**NAME :** D. BHANU SRI **GROUP :** DSSTCS **YEAR :** 2ND YEAR  
**COLLEGE :** ADITYA WOMEN’S DEGREE COLLEGE RAJAHMUNDRY



**DATA ANALYSIS   
ON  
TOURISM WEBSITE ENGAGEMENT**

* **BEGINNING OF THE PROJECT:**

Before starting the actual data analysis there are few steps to follow. There are :

**STEP 1:** First load the dataset into the kernal or studio.

**STEP 2**:Import all the libraries and packages used for data analysis.

**STEP 3:** Then clean and prepare the data for actual analysis. It again contains many steps like removing duplicate values, finding out null values and replacing it with mean values or most frequent values.

* Also we can find the unique values in the columns.
* We should find the number of rows and columns in the given dataset.
* We should find out the basic statistical information about the data by using describe().

***SOURCE CODE:***

library(dplyr) #Used for data manipulation

library(ggplot2) # Used for Elegant data visualization and Grammer of Graphics

library(plotly) #Create interactive Web graphics and used for data visualization

library(forcats) # Tools for working with Categorical Data

library(viridis) #To have different colors in plots

library(moments) #To find skewness and kurtosis

library(cowplot) # Plot Theme and plot Annotations for ggplot2

###**TOURISM DATASET**

##1.Loading the data into R studio:

#Data Collection:

df = read.csv(file.choose(), header = T)

df

##To view dataset in another file

View(df)

**ABOUT THE DATASET:**

#**UserID**: Unique ID of the user

#**Buy\_ticket**: Buy a ticket in the next month (target variable)

#**Yearly\_avg\_view\_on\_travel\_page**: Average yearly views on any travel-related page by the user

#**preferred\_device**:Preferred device for user login

#**total\_likes\_on\_outstation\_checking\_given:** Total number of likes given by the user on out-of-station check-ins in the last year

#**yearly\_avg\_Outstation\_checkins**:Average number of out-of-station check-ins done by the user

#**member\_in\_family:**Total number of relationships mentioned by the user in the account

#**preferred\_location\_type:**Preferred type of location for traveling by the user

#**Yearly\_avg\_comment\_on\_travel\_page**:Average yearly comments on any travel-related page by the user

#**total\_likes\_on\_outofstation\_checkin\_received**: Total number of likes received by the user on out-of-station check-ins in the last year

#**week\_since\_last\_outstation\_checkin**:Number of weeks since the last out-of-station check-in update by the user

#**following\_company\_page**:Whether the customer is following the company page (Yes or No)

#**montly\_avg\_comment\_on\_company\_page**:Average monthly comments on the company page by the user

#working\_flag: Whether the customer is working or not

#**travelling\_network\_rating**:The rating indicating if the user has close friends who also like traveling. 1 is high, 4 is lowest

#**Adult\_flag**: Whether the customer is an adult or not

#**Daily\_Avg\_mins\_spend\_on\_traveling\_page :**Average time spent on the company's travel page by the user

**BASIC OPERATION TO KNOW ABOUT THE DATA**

#To find top rows in dataset

head(df)

#To find last rows in dataset

tail(df)

##To know the rows and columns in dataset (dimensions)

dim(df)

**##To find out if there are any null values in the dataset**

is.na(df)

**##To find out null values count in every column in the dataset**

colSums(is.na(df))

**#To find the duplicate values in the given dataset**

sum(duplicated(df$UserID))

**#As there is no duplicate values in UserID means there is no repetition of same customer**

**#To find the datatype of each column in the dataset**

str(df) ##This functions shows data type along with some data in the column like structure

c<-sapply(df,class)

c

**##To fill missing values with 'NA' in a dataset**

df[df=='']<-NA

print(df)

**#To know any unique values present in the given dataset**

unique(df$Taken\_product) #It should have only two values-"YES" or "NO"

unique(df$preferred\_device) #It should have 7 categories like ios,Android,others,Laptop,Mobile,ios and android and tab

unique(df$member\_in\_family) #In given data there are maximum range of 10 members in a family

unique(df$preferred\_location\_type) #There are mainly 14 categories in this column

unique(df$following\_company\_page) #It should has only two values like - "YES" or "NO"

unique(df$working\_flag) #It should has only two values like - "YES" or "NO"

unique(df$travelling\_network\_rating) #It should have the range of 1 to 4

unique(df$Adult\_flag) ##It should have only 2 values 0 and 1.

**DATA CLEANING:**

Data cleaning, also known as data cleansing or data scrubbing, is the process of identifying and correcting (or removing) errors, inconsistencies, and inaccuracies in datasets. The goal of data cleaning is to improve the quality of data, ensuring that it is accurate, reliable, and suitable for analysis. Data cleaning is a crucial step in the data preparation process and is essential for obtaining meaningful and reliable insights from the data.  
**Key aspects of data cleaning include:**

1. **Handling Missing Values**
2. **Correcting Inaccuracies**
3. **Dealing with Duplicates**
4. **Standardizing Formats**
5. **Handling Outliers**
6. **Converting Data Types**

***SOURCE CODE:***

#Transforming each column data into its actual/preferred data values

#To replace NA values in this column with mean of the column

**Yearly\_avg\_view\_on\_travel\_page :**

df$Yearly\_avg\_view\_on\_travel\_page[is.na(df$Yearly\_avg\_view\_on\_travel\_page)]<-mean(df$Yearly\_avg\_view\_on\_travel\_page,na.rm=TRUE)

df$Yearly\_avg\_view\_on\_travel\_page <- floor(df$Yearly\_avg\_view\_on\_travel\_page)

##To convert the data values into proper lower case in this column

**preferred\_device column:**

table(df$preferred\_device) #To know the count of the datavalues in a column

df$preferred\_device[df$preferred\_device == 'ANDROID'] <- 'Android'

df$preferred\_device[df$preferred\_device == 'Other'] <- 'Others'

df$preferred\_device[df$preferred\_device == 'Android OS'] <- 'Android'

table(df$preferred\_device) #To check once again

unique(df$preferred\_device) #To check whether there are still any NA values in the column

df

sum(is.na(df$preferred\_device)) #To check no.of NA values in this column

# Find the indices of NA values in preferred\_device column

na\_indices <- which(is.na(df$preferred\_device))

na\_indices

# Determine the replacement values

replacement\_values <- c("Android", "iOS", "iOS and Android", "Mobile","Tab","Laptop","Others") # Add more values as needed

# Specify the number of replacement values in each group

replacement\_count <- 7

#As there are 53 NA values in preferred\_device column then I replaced every 7 NA values with all the different categories in this column

# Loop through replacement\_values

start\_index <- 1

for (replacement\_value in replacement\_values) {

# Replace NA values with the current replacement value

df$preferred\_device[na\_indices[start\_index:(start\_index + replacement\_count - 1)]] <- replacement\_value

# Update the starting index for the next set of replacement values

start\_index <- start\_index + replacement\_count

# Break the loop if all NA values are replaced

if (start\_index > length(na\_indices)) {

break

}

}

#Again check no.of NA values in the column

sum(is.na(df$preferred\_device))

table(df$preferred\_device)

#As 4 NA values left in this column we will replace the values with more frequent value "Tab"

your\_value <- 'Tab'

num\_na\_to\_replace <- 4

# Find the last 4 NA values in the column

last\_na\_indices <- tail(which(is.na(df$preferred\_device)), n = num\_na\_to\_replace)

# Replace the NA values with the specified value

df$preferred\_device[last\_na\_indices] <- your\_value

#So the preferred\_device column in data set cleaned.

**total\_likes\_on\_outstation\_checkin\_given:**

#Cleaning of total\_likes\_on\_outstation\_checkin\_given column

#To find NA values in Yearly\_avg\_view\_on\_travel\_page column

sum(is.na(df$total\_likes\_on\_outstation\_checkin\_given))

#We can replace all the 381 NA values with the mean of this column so that there will be no change in the data

df$total\_likes\_on\_outstation\_checkin\_given[is.na(df$total\_likes\_on\_outstation\_checkin\_given)]<-mean(df$total\_likes\_on\_outstation\_checkin\_given,na.rm=TRUE)

#As there is no use of float values in likes column. So we will remove the decimal values.

df$total\_likes\_on\_outstation\_checkin\_given <- as.numeric(df$total\_likes\_on\_outstation\_checkin\_given)

df$total\_likes\_on\_outstation\_checkin\_given <- floor(df$total\_likes\_on\_outstation\_checkin\_given)

**yearly\_avg\_Outstation\_checkins:**

#Cleaning of yearly\_avg\_Outstation\_checkins column

sum(is.na(df$yearly\_avg\_Outstation\_checkins))

#To check the indices of NA values in this column

na\_indices<-which(is.na(df$yearly\_avg\_Outstation\_checkins))

na\_indices

#table function is used to show the count of each unique value in that column

table(df$yearly\_avg\_Outstation\_checkins)

#To convert '\*' value into NA

df$yearly\_avg\_Outstation\_checkins[df$yearly\_avg\_Outstation\_checkins == "\*"] <- NA

#As the given column is already the average no.of out station checkins, So we will replace the NA values with the mode of same column

# Calculate the mode of the 'yearly\_avg\_Outstation\_checkins' column

mode\_value <- as.numeric(names(sort(table(df$yearly\_avg\_Outstation\_checkins), decreasing = TRUE)[1]))

# Replace NA values in the 'yearly\_avg\_Outstation\_checkins' column with the mode value

df$yearly\_avg\_Outstation\_checkins[is.na(df$yearly\_avg\_Outstation\_checkins)] <- mode\_value

#To check if there are any NA values left or not

sum(is.na(df$yearly\_avg\_Outstation\_checkins))

**member\_in\_family:**

#There is no NA value in the "member\_in\_family" column

table(df$member\_in\_family)

sum(is.na(df$member\_in\_family))

#replacing "Three" with 3

df$member\_in\_family[df$member\_in\_family == "Three"] <- 3

#To check again if there "Three" is added to 3 or not

table(df$member\_in\_family)

