Survey Work

2024-05-03

Mean, Standard Deviation and Factoring the Data

Author: Randell Eduarth Soteo

```
# Import CSV file to R
data <- read.csv("User Satisfaction on Popular Mobile Games_ A Survey_COPY.csv")
```

```
# Performance Expectancy
# Find mean and standard deviation for PE
PE <- data$How.do.you.perceive.the.performance..graphics..gameplay..etc...of.the.games.you.play.
PE_mean <- mean(PE)
PE_sd <- sd(PE)

# Print the mean and standard deviation for PE
PE_mean
```

```
## [1] 4.214286
```

```
PE_sd
```

```
## [1] 0.8981204
```

```
# Find the mean and standard deviation for EE
EE <- data$How.easy.or.difficult.do.you.find.it.to.navigate.and.use.the.features.in.the.games.
EE_mean <- mean(EE)
EE_sd <- sd(EE)

# Print the mean and standard deviation for EE
EE_mean
```

```
## [1] 3.880952
```

```
EE_sd
```

```
## [1] 0.9927145
```

```
# Find the mean and standard deviation for EE
FC <- data$Do.you.believe.that.you.have.the.necessary.resources.and.support.to.effectively.use.mobile.ga
FC_mean <- mean(FC)
FC_sd <- sd(FC)

# Print the mean and standard deviation for EE
FC_mean
```

```
## [1] 3.928571
```

```
FC_sd
```

```
## [1] 0.8942324
```

```r
BI <- data$How.likely.are.you.to.continue.playing.mobile.games.in.the.future.
BI_mean <- mean(BI)
BI_sd <- sd(BI)

# Print the mean and standard deviation for EE
BI_mean
```

```
## [1] 3.857143
```

```r
BI_sd
```

```
## [1] 1.298619
```

```r
# Factoring the responses
SI <- data$Have.you.started.playing.any.of.these.games.because.of.recommendations.from.friends.or.social
SI_fr <- factor(data$Have.you.started.playing.any.of.these.games.because.of.recommendations.from.friends

FC <- data$Do.you.believe.that.you.have.the.necessary.resources.and.support.to.effectively.use.mobile.ga
FC_fr <- factor(data$Do.you.believe.that.you.have.the.necessary.resources.and.support.to.effectively.use

BI <- data$How.likely.are.you.to.continue.playing.mobile.games.in.the.future.
BI_fr <- factor(data$How.likely.are.you.to.continue.playing.mobile.games.in.the.future., levels = c(1,2
```

```r
# Merge responses to dataframe
mergeData_PE <- data.frame(
  Poor = sum(PE == 1),
  Fair = sum(PE == 2),
  Neutral = sum(PE == 3),
  Good = sum(PE == 4),
  Excellent = sum(PE == 5),
  Mean = PE_mean,
  SD = PE_sd
)
# Assigning row name
row.names(mergeData_PE) <- "Performance Expectancy"


# Merge responses to dataframe
mergeData_EE <- data.frame(
  ID = "EE",
  Very_Difficult = sum(EE == 1),
  Difficult = sum(EE == 2),
  Neutral = sum(EE == 3),
  Easy = sum(EE == 4),
  Very_Easy = sum(EE == 5),
  Mean = EE_mean,
  SD = EE_sd
)

# Assigning row name
row.names(mergeData_EE) <- "Effort Expectancy"

#Merge responses to data frame
mergeData_SI <- data.frame(
  ID = "SI",
  Yes = sum(SI == "Yes"),
  No = sum(SI == "No")
)

# Assigning row name
row.names(mergeData_SI) <- "Social Influence"


# Merge responses to dataframe
mergeData_FC <- data.frame(
  ID = "FC",
  Strongly_Disagree = sum(FC == 1),
  Disagree = sum(FC == 2),
  Neutral = sum(FC == 3),
  Agree = sum(FC == 4),
  Strongly_Agree = sum(FC == 5),
  Mean = FC_mean,
  SD = FC_sd
)

# Assigning row name
row.names(mergeData_FC) <- "Facilitating Conditions"


# Merge responses to dataframe
mergeData_BI <- data.frame(
  ID = "BI",
  Very_Unlikely = sum(BI == 1),
  Unlikely = sum(BI == 2),
  Neutral = sum(BI == 3),
  Likely = sum(BI == 4),
  Very_Likely = sum(BI == 5),
  Mean = BI_mean,
  SD = BI_sd
)

# Assigning row name
row.names(mergeData_FC) <- "Behavioral Intentions"
```

```r
# Write CSV file
write.csv(mergeData_PE, file = "Survey Data (PE).csv", row.names = FALSE)
write.csv(mergeData_EE, file = "Survey Data (EE).csv", row.names = FALSE)
write.csv(mergeData_SI, file = "Survey Data (SI).csv", row.names = FALSE)
write.csv(mergeData_FC, file = "Survey Data (FC).csv", row.names = FALSE)
write.csv(mergeData_BI, file = "Survey Data (BI).csv", row.names = FALSE)
```

Plotting

Author: Randell Eduarth Soteo

```r
library(plotly)
```

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

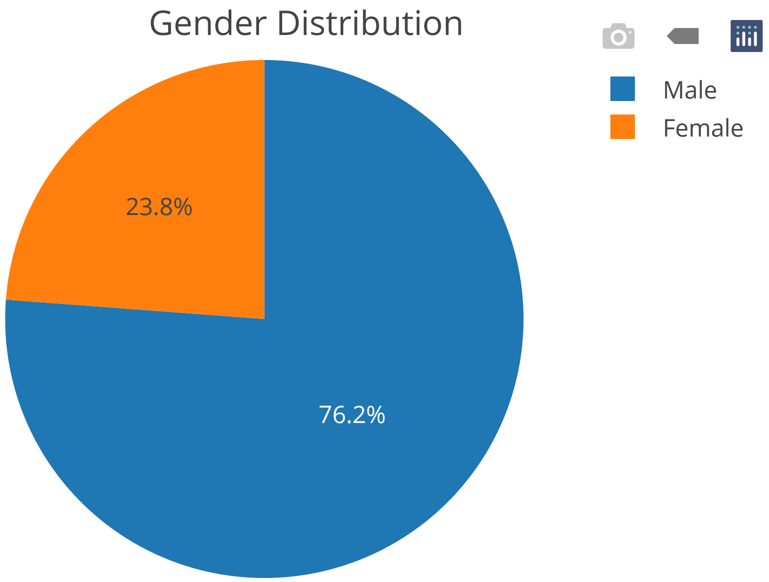
```
##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout
```

```r
library(dplyr)
```

