Assignment 3

Name: Vikrant Singh Jamwal

Student ID: 23104534

Goal:

- To analyse Apple App store data using R and perform several tasks.

Importing Tidyverse Library

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.3     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.4     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.0
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Task 1: Reading Apple Store data file.

- read_csv ( ) creates a tibble of the csv file.
- head ( ) visualizes n rows of the tibble.

```
## 1
apple_store <- read_csv("AppleStore.csv")
```

```
## New names:
## Rows: 7197 Columns: 17
## ── Column specification ──────────────────────────────────────────────────────── Delimiter: "," chr
## (5): track_name, currency, ver, cont_rating, prime_genre dbl (12): ...1, id,
## size_bytes, price, rating_count_tot, rating_count_ver, u...
## ℹ Use `spec()` to retrieve the full column specification for this data. ℹ
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## • `` -> `...1`
```

```
head(apple_store, n=5)
```

```
## # A tibble: 5 × 17
##    ...1        id track_name          size_bytes currency price rating_count_tot
##   <dbl>     <dbl> <chr>                    <dbl> <chr>    <dbl>            <dbl>
## 1     1 281656475 PAC-MAN Premium      100788224 USD       3.99            21292
## 2     2 281796108 Evernote - stay or…  158578688 USD       0              161065
## 3     3 281940292 WeatherBug - Local…  100524032 USD       0              188583
## 4     4 282614216 eBay: Best App to …  128512000 USD       0              262241
## 5     5 282935706 Bible                 92774400 USD       0              985920
## # ℹ 10 more variables: rating_count_ver <dbl>, user_rating <dbl>,
## #   user_rating_ver <dbl>, ver <chr>, cont_rating <chr>, prime_genre <chr>,
## #   sup_devices.num <dbl>, ipadSc_urls.num <dbl>, lang.num <dbl>, vpp_lic <dbl>
```

Task 2: Creating a column for frequency of different genres

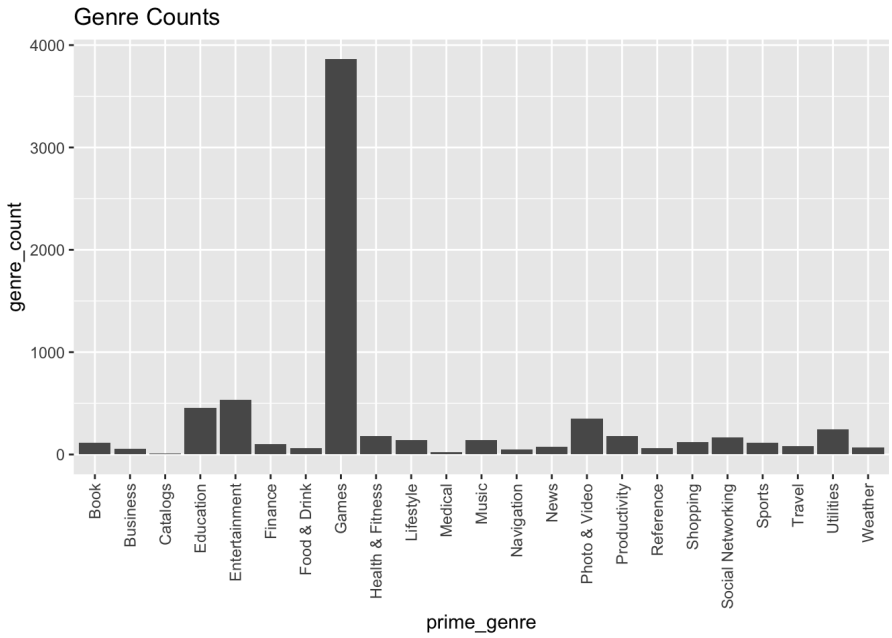
- genre_count holds the count of each genre.

```
## 2
apple_store_t1 <- group_by(apple_store, prime_genre) %>%
             # grouped by prime_genre.
             summarise( genre_count = n())
             # summarise is used to summarise the data in each group (here count is used)
head(apple_store_t1, n=10)
```

```
## # A tibble: 10 × 2
##    prime_genre      genre_count
##    <chr>                  <int>
##  1 Book                     112
##  2 Business                  57
##  3 Catalogs                  10
##  4 Education                453
##  5 Entertainment            535
##  6 Finance                  104
##  7 Food & Drink              63
##  8 Games                   3862
##  9 Health & Fitness         180
## 10 Lifestyle                144
```

