Analysis on Social Media and Mental Health

041-Sai Hitesh, 048-Tharshith, 064-Aditya

2024-05-14

Installing all the required libraries

```
if (!require("readxl")) install.packages("readxl")
```

```
## Loading required package: readxl
```

```
if (!require("dplyr")) install.packages("dplyr")
```

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

```
if (!require("ggplot2")) install.packages("ggplot2")
```

```
## Loading required package: ggplot2
```

```
if (!require("tidyverse")) install.packages("tidyverse")
```

```
## Loading required package: tidyverse

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1
## v readr     2.1.5
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
if (!require("scales")) install.packages("scales")
```

```
## Loading required package: scales
##
## Attaching package: 'scales'
##
## The following object is masked from 'package:purrr':
##
##     discard
##
## The following object is masked from 'package:readr':
##
##     col_factor
```

```r
if (!require("corrplot")) install.packages("corrplot")
```

```
## Loading required package: corrplot
## corrplot 0.92 loaded
```

```r
if (!require("iterators")) install.packages("iterators")
```

```
## Loading required package: iterators
```

```r
if (!require("caret")) install.packages("caret")
```

```
## Loading required package: caret
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
if (!require("pROC")) install.packages("pROC")
```

```
## Loading required package: pROC
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
if (!require("Metrics")) install.packages("Metrics")
```

```
## Loading required package: Metrics
##
```

```
## Attaching package: 'Metrics'
##
## The following object is masked from 'package:pROC':
##
##     auc
##
## The following objects are masked from 'package:caret':
##
##     precision, recall
```

```
if (!require("rpart")) install.packages("rpart")
```

```
## Loading required package: rpart
```

```
if (!require("rattle")) install.packages("rattle")
```

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

```
if (!require("ROCR")) install.packages("ROCR")
```

```
## Loading required package: ROCR
```

```
if (!require("tinytex")) install.packages("tinytex")
```

```
## Loading required package: tinytex
```

```
# Load necessary packages
library(readxl)
library(dplyr)
library(ggplot2)
library(tidyverse)
library(scales)
library(corrplot)
library(iterators)
library(caret)
library(pROC)
library(Metrics)
library(rpart)
library(rattle)
library(ROCR)
library(tinytex)
```

Loading Datasets

```
data_path1="D:/social_mental/data1.xlsx"
data1=read_excel(data_path1)

data_path2="D:/social_mental/data2.xlsx"
data2=read_excel(data_path2)
```

```
## New names:
## * `If yes, then did you lend any support to such person?` -> `If yes, then did
##   you lend any support to such person?...15`
## * `If yes, then did you lend any support to such person?` -> `If yes, then did
##   you lend any support to such person?...17`
```

```
data_3_path="D:/social_mental/data3.csv"
data3=read.csv(data_3_path)
```

Preprocessing

Operations on Data1

Checking and removing Null values

```
times_used=data1$`Frequency of Social Media Interaction`
summary(times_used)
```

```
##    Length     Class      Mode
##       300 character character
```

```
null_values <-is.na(times_used)
num_null_values <- sum(null_values)
num_null_values  #0 null values
```

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

Converting categorical values into numerical values

```
category_mapping=c("Frequently"=4,"Very Often"=3,"Occasionally"=2,"Rarely"=1)
freq=category_mapping[times_used]
```

Binding the newly created columns to the data1

```
data1_final <- cbind(data1,freq)
```

Operations on Data2

Converting text to numericals

```
k=data2$`How much time did you spend in Social media during the Lockdown?`
freq_mapping=c("Less than 2 hours"=1,"2-3 hours"=2,"3-6 hours"=4,"6-8 hours"=7)
freq=freq_mapping[k]
data2_final <- cbind(data2,freq)
View(data2_final)
```

Assigning age range a fixed value

```
age_check=c("18-21 years old"=19,"22-30 years old"=26,"41-60 years old"=50,"31-40 years old"=35)
l=data2$`Age Range`
Age <-age_check[l]
data2_final <- cbind(data2_final,Age)
```

Operations on Data3

Adding a frequency column and binding gender column of data2

```
fre_mapping=c("Between 2 and 3 hours"=2,"Between 3 and 4 hours"=3,"More than 5 hours"=5,"Less than an H
k<-data3$X8..What.is.the.average.time.you.spend.on.social.media.every.day.
#k
#p=data3$X9..How.often.do.you.find.yourself.using.Social.media.without.a.specific.purpose.
#data3_final <- subset(data3_final,select=-freq)
freq=fre_mapping[k]
data3_final <- cbind(data3,freq)
View(data3_final)
o=data3$X1..What.is.your.age.
colnames(data3_final)
```

```
##  [1] "Timestamp"
##  [2] "X1..What.is.your.age."
##  [3] "X2..Gender"
##  [4] "X3..Relationship.Status"
##  [5] "X4..Occupation.Status"
##  [6] "X5..What.type.of.organizations.are.you.affiliated.with."
##  [7] "X6..Do.you.use.social.media."
##  [8] "X7..What.social.media.platforms.do.you.commonly.use."
##  [9] "X8..What.is.the.average.time.you.spend.on.social.media.every.day."
## [10] "X9..How.often.do.you.find.yourself.using.Social.media.without.a.specific.purpose."
## [11] "X10..How.often.do.you.get.distracted.by.Social.media.when.you.are.busy.doing.something."
## [12] "X11..Do.you.feel.restless.if.you.haven.t.used.Social.media.in.a.while."
## [13] "X12..On.a.scale.of.1.to.5..how.easily.distracted.are.you."
## [14] "X13..On.a.scale.of.1.to.5..how.much.are.you.bothered.by.worries."
## [15] "X14..Do.you.find.it.difficult.to.concentrate.on.things."
## [16] "X15..On.a.scale.of.1.5..how.often.do.you.compare.yourself.to.other.successful.people.through.th
```

```
## [17] "X16..Following.the.previous.question..how.do.you.feel.about.these.comparisons..generally.speaki
## [18] "X17..How.often.do.you.look.to.seek.validation.from.features.of.social.media."
## [19] "X18..How.often.do.you.feel.depressed.or.down."
## [20] "X19..On.a.scale.of.1.to.5..how.frequently.does.your.interest.in.daily.activities.fluctuate."
## [21] "X20..On.a.scale.of.1.to.5..how.often.do.you.face.issues.regarding.sleep."
## [22] "freq"
```

```
data3_final$Age=data3$X1..What.is.your.age.
#gender
Gender=data3$X2..Gender
data3_final$Gender=data3$X2..Gender
write.csv(data3_final,file="D:/social_mental/data3_final.csv",row.names = FALSE)
```