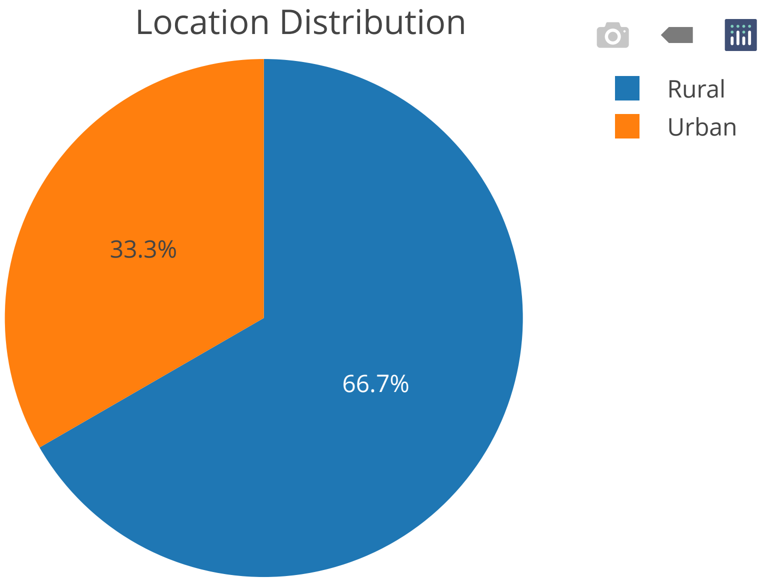
```
##
## Attaching package: 'dplyr'

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

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

```r
# Read the data
visual_data <- read.csv("User Satisfaction on Popular Mobile Games_ A Survey_COPY.csv")
```

```r
# Count gender responses
gender_counts <- table(data$Gender)

# Create pie chart using Plotly
pie_chart_gender <- plot_ly(labels = names(gender_counts), values = gender_counts, type = "pie") %>%
  layout(title = "Gender Distribution")

#Print
pie_chart_gender
```

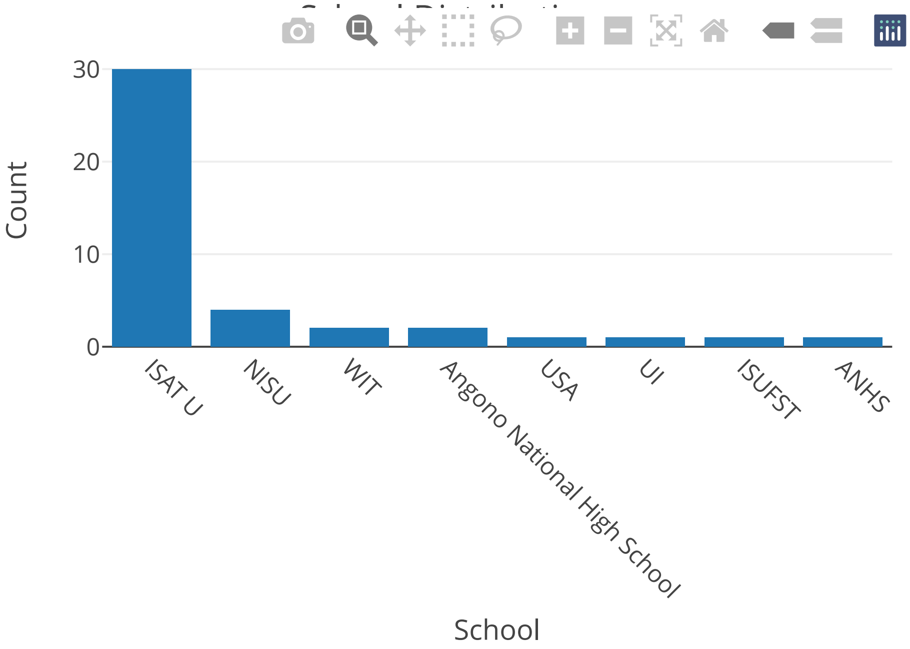


```r
#Count location responses
location_counts <- table(data$Location)

# Create pie chart using Plotly
pie_chart_location <- plot_ly(labels = names(location_counts), values = location_counts, type = "pie")
  layout(title = "Location Distribution")

#Print
pie_chart_location
```

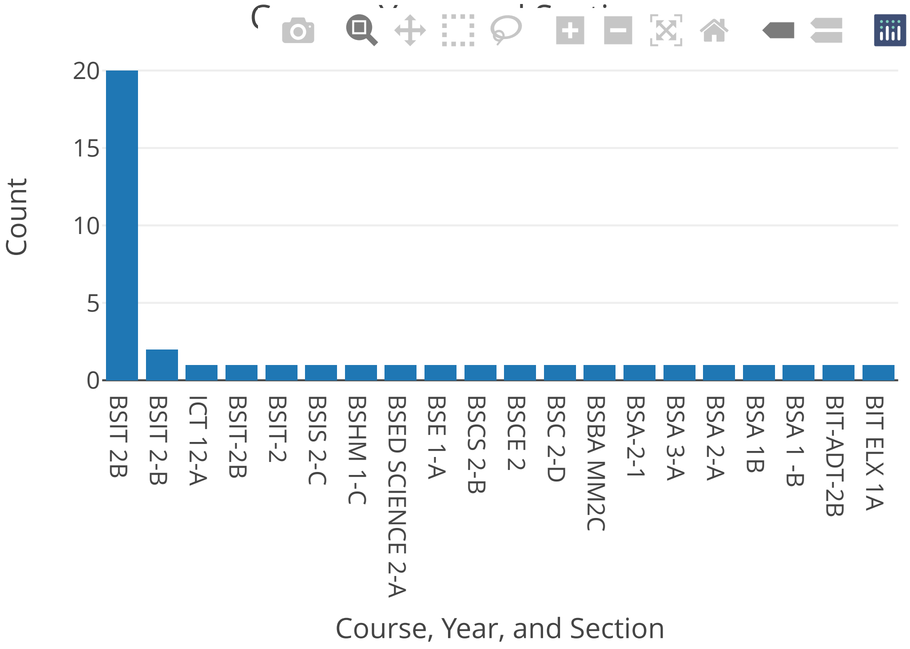


```r
# List of similar schools to merge for ISAT U
ISAT_U <- c("ISAT U", "ISATU")

