Deliverable 1

Brandon Leff

```
## -- Attaching packages --------------------------------------------------------------------------------

## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ------------------------------------------------------------------------------------ tidy
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

## Learn more about sjPlot with 'browseVignettes("sjPlot")'.
```

Introduction

In order to understand why I am diving into this topic, I need to explain the motivation behind it. Growing up, video games have been a big aspect of not only my life, but society in general. Lately there has been a change in the climate of gaming from single player relaxing games to the creation of competitve online gaming and eSports as we know it today. A very influential aspect of this change is the introduction of microtransactions and virtual currency. I want to research mobile games specifically to see the effect of microtransactions on things such as average ratings, game sales, and other mobile game statistics.

Data Set

```
apps <- read_csv("data/AppleStore.csv")
```

```
## Parsed with column specification:
## cols(
##   id = col_double(),
##   track_name = col_character(),
##   size_bytes = col_double(),
##   currency = col_character(),
##   price = col_double(),
##   rating_count_tot = col_double(),
##   rating_count_ver = col_double(),
##   user_rating = col_double(),
##   user_rating_ver = col_double(),
##   ver = col_character(),
##   cont_rating = col_character(),
##   prime_genre = col_character(),
##   sup_devices.num = col_double(),
##   ipadSc_urls.num = col_double(),
##   lang.num = col_double(),
##   vpp_lic = col_double(),
##   game_enab = col_double()
## )
```

The data set I will be using is `AppleStore.csv` which is a comma separated values data set about app store statistics. There are 11,100 observations of 17 variables. Every row is a specific app and there are 17 columns describing certain aspects of each app. Out of the 17 variables, there are 12 variables that are doubles and 5 that are characters. I will cover the variables deeper later.

Data Set Source

The source I got my data from was a Kaggle posting. The following github account is the creator.

https://github.com/ramamet/applestoreR)

The data was scraped using Linux and R web scraping tools from the iTunes Search API.

The strongsuits of this data set is that it comes from Apple themselves from their Apple Inc website using scraping tools to where there is less people who have potentially touched the data set. The potential shortcoming of the data set is that it was from July 2017 meaning that there could be updates of data in the current day that differ from the data set.

Variables

There are 17 variables in this data set.

`id` is the app ID which is a number uniquely assigned to the app.

`track_name` is the app name.

`size_bytes` is the size of the app in bytes.

`currency` is the type of currency for the price of the app.

`price` is the price of the app.

`rating_count_tot` is the user rating counts for all versions of the app.

`rating_count_ver` is the user rating counts for the current version of the app.

`user_rating` is the average user rating value for all versions of the app.

`user_rating_ver` is the average user rating value for the current version of the app.

`ver` is the latest version code.

`cont_rating` is the content rating of the app.

`prime_genre` is the primary genre of the app.

`sup_devices.num` is the number of supporting devices for the app.

`ipadSc_urls.num` is the number of screenshots shown for display in the app store.

`lang.num` is the number of supported languages for the app.

`vpp_lic` is if the Vpp Device Based Licensing is enabled.

`game_enab` is not explained in the data set and is a column of 0's so we will delete this variable.

Data Cleaning and Preparing Tidy Data

In order to clean this data set I had to do deep research into the data set. I first realized that there were 11100 observations, but in the Kaggle data set description, I saw that there were 7197 observations. When looking at the data set, I realized that there were `NA` values for `track_name` which is the app name. I want to delete the rows where the `track_name` is `NA` and store that into a new data frame called `apps2`.

```r
apps2 <- apps %>%
  filter(!is.na(track_name))
nrow(apps2)
```

```
## [1] 7197
```

After deleting the rows with `NA` values for `track_name` we now have the correct number of observations (7197) for the data set.