Visualizing different genre's count in the dataset.

```
ggplot(apple_store_t1, aes(x = prime_genre, y = genre_count)) +
      geom_bar(stat = "identity")+ # bar plot with "identity" stat.
      labs(title="Genre Counts") + # Title of the plot
      theme(axis.text.x = element_text(angle=90, vjust = 0.5, hjust= 1))
```

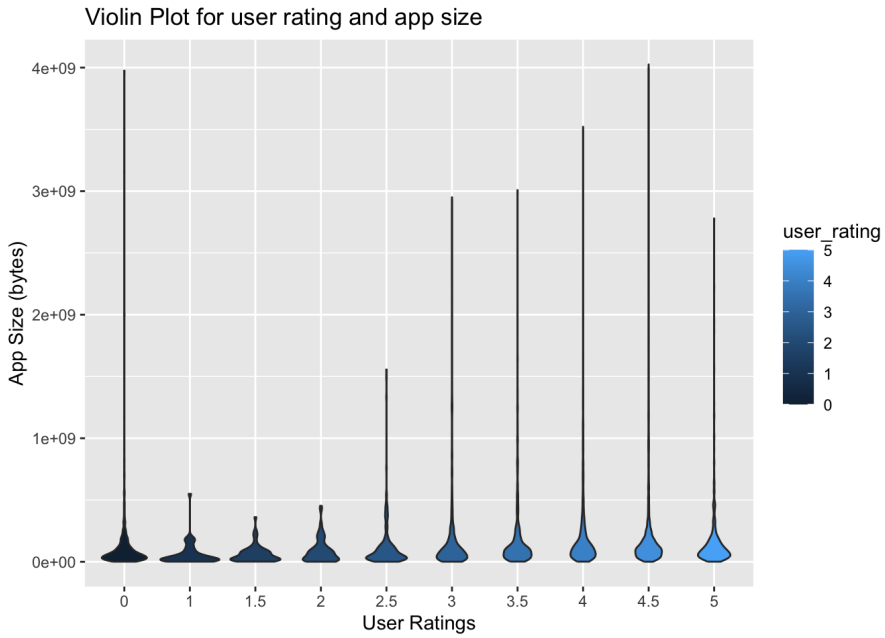



```
      # vjust, hjust -- vertical/height adjust of the x-axis text.
      # stat="count" can also be used to plot barplot without y argument.
```

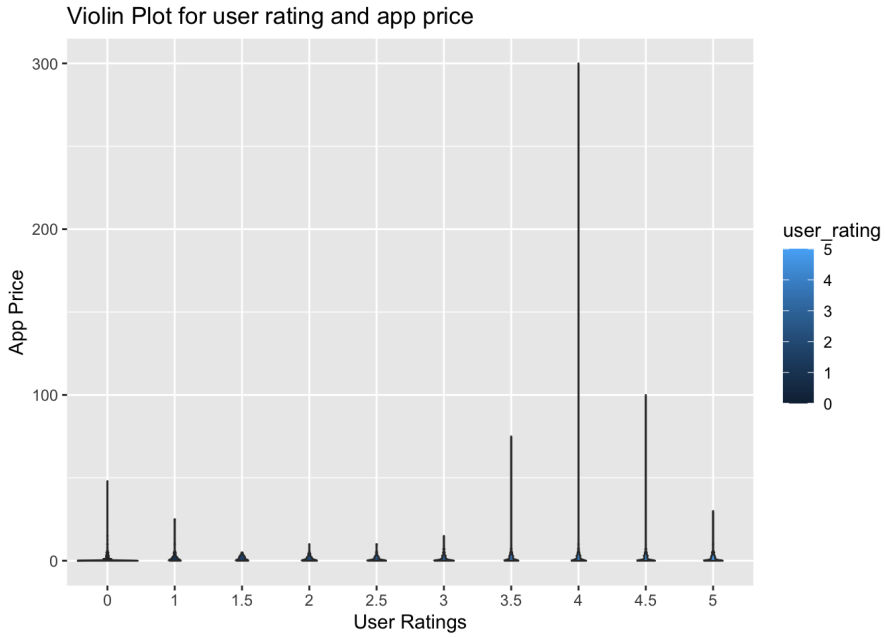
Task 3: Exploring and visualizing data with various plots