Finding frequency of males and females

```
gaf1=data.frame(data1_final$Age,data1_final$Gender,data1_final$freq)

names(gaf1)<- c('age', 'gender', 'freq')
gaf2=data.frame(data2_final$Age,data2_final$Gender,data2_final$freq)

names(gaf2)<- c('age', 'gender', 'freq')
gaf3=data.frame(data3_final$Age,data3_final$Gender,data3_final$freq)
names(gaf3)<- c('age', 'gender', 'freq')

GAF <- rbind(gaf1,gaf2,gaf3)
x<-7
GAF_filtered <- GAF[GAF$freq == x, ]
write.csv(GAF_filtered,file="D:/social_mental/my_dataset.csv",row.names = FALSE)

x <- max(GAF_filtered$freq)
cat("maX frequency:", x, "\n")
```

```
## maX frequency: 7
```

```
# Filter the dataset to get only the rows with the highest frequency count
highest_freq_data <- subset(GAF_filtered, freq == x)


# Count the number of males and females with the highest frequency count
num_males <- sum(highest_freq_data$gender == "Male")
num_females <- sum(highest_freq_data$gender == "Female")

# Print the number of males and females with the highest frequency count
print(paste("Number of males with the highest frequency count:", num_males))
```

```
## [1] "Number of males with the highest frequency count: 16"
```

```
print(paste("Number of females with the highest frequency count:", num_females))
```

```
## [1] "Number of females with the highest frequency count: 9"
```

```r
print(x)
```

```
## [1] 7
```

```r
h_f_c<-data.frame(num_males,num_females)

write.csv(h_f_c,file="D:/social_mental/hfc.csv",row.names = FALSE)
```

Operations using all datasets

Finding count of Social media platform users

```r
app_use=data2$`What Social Media Platforms do you use?`

app_usage=data.frame(app_use)

app_usage <- app_usage$app_use

arr <- c("Facebook", "Whatsapp", "Instagram", "Snapchat", "Telegram", "Discord", "YouTube")
count_vector <- rep(0, length(arr))
names(count_vector) <- arr
```

Finding the count of each app users

```r
data_app_gender <- data.frame(Gender = data2$Gender, app_usage = app_usage)
arr <- c("Facebook", "Whatsapp", "Instagram", "Snapchat", "Telegram", "Discord", "YouTube")
genders <- unique(data_app_gender$Gender)
count_matrix <- matrix(0, nrow = length(genders), ncol = length(arr), dimnames = list(genders, arr))

# Iterate over each value in the data frame "data_app_gender"
for (i in 1:nrow(data_app_gender)) {
  apps_used <- unlist(strsplit(as.character(data_app_gender$app_usage[i]), ", "))
  gender <- data_app_gender$Gender[i]
  for (j in 1:length(arr)) {
    if (arr[j] %in% apps_used) {
      count_matrix[gender, arr[j]] <- count_matrix[gender, arr[j]] + 1
    }
  }
}

# Print the counts for each gender and platform
print(count_matrix["Male",])
```

```
##  Facebook  Whatsapp Instagram  Snapchat  Telegram   Discord   YouTube
##        62        74        50        14        34        16        65
```

```r
print(count_matrix["Female",])
```

```
##   Facebook   Whatsapp  Instagram   Snapchat   Telegram    Discord    YouTube
##         43         61         37         15         21          8         47
```

```r
print(count_matrix)
```

```
##        Facebook Whatsapp Instagram Snapchat Telegram Discord YouTube
## Male         62       74        50       14       34      16      65
## Female       43       61        37       15       21       8      47
```

```r
gender_app_freq=data.frame(count_matrix)
View(gender_app_freq)
write.csv(gender_app_freq,file="D:/social_mental/gender_app_freq.csv",row.names = TRUE)
```

Finding frequency of age

```r
age_freq1=cbind(data1_final$Age,data1_final$freq)
age_freq2=cbind(data2_final$Age,data2_final$freq)
age_freq3=cbind(data3_final$Age,data3_final$freq)
age_freq=rbind(age_freq1,age_freq2,age_freq3)

colnames(age_freq)<-c("Age","freq")
write.csv(age_freq,file="D:/social_mental/age_freq.csv",row.names = TRUE)
```