**preferred\_location\_type :**

sum(is.na(df$preferred\_location\_type))

table(df$preferred\_location\_type)

#To convert "Tour Travel" into "Tour and Travel" because they both are same

df$preferred\_location\_type[df$preferred\_location\_type=="Tour Travel"]<-"Tour and Travel"

unique(df$preferred\_location\_type)

#We will find the mode of the given data to replace NA values with it

freq\_table<-table(df$preferred\_location\_type)

mode\_value <- names(freq\_table)[which.max(freq\_table)]

cat("Mode:", mode\_value, "\n")

#To replace the NA values with the mode value i.e Beach

df$preferred\_location\_type[is.na(df$preferred\_location\_type)]<-mode\_value

sum(is.na(df$preferred\_location\_type))

**Yearly\_avg\_comment\_on\_travel\_page :**

sum(is.na(df$Yearly\_avg\_comment\_on\_travel\_page))# There are 206 NA values in the given dataset

#Calculating the mean value of Yearly\_avg\_comment\_on\_travel\_page column

# Calculate the mean without decimals

mean\_value <- round(mean(df$Yearly\_avg\_comment\_on\_travel\_page, na.rm = TRUE))

mean\_value #75 is mean value

# Replace NA values with the rounded mean

df$Yearly\_avg\_comment\_on\_travel\_page[is.na(df$Yearly\_avg\_comment\_on\_travel\_page)] <- mean\_value

#To check if any NA values left

na\_indices=which(is.na(df$Yearly\_avg\_comment\_on\_travel\_page))

na\_indices

**total\_likes\_on\_outofstation\_checkin\_received :**

#To check if there are any null values

sum(is.na(df$total\_likes\_on\_outofstation\_checkin\_received))

#To check if there are any negative values in the column

print(subset(df, total\_likes\_on\_outofstation\_checkin\_received <0)$total\_likes\_on\_outofstation\_checkin\_received)

**week\_since\_last\_outstation\_checkin :**

sum(is.na(df$week\_since\_last\_outstation\_checkin))

#To check if there are any negative values in the column

print(subset(df, week\_since\_last\_outstation\_checkin < 0)$week\_since\_last\_outstation\_checkin)

**following\_company\_page :**

sum(is.na(df$following\_company\_page))

table(df$following\_company\_page)

##To convert the data into mainly 2 categories i.e "Yes" or "No"

df$following\_company\_page[df$following\_company\_page=='Yeso']<-'Yes'

df$following\_company\_page[df$following\_company\_page==1]<-'Yes'

df$following\_company\_page[df$following\_company\_page==0]<-'No'

table(df$following\_company\_page)

**montly\_avg\_comment\_on\_company\_page :**

sum(is.na(df$montly\_avg\_comment\_on\_company\_page))

# Find the range of the column

column\_range <- range(df$montly\_avg\_comment\_on\_company\_page)

# Print the range

cat("Range of the column:", column\_range, "\n")

#There highest commenting range is 500 and lowest is 4

sum(is.na(df$montly\_avg\_comment\_on\_company\_page))

table(df$working\_flag)

#**TRANSFORMATION** of data into 0's and 1's:

df$working\_flag[df$working\_flag=='Yes']<-1

df$working\_flag[df$working\_flag=="No"]<-0

**travelling\_network\_rating :**

sum(is.na(df$travelling\_network\_rating))

# Find the range of the column

column\_range <- range(df$travelling\_network\_rating)

# Print the range

cat("Range of the column:", column\_range, "\n")

#The range of the friends who interested in tourism along with the user is from 1 to 10

**Adult\_flag :**

sum(is.na(df$Adult\_flag))

table(df$Adult\_flag)

#There is only 1 NA value and replacing it with 0

df$Adult\_flag[is.na(df$Adult\_flag)] <- 0

**OUTLIER DETECTION:**

#Finding out the outliers in the data

Q1 <- quantile(df$Adult\_flag, 0.25)

Q3 <- quantile(df$Adult\_flag, 0.75)

Q1

Q3

IQR <- Q3 - Q1

IQR

# Define the lower and upper bounds for potential outliers

lower\_bound <- Q1

upper\_bound <- Q3

# Identify potential outliers

outliers <- df$Adult\_flag < lower\_bound | df$Adult\_flag> upper\_bound

# Print the results

cat("Lower bound:", lower\_bound, "\n")

cat("Upper bound:", upper\_bound, "\n")

cat("Potential outliers:", df$Adult\_flag[outliers], "\n")

#Create a boxplot highlighting outliers

boxplot(df$Adult\_flag, main = "Boxplot with Outliers Highlighted", ylab = "Values", outline = TRUE)

points(which(outliers), df$Adult\_flag[outliers], col = "red", pch = 16)

#This plot shows us the outliers that is 2,3 and the IQR = 1 i.e Q2 median as the lower bound is 0.

#We will replace the outlier values with the main values i.e 1 and 0

df$Adult\_flag[df$Adult\_flag==2]<-"Outlier"

df$Adult\_flag[df$Adult\_flag==3]<-"Outlier"

##Now divide it with two

table\_result <- table(df$Adult\_flag)

outlier\_count <- table\_result["Outlier"]

# Divide the outlier count by 2

divided\_outlier\_count <- outlier\_count / 2

# Update the counts in the "0" and "1" categories

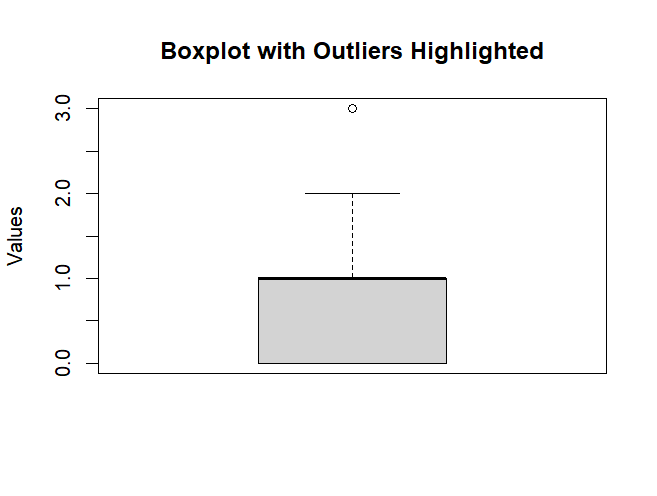
table\_result["0"] <- table\_result["0"] + divided\_outlier\_count

table\_result["1"] <- table\_result["1"] + divided\_outlier\_count

# Update the original data frame with the new counts

df$Adult\_flag[df$Adult\_flag == "Outlier"] <- rep(c("0", "1"), each = divided\_outlier\_count)

table(df$Adult\_flag)



**Daily\_Avg\_mins\_spend\_on\_traveling\_page :**

#To find out NA values

sum(is.na(df$Daily\_Avg\_mins\_spend\_on\_traveling\_page))

# Calculate the mean without decimals.

mean\_value <- round(mean(df$Daily\_Avg\_mins\_spend\_on\_traveling\_page, na.rm = TRUE))

# Print the range

column\_range <- range(df$Daily\_Avg\_mins\_spend\_on\_traveling\_page)

cat("Range of the column:", column\_range, "\n")

# Replace NA values with the rounded mean

df$Daily\_Avg\_mins\_spend\_on\_traveling\_page[is.na(df$Daily\_Avg\_mins\_spend\_on\_traveling\_page)] <- mean\_value

**DESCRIBE ():**

#To know the basic statistical values about all the columns in the given data set

describe(df)

**#ADDING STRING TO WHOLE COLUMN :**

##Adding "min" string to the Daily\_Avg\_mins\_spend\_on\_traveling\_page column

df$Daily\_Avg\_mins\_spend\_on\_traveling\_page<- paste(df$Daily\_Avg\_mins\_spend\_on\_traveling\_page,"min")

#To know the data type of column after adding string.

class(df$Daily\_Avg\_mins\_spend\_on\_traveling\_page)

**Changing of Column indices for our convenience**

# Specify the desired order of column indices

column\_indices <- c(1,2,12,4,8,3,5,6,7,9,10,11,13,14,15,16,17)

# Reorder the columns using column indices

df <- df[, column\_indices]

**NORMALIZATION:**

#Performing Normalization on week\_since\_last\_outstation\_checkin

df <- df %>%

mutate(normalized\_column = (week\_since\_last\_outstation\_checkin - min(week\_since\_last\_outstation\_checkin)) /

(max(week\_since\_last\_outstation\_checkin) - min(week\_since\_last\_outstation\_checkin)))

**STANDARDIZATION:**

#Performing Standardization on week\_since\_last\_outstation\_checkin

df <- df %>%

mutate(standardized\_column = (week\_since\_last\_outstation\_checkin - mean(week\_since\_last\_outstation\_checkin)) / sd(week\_since\_last\_outstation\_checkin))

**DATA VISUALIZATION:**

Data visualization is the graphical representation of data to help users understand patterns, trends, and insights within datasets. It involves creating visual representations, such as charts, graphs, and maps, to make complex data more accessible and comprehensible. The primary goal of data visualization is to communicate information clearly and effectively, allowing individuals to explore and interpret data visually.

Common types of data visualizations include:

* **Bar Charts and Column Charts:** Used to compare categorical data.
* **Line Charts:** Suitable for showing trends over time.
* **Scatter Plots:** Display relationships between two numerical variables.
* **Pie Charts:** Show the proportion of parts to a whole.
* **Heatmaps:** Visualize the intensity of data values in a matrix.
* **Histograms:** Depict the distribution of continuous data.
* **Box Plots:** Represent the distribution of data and identify outliers.
* **Maps:** Display spatial patterns and geographical data.

1.**PIE CHART :**

This pie chart representing no.of users who are willing to buy the ticket next month from the column “Taken Product” .

