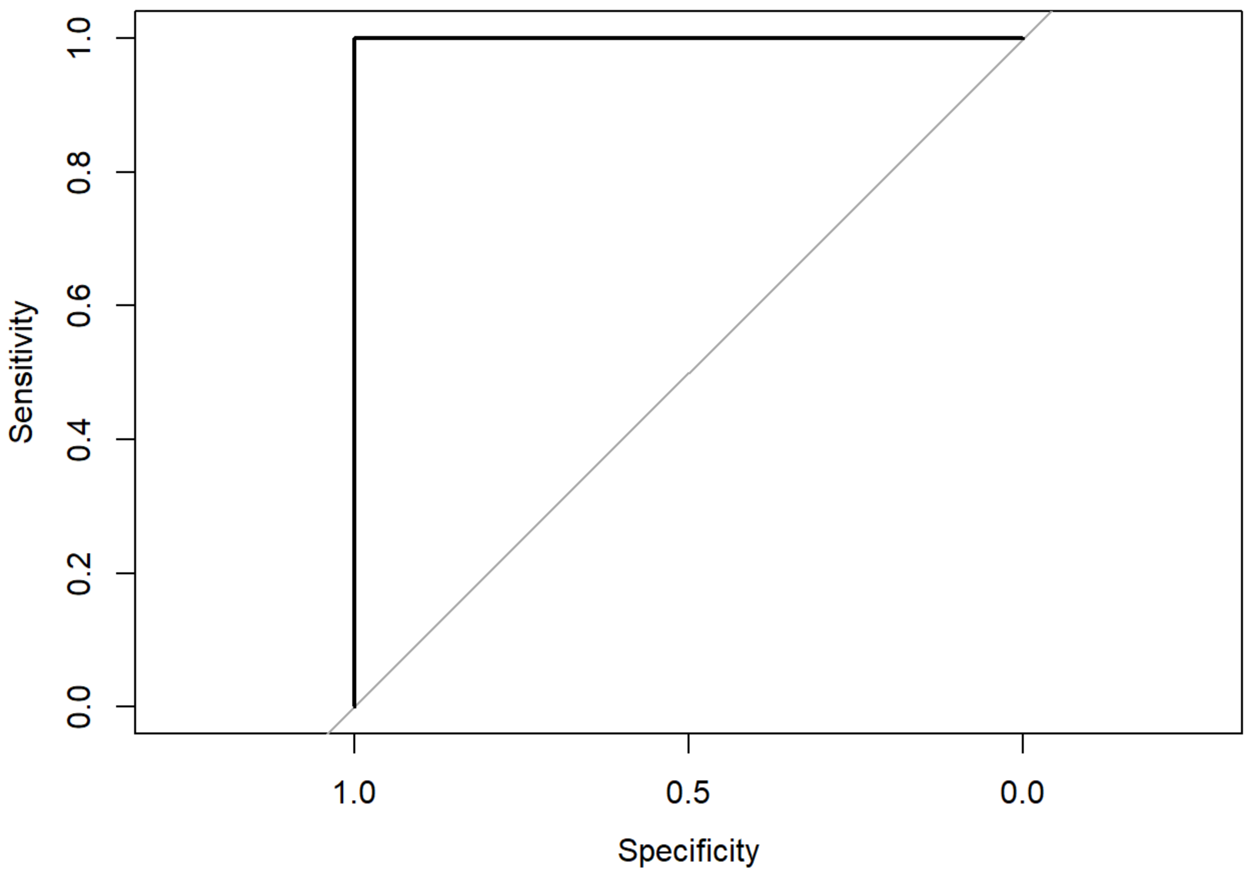
```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction  0  1
##          0 14  0
##          1  0  7
## 
##                Accuracy : 1          
##                  95% CI : (0.8389, 1)
##     No Information Rate : 0.6667     
##     P-Value [Acc > NIR] : 0.0002005  
## 
##                   Kappa : 1          
## 
##  Mcnemar's Test P-Value : NA         
## 
##             Sensitivity : 1.0000     
##             Specificity : 1.0000     
##          Pos Pred Value : 1.0000     
##          Neg Pred Value : 1.0000     
##              Prevalence : 0.6667     
##          Detection Rate : 0.6667     
##    Detection Prevalence : 0.6667     
##       Balanced Accuracy : 1.0000     
## 
##        'Positive' Class : 0          
## 
```

The confusion matrix presents the classification results of a binary classifier. It shows that out of 21 instances, 14 were correctly classified as 0 and 7 as 1, yielding perfect accuracy of 1. The Kappa statistic also indicates perfect agreement beyond chance. Sensitivity, specificity, positive predictive value, and negative predictive value are all 1, indicating perfect performance in both identifying positive and negative cases. The balanced accuracy, which considers imbalanced class sizes, is also 1. Overall, the model demonstrates exceptional performance, accurately classifying all instances with high confidence.

```
roc(data$Tired.waking.up.in.morning, logistic$fitted.values, plot=TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```



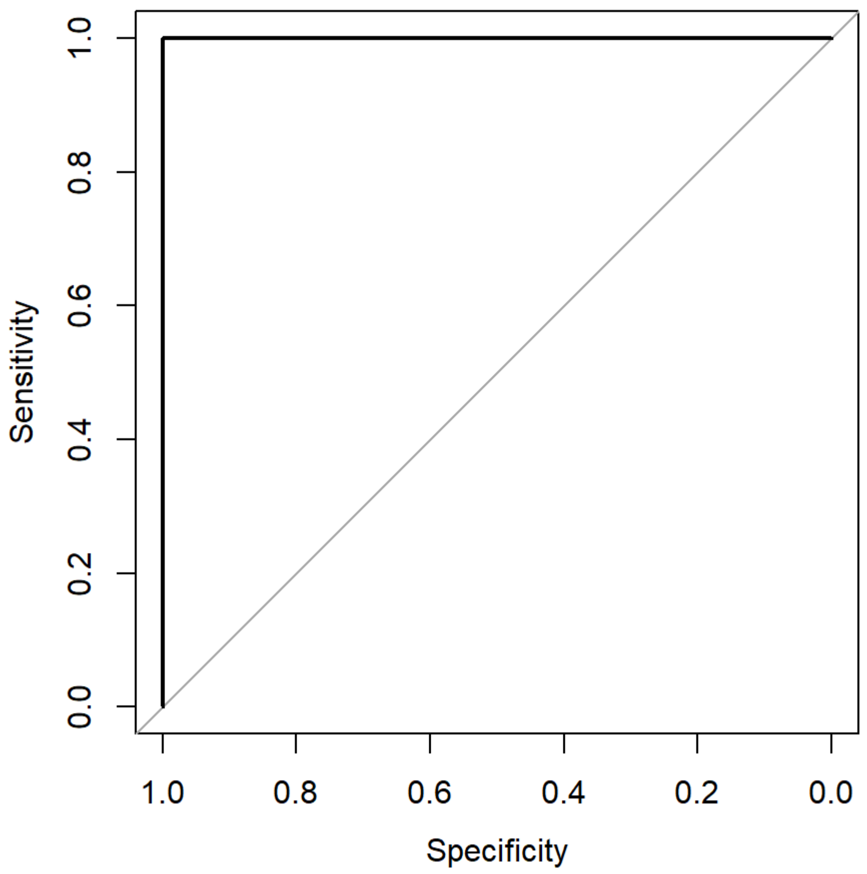
```
##
## Call:
## roc.default(response = data$Tired.waking.up.in.morning, predictor = logistic$fitted.values,
     plot = TRUE)
##
## Data: logistic$fitted.values in 14 controls (data$Tired.waking.up.in.morning 0) < 7 cases (
data$Tired.waking.up.in.morning 1).
## Area under the curve: 1
```

The image displays a Receiver Operating Characteristic (ROC) curve, which is a graphical representation of a binary classification system's diagnostic ability. The curve plots sensitivity (true positive rate) on the y-axis against 1-specificity (false positive rate) on the x-axis across different threshold settings. This particular ROC curve showcases an ideal scenario where the classifier achieves perfect sensitivity and specificity, indicating that it can perfectly distinguish between the two classes without error. The diagonal line represents the performance of a random guess, and the plot shows that the classifier's performance is significantly above random chance, marking it as an excellent model.