Finding the overall impact of social media usage on mental health

Here, we are calculating means for all datasets and applying preprocessing techniques to obtain gender and age proportions from the datasets.

```r
a_g_m_1=data.frame(data1$Age,data1$Gender,data1$`Impact on Mental Health (Score)`)
View(a_g_m_1)
assign_label <- function(value){
  if(value<temp_1){
    return ("negative")

  }
  else{
    return ("positive")
  }
}
temp_1 <- mean(a_g_m_1$data1..Impact.on.Mental.Health..Score..)#---------------------------------------
temp_1
```

```
## [1] 3.621333
```

```r
a_g_m_1$mental_health=sapply(a_g_m_1$data1..Impact.on.Mental.Health..Score..,assign_label)
View(a_g_m_1)
a_g_m_1_final <- a_g_m_1[,c("data1.Age","data1.Gender","mental_health")]
colnames(a_g_m_1_final) <-c("age","gender","mental_health")
View(a_g_m_1_final)

a_g_m_2=data.frame(data2_final$Age,data2_final$Gender,data2$`Have you faced a Mental health Issue?`)
View(a_g_m_2)
assign_label2 <- function(value){
  if(value=="Yes"){
    return ("positive")
  }
  else if(value=="No"){
    return ("negative")
  }
  else{
    return("NA")
  }
}
a_g_m_2$mental_health<- sapply(data2$`Have you faced a Mental health Issue?`, assign_label2)
View(a_g_m_2)
#remove na values
a_g_m_2_final <-na.omit(a_g_m_2)
View(a_g_m_2_final)
a_g_m_2_final<- a_g_m_2[a_g_m_2$mental_health%in% c("positive", "negative"), ]
a_g_m_2_final <- a_g_m_2_final[,c("data2_final.Age","data2_final.Gender","mental_health")]
colnames(a_g_m_2_final) <- c("age", "gender", "mental_health")
View(a_g_m_2_final)

a_g_m_3=data.frame(data3$X1..What.is.your.age.,data3$X2..Gender,data3$X18..How.often.do.you.feel.depres
View(a_g_m_3)
assign_label3 <- function(value){
  if(value>temp_3){
    return("positive")
  }
  else{
    return("negative")
  }
}
temp_3 = mean(data3$X18..How.often.do.you.feel.depressed.or.down.)#-----------------------------------
temp_3
```

```
## [1] 3.255717
```

```r
colnames(a_g_m_3)<-c("age","gender","mental_health")
a_g_m_3$mental_health=sapply(a_g_m_3$mental_health,assign_label3)
View(a_g_m_3)
a_g_m_3_final=data.frame(a_g_m_3$age,a_g_m_3$gender,a_g_m_3$mental_health)
View(a_g_m_3_final)
colnames(a_g_m_3_final) <- c("age","gender","mental_health")
data3_final$mental_health=a_g_m_3_final$mental_health
View(data3_final)
Age_gender_mental <- rbind(a_g_m_1_final,a_g_m_2_final,a_g_m_3_final)
```

```r
View(Age_gender_mental)
#---------------------------------------------------------------------------------------------------t
Age_gender_mental_final <- Age_gender_mental[Age_gender_mental$gender %in% c("Male", "Female"), ]
View(Age_gender_mental_final)
summary(Age_gender_mental)
```

```
##       age           gender          mental_health
##  Min.   :13.00   Length:897         Length:897
##  1st Qu.:21.00   Class :character   Class :character
##  Median :26.00   Mode  :character   Mode  :character
##  Mean   :29.55
##  3rd Qu.:36.00
##  Max.   :91.00
```

```r
write.csv(Age_gender_mental_final,file="D:/social_mental/Age_gender_mental.csv",row.names = FALSE)
unique_values<- unique(Age_gender_mental_final$mental_health)
print(length(unique_values))
```

```
## [1] 2
```

```r
unique_values
```

```
## [1] "negative" "positive"
```

```r
count_mental_health <- table(Age_gender_mental$mental_health)
count_mental_health
```

```
##
## negative positive
##      463      434
```

```r
count_male_mental_health <- table(subset(Age_gender_mental, gender == "Male")$mental_health)
print("Counts of Mental Health for Males:")
```

```
## [1] "Counts of Mental Health for Males:"
```

```r
print(count_male_mental_health)
```

```
##
## negative positive
##      221      201
```

```r
cc<-data.frame(count_male_mental_health)
View(cc)


count_female_mental_health <- table(subset(Age_gender_mental, gender == "Female")$mental_health)
cc1<-data.frame(count_female_mental_health)
View(cc1)
```

```r
merged_counts <- cbind(male_positive = count_male_mental_health["positive"],
                       male_negative = count_male_mental_health["negative"],
                       female_positive = count_female_mental_health["positive"],
                       female_negative = count_female_mental_health["negative"])
View(merged_counts)
gender_counts <- data.frame(
  gender = c("Male", "Female"),
  positive = c(count_male_mental_health["positive"], count_female_mental_health["positive"]),
  negative = c(count_male_mental_health["negative"], count_female_mental_health["negative"])
)
View(gender_counts)
write.csv(gender_counts,file="D:/social_mental/gender_counts.csv",row.names = FALSE)
#-------------------------------------------------------------------------------------gender proportio
gender_proportions <- Age_gender_mental_final %>%
  group_by(gender) %>%
  summarise(prop_positive = mean(mental_health == "positive"))

# Print proportions by gender
print(gender_proportions)
```

```
## # A tibble: 2 x 2
##   gender prop_positive
##   <chr>          <dbl>
## 1 Female         0.487
## 2 Male           0.476
```