***SOURCE CODE:***

#As the Taken product is a target variable so, we can show the percentage of users going to buy ticket next month

counts <- table(df$Taken\_product)

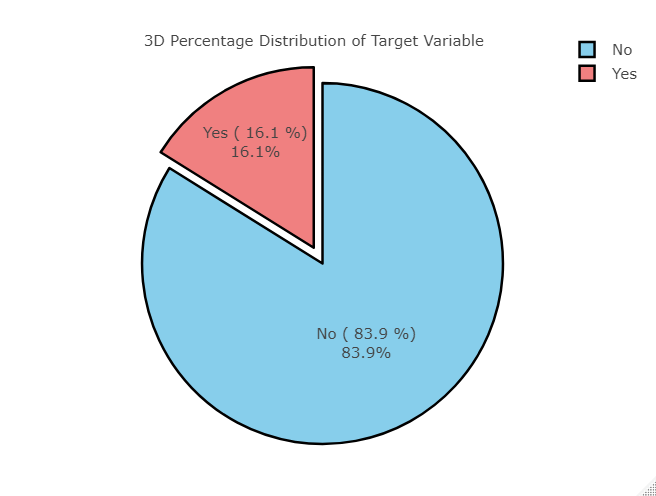
counts

# Create a 3D pie chart with plotly

pie\_chart\_3d <- plot\_ly(labels = names(counts), values = counts, type = 'pie', pull = c(0.1, 0), text = paste(names(counts), "(", round(prop.table(counts) \* 100, 1), "%)"),marker = list(colors = c("skyblue", "lightcoral"),line = list(color = 'black', width = 2)), textinfo = "text+percent",title = "3D Percentage Distribution of Target Variable")

#To plot a pie chart

print(pie\_chart\_3d)



**Observation:** By this we can say that there is only 16% of users who are interested in buying tickets and 89% of users are not interested in buying tickets.

2.**BAR PLOT :**

This Bar plot that represents the percentage of users who are following the Tourism page and who are not following this Tourism page.

***SOURCE CODE:***

count <- table(df$following\_company\_page)

count

# Create a bar plot

bar\_colors <- c("red", "green")

bp<-barplot(count, col = c("red", "green"), main = "Following Company page",

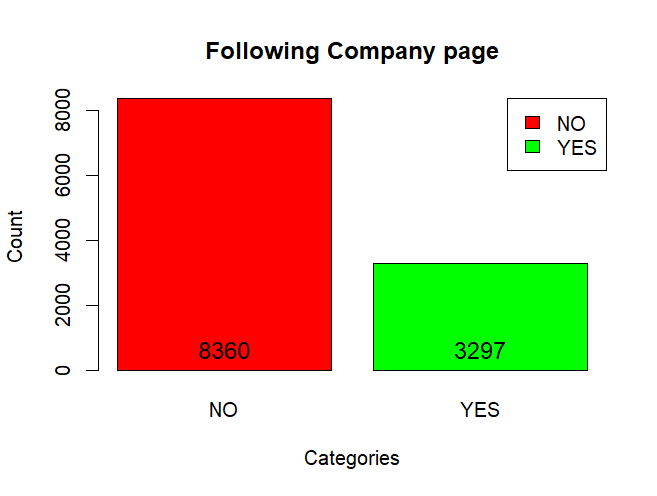
xlab = "Categories", ylab = "Count", names.arg = c("NO", "YES"))

# Add legend

legend("topright", legend = c("NO", "YES"), fill = bar\_colors,text(x = barplot(count) - 0.2, y = count + 0.2, labels = count))

#Add text labels for each bar with matching colors

text(bp,x = bp, y =bp, labels = count,pos = 3, col = "black", cex = 1.2)



**Observation:** In this bar plot we can see that the red bar indicates the number of users not following page and green bar indicates number of users are following company's page.

**3. BAR CHART:**

This Bar plot that represents the usage of different devices by the users of Tourism Website in order to login into this page.

***SOURCE CODE:***

# Create a bar plot

device\_counts = table(df$preferred\_device)

device\_counts

bar\_colors<-c('orange','forestgreen','slategrey','magenta','lightgreen','blue','purple')

bp<-barplot(device\_counts, col =bar\_colors , main = "Devices Used by Users",

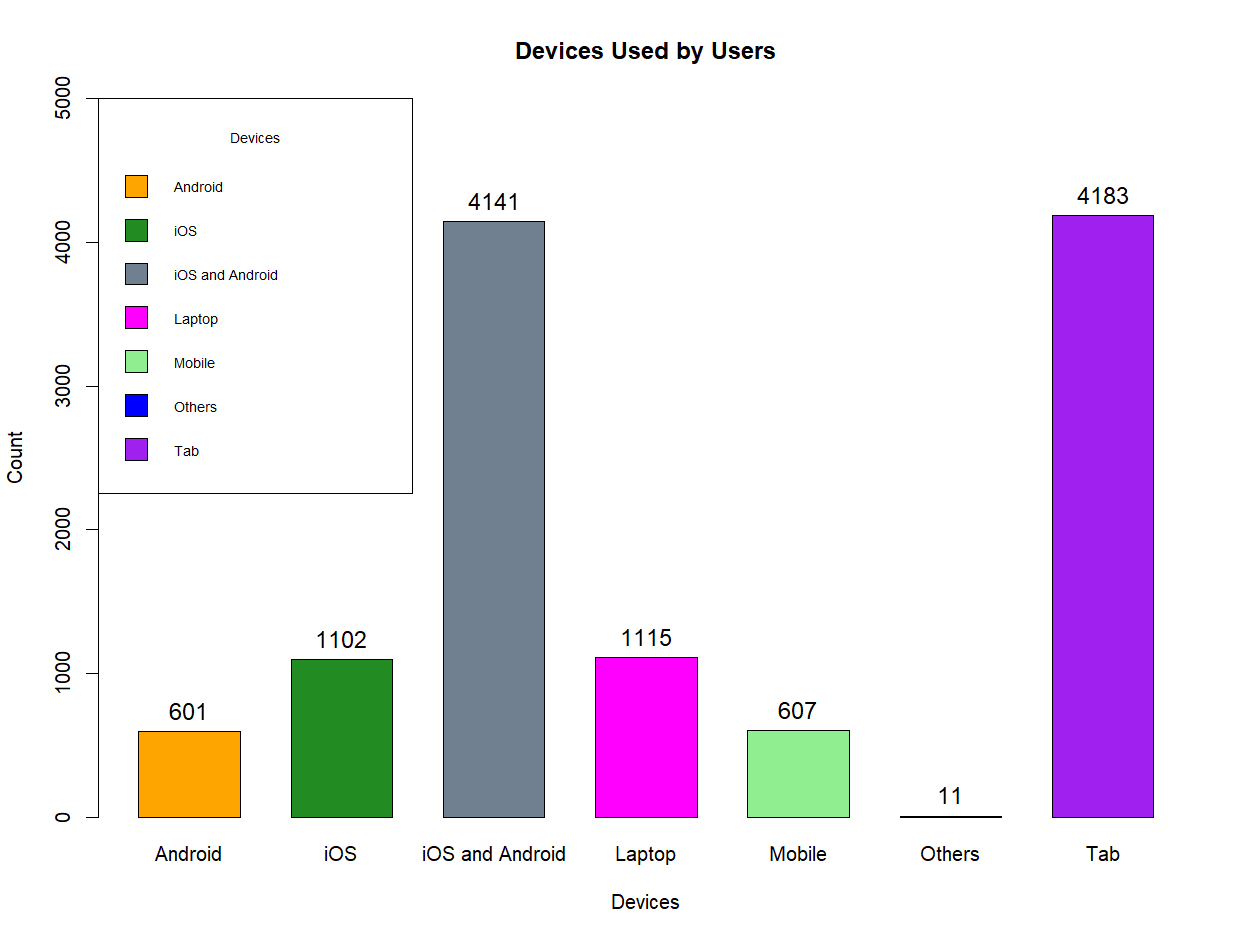
xlab = "Devices", ylab = "Count", ylim = c(0, 5000), space = 0.5)

# Add data labels near the bars

text(x = bp, y = device\_counts + 0.2, labels = device\_counts, pos = 3, col = "black", cex = 1.2)

# Add legend with reduced size

legend("topleft", legend = names(device\_counts), fill =bar\_colors, title = "Devices", cex = 0.7)



**Observation:** By the above bar plot we can observe that users mostly preferred "Tab" to log in into the tourism page.

**4. PIE CHART :**

This pie chart gives us an representation of users interest in selection of location for tourism through this website.

***SOURCE CODE:***

category<-unique(df$preferred\_location\_type)

count<-table(df$preferred\_location\_type)

count

# Create a data frame for plotting

plot\_data <- data.frame(

category = as.factor(names(count)),

count = as.numeric(count)