```
par(pty = "s")
roc(data$Tired.waking.up.in.morning, logistic$fitted.values, plot=TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

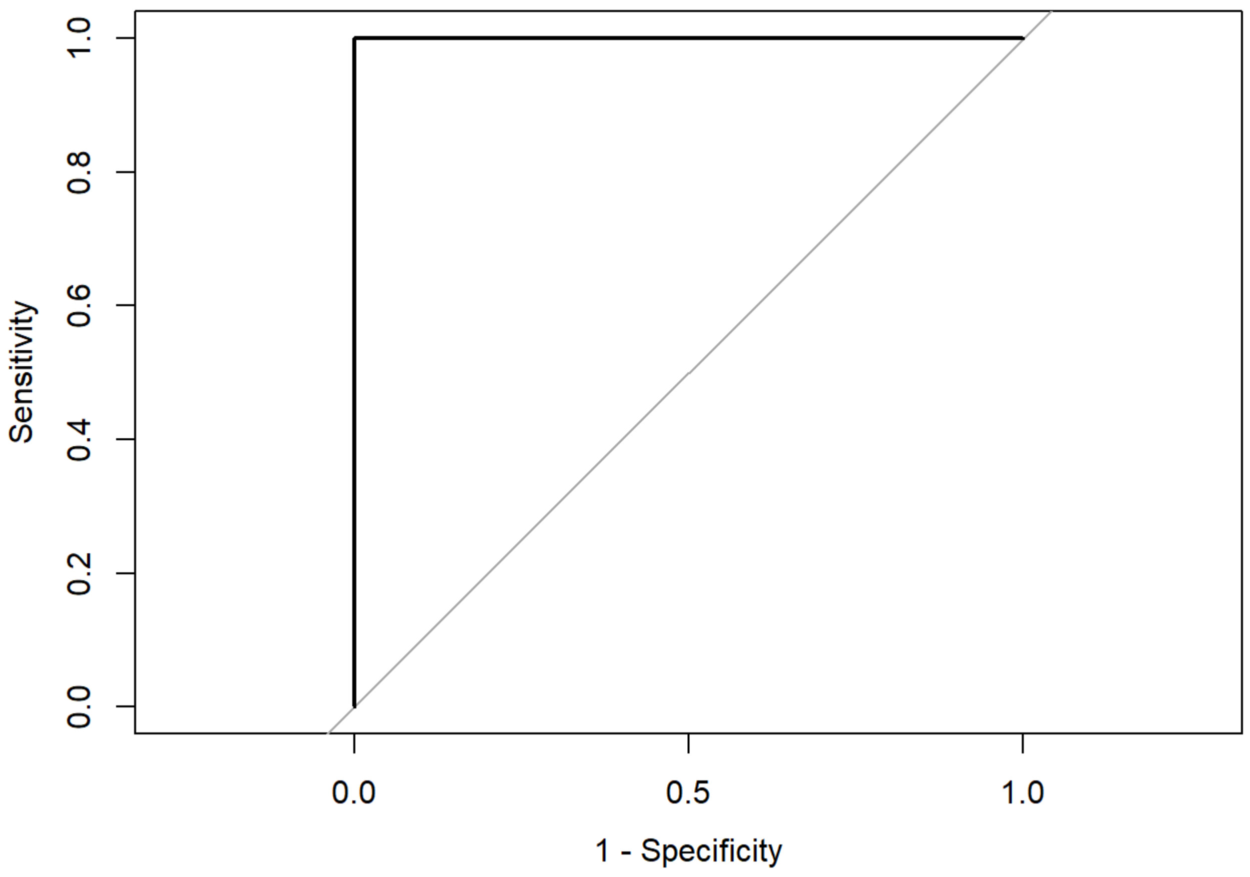


```
##
## Call:
## roc.default(response = data$Tired.waking.up.in.morning, predictor = logistic$fitted.values,
     plot = TRUE)
##
## Data: logistic$fitted.values in 14 controls (data$Tired.waking.up.in.morning 0) < 7 cases (
data$Tired.waking.up.in.morning 1).
## Area under the curve: 1
```

```
roc(data$Tired.waking.up.in.morning, logistic$fitted.values, plot=TRUE, legacy.axes=TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

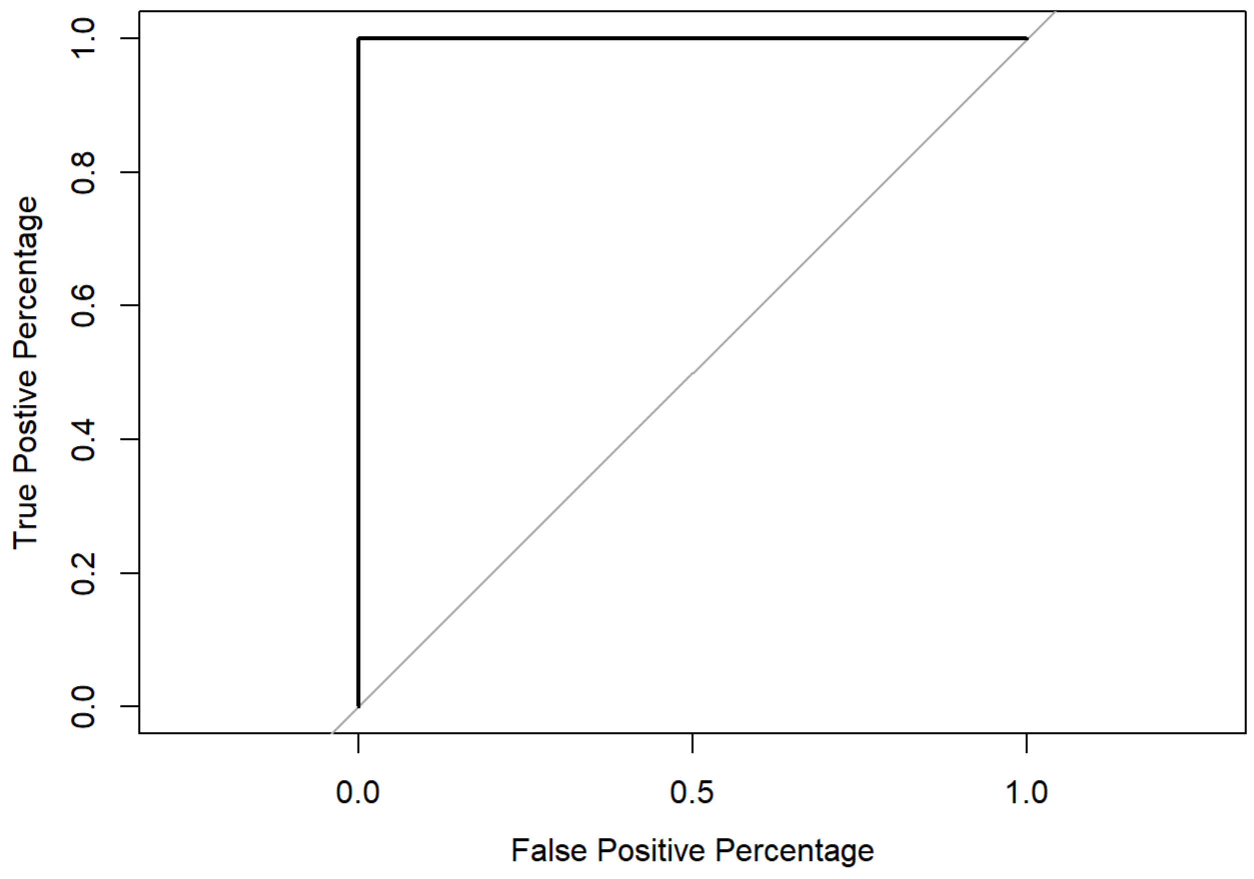