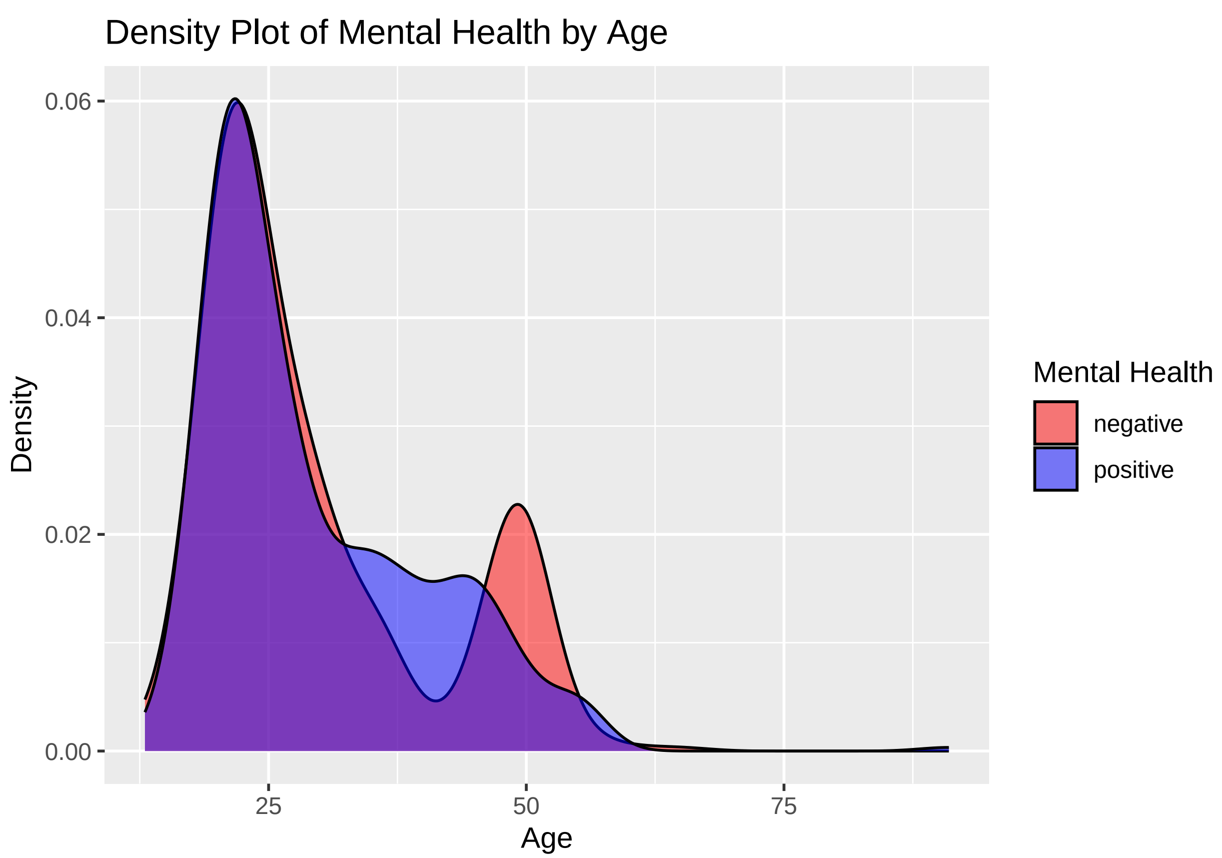
```r
g_p<-data.frame(gender_proportions)
View(g_p)
write.csv(g_p,file="D:/social_mental/g_p.csv",row.names = FALSE)

#-----------------------------------------------------------------------------------------------------
age_proportions <- Age_gender_mental_final %>%
  group_by(age) %>%
  summarise(prop_positive = mean(mental_health == "positive"))
#mean of all the values based

print(age_proportions)
```

```
## # A tibble: 46 x 2
##      age prop_positive
##    <dbl>         <dbl>
##  1    13         0
##  2    14         0.5
##  3    15         0.5
##  4    16         0
##  5    17         0.5
##  6    18         0.5
##  7    19         0.478
##  8    20         0.6
##  9    21         0.495
## 10    22         0.387
## # i 36 more rows
```

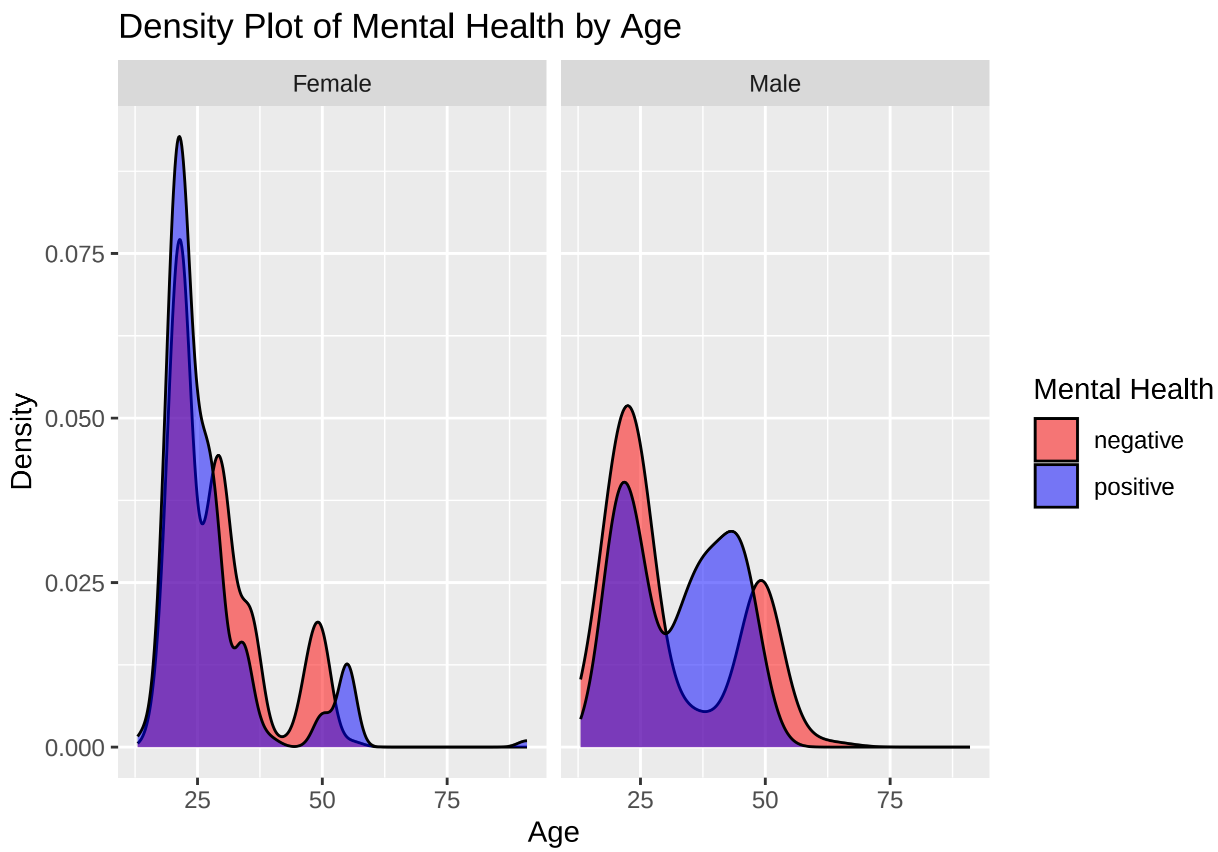
```r
age_prop<-data.frame(age_proportions)
View(age_prop)
write.csv(age_prop,file="D:/social_mental/age_prop.csv",row.names = FALSE)
# Define age ranges
age_ranges <- cut(age_proportions$age, breaks = c(0, 20, 30, 40, 50, 60, max(age_proportions$age)), labe

# Aggregate data by age range
age_range_proportions <- aggregate(prop_positive ~ age_range,
                                   data = data.frame(age_range = age_ranges, prop_positive = age_propor
                                   FUN = mean)

unique_levels <- unique(age_prop$prop_positive)

# Generate a color palette with sufficient colors
color_palette <- rainbow(length(unique_levels))
```