The next step in the data cleaning process is to change certain variables from character data to factors. Factors are a statistical data type used to store categorical data. I am specifically changing `cont_rating`, which is the content rating with units of age, to a factor with the levels of 4+, 9+, 12+, and 17+.

```r
apps2$cont_rating <- factor(apps2$cont_rating, levels = c("12+", "17+", "4+", "9+"))
apps2$cont_rating <- apps2$cont_rating %>% fct_relevel("4+","9+","12+","17+")
```

After making those changes to the `apps2` data frame, I now want to break up the data frame into smaller tibbles specifically defined for different aspects of the project. `basic_info` is a data frame including the apps' identification and basic information that doesn't involve people's ratings on the app. In this process, I am taking previous variables from the `apps2` data frame and renaming them into this new categorized tibble. The new variables include:

`app_id`

`app_name`

`size_bytes`

`currency_type`

`app_price`

`app_version`

`age_rating`

`app_genre`

`devices_supported`

`number_screenshots`

`languages_supported`

`vpp_lic`

```r
basic_info <- tibble(app_id = apps2$id,
                     app_name = apps2$track_name,
                     size_bytes = apps2$size_bytes,
                     currency_type = apps2$currency,
                     app_price = apps2$price,
                     app_version = apps2$ver,
                     age_rating = apps2$cont_rating,
                     app_genre = apps2$prime_genre,
                     devices_supported = apps2$sup_devices.num,
                     number_screenshots = apps2$ipadSc_urls.num,
                     languages_supported = apps2$lang.num,
                     vpp_lic = apps2$vpp_lic)
```

The second tibble I am making is called `ratings` which is a data frame including the app's ID and then the rating variables. This process is the same as above where I am taking variabels from `apps2` and changing the names. The new variables include:

`app_id`

`rating_count_tot`

`rating_vount_ver`

`avg_rating_tot`

`avg_rating_ver`

```r
ratings <- tibble(app_id = apps2$id,
                  rating_count_tot = apps2$rating_count_tot,
                  rating_count_ver = apps2$rating_count_ver,
                  avg_rating_tot = apps2$user_rating,
                  avg_rating_ver = apps2$user_rating_ver)
```

The last thing I want to do for data cleaning is to create a new variable in `ratings` called `diff_rating_ver` which will be the difference between the average rating for the current version and the average rating for all versions. This will be potentially useful for future analysis to see if an app has improved recently.

```r
ratings$diff_rating_ver <- ratings$avg_rating_ver - ratings$avg_rating_tot
head(ratings[,c(4,5,6)])
```

```
## # A tibble: 6 x 3
##   avg_rating_tot avg_rating_ver diff_rating_ver
##            <dbl>          <dbl>           <dbl>
## 1            4              4.5             0.5
## 2            4              3.5            -0.5
## 3            3.5            4.5             1
## 4            4              4.5             0.5
## 5            4.5            5               0.5
## 6            4              4               0
```

After creating this new variable, I am using the `head()` function to see the first 5 rows of `ratings` but specifically only `avg_rating_tot`, `avg_rating_ver`, and the new variable `diff_rating_ver`. Doing so allow us to make sure that the new variable was made correctly and in this case we can confirm that the variable was made correctly.

Now that I have completed the data cleaning, I am going to export the tibbles I have created and put them into my `/data` folder for future use. Doing so will allow for future use without having to run this document and when I come back to this document in the future to make changes, I can update the files in the `/data` folder.

```r
write.csv(basic_info,"data/basic_info.csv", row.names = FALSE)
write.csv(ratings,"data/ratings.csv", row.names = FALSE)
```

Data Visualization

Now that the data is tidy, we are ready to show simple descriptive statistics and plots. I want to first dive into `ratings`.

I am going to use the `summary()` function to show the 5 number summary and the mean for `ratings$rating_count_tot`.

```r
summary(ratings$rating_count_tot)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0      28     300   12893    2793 2974676
```

After seeing, the results I see there is a huge discrepency from the 3rd quartile and the maximum value and I also want to see how many apps have 0 ratings so next I will find the count of how many apps have 0 ratings using the `match()` function.