)

my\_palette <- viridis\_pal()(14)

my\_palette

# Calculate midpoints for labeling

midpoints <- cumsum(plot\_data$count) - plot\_data$count / 2

# Create a donut chart using ggplot2

ggplot(plot\_data, aes(x = "", y = count, fill = category)) +

geom\_bar(stat = "identity", width = 1, color = "white") +

geom\_bar(stat = "identity", width = 0.3, color = "white", fill = "white") +

coord\_polar(theta = "y",start = 0) +

theme\_void() +

theme(

legend.position = "right",

legend.key.size = unit(2, "lines"), # Adjust the size of the legend key

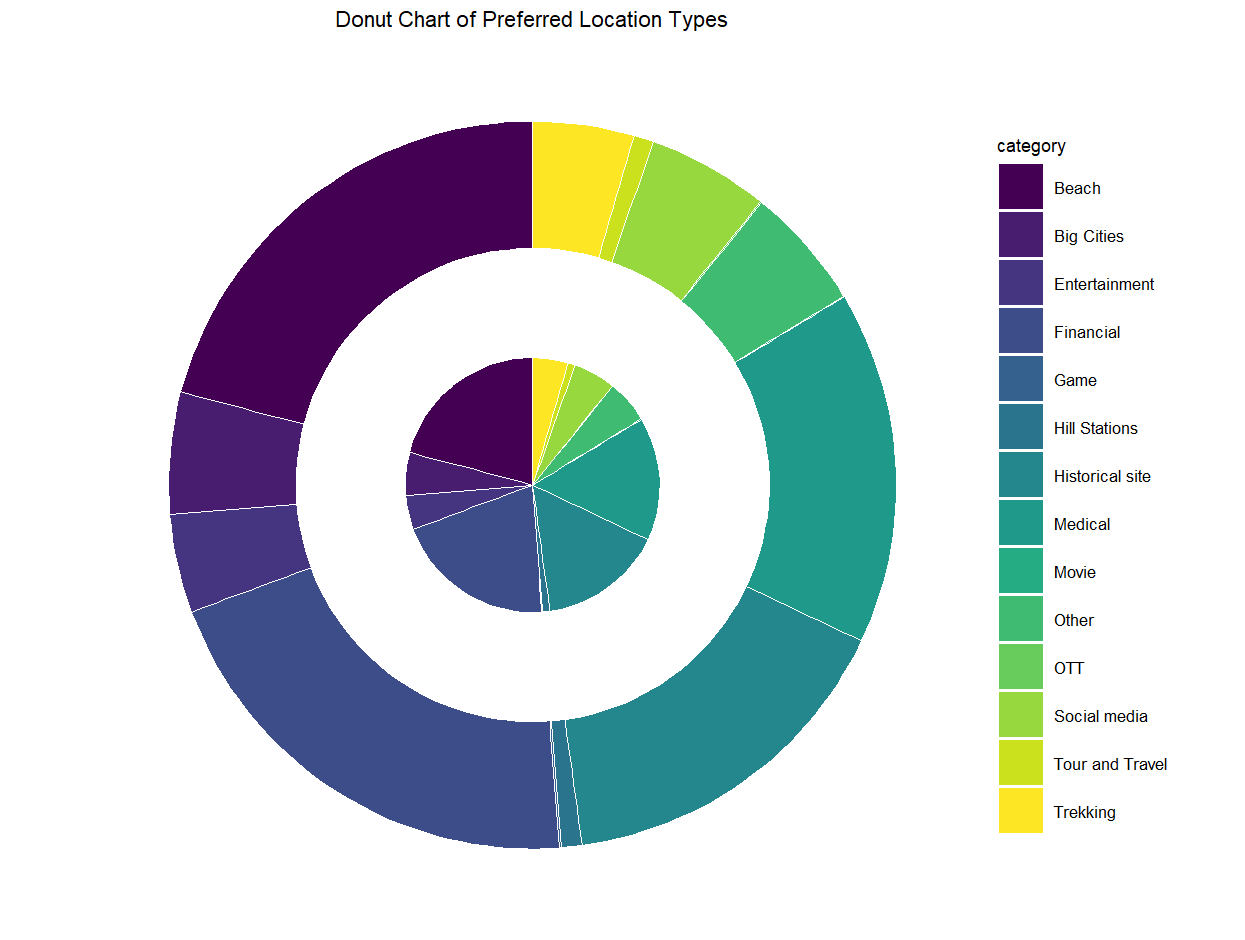
legend.text = element\_text(size = 10),

plot.title = element\_text(hjust = 0.5) # Adjust the size of the legend text

) +

scale\_fill\_manual(values = my\_palette)+

ggtitle( "Donut Chart of Preferred Location Types")



**Observation**: From the above graph we can say that most of the users are interested in "Beach" location.

**5. HISTOGRAM :**

This histogram represents us the yearly average views of all the users who are following this tourism page.

***SOURCE CODE:***

#To find the least viewers

least\_avg\_views <- df %>%

filter(Yearly\_avg\_view\_on\_travel\_page == min(Yearly\_avg\_view\_on\_travel\_page)) %>%

select(UserID, Yearly\_avg\_view\_on\_travel\_page)

print(least\_avg\_views)

#To find the users who view maximum

maximum\_avg\_views <- df %>%

filter(Yearly\_avg\_view\_on\_travel\_page == max(Yearly\_avg\_view\_on\_travel\_page)) %>%

select(UserID, Yearly\_avg\_view\_on\_travel\_page)

print(maximum\_avg\_views)

# Plot the histogram

data=data.frame(df$Yearly\_avg\_view\_on\_travel\_page)

# Create a histogram with density

ggplot(data, aes(x = df$Yearly\_avg\_view\_on\_travel\_page)) +

geom\_histogram(aes(y = ..density..), binwidth = 5, fill = "blue", color = "black", alpha = 2) +

geom\_vline(aes(xintercept = mean(df$Yearly\_avg\_view\_on\_travel\_page)), color = "red", linetype = "dashed", size = 1,show.legend = TRUE) +

geom\_text(aes(x = mean(df$Yearly\_avg\_view\_on\_travel\_page) + 2,y=0.02, label = "Mean"), color = "red", vjust = -2, size = 3) +

geom\_vline(aes(xintercept = quantile(df$Yearly\_avg\_view\_on\_travel\_page, 0.25)), color = "green", linetype = "dashed", size = 1,show.legend = TRUE) +

geom\_text(aes(x = quantile(df$Yearly\_avg\_view\_on\_travel\_page, 0.25), y = 0.02, label = "Q1"), color = "green", vjust = -2, size = 3) +

geom\_vline(aes(xintercept = median(df$Yearly\_avg\_view\_on\_travel\_page)), color = "purple", linetype = "dashed", size = 1,show.legend = TRUE) +

geom\_text(aes(x = median(df$Yearly\_avg\_view\_on\_travel\_page), y = 0.02, label = "Median"), color = "purple", vjust = -0.5, size = 3) +

geom\_vline(aes(xintercept = quantile(df$Yearly\_avg\_view\_on\_travel\_page, 0.75)), color = "green", linetype = "dashed", size = 1,show.legend = TRUE) +

geom\_text(aes(x = quantile(df$Yearly\_avg\_view\_on\_travel\_page, 0.75), y = 0.02, label = "Q3"), color = "green", vjust = -2, size = 3) +

labs(title = "Distribution of Yearly Average Views per User",

x = "Average Views per User",

y = "Density",

caption =paste(

"Skewness: ", skewness(df$Yearly\_avg\_view\_on\_travel\_page),

"\nKurtosis: ", kurtosis(df$Yearly\_avg\_view\_on\_travel\_page),

"\nMean: ", mean(df$Yearly\_avg\_view\_on\_travel\_page),

"\nQ1: ", quantile(df$Yearly\_avg\_view\_on\_travel\_page, 0.25),

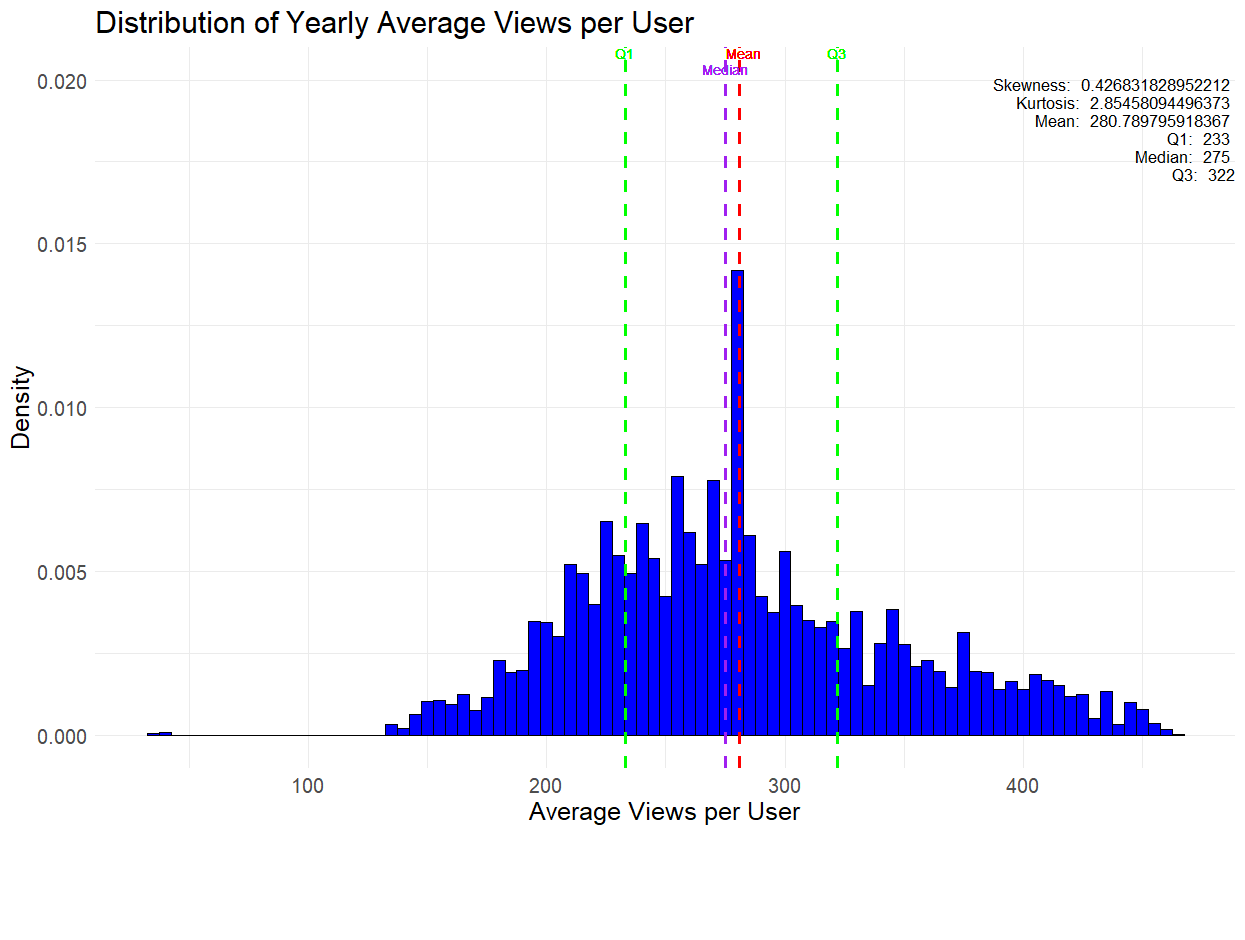
"\nMedian: ", median(df$Yearly\_avg\_view\_on\_travel\_page),

"\nQ3: ", quantile(df$Yearly\_avg\_view\_on\_travel\_page, 0.75)

)) +

theme\_minimal()+

theme(plot.caption = element\_text(hjust = 1,vjust = 250, margin = margin(t = 0, r = 10),size = 10),text = element\_text(size = 15))

**Observation**: In this plot we can see the avg views (highest and lowest) and its statistical measurements and it is positively skewed, and also leptokurtic curve. This plot shows us the distribution of data Yearly\_avg\_view\_on\_travel\_page.