Plots

```r
ggplot(Age_gender_mental_final, aes(x = age, fill = mental_health)) +
  geom_density(alpha = 0.5) +
  scale_fill_manual(values = c("positive" = "blue", "negative" = "red")) +
  labs(title = "Density Plot of Mental Health by Age", x = "Age", y = "Density", fill = "Mental Health")
```

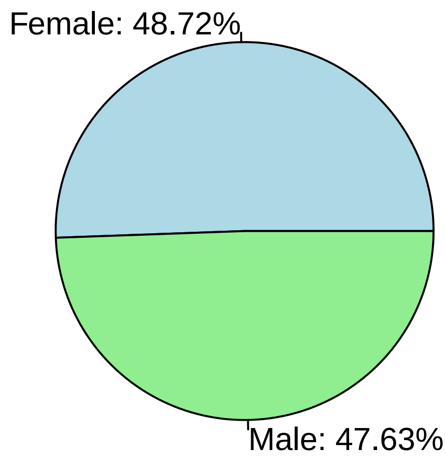


```r
ggplot(Age_gender_mental_final, aes(x = age, fill = mental_health)) +
  geom_density(alpha = 0.5) +
  scale_fill_manual(values = c("positive" = "blue", "negative" = "red")) +
  labs(title = "Density Plot of Mental Health by Age", x = "Age", y = "Density", fill = "Mental Health")
  facet_wrap(~ gender)
```




```r
pie(g_p$prop_positive,
    labels = sprintf("%s: %.2f%%", g_p$gender, g_p$prop_positive * 100),
    col = c("lightblue", "lightgreen"),
    main = "Proportion of Positive Mental Health by Gender")
```

Proportion of Positive Mental Health by Gender



Finding Age and Gender relationship

Here, we consider relationship status like single,relationship,married,divorced.

```
pd=data.frame(data3_final$X3..Relationship.Status,data3_final$Age,data3_final$Gender,data3_final$X7..Wha

colnames(pd)=c("relationshipStatus","Age","Gender","App")
write.csv(pd,file="D:/social_mental/age_gender_relation_app.csv",row.names = FALSE)
summary(pd)
```

```
##  relationshipStatus      Age            Gender              App
##  Length:481         Min.   :13.00   Length:481         Length:481
##  Class :character   1st Qu.:21.00   Class :character   Class :character
##  Mode  :character   Median :22.00   Mode  :character   Mode  :character
##                     Mean   :26.14
##                     3rd Qu.:26.00
##                     Max.   :91.00
```

```
unique(pd$Gender)
```

```
## [1] "Male"              "Female"            "Nonbinary "
## [4] "Non-binary"        "NB"                "unsure "
## [7] "Trans"             "Non binary "       "There are others???"
```

```r
unique(pd$relationshipStatus)   #-----------------------------------------------------------------
```

```
## [1] "In a relationship" "Single"            "Married"
## [4] "Divorced"
```

```r
pd <- pd[pd$Gender %in% c("Male", "Female", "Trans"), ]
```

relationship status of mentally affected people:

Single

```r
#-------------------------------------------------------------------------------------single
filtered_pd <- pd[pd$relationshipStatus %in% c("Single"), ]
val<-mean(filtered_pd$Age)

data <- subset(pd,relationshipStatus=="Single")
apps <- strsplit(trimws(data$App),",")
apps <- unlist(apps)
app_counts <- table(apps)
sorted_counts_single <- sort(app_counts,decreasing = TRUE)
n_s<-nrow(filtered_pd)
sorted_counts_single<-sorted_counts_single/n_s
print(sorted_counts_single)
```

```
## apps
##    Facebook     YouTube   Instagram     Discord    Snapchat   Pinterest
## 0.835714286 0.825000000 0.732142857 0.492857143 0.428571429 0.335714286
##      Reddit     Twitter      TikTok   Instagram     Twitter     YouTube
## 0.285714286 0.235714286 0.210714286 0.057142857 0.039285714 0.035714286
##      Reddit     Discord   Pinterest
## 0.014285714 0.010714286 0.007142857
```

```r
write.csv(sorted_counts_single,file="D:/social_mental/single_app.csv",row.names = FALSE)
```

Relationship

```r
#--------------------------------------------------------In a relationship
filtered_pd_relation <- pd[pd$relationshipStatus %in% c("In a relationship"), ]
val_relationship<-mean(filtered_pd_relation$Age)


data_relationship <- subset(pd,relationshipStatus=="In a relationship")
apps <- strsplit(trimws(data_relationship$App),",")

apps_relation <- unlist(apps)
app_counts_relation <- table(apps_relation )

sorted_counts_relation <- sort(app_counts_relation,decreasing = TRUE)
n_r<-nrow(filtered_pd_relation)
sorted_counts_relation <-sorted_counts_relation/n_r
write.csv(sorted_counts_relation,file="D:/social_mental/relation_app.csv",row.names = FALSE)
sorted_counts_relation
```

```
## apps_relation
##   Facebook    YouTube  Instagram   Snapchat    Discord  Pinterest    Twitter
## 0.91954023 0.86206897 0.82758621 0.49425287 0.45977011 0.25287356 0.25287356
##     Reddit     TikTok  Instagram    YouTube
## 0.24137931 0.21839080 0.04597701 0.03448276
```

Married

```r
#------------------------------------------------------------------Married
filtered_pd_marrige <- pd[pd$relationshipStatus %in% c("Married"), ]

val_marrige<-mean(filtered_pd_marrige$Age)

data_marrige <- subset(pd,relationshipStatus=="Married")
apps_marrige <- strsplit(trimws(data_marrige$App),",")

apps_marrige<- unlist(apps_marrige)
app_counts_marrige <- table(apps_marrige)
sorted_counts_marrige <- sort(app_counts_marrige,decreasing = TRUE)
n_m<-nrow(filtered_pd_marrige)
sorted_counts_marrige<-sorted_counts_marrige/n_m
print(sorted_counts_marrige)
```

```
## apps_marrige
##   Facebook    YouTube  Instagram  Pinterest    Twitter     Reddit    Discord
## 0.84158416 0.77227723 0.48514851 0.25742574 0.17821782 0.15841584 0.12871287
##   Snapchat     TikTok    Twitter  Instagram    YouTube    Discord     Reddit
## 0.12871287 0.11881188 0.07920792 0.02970297 0.02970297 0.00990099 0.00990099
```

```r
write.csv(sorted_counts_marrige,file="D:/social_mental/married_app.csv",row.names = FALSE)
```