# Create a new column to map similar schools to a common category for ISAT U
data$Merged_School <- ifelse(data$School %in% ISAT_U, "ISAT U", data$School)

# List of similar schools to merge for NISU
```

```r
NISU <- c("NISU Main", "NISU BVC", "NISU AJUY")

# Update the Merged_School column to map similar schools to a common category for NISU
data$Merged_School <- ifelse(data$School %in% NISU, "NISU", data$Merged_School)

# Aggregate counts for the common categories
merged_school_counts <- aggregate(School ~ Merged_School, data, length)

# Create bar chart using Plotly with merged data
bar_chart_merged_schools <- plot_ly(x = merged_school_counts$Merged_School, y = merged_school_counts$Scl
  layout(title = "School Distribution", xaxis = list(title = "School", tickangle = 45, categoryorder =

#Print
bar_chart_merged_schools
```

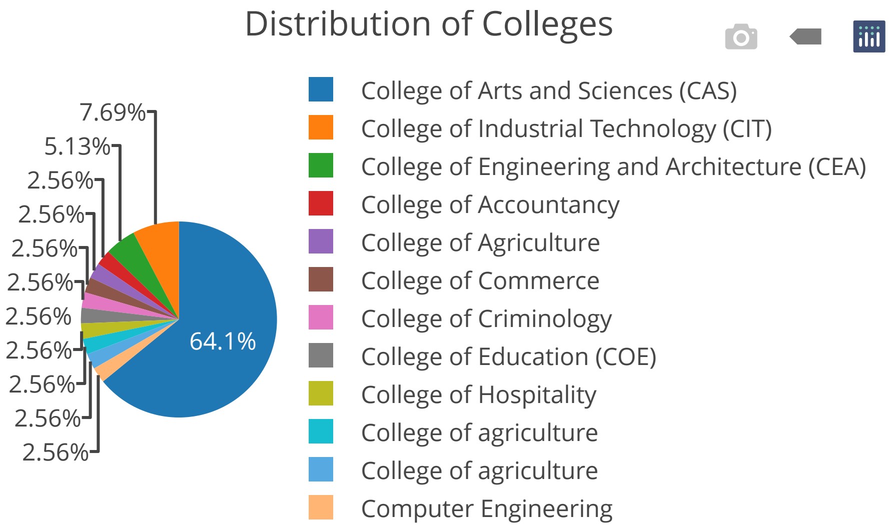


```r
# Define the similar responses to be merged
similar_responses <- list(
  c("BSIT 2B", "BSIT -2B", "BSIT 2-B",  "BS IT 2-B" )
)

# Function to merge similar responses
merge_similar <- function(x) {
  for (pair in similar_responses) {
    if (x %in% pair) {
      return(pair[[1]])
    }
  }
  return(x)
}

# Apply the merge_similar function to the combined column
data <- mutate(data, Course..Year.and.Section..ex..BSIT.2.B. = sapply(Course..Year.and.Section..ex..BSI'
# Remove N/A values
data <- data %>%
  filter(Course..Year.and.Section..ex..BSIT.2.B. != "N/A")

# Aggregate counts for each unique combination of the combined column
combined_column_counts <- data %>%
  group_by(Course..Year.and.Section..ex..BSIT.2.B.) %>%
  summarise(count = n()) %>%
  ungroup()

# Create bar chart using Plotly
bar_chart_combined_column <- plot_ly(data = combined_column_counts, x = ~Course..Year.and.Section..ex..
  layout(title = "Course, Year, and Section", xaxis = list(title = "Course, Year, and Section", categor

# Print the chart
bar_chart_combined_column
```



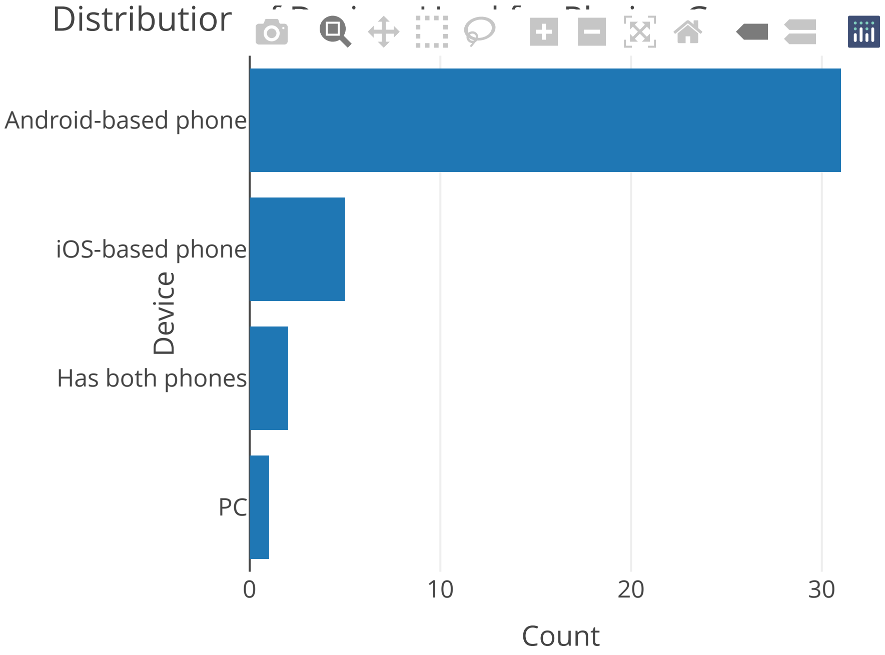
```
# Define the similar responses to be merged
similar_colleges <- list(
  c("College of agriculture", "College of Agriculture")
)