- **Violin plot:** A method of data distribution visualization containing aspects of both box plots and kernel density plots. The wider part tells the density of the data and the structure is based on the IQRs similar to box plots.
- **Histogram:** A frequency distribution visualization method. Involves range of continuous variable or factors of categorical variable and frequency of these range/factors.
- **BoxPlot:** Plot used to determine the outliers in the feature. The box represents the interquartile range (IQR), which is the range between the first quartile (Q1) and the third quartile (Q3).
- **Scatter Plot:** These Plots are useful for visualizing the correlation or pattern between two continuous variables.

```
## 3
ggplot(apple_store, aes(x = factor(user_rating), y = size_bytes , fill=user_rating)) +
      geom_violin()+
      labs(title="Violin Plot for user rating and app size") +
      xlab("User Ratings") +
      ylab("App Size (bytes)")
```

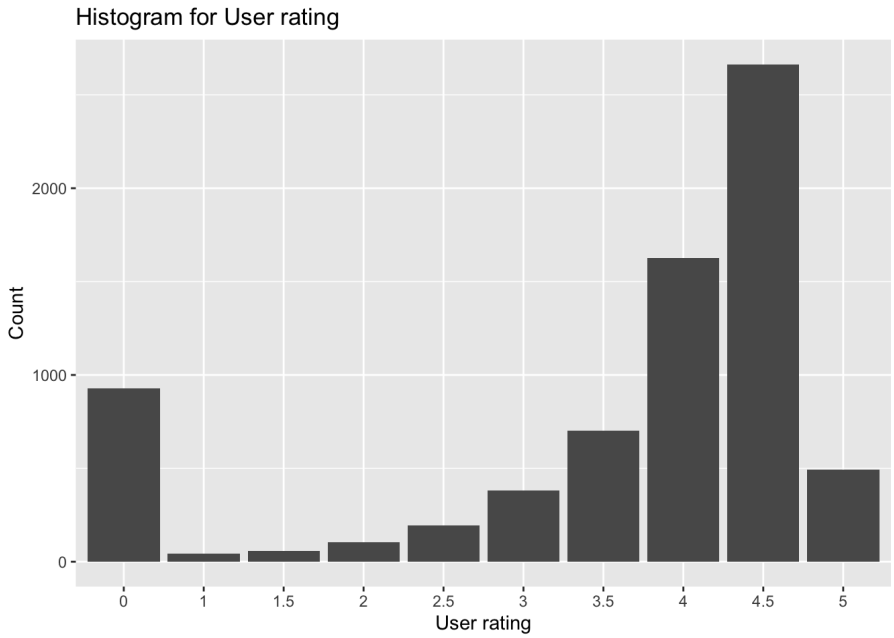



```
ggplot(apple_store, aes(x =  factor(user_rating), y = price , fill=user_rating)) +
      geom_violin()+
      labs(title="Violin Plot for user rating and app price") +
      xlab("User Ratings") +
      ylab("App Price")
```