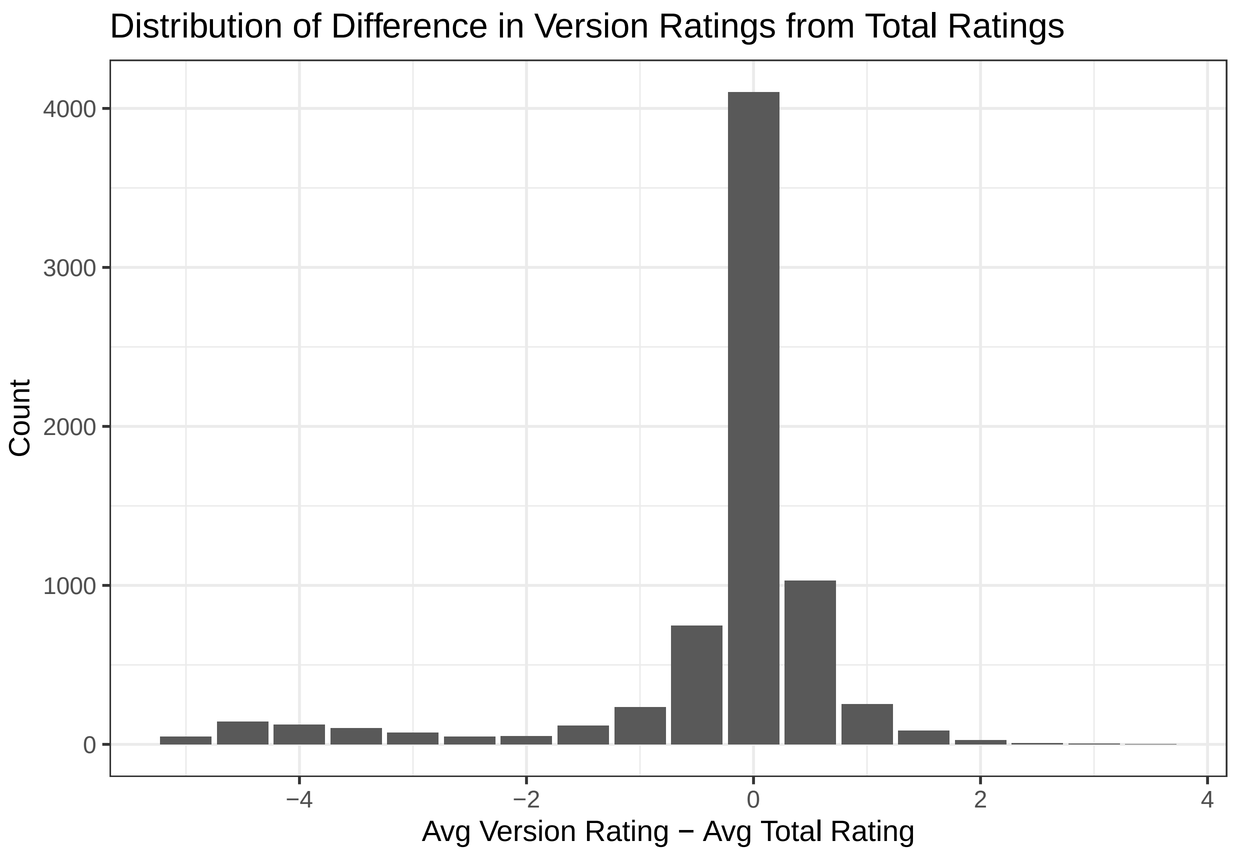
```
match(0,ratings$rating_count_tot)
```

```
## [1] 200
```

After seeing that 200 apps have 0 total ratings, I will have to be ready for that in the future for analysis.

The next thing I want to look into is the new variable I created `diff_rating_tot`

```
ggplot(ratings,aes(x=diff_rating_ver)) + geom_bar() + theme_bw() + labs(x = "Avg Version Rating - Avg To
```



Looking at the graph above, the large majority of apps have 0 difference between the average rating for the current version and the average rating of all versions. The are more apps that have a negative value for `diff_rating_ver` which means that the average rating for all versions is higher than the average rating for the current version. This could be due to the amount of ratings but that will be during analysis.

The next thing I want to look at is from within `basic_info` if the variable `age_rating` has a relationship with the variable `app_price`. I will use a grouped barchart for this visualization.