# Function to merge similar college names
merge_similar_colleges <- function(x) {
  for (pair in similar_colleges) {
    if (x %in% pair) {
      return(pair[[1]])
    }
  }
  return(x)
}

# Apply the merge_similar_colleges function to the college column
```

```
data <- mutate(data, Which.College.are.you.from. = sapply(Which.College.are.you.from., merge_similar_co

# Remove N/A values
data <- data %>%
  filter(Which.College.are.you.from. != "N/A")

# Aggregate counts for each unique college
college_counts <- data %>%
  group_by(Which.College.are.you.from.) %>%
  summarise(count = n()) %>%
  ungroup()

# Create pie chart using Plotly
pie_chart_colleges <- plot_ly(data = college_counts, labels = ~Which.College.are.you.from., values = ~c
  layout(title = "Distribution of Colleges")

# Print the chart
pie_chart_colleges
```

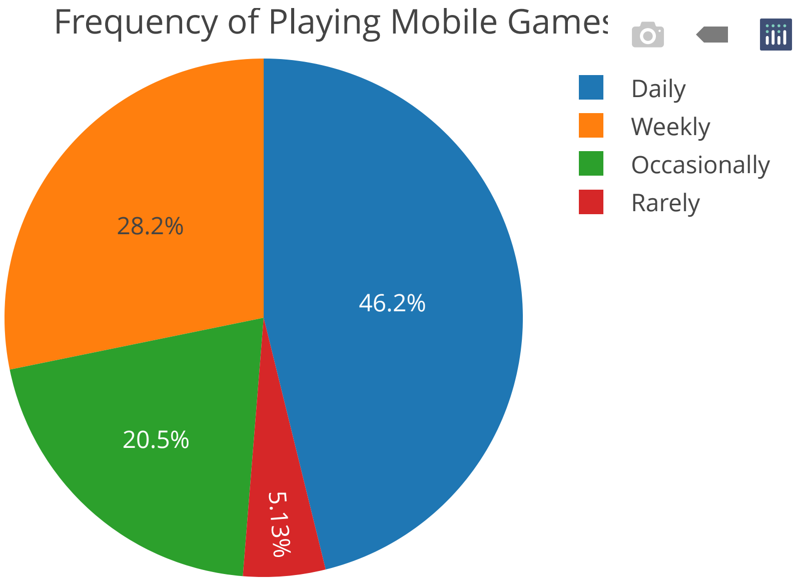


```
# Define the similar responses to be merged
similar_phones <- list(
  c("iOS-based phone", "Iphone")
)

# Function to merge similar phone models
merge_similar_phones <- function(x) {
  for (pair in similar_phones) {
    if (x %in% pair) {
      return(pair[[1]])
    }
  }
  return(x)
}

# Apply the merge_similar_phones function to the phone column
data <- mutate(data, `What.phone.are.you.using.to.play.mobile.games...Select.all.that.applies.` = sappl
```

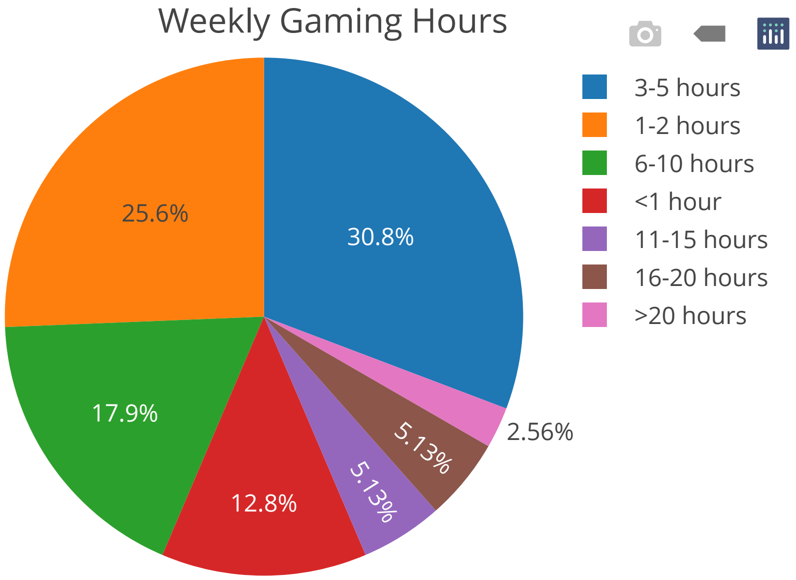
```r
# Recode the category name from "iOS-based phone" to "Iphone"
data <- data %>%
  mutate(`What.phone.are.you.using.to.play.mobile.games...Select.all.that.applies.` = recode(`What.phon

# Aggregate counts for each unique phone model
phone_counts <- data %>%
  group_by(`What.phone.are.you.using.to.play.mobile.games...Select.all.that.applies.`) %>%
  summarise(count = n()) %>%
  ungroup()

# Create horizontal bar graph using Plotly
horizontal_bar_chart <- plot_ly(data = phone_counts, y = ~`What.phone.are.you.using.to.play.mobile.game
  layout(title = "Distribution of Devices Used for Playing Games", yaxis = list(title = "Device", categ