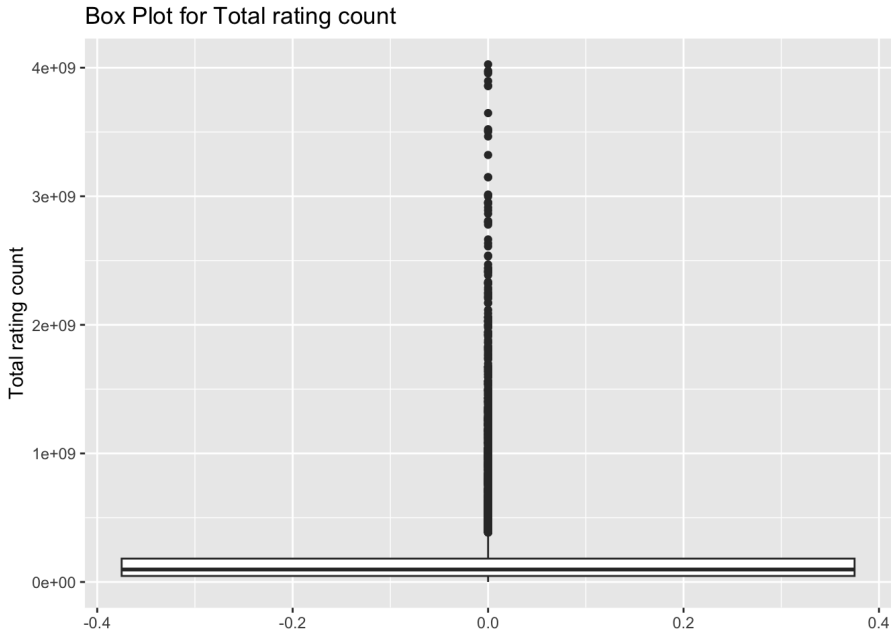



```
ggplot(apple_store, aes(x = factor(user_rating ))) +
      geom_histogram(stat = "count")+
      labs(title="Histogram for User rating") +
      xlab("User rating") +
      ylab("Count")
```

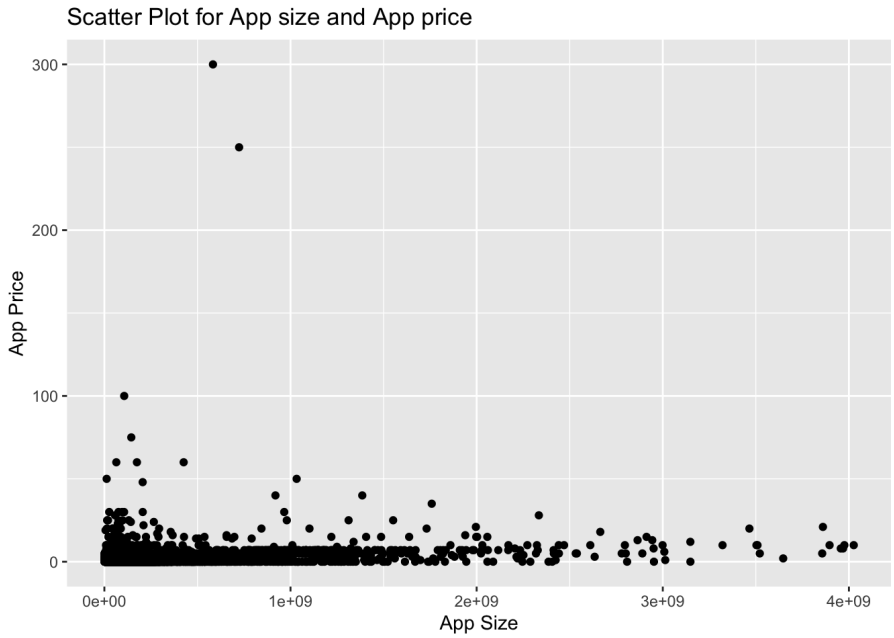
```
## Warning in geom_histogram(stat = "count"): Ignoring unknown parameters:
## `binwidth`, `bins`, and `pad`
```