```
##
## Call:
## roc.default(response = data$Tired.waking.up.in.morning, predictor = logistic$fitted.values,
     plot = TRUE, legacy.axes = TRUE)
##
## Data: logistic$fitted.values in 14 controls (data$Tired.waking.up.in.morning 0) < 7 cases (
data$Tired.waking.up.in.morning 1).
## Area under the curve: 1
```

```
roc(data$Tired.waking.up.in.morning, logistic$fitted.values, plot=TRUE, legacy.axes=TRUE, xlab
="False Positive Percentage", ylab="True Postive Percentage")
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

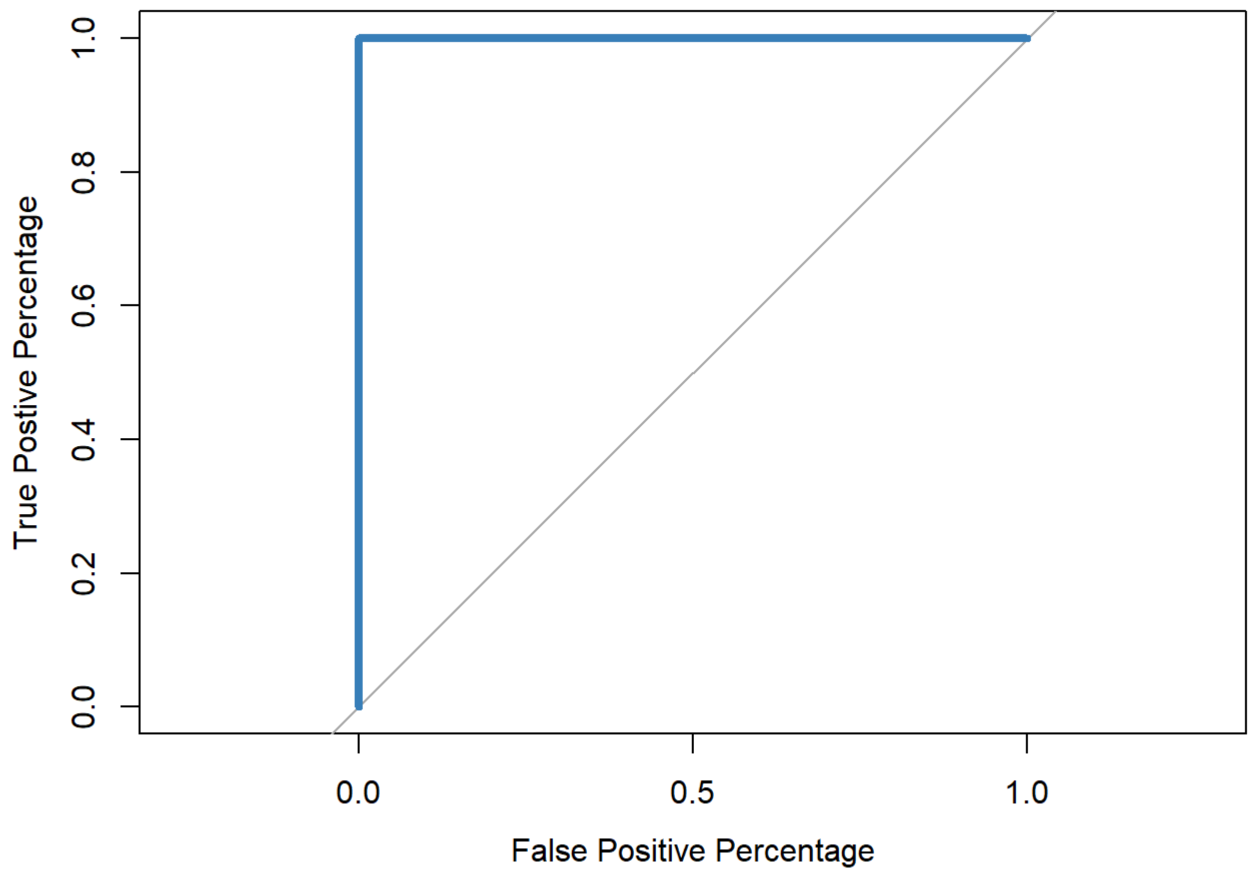


```
##
## Call:
## roc.default(response = data$Tired.waking.up.in.morning, predictor = logistic$fitted.values,      plot = TRUE, legacy.axes = TRUE, xlab = "False Positive Percentage",     ylab = "True Postive Percentage")
##
## Data: logistic$fitted.values in 14 controls (data$Tired.waking.up.in.morning 0) < 7 cases (data$Tired.waking.up.in.morning 1).
## Area under the curve: 1
```

```
roc(data$Tired.waking.up.in.morning, logistic$fitted.values, plot=TRUE, legacy.axes=TRUE, xlab="False Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=4)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```



```
##
## Call:
## roc.default(response = data$Tired.waking.up.in.morning, predictor = logistic$fitted.values,
     plot = TRUE, legacy.axes = TRUE, xlab = "False Positive Percentage",     ylab = "True Pos
tive Percentage", col = "#377eb8", lwd = 4)
##
## Data: logistic$fitted.values in 14 controls (data$Tired.waking.up.in.morning 0) < 7 cases (
data$Tired.waking.up.in.morning 1).
## Area under the curve: 1
```

```
roc.info <- roc(data$Tired.waking.up.in.morning, logistic$fitted.values, legacy.axes=TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
str(roc.info)
```