# Print the chart
horizontal_bar_chart
```



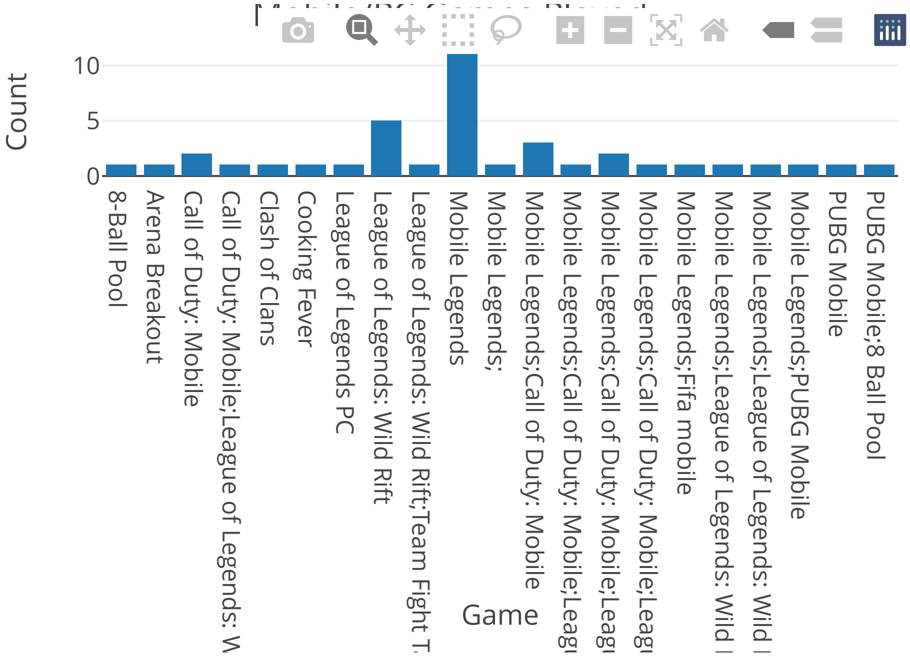
```r
# Aggregate counts for each unique response
frequency_counts <- data %>%
  group_by(How.often.do.you.play.mobile.games.) %>%
  summarise(count = n()) %>%
  ungroup()

# Create pie chart using Plotly
pie_chart <- plot_ly(data = frequency_counts, labels = ~How.often.do.you.play.mobile.games., values = ~
  layout(title = "Frequency of Playing Mobile Games")

# Print the chart
pie_chart
```



```r
# Aggregate counts for each unique response
hours_counts <- data %>%
  group_by(`On.average..how.many.hours.do.you.typically.spend.gaming.in.a.week.`) %>%
  summarise(count = n()) %>%
  ungroup()

# Create pie chart using Plotly
pie_chart_hours <- plot_ly(data = hours_counts, labels = ~`On.average..how.many.hours.do.you.typically.
  layout(title = "Weekly Gaming Hours")

# Print the chart
pie_chart_hours
```

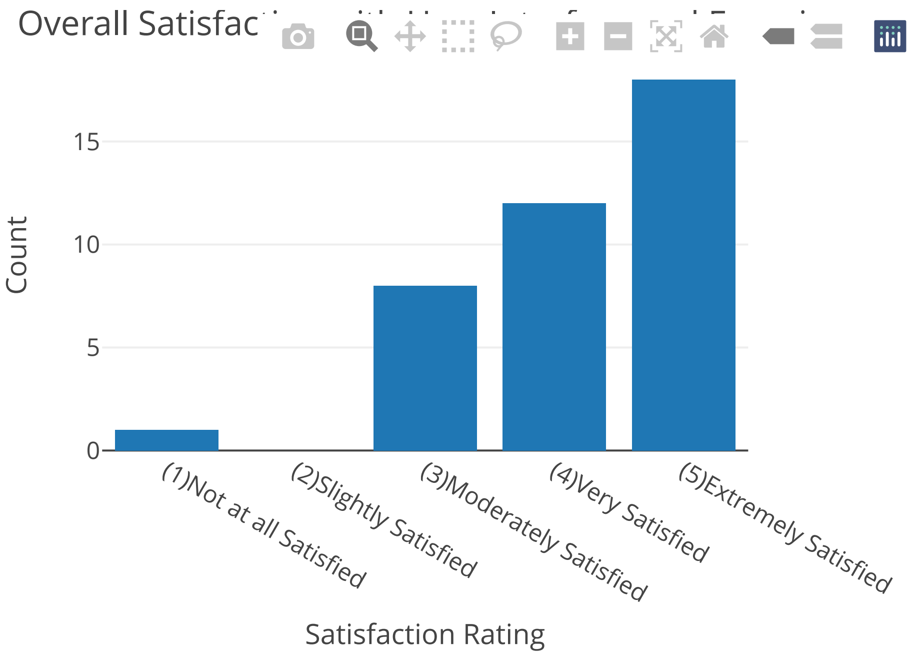


```r
# Define the recode template
recode_template <- c(
  "League of Legends " = "League of Legends PC",
  "coc" = "Clash of Clans",
  "Arenabreakout" = "Arena Breakout",
  "8-Pool" = "8-Ball Pool"
)

# Function to recode the values
recode_games <- function(x) {
  recode_value <- recode_template[x]
  if (!is.na(recode_value)) {
    return(recode_value)
  } else {
    return(x)
  }
}

# Apply the recode function to the column
data <- mutate(data, Which.of.the.following.mobile.games.do.you.play...Select.all.that.apply. = sapply(


# Aggregate counts for each unique game
game_counts <- data %>%
  group_by(Which.of.the.following.mobile.games.do.you.play...Select.all.that.apply.) %>%
  summarise(count = n()) %>%
  ungroup()

# Create bar chart using Plotly
bar_chart_games <- plot_ly(data = game_counts, x = ~Which.of.the.following.mobile.games.do.you.play...Se
  layout(title = "Mobile/PC Games Played", xaxis = list(title = "Game"), yaxis = list(title = "Count"))