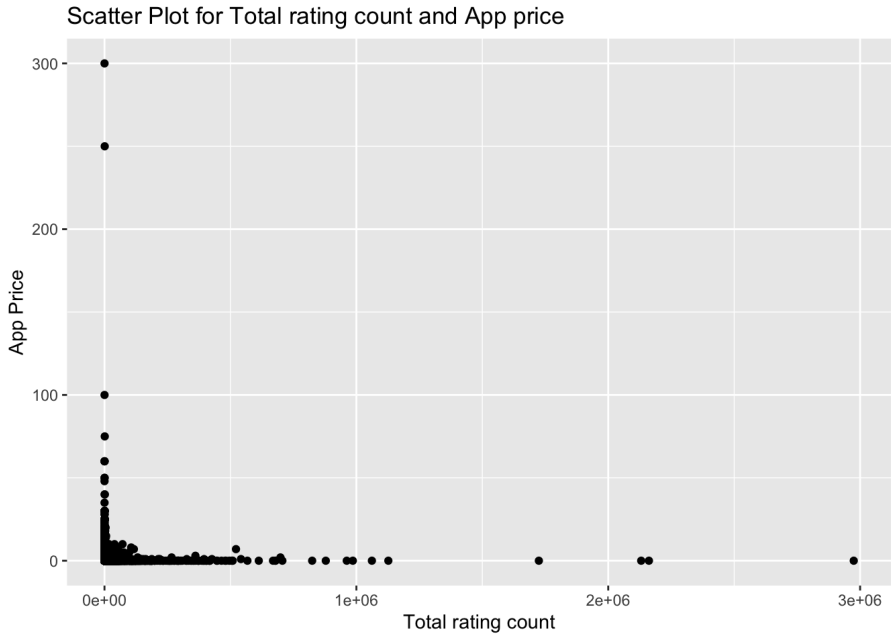

```
ggplot(apple_store, aes(y= size_bytes)) +
      geom_boxplot()+
      labs(title="Box Plot for Total rating count") +
      ylab("Total rating count")
```




```
ggplot(apple_store, aes(x= size_bytes, y= price)) +
      geom_point()+
      labs(title="Scatter Plot for App size and App price") +
      xlab("App Size") +
      ylab("App Price")
```




```
ggplot(apple_store, aes(x= rating_count_tot, y= price)) +
      geom_point()+
      labs(title="Scatter Plot for Total rating count and App price") +
      xlab("Total rating count") +
      ylab("App Price")
```



Task 4: Total size of each genre

```
## 4
apple_store_size <- group_by(apple_store, prime_genre) %>%
          summarise( "Size (bytes)" = sum(size_bytes),
                     "Size (Mb)" = (sum(size_bytes)/(1024*1024)),
                     "Size (Gb)" = (sum(size_bytes)/(1024*1024*1024)))
apple_store_size
```

```
## # A tibble: 23 × 4
##    prime_genre      `Size (bytes)` `Size (Mb)` `Size (Gb)`
##    <chr>                     <dbl>       <dbl>       <dbl>
##  1 Book                20027910144      19100.      18.7
##  2 Business             3657604096       3488.       3.41
##  3 Catalogs              501816320        479.       0.467
##  4 Education           81732172267      77946.      76.1
##  5 Entertainment       54291115746      51776.      50.6
##  6 Finance              8136529366       7760.       7.58
##  7 Food & Drink         4888484864       4662.       4.55
##  8 Games             1095488356523    1044739.    1020.
##  9 Health & Fitness    16219195392      15468.      15.1
## 10 Lifestyle            8972131321       8556.       8.36
## # ℹ 13 more rows
```

Task 5: Correlation between

* User Rating-Size

```
as_cor_size <- cor(apple_store$user_rating, apple_store$size_bytes)
print(as_cor_size)
```

```
## [1] 0.06625572
```

* User Rating-Total Rating Count

```
as_cor_count <- cor(apple_store$rating_count_tot, apple_store$user_rating)
print(as_cor_count)
```

```
## [1] 0.08330997
```

Results:

- As Both are Weak Correlations:
  - Are larger apps (larger size in bytes) higher rated by customers?