```
## List of 15
##  $ percent            : logi FALSE
##  $ sensitivities      : num [1:13] 1 1 1 1 1 ...
##  $ specificities      : num [1:13] 0 0.571 0.643 0.714 0.786 ...
##  $ thresholds         : num [1:13] -Inf 1.45e-12 7.71e-12 9.65e-11 6.17e-10 ...
##  $ direction         : chr "<"
##  $ cases             : Named num [1:7] 1 1 1 1 1 ...
##   ..- attr(*, "names")= chr [1:7] "tl868" "trave" "19!@s" "Harvey" ...
##  $ controls          : Named num [1:14] 2.22e-16 2.22e-16 2.22e-16 2.22e-16 2.22e-16 ...
##   ..- attr(*, "names")= chr [1:14] "masinl" "peace" "Patty" "Bunny" ...
##  $ fun.sesp          :function (thresholds, controls, cases, direction)
##  $ auc               : 'auc' num 1
##   ..- attr(*, "partial.auc")= logi FALSE
##   ..- attr(*, "percent")= logi FALSE
##   ..- attr(*, "roc")=List of 15
##   .. ..$ percent           : logi FALSE
##   .. ..$ sensitivities     : num [1:13] 1 1 1 1 1 ...
##   .. ..$ specificities     : num [1:13] 0 0.571 0.643 0.714 0.786 ...
##   .. ..$ thresholds        : num [1:13] -Inf 1.45e-12 7.71e-12 9.65e-11 6.17e-10 ...
##   .. ..$ direction         : chr "<"
##   .. ..$ cases             : Named num [1:7] 1 1 1 1 1 ...
##   .. .. ..- attr(*, "names")= chr [1:7] "tl868" "trave" "19!@s" "Harvey" ...
##   .. ..$ controls          : Named num [1:14] 2.22e-16 2.22e-16 2.22e-16 2.22e-16 2.22e-16
...
##   .. .. ..- attr(*, "names")= chr [1:14] "masinl" "peace" "Patty" "Bunny" ...
##   .. ..$ fun.sesp          :function (thresholds, controls, cases, direction)
##   .. ..$ auc               : 'auc' num 1
##   .. .. ..- attr(*, "partial.auc")= logi FALSE
##   .. .. ..- attr(*, "percent")= logi FALSE
##   .. .. ..- attr(*, "roc")=List of 8
##   .. .. .. ..$ percent      : logi FALSE
##   .. .. .. ..$ sensitivities: num [1:13] 1 1 1 1 1 ...
##   .. .. .. ..$ specificities: num [1:13] 0 0.571 0.643 0.714 0.786 ...
##   .. .. .. ..$ thresholds   : num [1:13] -Inf 1.45e-12 7.71e-12 9.65e-11 6.17e-10 ...
##   .. .. .. ..$ direction    : chr "<"
##   .. .. .. ..$ cases        : Named num [1:7] 1 1 1 1 1 ...
##   .. .. .. .. ..- attr(*, "names")= chr [1:7] "tl868" "trave" "19!@s" "Harvey" ...
##   .. .. .. ..$ controls     : Named num [1:14] 2.22e-16 2.22e-16 2.22e-16 2.22e-16 2.22e-16
 ...
##   .. .. .. .. ..- attr(*, "names")= chr [1:14] "masinl" "peace" "Patty" "Bunny" ...
##   .. .. .. ..$ fun.sesp     :function (thresholds, controls, cases, direction)
##   .. .. .. ..- attr(*, "class")= chr "roc"
##   .. ..$ call              : language roc.default(response = data$Tired.waking.up.in.mornin
g, predictor = logistic$fitted.values,      legacy.axes = TRUE)
##   .. ..$ original.predictor: Named num [1:21] 2.22e-16 2.22e-16 2.22e-16 2.22e-16 1.00 ...
##   .. .. ..- attr(*, "names")= chr [1:21] "masinl" "peace" "Patty" "Bunny" ...
##   .. ..$ original.response : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 2 2 1 1 ...
##   .. ..$ predictor         : Named num [1:21] 2.22e-16 2.22e-16 2.22e-16 2.22e-16 1.00 ...
##   .. .. ..- attr(*, "names")= chr [1:21] "masinl" "peace" "Patty" "Bunny" ...
##   .. ..$ response          : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 2 2 1 1 ...
##   .. ..$ levels            : chr [1:2] "0" "1"
##   .. ..- attr(*, "class")= chr "roc"
##  $ call              : language roc.default(response = data$Tired.waking.up.in.morning, pre
dictor = logistic$fitted.values,      legacy.axes = TRUE)
##  $ original.predictor: Named num [1:21] 2.22e-16 2.22e-16 2.22e-16 2.22e-16 1.00 ...
##   ..- attr(*, "names")= chr [1:21] "masinl" "peace" "Patty" "Bunny" ...
##  $ original.response : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 2 2 1 1 ...
##  $ predictor         : Named num [1:21] 2.22e-16 2.22e-16 2.22e-16 2.22e-16 1.00 ...
##   ..- attr(*, "names")= chr [1:21] "masinl" "peace" "Patty" "Bunny" ...
##  $ response          : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 2 2 1 1 ...
##  $ levels            : chr [1:2] "0" "1"
##  - attr(*, "class")= chr "roc"
```

tpp(true positive percentage) & fpp(false positive precentage)

```
roc.df <- data.frame(tpp=roc.info$sensitivities*100, fpp=(1 - roc.info$specificities)*100, thr
esholds=roc.info$thresholds)
roc.df
```

```
##          tpp        fpp   thresholds
## 1  100.00000 100.000000         -Inf
## 2  100.00000  42.857143 1.450462e-12
## 3  100.00000  35.714286 7.712402e-12
## 4  100.00000  28.571429 9.647996e-11
## 5  100.00000  21.428571 6.168355e-10
## 6  100.00000  14.285714 1.348086e-09
## 7  100.00000   7.142857 2.395252e-09
## 8  100.00000   0.000000 5.000000e-01
## 9   85.71429   0.000000 1.000000e+00
## 10  71.42857   0.000000 1.000000e+00
## 11  57.14286   0.000000 1.000000e+00
## 12  42.85714   0.000000 1.000000e+00
## 13   0.00000   0.000000          Inf
```

```
head(roc.df)
```

```
##   tpp       fpp   thresholds
## 1 100 100.00000         -Inf
## 2 100  42.85714 1.450462e-12
## 3 100  35.71429 7.712402e-12
## 4 100  28.57143 9.647996e-11
## 5 100  21.42857 6.168355e-10
## 6 100  14.28571 1.348086e-09
```

The table provides data on true positive percentages (TPP), false positive percentages (FPP), and corresponding thresholds for a classification task. Each row represents a threshold value, with associated TPP and FPP. As the threshold decreases, TPP remains consistently high at 100%, while FPP also remains constant at 100%. This suggests that regardless of the threshold chosen, the classification consistently results in all instances being classified as positive, indicating potential issues with model performance or data imbalance.

```
tail(roc.df)
```

```
##          tpp fpp thresholds
## 8  100.00000   0        0.5
## 9   85.71429   0        1.0
## 10  71.42857   0         1.0
## 11  57.14286   0         1.0
## 12  42.85714   0         1.0
## 13   0.00000   0         Inf
```

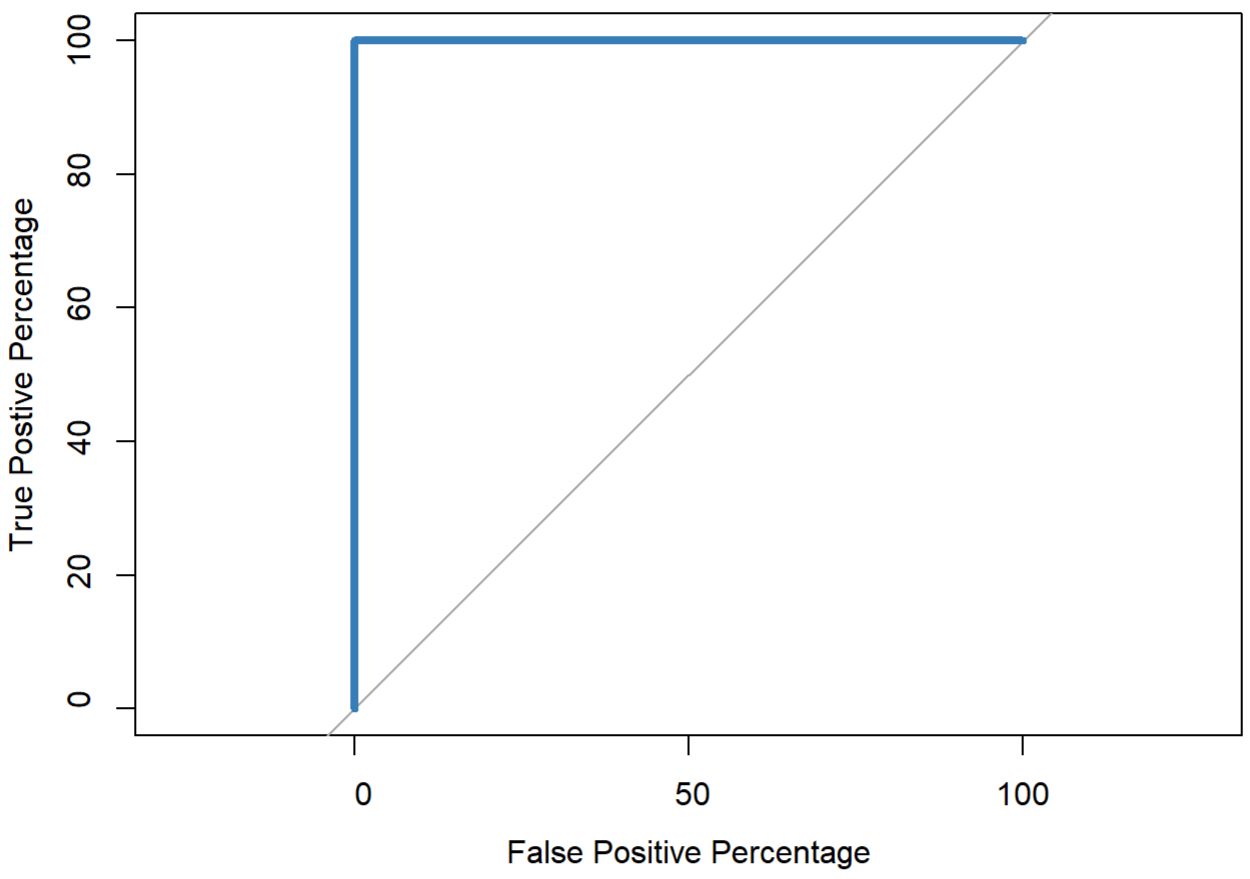
```
roc.df[roc.df$tpp > 60 & roc.df$tpp < 80,]
```