**6.FREQUENCY POLYGON:**

This frequency polygon graph shows the total likes given by the user who are following the tourism page.

***SOURCE CODE:***

#Maximum likes given by user

maximum\_likes <- df %>%

filter(total\_likes\_on\_outstation\_checkin\_given == max(total\_likes\_on\_outstation\_checkin\_given)) %>%

select(UserID, total\_likes\_on\_outstation\_checkin\_given)

print(maximum\_likes)

#Minimum likes given by user

minimum\_likes <- df %>%

filter(total\_likes\_on\_outstation\_checkin\_given == min(total\_likes\_on\_outstation\_checkin\_given)) %>%

select(UserID, total\_likes\_on\_outstation\_checkin\_given)

print(minimum\_likes)

######################################

data<- data.frame(df$total\_likes\_on\_outstation\_checkin\_given)

# Set the binwidth based on your data

binwidth <- 10# You can adjust this based on your data distribution

# Create a ggplot with a frequency polygon

plot1<-ggplot(data, aes(x = df$total\_likes\_on\_outstation\_checkin\_given)) +

geom\_freqpoly( color = "blue", size = 1) +

labs(title = "Frequency Polygon of Total Likes Given by Users",

x = "Total Likes Given by Users",

y = "Density",

caption = paste(

"Highest Likes: 252430",

"\nLowest Likes: 3570"

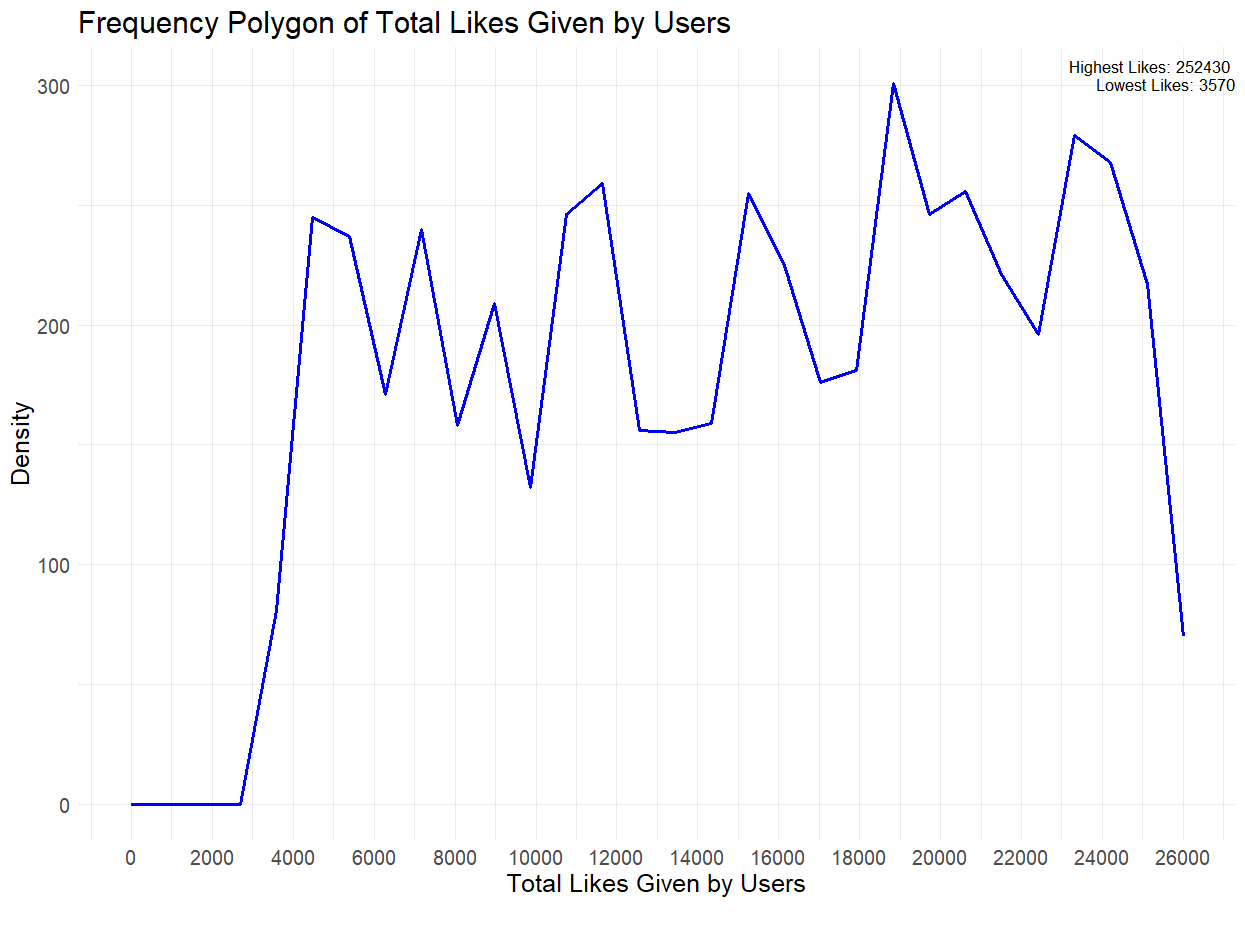
)) +

scale\_x\_continuous(limits = c(0, 26000), breaks = seq(0, 26000, by = 2000)) +# Adjust limits and breaks based on your data

theme\_minimal()+

theme(plot.caption = element\_text(hjust = 1,vjust = 280, margin = margin(t = 0, r = 10),size = 10),text = element\_text(size = 15))

plot1



**OBSERVATION:**

* On the y- axis we can see the density of the users who gives likes and on x-axis we can see the number of counts that the user given to the tourism page.
* In legend we can see the highest count of likes and also lowest count of likes.

**7.FREQUENCY POLYGON :**

This frequency polygon shows the total number of likes received from the user to the tourism page.

***SOURCE CODE***:

maximum\_likes <- df %>%

filter(total\_likes\_on\_outofstation\_checkin\_received == max(total\_likes\_on\_outofstation\_checkin\_received)) %>%

select(UserID, total\_likes\_on\_outofstation\_checkin\_received)

print(maximum\_likes)

minimum\_likes <- df %>%

filter(total\_likes\_on\_outofstation\_checkin\_received == min(total\_likes\_on\_outofstation\_checkin\_received)) %>%

select(UserID, total\_likes\_on\_outofstation\_checkin\_received)

print(minimum\_likes)

data<- data.frame(df$total\_likes\_on\_outstation\_checkin\_given)

# Set the bin width based on your data

binwidth <- 10# You can adjust this based on your data distribution

# Create a ggplot with a frequency polygon

plot2<-ggplot(data, aes(x = df$total\_likes\_on\_outofstation\_checkin\_received)) +

geom\_freqpoly( color = "blue", size = 1) +

labs(title = "Frequency Polygon of Total Likes Received from Users",

x = "Total Likes Received from Users",

y = "Density",

caption = paste(

"Highest Likes: 20065",

"\nLowest Likes: 1009"