# Print the chart
bar_chart_games
```

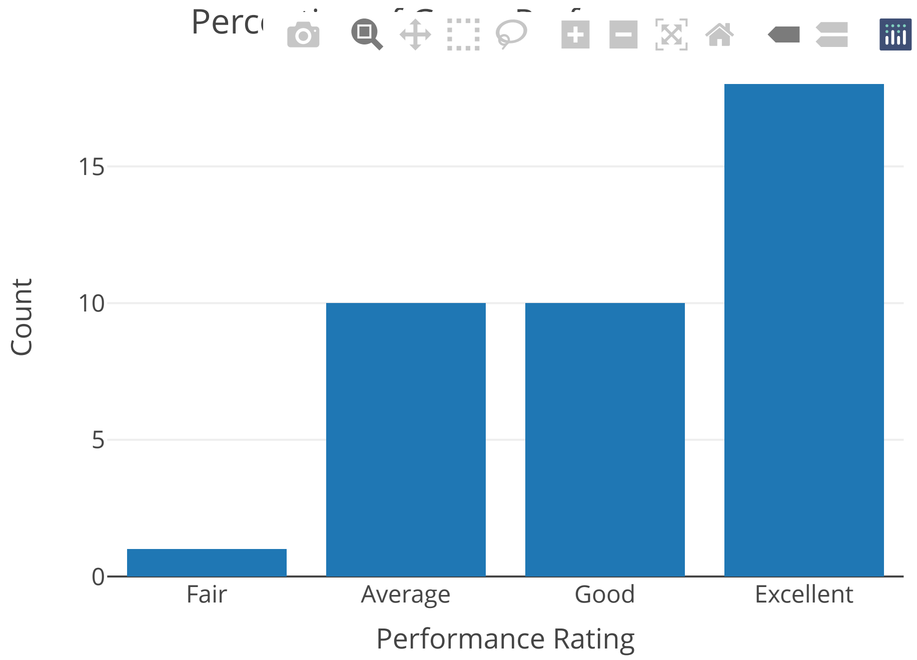


```
# Define the satisfaction labels corresponding to the ratings
satisfaction_labels <- c("(1)Not at all Satisfied", "(2)Slightly Satisfied", "(3)Moderately Satisfied",
```

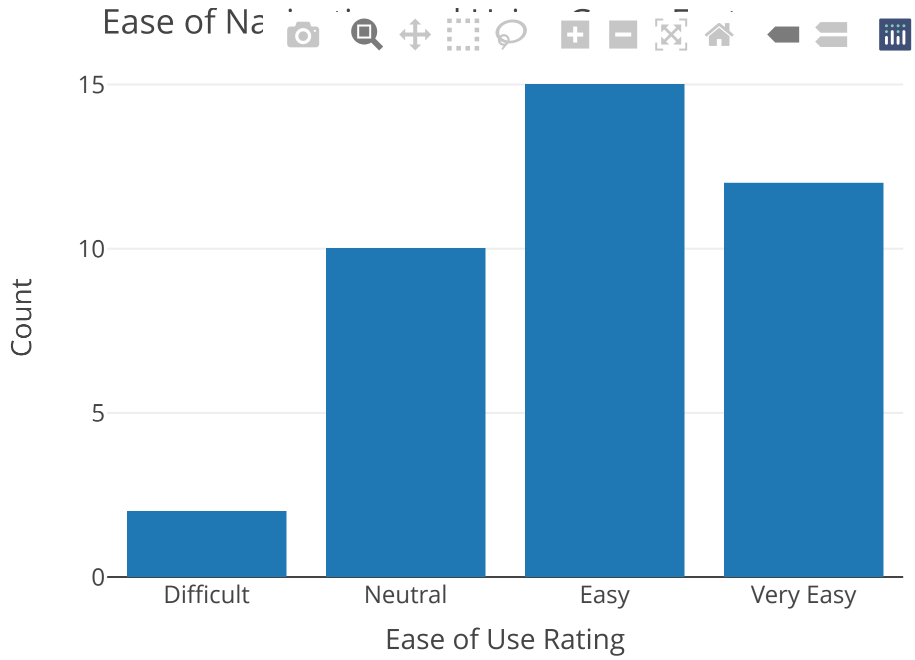
```r
# Aggregate counts for each unique rating
satisfaction_counts <- data %>%
  group_by(Rate.your.overall.satisfaction.with.the.user.interface.and.experience.of.the.games.you.play.
  summarise(count = n()) %>%
  ungroup()

# Create bar chart using Plotly with customized x-axis labels
bar_chart_satisfaction <- plot_ly(data = satisfaction_counts, x = ~factor(Rate.your.overall.satisfactio
  layout(title = "Overall Satisfaction with User Interface and Experience", xaxis = list(title = "Satis

# Print the chart
bar_chart_satisfaction
```



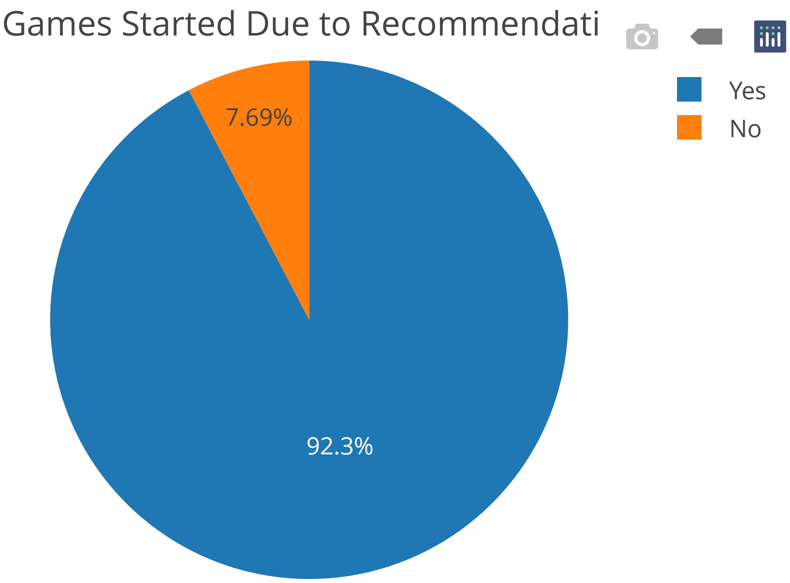
```r
# Define the performance labels corresponding to the ratings
performance_labels <- c("Poor", "Fair", "Average", "Good", "Excellent")

# Aggregate counts for each unique rating
performance_counts <- data %>%
  group_by(`How.do.you.perceive.the.performance..graphics..gameplay..etc...of.the.games.you.play.`) %>%
  summarise(count = n()) %>%
  ungroup()

# Create bar chart using Plotly with customized x-axis labels
bar_chart_performance <- plot_ly(data = performance_counts, x = ~factor(`How.do.you.perceive.the.perfor
  layout(title = "Perception of Game Performance", xaxis = list(title = "Performance Rating", tickvals =