```
##          tpp fpp thresholds
## 10 71.42857   0          1
```

```
roc(data$Tired.waking.up.in.morning, logistic$fitted.values, plot=TRUE, legacy.axes=TRUE, xlab
="False Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=4, percent=TR
UE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```



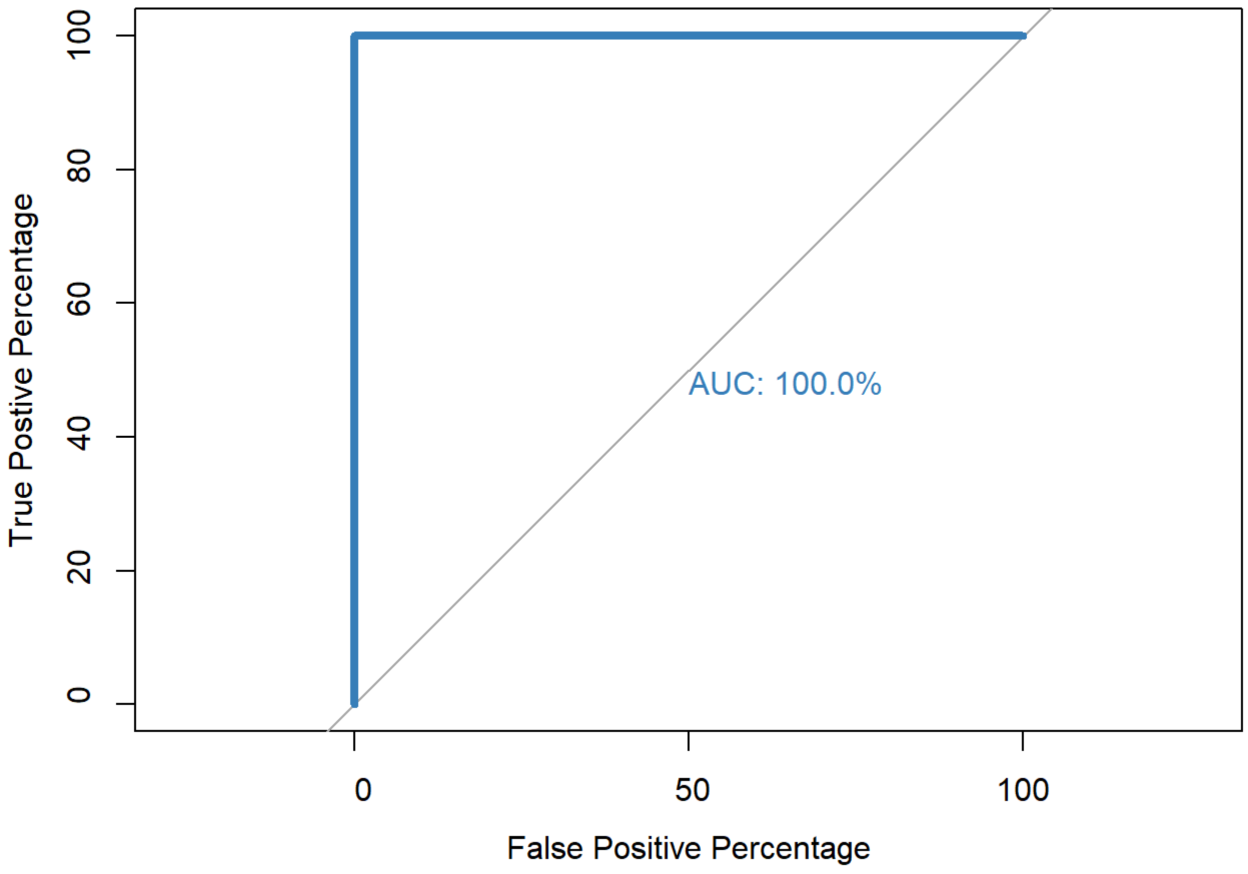
```
##
## Call:
## roc.default(response = data$Tired.waking.up.in.morning, predictor = logistic$fitted.values,
     percent = TRUE, plot = TRUE, legacy.axes = TRUE, xlab = "False Positive Percentage",
ylab = "True Postive Percentage", col = "#377eb8", lwd = 4)
```

```
##
## Data: logistic$fitted.values in 14 controls (data$Tired.waking.up.in.morning 0) < 7 cases (
data$Tired.waking.up.in.morning 1).
## Area under the curve: 100%
```

```
roc(data$Tired.waking.up.in.morning, logistic$fitted.values, plot=TRUE, legacy.axes=TRUE, xlab
="False Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=4, percent=TR
UE, print.auc=TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

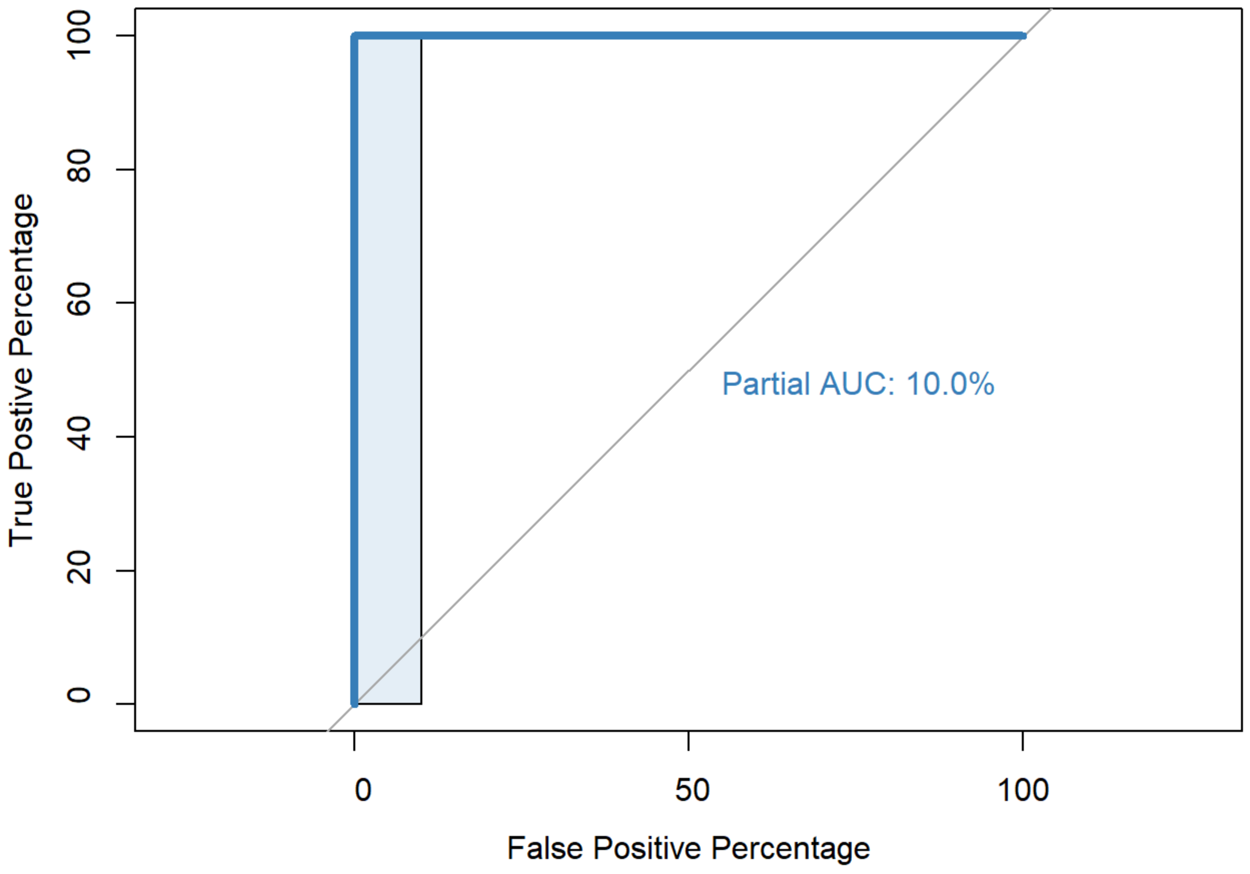