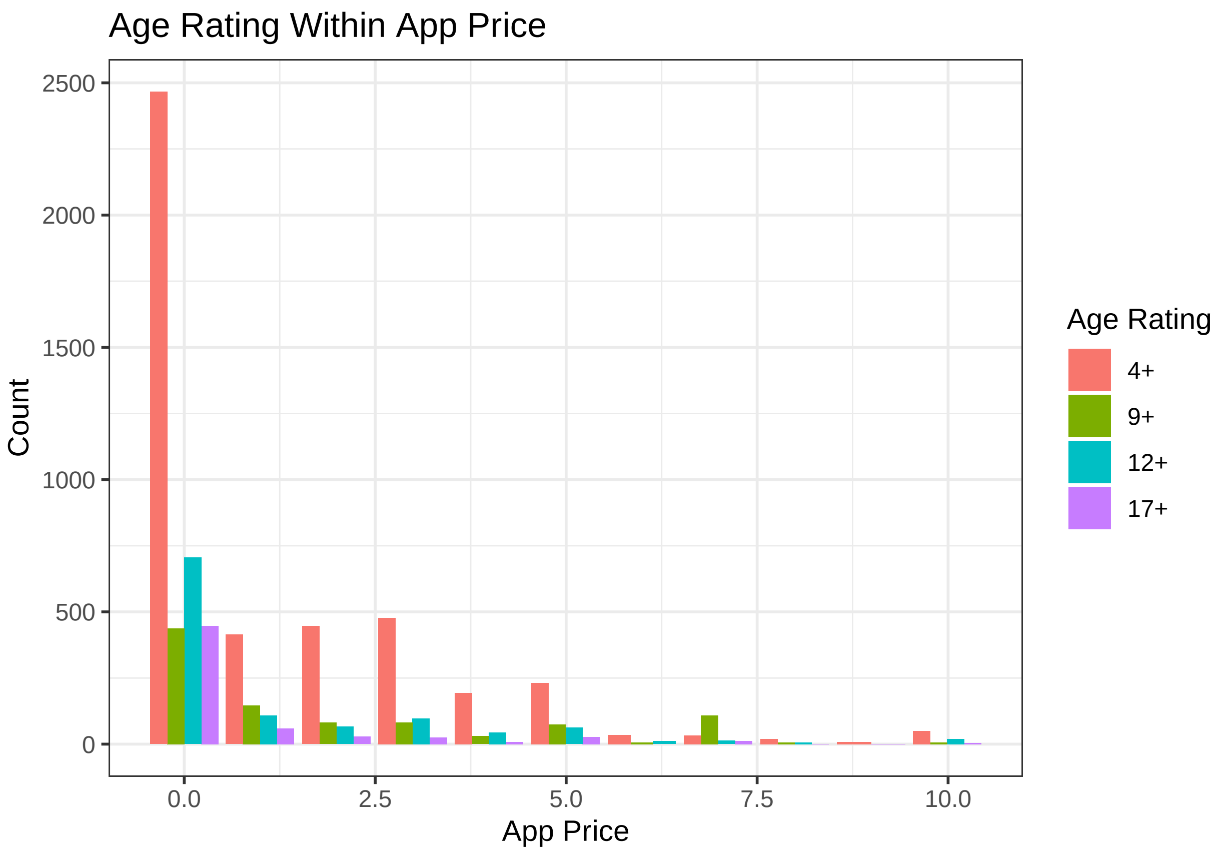
The first thing I want to do is check the counts of `app_price` to see where the data mostly lives.

```
table(basic_info$app_price)
```

```
##
##      0   0.99   1.99   2.99   3.99   4.99   5.99   6.99   7.99   8.99
##   4056    728    621    683    277    394     52    166     33      9
##   9.99  11.99  12.99  13.99  14.99  15.99  16.99  17.99  18.99  19.99
##     81      6      5      6     21      4      2      3      1     13
##  20.99  21.99  22.99  23.99  24.99  27.99  29.99  34.99  39.99  47.99
##      2      1      2      2      8      2      6      1      2      1
##  49.99  59.99  74.99  99.99 249.99 299.99
##      2      3      1      1      1      1
```

The large majority of the data is less than 10.00 therefore for the graph I will only include the apps with prices strictly under 10.00.

```
basic_info %>% filter(app_price < 10) %>% ggplot(aes(x = app_price, fill = age_rating)) + geom_bar(posi
```