# Print the chart
bar_chart_performance
```



```
# Define the ease of use labels corresponding to the ratings
ease_of_use_labels <- c("Very Difficult", "Difficult", "Neutral", "Easy", "Very Easy")

# Aggregate counts for each unique rating
ease_of_use_counts <- data %>%
  group_by(`How.easy.or.difficult.do.you.find.it.to.navigate.and.use.the.features.in.the.games.`) %>%
  summarise(count = n()) %>%
  ungroup()

# Create bar chart using Plotly with customized x-axis labels
bar_chart_ease_of_use <- plot_ly(data = ease_of_use_counts, x = ~factor(`How.easy.or.difficult.do.you.f
  layout(title = "Ease of Navigating and Using Game Features", xaxis = list(title = "Ease of Use Rating

# Print the chart
bar_chart_ease_of_use
```

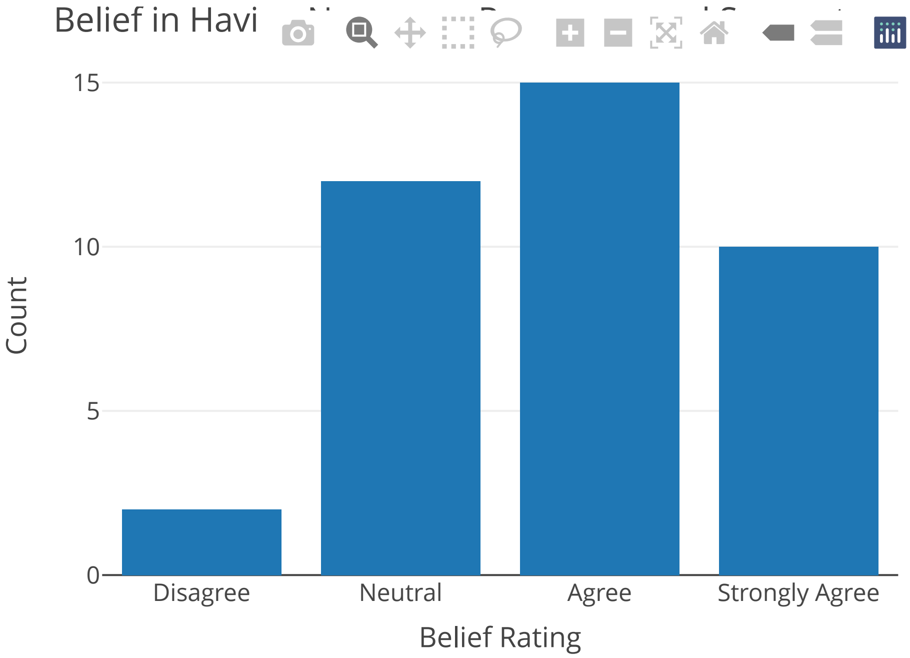


```r
# Aggregate counts for each unique response
recommendation_counts <- data %>%
  group_by(Have.you.started.playing.any.of.these.games.because.of.recommendations.from.friends.or.social
  summarise(count = n()) %>%
  ungroup()

# Create pie chart using Plotly
pie_chart_recommendations <- plot_ly(data = recommendation_counts, labels = ~Have.you.started.playing.an
  layout(title = "Games Started Due to Recommendations", showlegend = TRUE)