Divorced

```r
#------------------------------------------------------------------Divorced
filtered_pd_Divorced <- pd[pd$relationshipStatus %in% c("Divorced"), ]

val_Divorced<- mean(filtered_pd_Divorced$Age)

data_Divorced <- subset(pd,relationshipStatus=="Divorced")
apps_Divorced <- strsplit(trimws(data_Divorced$App),",")
apps_Divorced<- unlist(apps_Divorced)
app_counts_Divorced <- table(apps_Divorced)
sorted_counts_Divorced <- sort(app_counts_Divorced,decreasing = TRUE)
print(sorted_counts_Divorced)
```

```
## apps_Divorced
##    YouTube   Facebook  Instagram    Twitter   Snapchat    Discord  Instagram
##          6          5          4          3          2          1          1
##     TikTok
##          1
```

```r
num_val<-nrow(filtered_pd_Divorced)
sorted_counts_Divorced <- sorted_counts_Divorced / num_val
df_sorted_counts_Divorced <- as.data.frame(sorted_counts_Divorced)


write.csv(df_sorted_counts_Divorced,file="D:/social_mental/divorced_app.csv",row.names = FALSE)
```

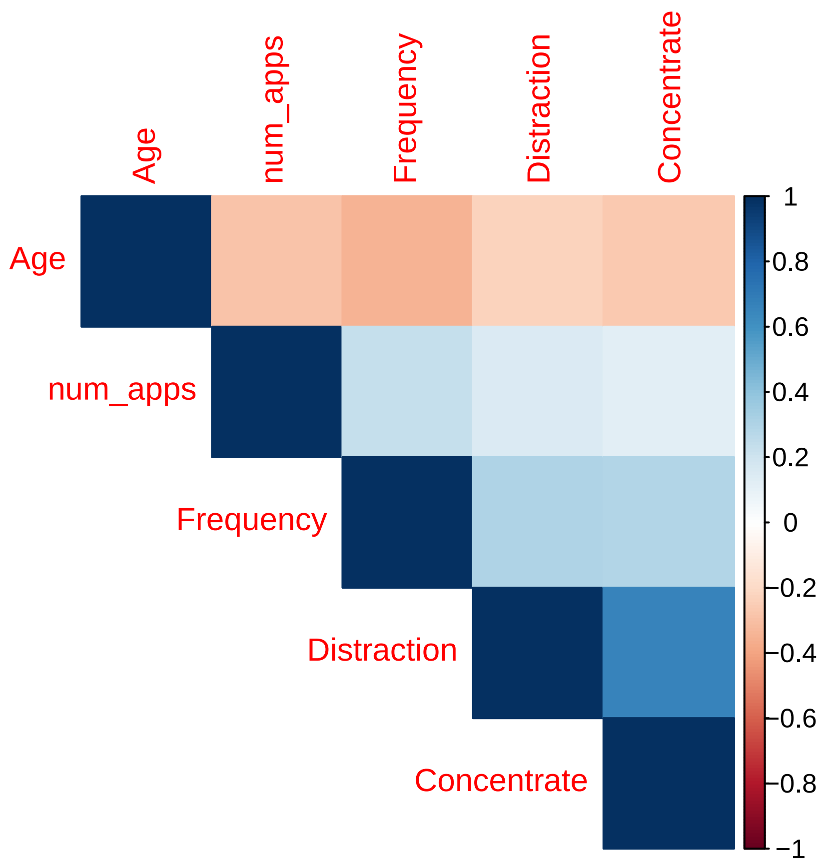
Model

Correlation matrix

```r
colnames(main_data) <- c("Age", "Gender", "Relation", "Apps", "Distraction", "Concentrate", "Frequency"
main_data$num_apps<-sapply(strsplit(main_data$Apps,","),length)
cor_matrix <- cor(main_data[, c("Age", "Distraction", "Concentrate", "Frequency", "num_apps")])

# View the correlation matrix
print(cor_matrix)
```

```
##                    Age Distraction Concentrate  Frequency   num_apps
## Age          1.0000000  -0.2223182  -0.2638628 -0.3455883 -0.2852126
## Distraction -0.2223182   1.0000000   0.6643524  0.3085896  0.1562933
## Concentrate -0.2638628   0.6643524   1.0000000  0.2905710  0.1252278
## Frequency   -0.3455883   0.3085896   0.2905710  1.0000000  0.2368250
## num_apps    -0.2852126   0.1562933   0.1252278  0.2368250  1.0000000
```

```r
corrplot(cor_matrix, method = "color", type = "upper", order = "hclust")
```



Logistic Regression model

```
main_data <- na.omit(main_data)
main_data$Gender <- factor(main_data$Gender, levels = c("Male", "Female","Trans"), labels = c(1, 0,2))
main_data$Relation<-factor(main_data$Relation,levels = c("In a relationship","Single","Married","Divorc
main_data$Mental_Health<-factor(main_data$Mental_Health,levels=c("positive","negative"),labels = c(1,0)

X<-main_data[,c("Age","Gender","Relation","Distraction","Concentrate","Frequency","num_apps")]
Y<-as.numeric(main_data$Mental_Health)-1
str(main_data$Mental_Health)
```

```
##  Factor w/ 2 levels "1","0": 1 1 1 1 1 2 1 1 1 2 ...
```

```
model <- glm(Mental_Health ~ Age + Gender + Relation + Distraction + Concentrate + Frequency + num_apps

model$rank
```

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

```
summary(model$coefficients)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -12.70913 -0.60007  -0.04307  -1.07799   0.02149   2.49830
```