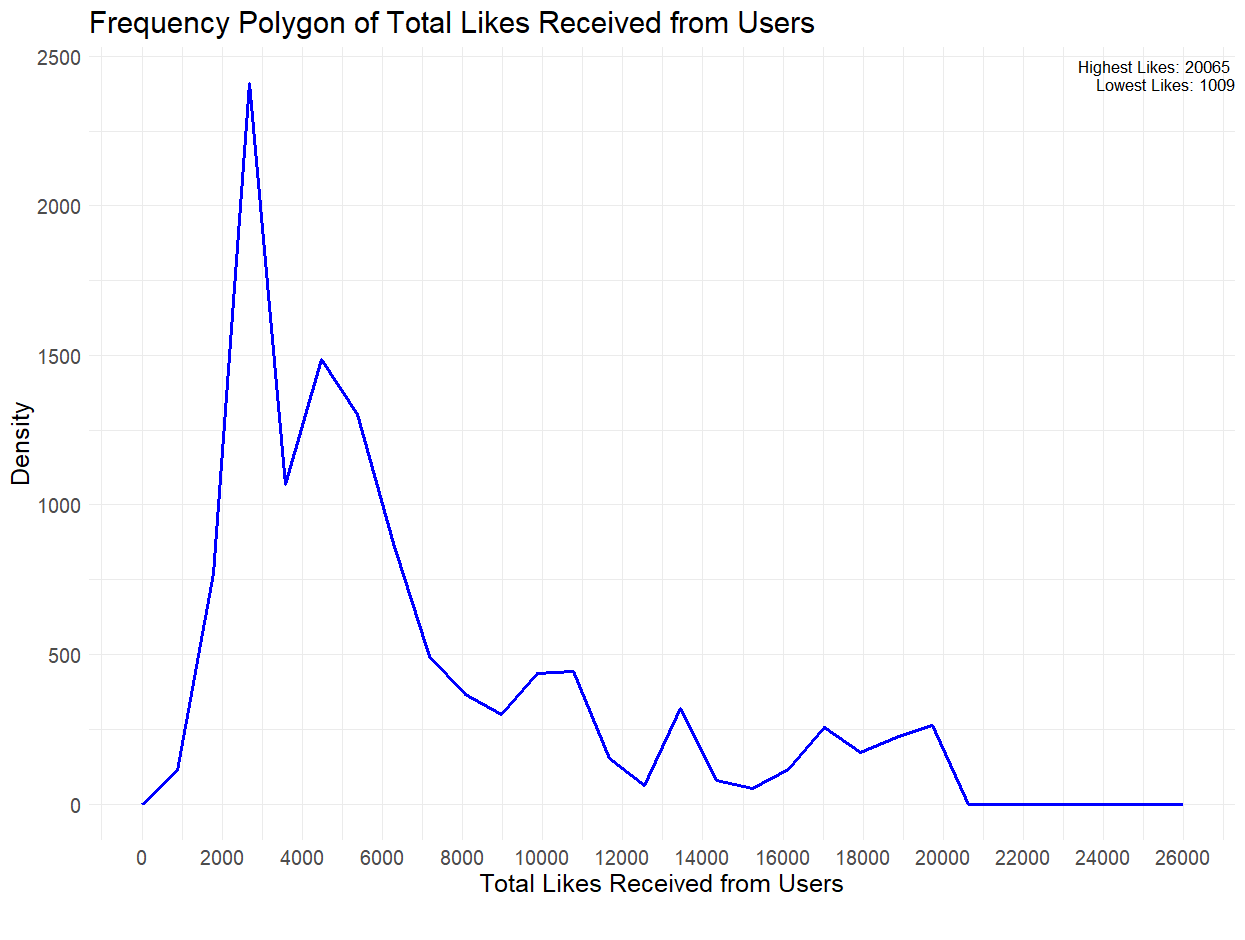
)) +

scale\_x\_continuous(limits = c(0, 26000), breaks = seq(0, 26000, by = 2000)) +# Adjust limits and breaks based on your data

theme\_minimal()+

theme(plot.caption = element\_text(hjust = 1,vjust = 280, margin = margin(t = 0, r = 10),size = 10),text = element\_text(size = 15))

plot2



**OBSERVATION:**

* On x-axis we can see the number of like that are received from the user to the tourism page.
* On y-axis we can see the density of the users i.e from how many number of users the likes are received.

**8.MERGED FREQUENCY POLYGON:**

In this we can see the merged plot between both the graphs like total number of likes given to user and total number of likes received from user.

***SOURCE CODE:***

#Merging of both the plots

#Use the same data frame for both plots

data <- data.frame(total\_likes = c(df$total\_likes\_on\_outstation\_checkin\_given, df$total\_likes\_on\_outofstation\_checkin\_received),

type = rep(c("Likes Given", "Likes Received"), each = nrow(df)))

# Overlay the plots

merged\_plot <- ggplot(data, aes(x = total\_likes, color = type)) +

geom\_freqpoly(binwidth = 10, size = 1) +

labs(title = "Merged Frequency Polygons",

x = "Total Likes",

y = "Density",

caption = paste(

"Highest Likes Given Count: 252430",

"\nLowest Likes Given Count: 3570",

"\nHighest Likes Received Count: 20065",

"\nLowest Likes Received Count: 1009"

)) +

scale\_x\_continuous(limits = c(0, 26000), breaks = seq(0, 26000, by = 2000)) +

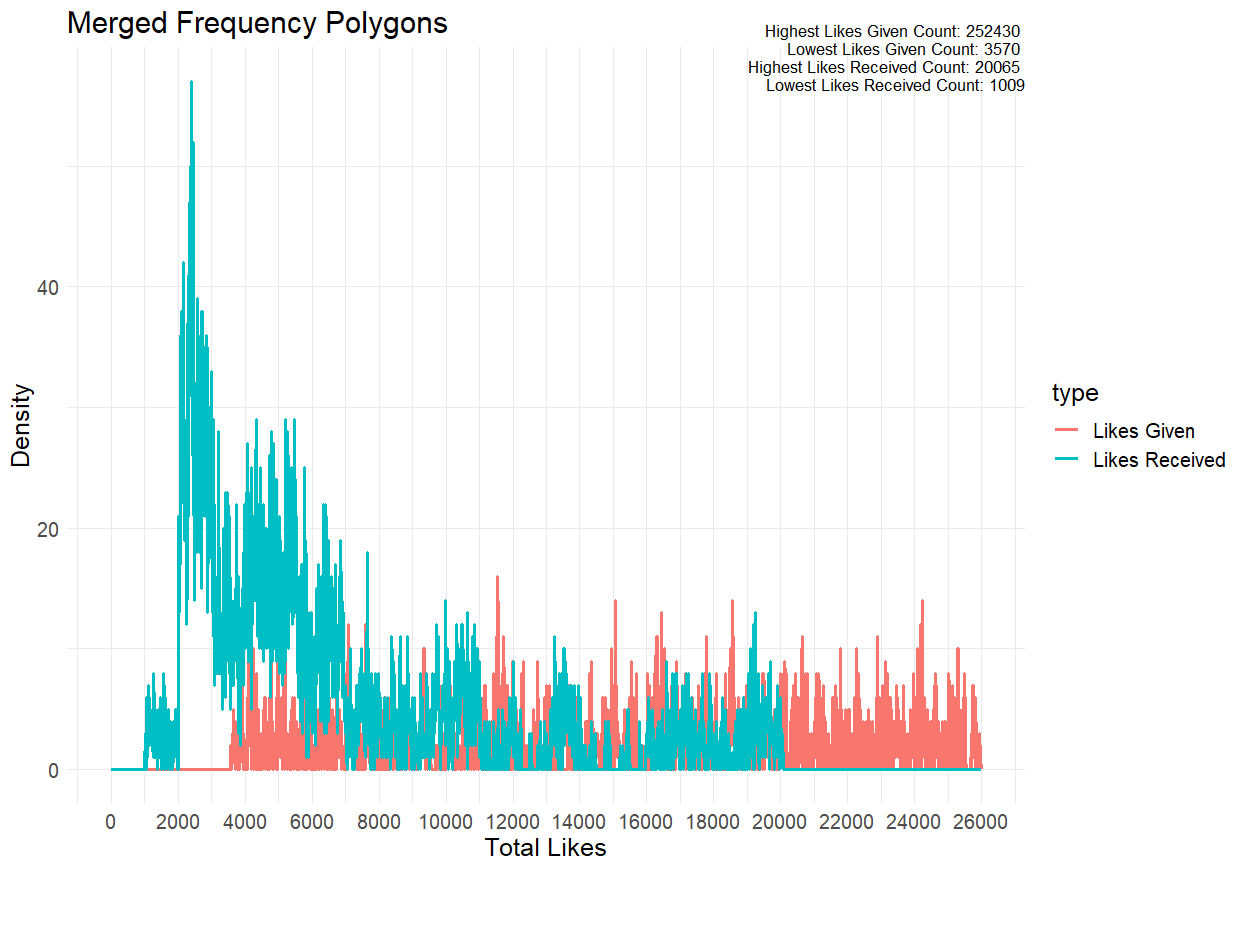
theme\_minimal() +

theme(plot.caption = element\_text(hjust = 1, vjust = 280, margin = margin(t = 0, r = 10), size = 10),

text = element\_text(size = 15))

# Show the merged plot

print(merged\_plot)



**OBSERVATION:**

In this graph we can find out the difference between both the graphs which shows the highest and lowest number of likes in the legend.

**9.HISTOGRAM:**

This histogram represents the graph between the number of users and yearly outstation checkins of all the users.

***SOURCE CODE:***

table(df$yearly\_avg\_Outstation\_checkins)

result <- df %>%

group\_by(yearly\_avg\_Outstation\_checkins) %>%

summarise(number\_of\_users = n\_distinct(UserID)) %>%

mutate(yearly\_avg\_Outstation\_checkins = order(factor(yearly\_avg\_Outstation\_checkins), -number\_of\_users))

# Print the result

print(result)

View(result)

result$yearly\_avg\_Outstation\_checkins <- as.numeric(as.character(result$yearly\_avg\_Outstation\_checkins))

############# Calculate summary statistics

skewness\_val <- skewness(result$number\_of\_users)

kurtosis\_val <- kurtosis(result$number\_of\_users)

mean\_val <- mean(result$number\_of\_users)

median\_val <- median(result$number\_of\_users)

# Assuming result is your dataframe

ggplot(result, aes(x = yearly\_avg\_Outstation\_checkins, y = number\_of\_users)) +

geom\_histogram(stat = "identity", fill = "skyblue", color = "white") +

geom\_text(aes(label = number\_of\_users), vjust = -0.5, color = "white", size = 3) +