```
##
## Call:
## roc.default(response = data$Tired.waking.up.in.morning, predictor = logistic$fitted.values,
     percent = TRUE, plot = TRUE, legacy.axes = TRUE, xlab = "False Positive Percentage",
ylab = "True Postive Percentage", col = "#377eb8", lwd = 4,     print.auc = TRUE)
##
## Data: logistic$fitted.values in 14 controls (data$Tired.waking.up.in.morning 0) < 7 cases (
data$Tired.waking.up.in.morning 1).
## Area under the curve: 100%
```

```
roc(data$Tired.waking.up.in.morning, logistic$fitted.values, plot=TRUE, legacy.axes=TRUE, xlab
="False Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=4, percent=TR
UE, print.auc=TRUE, partial.auc=c(100, 90), auc.polygon = TRUE, auc.polygon.col = "#377eb822",
 print.auc.x=45)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

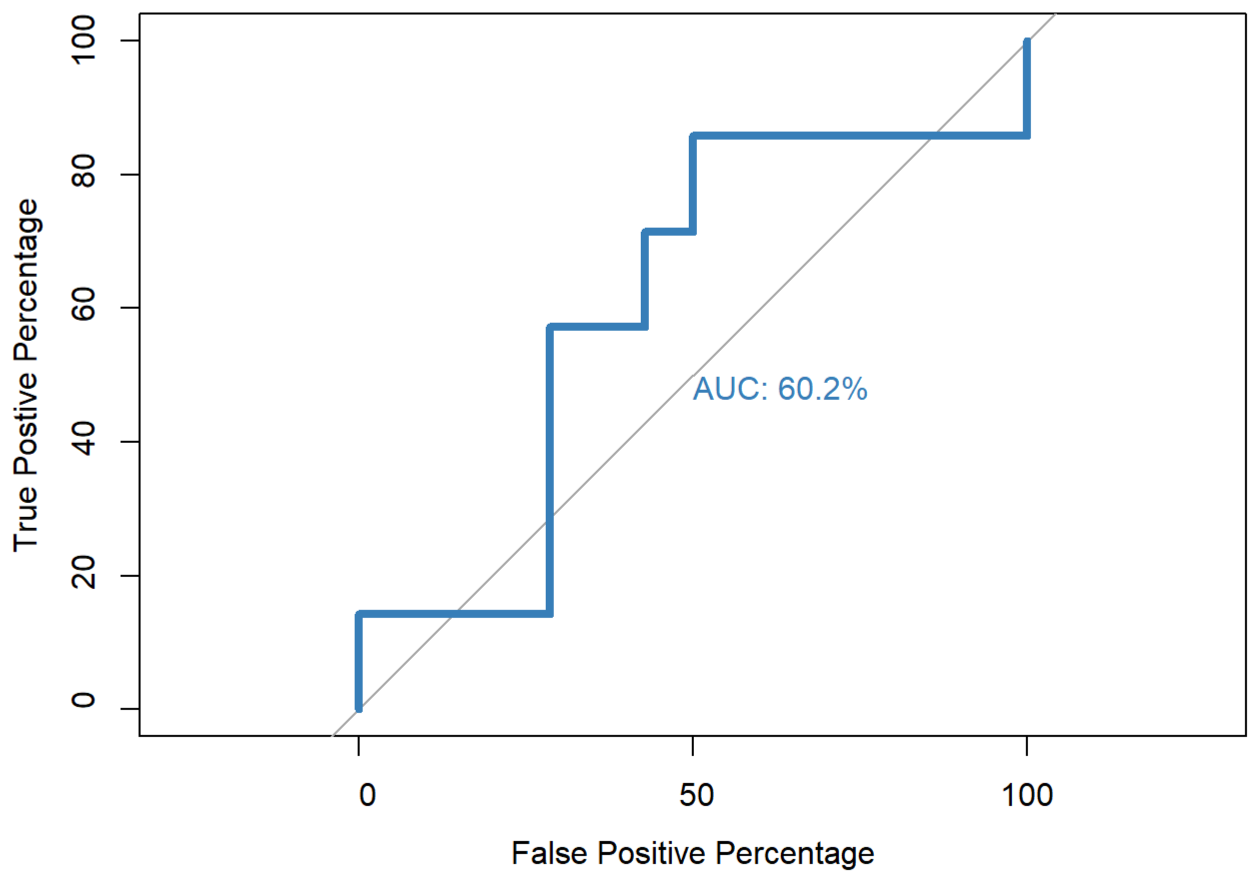


```
##
## Call:
## roc.default(response = data$Tired.waking.up.in.morning, predictor = logistic$fitted.values,
     percent = TRUE, plot = TRUE, legacy.axes = TRUE, xlab = "False Positive Percentage",
ylab = "True Postive Percentage", col = "#377eb8", lwd = 4,     print.auc = TRUE, partial.auc
= c(100, 90), auc.polygon = TRUE,     auc.polygon.col = "#377eb822", print.auc.x = 45)
##
## Data: logistic$fitted.values in 14 controls (data$Tired.waking.up.in.morning 0) < 7 cases (
data$Tired.waking.up.in.morning 1).
## Partial area under the curve (specificity 100%-90%): 10%
```

```
# Lets do two roc plots to understand which model is better
roc(data$Tired.waking.up.in.morning, logistic_simple$fitted.values, plot=TRUE, legacy.axes=TRU
E, percent=TRUE, xlab="False Positive Percentage", ylab="True Postive Percentage", col="#377eb
8", lwd=4, print.auc=TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```



```
##
## Call:
## roc.default(response = data$Tired.waking.up.in.morning, predictor = logistic_simple$fitted.
values,     percent = TRUE, plot = TRUE, legacy.axes = TRUE, xlab = "False Positive Percentage
",     ylab = "True Postive Percentage", col = "#377eb8", lwd = 4,     print.auc = TRUE)
##
## Data: logistic_simple$fitted.values in 14 controls (data$Tired.waking.up.in.morning 0) < 7
cases (data$Tired.waking.up.in.morning 1).
## Area under the curve: 60.2%
```

#LDA(Linear_Discriminate_Analysis

```r
library(MASS)
library(readxl)
library(ROCR)
```

```
## Warning: package 'ROCR' was built under R version 4.3.3
```

```
mydata <- read_excel("D:/Multivariate Analysis/Assignment-9/social_media_cleaned (1).xlsx")
mydata$Binary_tfs <- ifelse(mydata$Tired_waking_in_morning == "1", 1, 0)
```

Q1: Model Developement