Confusion matrix

```r
# Create confusion matrix
conf_matrix <- table(predict(model, type = "response") > 0.5, Y)
print("Confusion Matrix:")
```

```
## [1] "Confusion Matrix:"
```

```r
print(conf_matrix)
```

```
##        Y
##           0   1
##   FALSE 147  73
##   TRUE   70 185
```

Training and Testing the model

```r
set.seed(123)
train_index <- createDataPartition(main_data$Mental_Health, p = 0.8, list = FALSE)
train_data <- main_data[train_index, ]
test_data <- main_data[-train_index, ]

Y_train <- as.numeric(train_data$Mental_Health) - 1
Y_test <- as.numeric(test_data$Mental_Health) - 1

formula <- as.formula("Mental_Health ~ Age + Gender + Relation + Distraction + Concentrate + Frequency +
model <- glm(formula, data = train_data, family = binomial)
predicted_values <- predict(model, newdata = test_data, type = "response")

predicted_classes <- ifelse(predicted_values > 0.5, 1, 0)

accuracy <- mean(predicted_classes == Y_test)
print(paste("Accuracy:", accuracy))
```

```
## [1] "Accuracy: 0.723404255319149"
```

Confusion matrix

```r
conf_matrix <- confusionMatrix(factor(predicted_classes, levels = c(0, 1)), factor(Y_test, levels = c(0
conf_matrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 31 14
##          1 12 37
```

```
##
##                Accuracy : 0.7234
##                  95% CI : (0.6215, 0.8107)
##     No Information Rate : 0.5426
##     P-Value [Acc > NIR] : 0.0002465
##
##                   Kappa : 0.4448
##
##  Mcnemar's Test P-Value : 0.8445193
##
##             Sensitivity : 0.7209
##             Specificity : 0.7255
##          Pos Pred Value : 0.6889
##          Neg Pred Value : 0.7551
##              Prevalence : 0.4574
##          Detection Rate : 0.3298
##    Detection Prevalence : 0.4787
##       Balanced Accuracy : 0.7232
##
##        'Positive' Class : 0
##
```

Precision, Recall, f1 Score

```r
precision <- conf_matrix$byClass["Pos Pred Value"]
recall <- conf_matrix$byClass["Sensitivity"]



#-------------------------------------------------------------------metrics
f1_score <- 2 * (precision * recall) / (precision + recall)
metrics_df <- data.frame(
  Metric = c("Precision", "Recall", "F1 Score"),
  Value = c(precision, recall, f1_score)
)

# Print the data frame
print(metrics_df)
```

```
##      Metric     Value
## 1 Precision 0.6888889
## 2    Recall 0.7209302
## 3  F1 Score 0.7045455
```

plot

```r
Y_test
```

```
##  [1] 0 1 1 0 0 0 0 0 1 1 1 0 0 0 1 0 1 1 0 0 1 1 0 0 1 1 1 1 1 1 0 1 1 1 1 0 1 0
## [39] 0 1 1 1 1 1 1 0 1 0 1 0 0 0 1 0 0 1 0 0 1 0 1 1 0 1 0 0 1 0 1 0 0 0 1 1 0 0
## [77] 1 0 0 1 1 0 1 1 1 1 0 0 1 0 1 1 1 1
```

```r
predicted_values
```

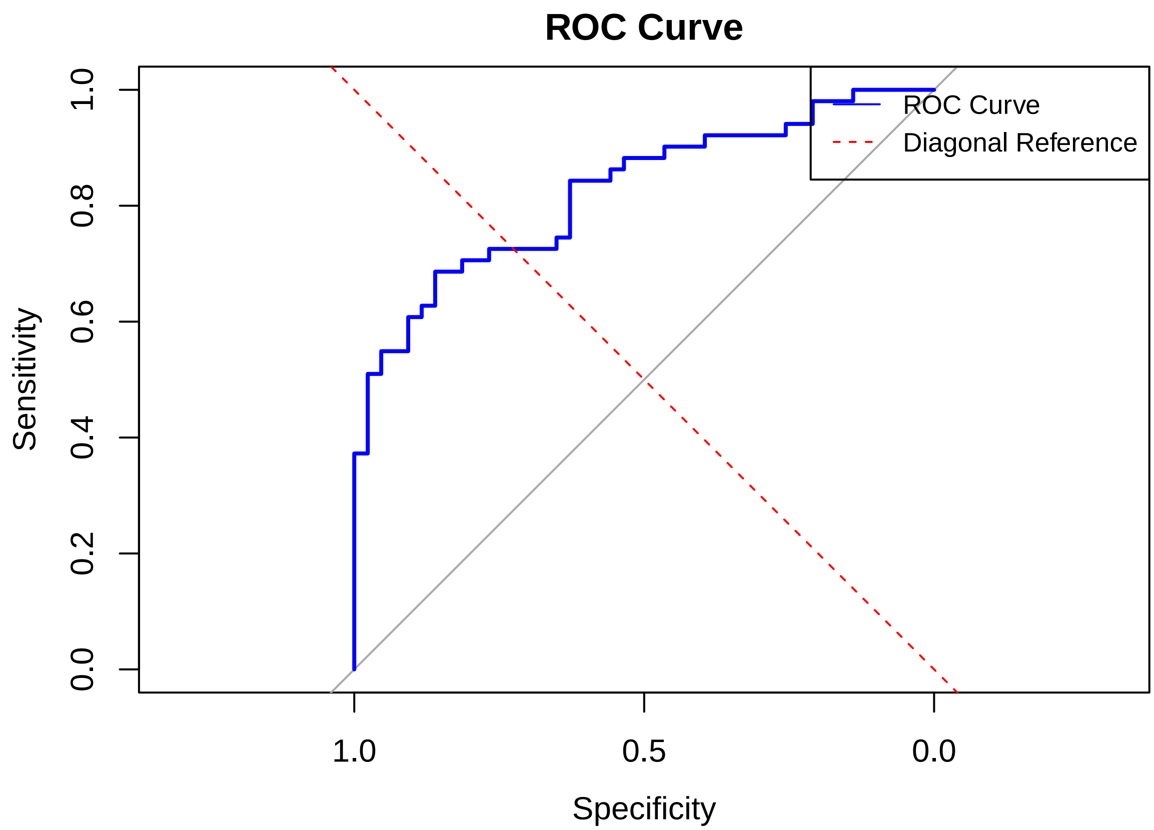
```
##         3         6        14        15        23        27        30        31
## 0.3334071 0.5187699 0.3189007 0.3019944 0.3636274 0.4879847 0.3550046 0.3500639
##        38        39        59        60        61        63        65        67
## 0.3514082 0.4072992 0.7767904 0.2919073 0.5559435 0.6016804 0.8244967 0.6845923
##        72        74        75        90       112       114       120       124
## 0.9437956 0.8970899 0.8017481 0.2799222 0.8250542 0.2882469 0.1858727 0.4029782
##       127       130       133       134       135       140       141       154
## 0.7722901 0.8991033 0.8367885 0.9012668 0.7894472 0.8300202 0.5797561 0.5619075
##       158       167       170       174       178       189       191       203
## 0.9475898 0.4072243 0.7037683 0.2441420 0.9197430 0.3166928 0.4916944 0.6387643
##       210       213       214       219       233       242       245       246
## 0.8803690 0.9510355 0.7481094 0.5677580 0.9397133 0.3426304 0.4409516 0.4029782
##       256       272       274       275       277       285       290       292
## 0.3781494 0.3547088 0.3165281 0.5312071 0.4239670 0.7334151 0.3853685 0.3894007
##       299       300       308       311       312       313       315       320
## 0.4061591 0.2947001 0.2769561 0.2571150 0.2668590 0.8391487 0.1827954 0.7964218
##       321       325       326       327       328       331       333       338
## 0.2861687 0.5174474 0.8290824 0.6937497 0.8735140 0.2651639 0.2291517 0.5242995
##       342       356       361       363       367       370       372       380
## 0.9462725 0.5851900 0.2013354 0.1471927 0.8016645 0.4479896 0.3074061 0.5417275
##       381       384       385       393       397       405       410       413
## 0.6051586 0.1992271 0.5651051 0.6475345 0.4383651 0.2359767 0.4881806 0.5089284
##       419       426       432       439       458       462
## 0.8035028 0.5465048 0.4789937 0.7214513 0.8009810 0.8806404
```

```r
roc_curve <- roc(Y_test, predicted_values)
```