labs(title = "Number of Users in Each Check-in Category",

x = "Check-in Category",

y = "Number of Users") +

scale\_x\_continuous(limits = c(0, 30), breaks = seq(0, 30, by = 2)) +

theme\_minimal()+

theme(plot.background = element\_rect(fill = "black"),

axis.text.x = element\_text(color = "white"), # Change color of x-axis labels

axis.text.y = element\_text(color = "white"),

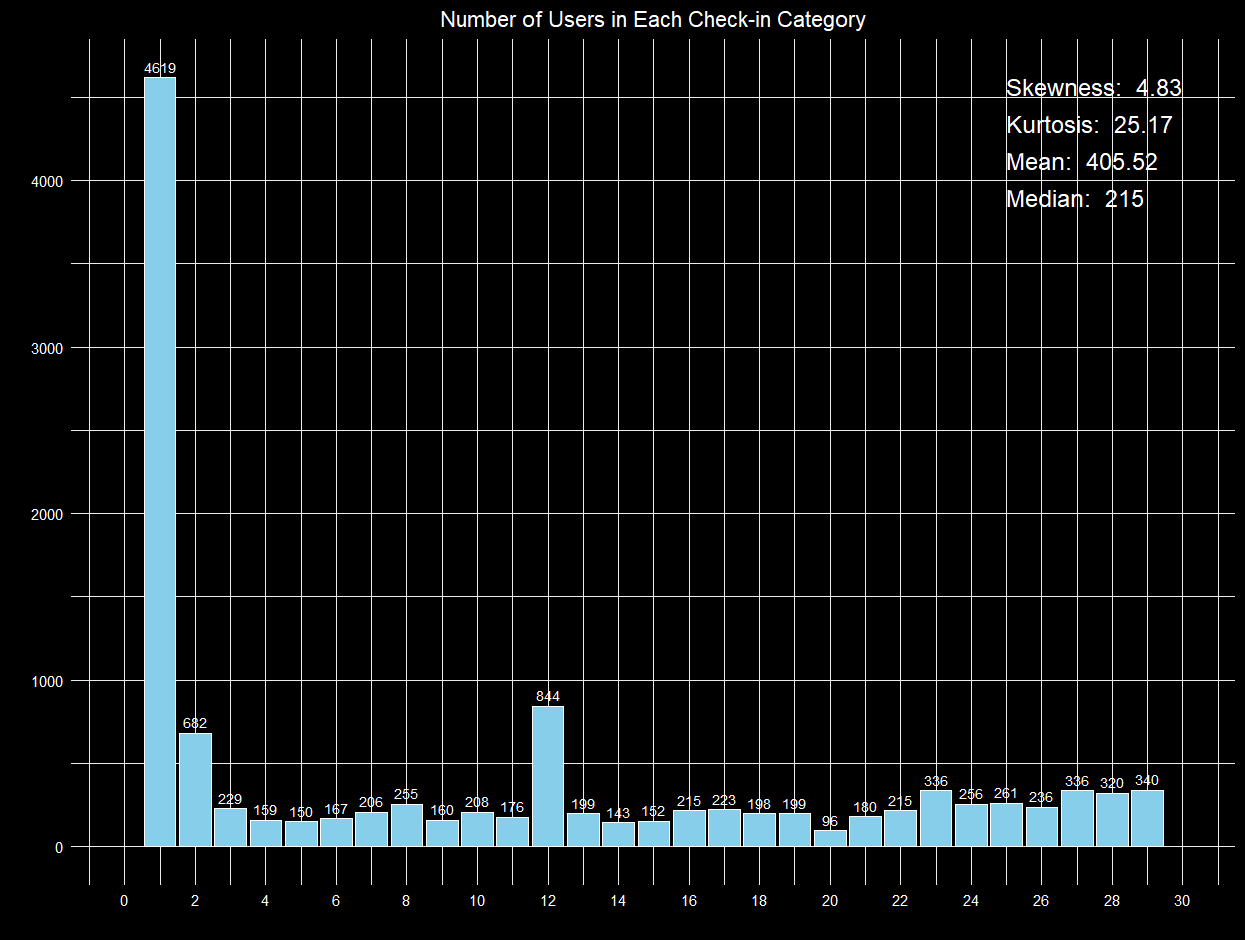
plot.title = element\_text(color = "white",hjust = 0.5))+

annotate("text", x = 25, y = max(result$number\_of\_users), label = paste("Skewness: ", round(skewness\_val, 2)), hjust = 0, vjust = 1, color = "white",size = 5) +

annotate("text", x = 25, y = max(result$number\_of\_users) - 5, label = paste("Kurtosis: ", round(kurtosis\_val, 2)), hjust = 0, vjust = 3, color = "white",size=5) +

annotate("text", x = 25, y = max(result$number\_of\_users) - 10, label = paste("Mean: ", round(mean\_val, 2)), hjust = 0, vjust = 5, color = "white",size=5) +

annotate("text", x = 25, y = max(result$number\_of\_users) - 15, label = paste("Median: ", round(median\_val, 2)), hjust = 0, vjust = 7, color = "white",size=5)



**OBSERVATION :**

In this histogram we can see the count of users who done different categories like yearly outstation checkins along with the skewness and kurtosis of the column.

**10. BARPLOT :**

This plot represents the data between the users and the members of family goes along with the user for vacation.

***SOURCE CODE:***

table(df$member\_in\_family)

result <- df %>%

group\_by(member\_in\_family) %>%

summarise(number\_of\_users = n\_distinct(UserID)) %>%

mutate(member\_in\_family = reorder(factor(member\_in\_family), -number\_of\_users))

result

#Ploting a bar

result\_bar\_plot <- ggplot(result, aes(x = member\_in\_family, y = number\_of\_users, fill =member\_in\_family)) +

geom\_bar(stat = "identity", position = "dodge") +

geom\_text(aes(label = number\_of\_users),

vjust = -0.5,

size = 3) +

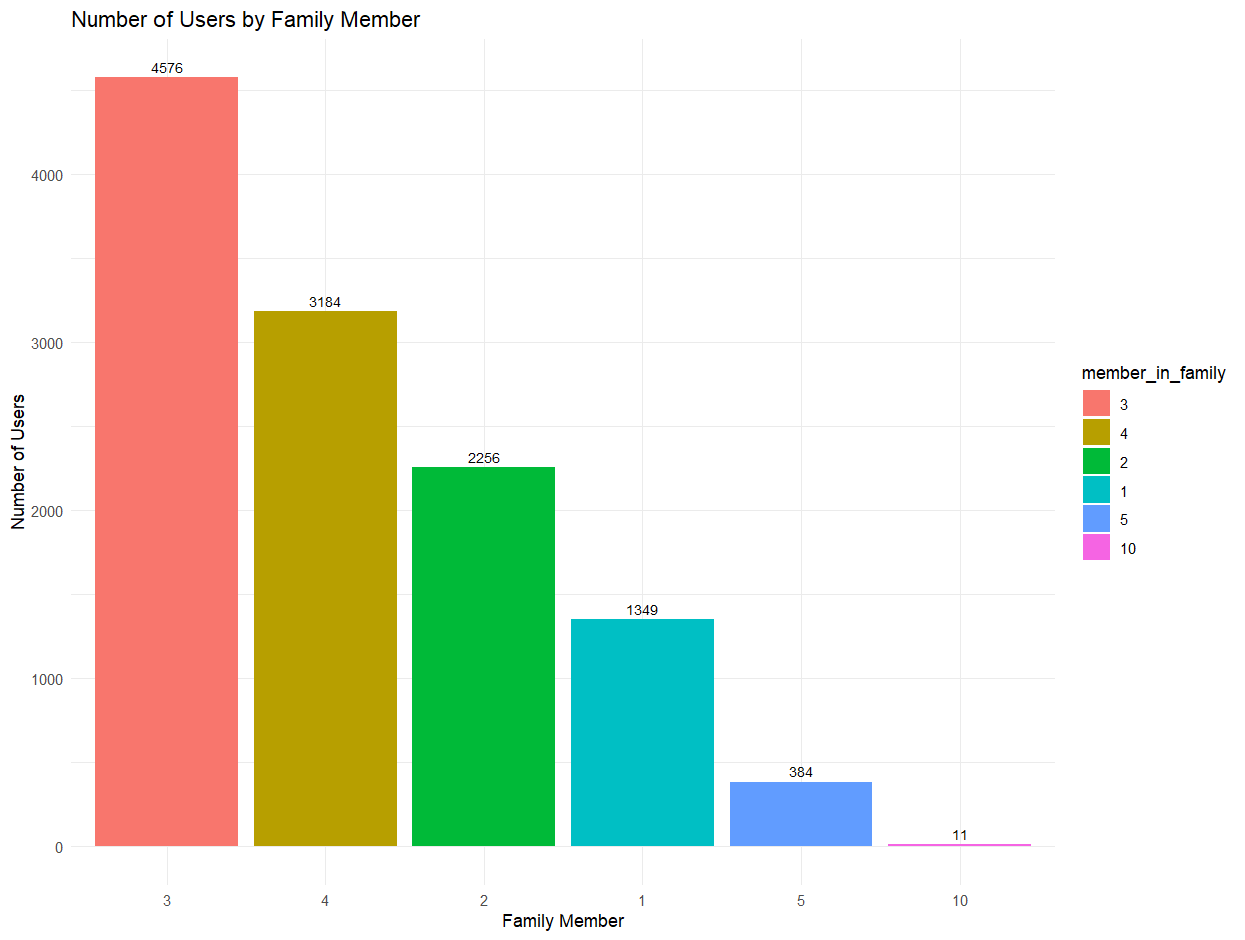
labs(title = "Number of Users by Family Member",

x = "Family Member",

y = "Number of Users") +

theme\_minimal()

print(result\_bar\_plot)



**OBSERVATION:**

This graph shows the representation of all 4 categories in members of family column like 1,2,3,4,5,10, and the users on y-axis.

**11. BAR PLOT:**

This bar plot shows the how many number of users are adults and how many number of users are not adults.