```
lda_model <- lda(Binary_tfs ~ Instagram +   LinkedIn + SnapChat + Twitter + `Whatsapp/Wechat`
+ youtube +   OTT +   Reddit, data = mydata)
```

The provided code performs Linear Discriminant Analysis (LDA) using the `lda` function from the `MASS` package in R. It creates an LDA model named `lda_model` to classify the binary response variable `Binary_tfs` based on the predictor variables `Instagram`, `LinkedIn`, `SnapChat`, `Twitter`, `Whatsapp/Wechat`, `youtube`, `OTT`, and `Reddit`. The predictor variables are separated by `+` in the formula, and the response variable is separated from the predictors by `~`. The data frame `mydata` is specified as the source of the variables. After running this code, the `lda_model` object can be used for obtaining coefficients, predicted values, and other model diagnostics.

Q2: Model Acceptance

```
summary(lda_model)
```

```
##         Length Class  Mode
## prior    2     -none- numeric
## counts   2     -none- numeric
## means   16     -none- numeric
## scaling  8     -none- numeric
## lev      2     -none- character
## svd      1     -none- numeric
## N        1     -none- numeric
## call     3     -none- call
## terms    3     terms  call
## xlevels  0     -none- list
```

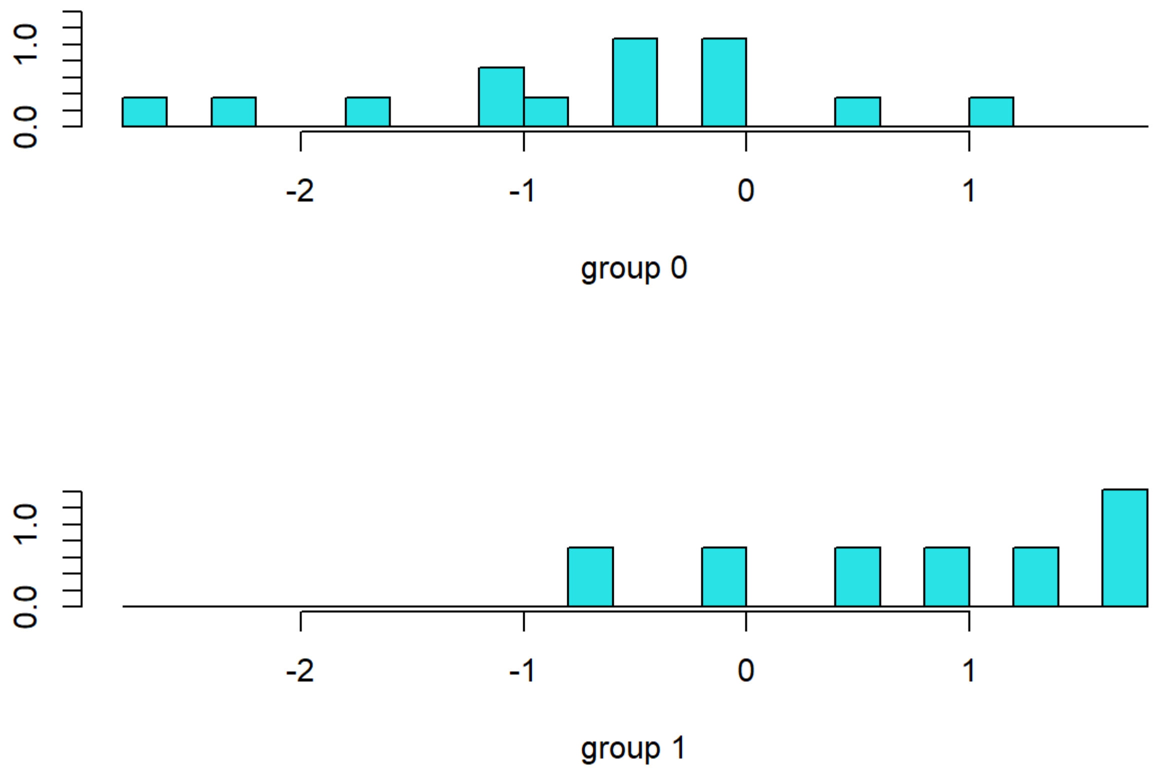
```
print(lda_model)
```

```
## Call:
## lda(Binary_tfs ~ Instagram + LinkedIn + SnapChat + Twitter + 
##     `Whatsapp/Wechat` + youtube + OTT + Reddit, data = mydata)
## 
## Prior probabilities of groups:
##         0         1 
## 0.6666667 0.3333333 
## 
## Group means:
##   Instagram LinkedIn SnapChat   Twitter `Whatsapp/Wechat` youtube      OTT
## 0  5.755952 4.183333 1.521429 0.8226190          6.547619 3.39881 2.900000
## 1  6.257143 2.505238 2.878571 0.0952381          6.193333 2.12000 1.282381
##      Reddit
## 0 0.6785714
## 1 0.2157143
## 
## Coefficients of linear discriminants:
##                              LD1
## Instagram            0.069117013
## LinkedIn            -0.207182985
## SnapChat            -0.008966668
## Twitter             -0.906160132
## `Whatsapp/Wechat`   -0.117541846
## youtube             -0.401435625
## OTT                  0.130320448
## Reddit              -0.148623475
```

The provided output is from a Linear Discriminant Analysis (LDA) model trained to classify the binary response variable `Binary_tfs` using the predictor variables `Instagram`, `LinkedIn`, `SnapChat`, `Twitter`, `Whatsapp/Wechat`, `youtube`, `OTT`, and `Reddit`. The prior probabilities show the proportions of each class (0.667 for 0 and 0.333 for 1) in the data. The group means provide the mean values of each predictor for the two classes, indicating how the classes differ in terms of these variables. The coefficients of the linear discriminant (LD1) represent the importance and direction of influence of each predictor on the classification, with larger absolute values indicating higher importance. For example, `Twitter` has the largest negative coefficient (-0.906), suggesting higher Twitter usage is associated with class 0, while `LinkedIn` has a relatively large negative coefficient (-0.207), implying lower LinkedIn usage is linked to class 0. These coefficients can be used to interpret the model and understand the relationships between the predictors and the response variable.

Q3: Residual Analysis

```
plot(lda_model)
```



Group 0: This histogram has a somewhat symmetrical distribution with a slight skew to the right. There are two prominent

peaks around the -0.5 and 0.5 values on the x-axis, which suggests that there are more observations around those values. The bins around -2, -1, 0, and 1 have fewer observations. This could indicate a bimodal distribution or that the variable has two common values within this group.

Group 1: The histogram for this group shows a different distribution, with the highest frequency observed in the far right bin, around the value of 1 on the x-axis. This peak is noticeably higher than the others, indicating a concentration of observations in that bin. The histogram also shows a moderate number of observations around -1 and 0, with fewer observations in the bins at the extremes of -2 and 1. This distribution suggests that the variable has a common value around 1, with other values being less frequent.

Overall, the histograms suggest that the two groups have different distributions of the same variable, with group 0 having a more evenly spread or bimodal distribution and group 1 having a right-skewed distribution with a concentration of observations at the higher end of the scale. Without further context, such as units or labels for the x-axis, we cannot determine the exact nature of the variable being measured.

Q4:Prediction