    - Can not determine Accurately as correlation is weak and close to 0.

  - Are apps with more ratings higher rated?

    - Can not determine Accurately as correlation is weak and close to 0.

Task 6: Creating a “paid” column

- Two columns are created “Paid”- (0,1) and “Paid Description”- (Free, Paid)

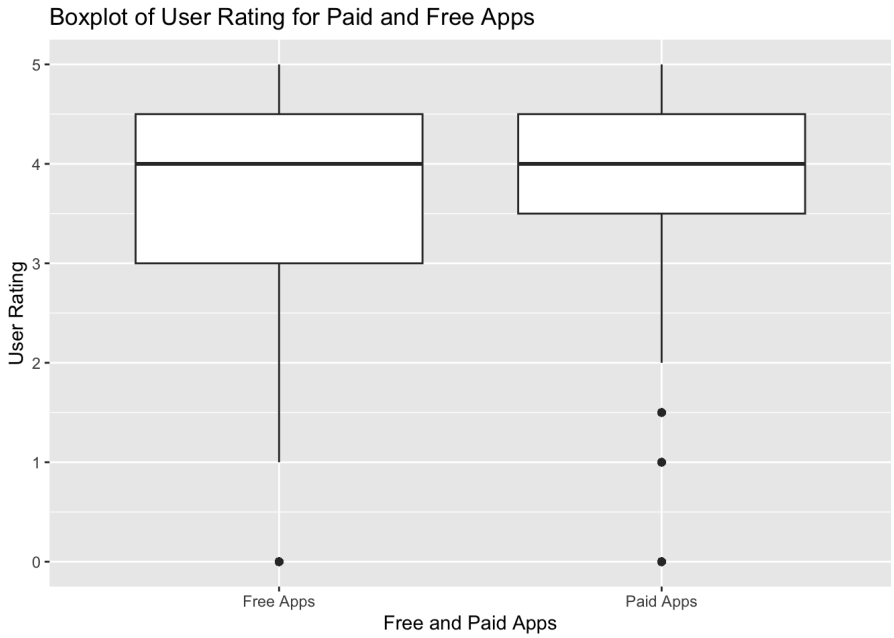
```
## 6
apple_store_paid <- apple_store %>%
          mutate(paid = ifelse (price>0 , 1, 0))

apple_store_paid <- apple_store_paid %>%
          mutate(paid_description = ifelse (paid==0 , "Free", "Paid"));

paid_mean <- apple_store_paid %>%
            group_by("Paid Description"= paid_description) %>%
            summarize("User Rating Mean" = mean(user_rating));
paid_mean
```

```
## # A tibble: 2 × 2
##   `Paid Description` `User Rating Mean`
##   <chr>                           <dbl>
## 1 Free                             3.38
## 2 Paid                             3.72
```

- As the mean of User ratings for Paid Applications is higher than that of Free Applications, we can say that Paid Applications are rated higher than Free Applications in general.

Visualizing User ratings for Free and Paid Apps using Box Plots

```
ggplot(apple_store_paid, aes(x = factor(paid, labels = c("Free Apps", "Paid Apps")), y = user_rating)) +
      geom_boxplot() +
      labs(title = "Boxplot of User Rating for Paid and Free Apps" )+
      xlab("Free and Paid Apps") +
      ylab("User Rating");
```



Task 7: App with highest User Rating per byte

```
##7
dt_max_rpb <- apple_store %>% mutate(rpb = user_rating/size_bytes) %>%
              arrange(desc(rpb)) %>%
              select("Application name" = track_name) %>%
              slice(1)
dt_max_rpb
```

```
## # A tibble: 1 × 1
##   `Application name`
##   <chr>
## 1 GraphModeling
```

- **Graph Modeling** Application has the highest User Rating per byte value.