***SOURCE CODE:***

result <- df %>%

group\_by(Adult\_flag) %>%

summarise(number\_of\_users = n\_distinct(UserID)) %>%

mutate(Adult\_flag = reorder(factor(Adult\_flag), -number\_of\_users))

result

#To print plot

result\_area\_plot <- ggplot(result, aes(x = Adult\_flag, y = number\_of\_users, fill = Adult\_flag)) +

geom\_col(alpha = 0.7) +

geom\_text(aes(label = number\_of\_users), position = position\_stack(vjust = 0.5), size = 3, color = "black") +

labs(title = "Number of Users by Adult Flag",

x = "Adult Flag",

y = "Number of Users") +

theme\_minimal()

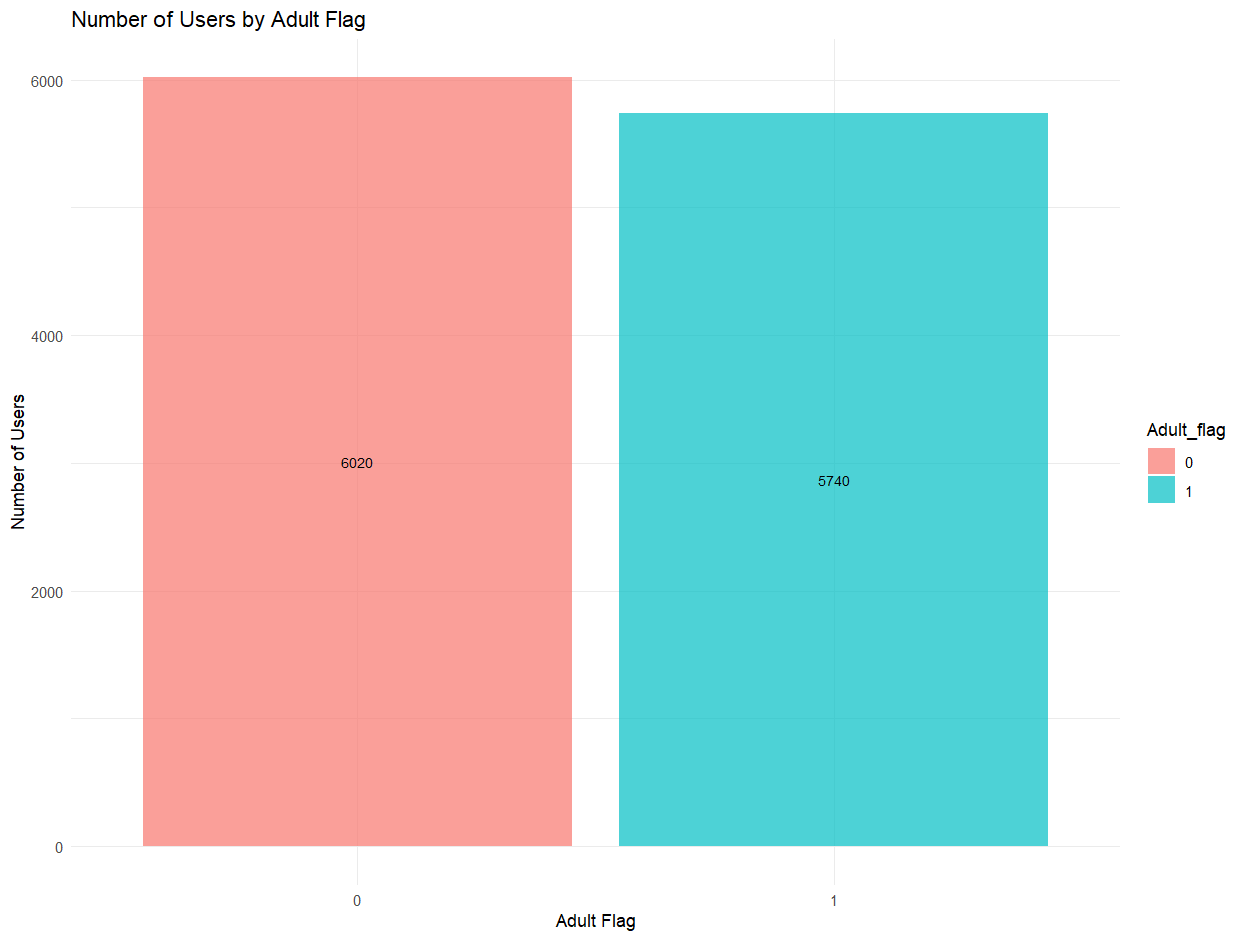
print(result\_area\_plot)

**OBSERVATION:**

* In this bar graph we can see that x-axis represents the 0 and 1 which mean “NOT ADULT” and

“ADULT” and y-axis represents the number of users.

* In this column there are outliers present like 2 and 3. I used some methods to find out outliers like Q1 and Q2 and drawn a box plot for outliers which shown the dots that represents in dots.



**12. SCATTER PLOT:**

This graph shows us the data between the users and the daily number of minutes there are viewing the page.

***SOURCE CODE:***

table(df$Daily\_Avg\_mins\_spend\_on\_traveling\_page)

result <- df %>%

group\_by(Daily\_Avg\_mins\_spend\_on\_traveling\_page) %>%

summarise(number\_of\_users = n\_distinct(UserID)) %>%

mutate(Daily\_Avg\_mins\_spend\_on\_traveling\_page = reorder(factor(Daily\_Avg\_mins\_spend\_on\_traveling\_page), -number\_of\_users))

result

############

ggplot(result, aes(x = Daily\_Avg\_mins\_spend\_on\_traveling\_page, y = number\_of\_users)) +

geom\_point(stat = "identity",color="gold") +

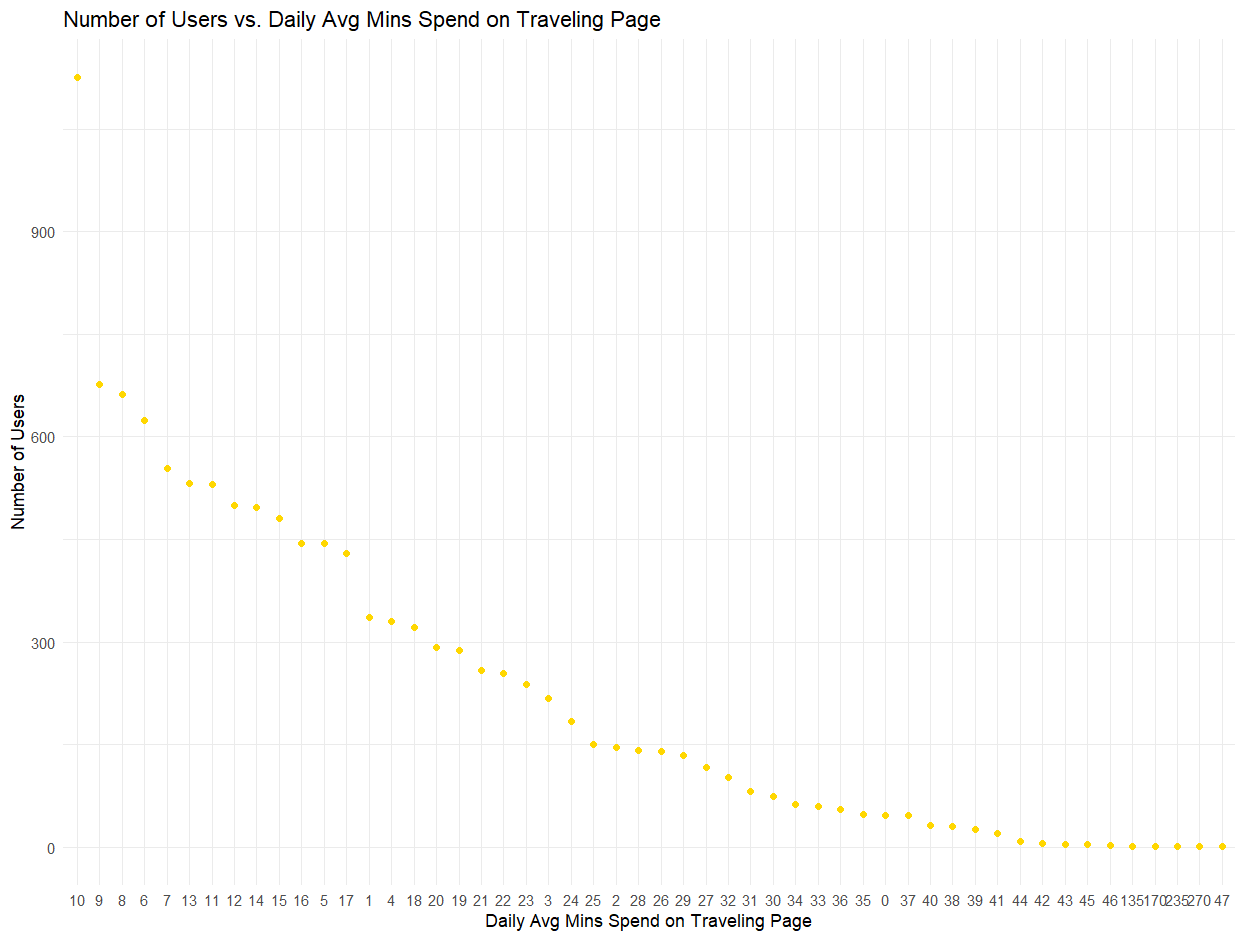
labs(title = "Number of Users vs. Daily Avg Mins Spend on Traveling Page",

x = "Daily Avg Mins Spend on Traveling Page",

y = "Number of Users") +

theme\_minimal()+

scale\_x\_discrete(labels = function(x) gsub(" min", "", x))



**OBSERVATION:**

In this above graph we can see the number of minutes on X-axis and the users on y-axis.

* **CONCLUSION:**
* From all the above graphs we can see the relation between the users and the tourism website page.
* In Data visualization there are total 12 graphs which consists of both Categorical as well as Numerical graphs. There are :
* Bar Plot
* Histogram
* Frequency Polygon
* Pie Chart
* Scatter Plot

**ABOUT COMPANY PAGE ANALYSIS:**

* In this company page the users there are most users not following the page like 8360 and 3297 are following the page.
* Most of the users logged into this website by using Tabs and IOS/ Android devices.
* The likes given to the users page is more when compared to the likes received from the users to the tourism page.

**ABOUT THE USERS:**

* In this Tourism data set there are 11760 unique user id’s which mean there are total 11760 users who are using this Tourism website.
* There are only two types of users in which 1 represent an Adult User and 0 represents Not an Adult.
* There are different locations present in this Tourism website but most of the users prefer “Beach” location
* There are mainly different categories of yearly outstation check-ins done by the users in which the highest count of check-in is 4619 i.e 1 time.
* The users go along with the members of their family which consists of different count like 1 to 10 members in that the highest count is 4576 that is 3 members in family along with user.
* In the dataset the data has Daily number of minutes the user viewing this tourism website and the highest number of minutes the user viewing this page is 10 minutes the count is 1125.
* There is column of working flag which contains Boolean values like 0 and 1 which indicates the user is working or not.