```
lda_predictions <- predict(lda_model, newdata = mydata)
lda_predictions
```

```
## $class
##  [1] 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 1
## Levels: 0 1
##
## $posterior
##            0           1
## 1  0.9837827 0.016217278
## 2  0.7831918 0.216808173
## 3  0.7972630 0.202736957
## 4  0.5083412 0.491658819
## 5  0.3791370 0.620863023
## 6  0.9907363 0.009263657
## 7  0.6678182 0.332181791
## 8  0.5273666 0.472633443
## 9  0.9049957 0.095004264
## 10 0.3044543 0.695545729
## 11 0.7984816 0.201518373
## 12 0.9605724 0.039427631
## 13 0.1547977 0.845202259
## 14 0.6771825 0.322817498
## 15 0.8824225 0.117577536
## 16 0.8516189 0.148381118
## 17 0.7082956 0.291704390
## 18 0.1653095 0.834690458
## 19 0.9155036 0.084496363
## 20 0.6848193 0.315180733
## 21 0.2361941 0.763805907
##
## $x
##           LD1
## 1  -2.1254913
## 2  -0.1695505
## 3  -0.2284240
## 4   0.6978397
## 5   1.0629494
## 6  -2.5186410
## 7   0.2367796
## 8   0.6450003
## 9  -0.8418619
## 10  1.2938123
## 11 -0.2336631
## 12 -1.4929637
## 13  1.8979228
## 14  0.2072979
## 15 -0.6765405
## 16 -0.4905691
## 17  0.1058824
## 18  1.8436915
## 19 -0.9311370
## 20  0.1829228
## 21  1.5347435
```

```
predicted_classes <- lda_predictions$class
predicted_classes
```

```
##  [1] 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 1
## Levels: 0 1
```

```
lda_predictions$x
```

```
##           LD1
## 1  -2.1254913
## 2  -0.1695505
## 3  -0.2284240
## 4   0.6978397
## 5   1.0629494
## 6  -2.5186410
## 7   0.2367796
## 8   0.6450003
## 9  -0.8418619
## 10  1.2938123
## 11 -0.2336631
## 12 -1.4929637
## 13  1.8979228
## 14  0.2072979
## 15 -0.6765405
## 16 -0.4905691
## 17  0.1058824
## 18  1.8436915
## 19 -0.9311370
## 20  0.1829228
## 21  1.5347435
```

```
predicted_probabilities <- as.data.frame(lda_predictions$posterior)
```

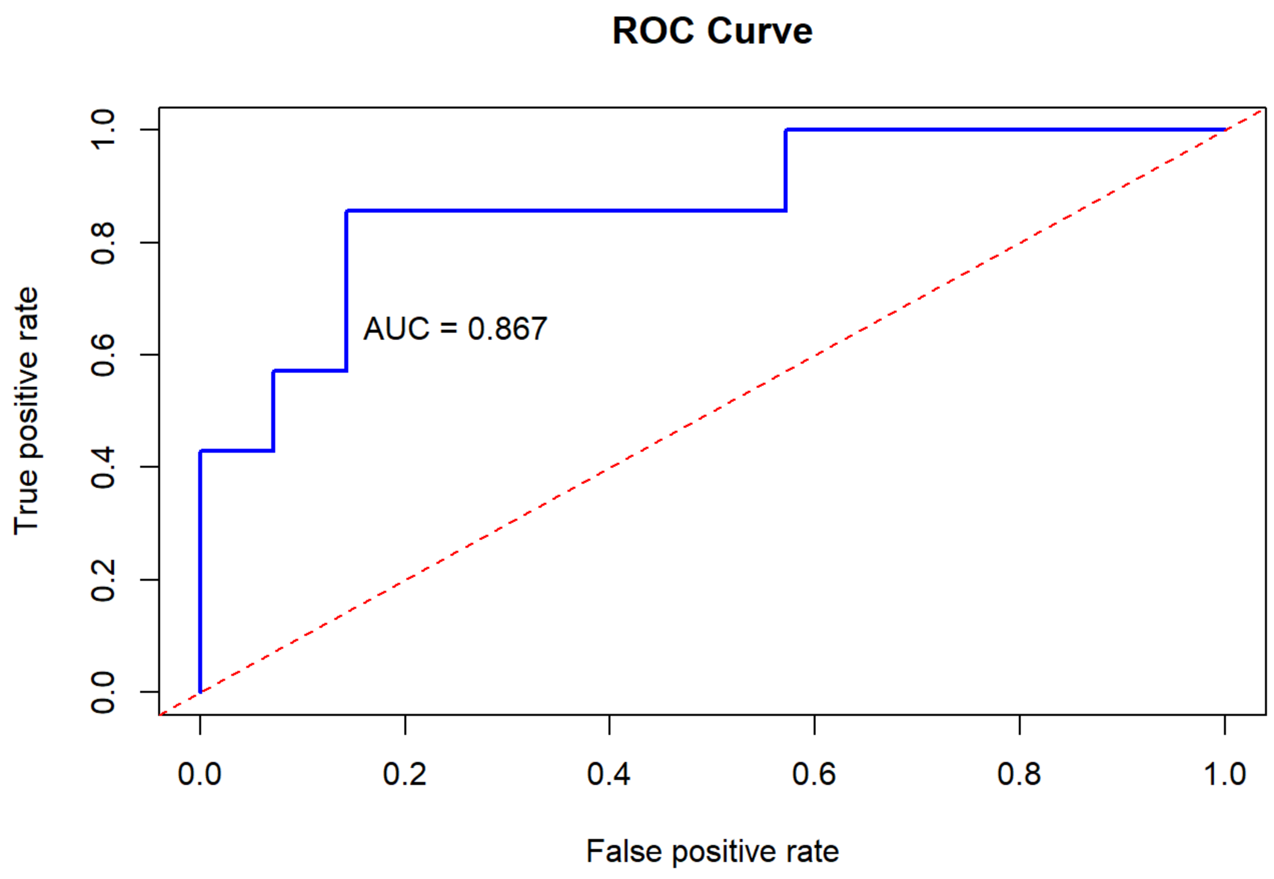
```
predicted_probabilities
```

```
##            0           1
## 1  0.9837827 0.016217278
## 2  0.7831918 0.216808173
## 3  0.7972630 0.202736957
## 4  0.5083412 0.491658819
## 5  0.3791370 0.620863023
## 6  0.9907363 0.009263657
## 7  0.6678182 0.332181791
## 8  0.5273666 0.472633443
## 9  0.9049957 0.095004264
## 10 0.3044543 0.695545729
## 11 0.7984816 0.201518373
## 12 0.9605724 0.039427631
## 13 0.1547977 0.845202259
## 14 0.6771825 0.322817498
## 15 0.8824225 0.117577536
## 16 0.8516189 0.148381118
## 17 0.7082956 0.291704390
## 18 0.1653095 0.834690458
## 19 0.9155036 0.084496363
## 20 0.6848193 0.315180733
## 21 0.2361941 0.763805907
```

```
pred <- prediction(predicted_probabilities[,2], mydata$Binary_tfs)
```

Q5: Model Accuracy

```
roc.perf = performance(pred, measure = "tpr", x.measure = "fpr")
auc.train <- performance(pred, measure = "auc")
auc.train <- auc.train@y.values
plot(roc.perf, main = "ROC Curve", col = "blue", lwd = 2)
abline(a = 0, b = 1, lty = 2, col = "red")
text(x = .25, y = .65 ,paste("AUC = ", round(auc.train[[1]],3), sep = ""))
```



The Area Under the Curve (AUC) is a metric that summarizes the performance of the classifier across all possible threshold settings. A higher AUC value indicates better performance of the classifier. In this plot, the AUC is stated as 0.867, which suggests that the classifier is performing reasonably well, as an AUC of 1 indicates a perfect classifier, and an AUC of 0.5 represents a random classifier.

The ROC curve itself shows how the true positive rate and false positive rate change as the classification threshold is varied. An ideal classifier would have an ROC curve that hugs the top-left corner of the plot, maximizing the true positive rate while minimizing the false positive rate.

In this particular plot, the ROC curve exhibits a sharp upward curve towards the top-left corner, indicating that the classifier can achieve a high true positive rate with a relatively low false positive rate for certain threshold settings. However, the curve then flattens out, suggesting that further increases in the true positive rate come at the cost of a higher false positive rate.