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

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

```r
plot(roc_curve, main = "ROC Curve", col = "blue", lwd = 2)
abline(a = 0, b = 1, lty = 2, col = "red")
legend("topright", legend = c("ROC Curve", "Diagonal Reference"), col = c("blue", "red"), lty = 1:2, ce
```



Decision tree

Training and Testing

```
formula <- Mental_Health ~ Age + Gender + Relation + Distraction + Concentrate + Frequency
fit <- rpart(formula,data=main_data,cp=0.01)


train_index <- createDataPartition(main_data$Mental_Health, p = 0.8, list = FALSE)
train_data <- main_data[train_index, ]
test_data <- main_data[-train_index, ]

Y_train <- as.numeric(train_data$Mental_Health) - 1
Y_test <- as.numeric(test_data$Mental_Health) - 1

train_pred <- predict(fit, newdata = train_data, type = "class")

# Predicting on testing data
test_pred <- predict(fit, newdata = test_data, type = "class")
test_pred
```

```
##   1   7   8  23  36  44  56  67  76  78  82  85 105 110 112 113 118 123 124 126
##   1   0   0   1   0   0   0   1   1   1   1   1   0   0   0   1   0   1   1   0
```

```
## 128 145 151 153 154 157 161 168 172 174 179 182 190 191 194 196 200 201 204 205
##   0   0   0   0   0   0   0   1   0   1   0   0   1   1   0   0   1   1   1   0
## 206 207 214 215 218 219 224 251 259 260 271 272 277 280 283 288 289 290 295 297
##   1   0   0   1   0   0   1   0   0   1   1   1   1   1   0   0   1   1   1   0
## 304 306 307 313 318 320 326 328 329 331 335 336 339 346 355 360 366 372 374 387
##   0   0   1   0   1   0   0   0   0   1   1   1   0   1   0   0   1   1   0   1
## 388 393 398 399 402 404 427 434 438 456 457 462 466 469
##   1   0   0   0   1   0   1   0   1   1   1   0   1   1
## Levels: 1 0
```

Performance

```
train_accuracy <- mean(train_pred == Y_train)


# Calculate accuracy for testing data
test_accuracy <- mean(test_pred == Y_test)

confusion_test <- table(Actual = Y_test, Predicted = test_pred)

precision_test <- confusion_test[2, 2] / sum(confusion_test[, 2])
recall_test <- confusion_test[2, 2] / sum(confusion_test[2, ])
f1_score_test <- 2 * precision_test * recall_test / (precision_test + recall_test)


predictions <- predict(fit, test_data, type = "prob")

prediction_obj <- prediction(predictions[,2], Y_test)
results_df <- data.frame(
  Metric = c("Training Accuracy", "Testing Accuracy", "Testing Precision", "Testing Recall", "Testing F1
  Value = c(train_accuracy, test_accuracy, precision_test, recall_test, f1_score_test)
)

# Print the data frame
print(results_df)
```

```
##              Metric     Value
## 1 Training Accuracy 0.2939633
## 2  Testing Accuracy 0.2127660
## 3 Testing Precision 0.8163265
## 4    Testing Recall 0.7843137
## 5  Testing F1 Score 0.8000000
```

```
# Create ROC curve
roc_curve <- performance(prediction_obj, "tpr", "fpr")

# Plot ROC curve
plot(roc_curve, main = "ROC Curve for Decision Tree Model", col = "blue", lwd = 2,
     xlab = "False Positive Rate (1 - Specificity)", ylab = "True Positive Rate (Sensitivity)",
     xlim = c(0, 1), ylim = c(0, 1))
abline(a = 0, b = 1, col = "red", lwd = 2, lty = 2)
```

```r
legend("bottomright", legend = c("ROC Curve", "Random Classifier"),
       col = c("blue", "red"), lty = c(1, 2), lwd = c(2, 2))
```