# Print the chart
pie_chart_recommendations
```

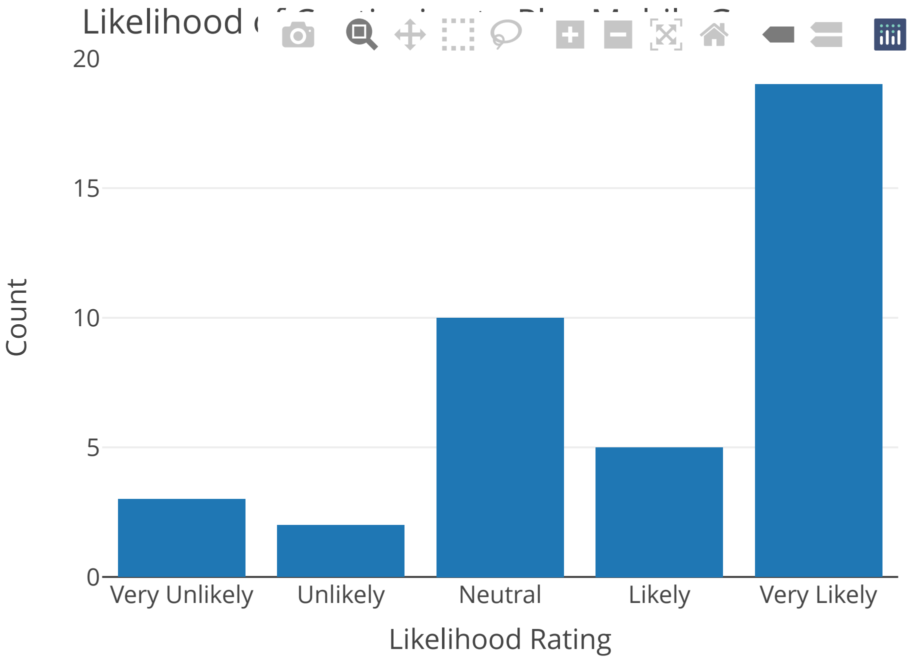


```r
# Define the scale labels corresponding to the ratings
scale_labels <- c("Strongly Disagree", "Disagree", "Neutral", "Agree", "Strongly Agree")
```

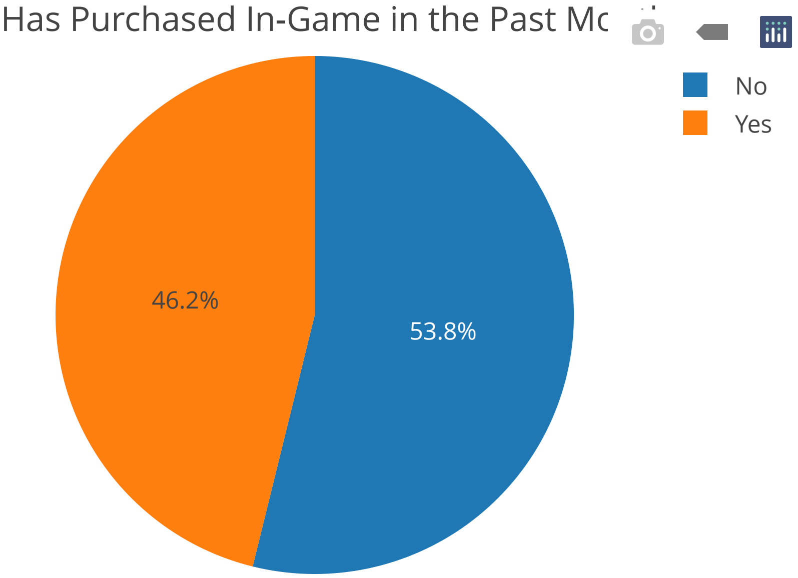
```r
# Aggregate counts for each unique rating
resources_support_counts <- data %>%
  group_by(Do.you.believe.that.you.have.the.necessary.resources.and.support.to.effectively.use.mobile.ga
  summarise(count = n()) %>%
  ungroup()

# Create bar chart using Plotly with customized x-axis labels
bar_chart_resources_support <- plot_ly(data = resources_support_counts, x = ~factor(Do.you.believe.that
  layout(title = "Belief in Having Necessary Resources and Support", xaxis = list(title = "Belief Rating

# Print the chart
bar_chart_resources_support
```



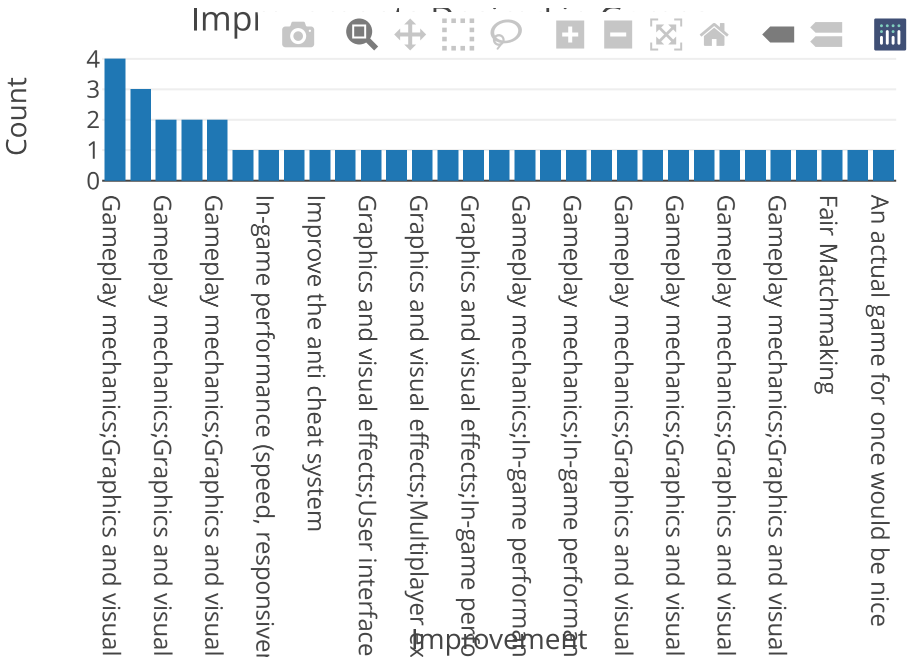
```r
# Define the likelihood labels corresponding to the ratings
likelihood_labels <- c("Very Unlikely", "Unlikely", "Neutral", "Likely", "Very Likely")

# Aggregate counts for each unique rating
likelihood_counts <- data %>%
  group_by(How.likely.are.you.to.continue.playing.mobile.games.in.the.future.) %>%
  summarise(count = n()) %>%
  ungroup()

# Create bar chart using Plotly with customized x-axis labels
bar_chart_likelihood <- plot_ly(data = likelihood_counts, x = ~factor(How.likely.are.you.to.continue.pla
  layout(title = "Likelihood of Continuing to Play Mobile Games", xaxis = list(title = "Likelihood Rati

# Print the chart
bar_chart_likelihood
```



```r
# Aggregate counts for each unique response
purchase_counts <- data %>%
  group_by(Have.you.made.any.in.game.purchases.in.the.past.month.) %>%
  summarise(count = n()) %>%
  ungroup()

# Create pie chart using Plotly with legends displayed
pie_chart_purchases <- plot_ly(data = purchase_counts, labels = ~Have.you.made.any.in.game.purchases.in
  layout(title = "Has Purchased In-Game in the Past Month", showlegend = TRUE)

# Print the chart
pie_chart_purchases
```



```r
data <- mutate(data, improvements = recode(What.improvements.would.you.like.to.see.in.these.games...Sel
                                           "FAIR MATCHMAKING" = "Fair Matchmaking",
                                                "anti cheat system" = "Anti-cheat System"))

# Aggregate counts for each unique improvement
improvement_counts <- data %>%
  group_by(improvements) %>%
  summarise(count = n()) %>%
  ungroup()

# Create vertical bar chart using Plotly
improvements_bar_chart <- plot_ly(data = improvement_counts, x = ~improvements, y = ~count, type = "bar
  layout(title = "Improvements Desired in Games", xaxis = list(title = "Improvement", categoryorder = "

# Print the chart
improvements_bar_chart
```


Impı
Count
Gameplay mechanics;Graphics and visual
Gameplay mechanics;Graphics and visual
Gameplay mechanics;Graphics and visual
In-game performance (speed, responsiven
Improve the anti cheat system
Graphics and visual effects;User interface
Graphics and visual effects;Multiplayer ex
Graphics and visual effects;In-game perfo
Gameplay mechanics;In-game performan
Gameplay mechanics;In-game performan
Gameplay mechanics;Graphics and visual
Gameplay mechanics;Graphics and visual
Gameplay mechanics;Graphics and visual
Gameplay mechanics;Graphics and visual
Fair Matchmaking
An actual game for once would be nice
Improvement