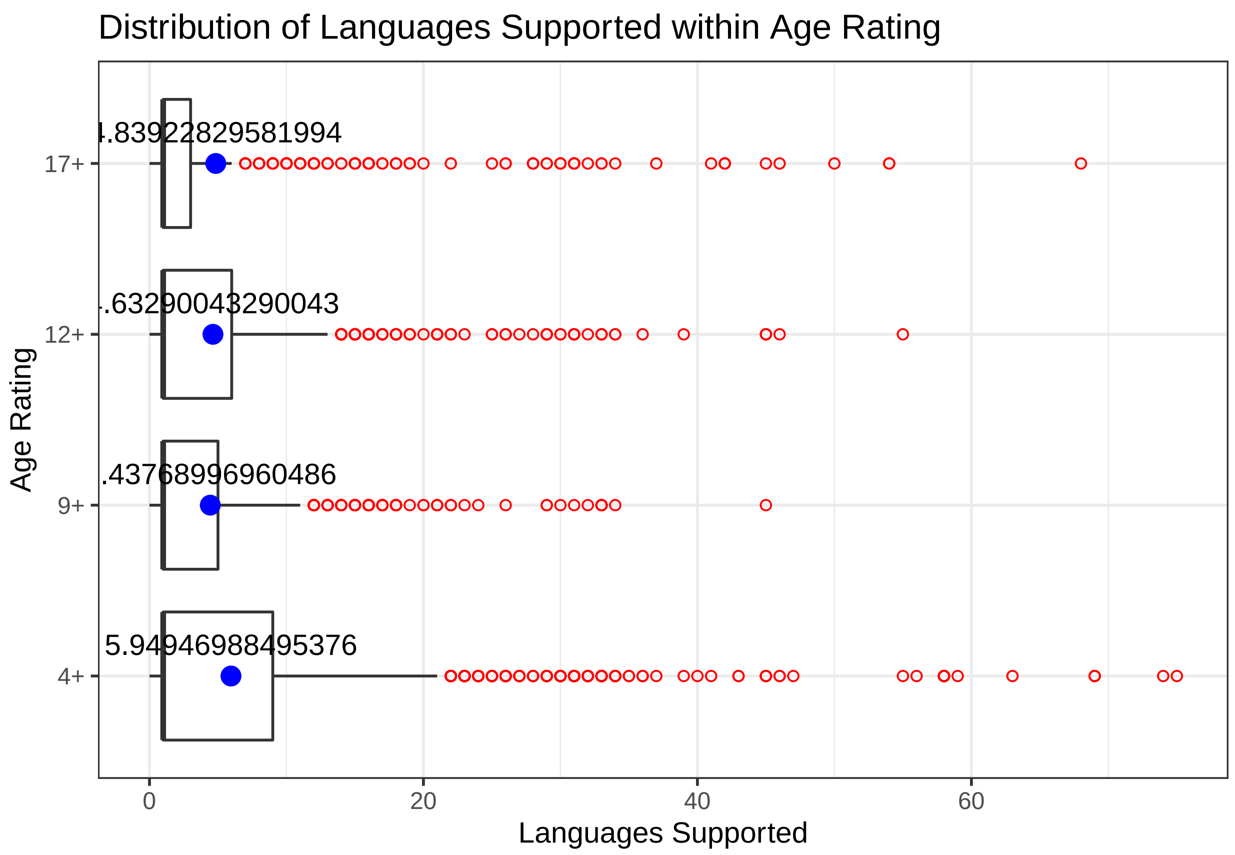
Looking at the graph abofe, within each app price, the majority of apps are 4+, but the largest amount of data are apps that are 0.0 dollars and 4+ for the age rating. The same pattern typically follows for all app prices where 4+ is the most common with 12+, 9+, and 17+ following in the respective order.

The last visualization I want to look at is a simple box plot in the `basic_info` data frame of the variable `languages_supported` within Age Rating. This will show some basic information about if language support is important to age rating.

I also want to show the mean for each boxplot so I am defining a new function `fun_mean` which will take the mean of a given data set.

```
fun_mean <- function(x){
  return(data.frame(y=mean(x),label=mean(x,na.rm=T)))}
```

```
basic_info %>% ggplot(aes(x = age_rating, y = languages_supported)) + geom_boxplot(outlier.colour = "re
```



Looking at the graph above, the medians of languages supported are all relatively similar for apps of all age rating. To better understand the distribution, I also included a blue dot for each category representing the mean. There are heavy amounts of outliers for all categories, but 4+ has the most amount of outliers which ultimately affect the mean.

Potential Research Questions

After looking into the data, I have seen a very interesting project in the `age_rating` variable. I am going to switch the focus of this project away from microtransactions and I would like to know if ratings for apps differ between age ratings.

1.) Do average ratings differ for each age group? Would there be a positive in making your app a certain age level?

2.) Do age groups have a specific price of the app? Are most apps that are 4+ free?

3.) Do age groups have a correlation between what genre the app is?

Conclusion

This data set is particularly interesting because it has aspects that are controlled and never touched such as the price or other aspects that the developer sets in the app store, but also has personal responses such as ratings for the current version or also the total average rating. There will be alot of potential areas for analysis and I am excited for it.