Analysis of Google Play Store Apps

Minhaz Khan

January 12, 2019

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.1.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.8
## v tidyr   0.8.2     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

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

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(klaR)
```

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

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(broom)
library(tidytext)
```

```
## Warning: package 'tidytext' was built under R version 3.5.3
```

```
library(rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 3.5.3
```

```
library(modelr)
```

```
## Warning: package 'modelr' was built under R version 3.5.3
```

```
##
## Attaching package: 'modelr'
```

```
## The following object is masked from 'package:broom':
##
##     bootstrap
```

Introduction

Have you ever come across an app, saw it had a 4.5 star rating and it described exactly what you were looking for but then found out the app was extremely buggy? After taking a closer look at the app page you find out it has about a thousand installs and only a couple of hundred reviews. Majority of apps in the Google Play store and any app store in general are misleading like this due to several factors such as number of installs and reviews as mentioned above along with when it was last updated, the OS version it supports and etc. In this analysis we will be exploring the true rating of the app.

Reading in the Data

```
apps = read_csv("googleplaystore.csv")
```

```
## Parsed with column specification:
## cols(
##   App = col_character(),
##   Category = col_character(),
##   Rating = col_double(),
##   Reviews = col_integer(),
##   Size = col_character(),
##   Installs = col_character(),
##   Type = col_character(),
##   Price = col_character(),
##   `Content Rating` = col_character(),
##   Genres = col_character(),
##   `Last Updated` = col_character(),
##   `Current Ver` = col_character(),
##   `Android Ver` = col_character()
## )
```

```
## Warning: 2 parsing failures.
## row # A tibble: 2 x 5 col     row col     expected              actual     file
expected   <int> <chr>   <chr>                 <chr>      <chr>                 actual 1 10473
Reviews no trailing characters .0M        'googleplaystore.csv' file 2 10473 <NA>    13 columns
12 columns 'googleplaystore.csv'
```

```
head(apps)
```

```
## # A tibble: 6 x 13
##   App   Category Rating Reviews Size  Installs Type  Price `Content Rating`
##   <chr> <chr>     <dbl>   <int> <chr> <chr>    <chr> <chr> <chr>
## 1 Phot~ ART_AND~    4.1     159 19M   10,000+  Free  0     Everyone
## 2 Colo~ ART_AND~    3.9     967 14M   500,000+ Free  0     Everyone
## 3 U La~ ART_AND~    4.7   87510 8.7M  5,000,0~ Free  0     Everyone
## 4 Sket~ ART_AND~    4.5  215644 25M   50,000,~ Free  0     Teen
## 5 Pixe~ ART_AND~    4.3     967 2.8M  100,000+ Free  0     Everyone
## 6 Pape~ ART_AND~    4.4     167 5.6M  50,000+  Free  0     Everyone
## # ... with 4 more variables: Genres <chr>, `Last Updated` <chr>, `Current
## #   Ver` <chr>, `Android Ver` <chr>
```

Average rating of each genre

Filtered out the missing rating values and stored them in a new data frame which we will use for the rest of the analysis.

```
apps %>% filter(Rating != 'NaN') -> true_apps
true_apps %>% group_by(Genres) %>% summarise(avg=mean(Rating))
```

```
## # A tibble: 116 x 2
##    Genres                          avg
##    <chr>                         <dbl>
##  1 Action                         4.29
##  2 Action;Action & Adventure      4.31
##  3 Adventure                      4.18
##  4 Adventure;Action & Adventure   4.42
##  5 Adventure;Brain Games          4.6
##  6 Adventure;Education            4.1
##  7 Arcade                         4.30
##  8 Arcade;Action & Adventure      4.35
##  9 Arcade;Pretend Play            4.5
## 10 Art & Design                   4.36
## # ... with 106 more rows
```

Most genres have a decent ratng around 4.0 but majority of these genres are similar to each other, something that we will simplify later on.

Getting a glimpse of the data to convert the numerical data from being characters

```
glimpse(true_apps)
```

```
## Observations: 9,367
## Variables: 13
## $ App              <chr> "Photo Editor & Candy Camera & Grid & ScrapBo...
## $ Category         <chr> "ART_AND_DESIGN", "ART_AND_DESIGN", "ART_AND_...
## $ Rating           <dbl> 4.1, 3.9, 4.7, 4.5, 4.3, 4.4, 3.8, 4.1, 4.4, ...
## $ Reviews          <int> 159, 967, 87510, 215644, 967, 167, 178, 36815...
## $ Size             <chr> "19M", "14M", "8.7M", "25M", "2.8M", "5.6M", ...
## $ Installs         <chr> "10,000+", "500,000+", "5,000,000+", "50,000,...
## $ Type             <chr> "Free", "Free", "Free", "Free", "Free", "Free...
## $ Price            <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", ...
## $ `Content Rating` <chr> "Everyone", "Everyone", "Everyone", "Teen", "...
## $ Genres           <chr> "Art & Design", "Art & Design;Pretend Play", ...
## $ `Last Updated`   <chr> "January 7, 2018", "January 15, 2018", "Augus...
## $ `Current Ver`    <chr> "1.0.0", "2.0.0", "1.2.4", "Varies with devic...
## $ `Android Ver`    <chr> "4.0.3 and up", "4.0.3 and up", "4.0.3 and up...
```

We have a lot of character variables here that we need as numbers, we will be converting them below

Seperating Numerical and Character values from columns

```
true_apps %>% filter(Size != "Varies with device") %>%
  separate(Size, c("Size","Type"), sep = -1, convert = TRUE) %>%
  separate(Installs, c("Installs","Symbol"), sep = -1, convert = TRUE) %>% drop_na() -> apps2

apps2$Price = parse_number(apps2$Price)

apps2$Symbol = NULL
apps2$Category = NULL
apps2$`Current Ver`= NULL
```

Removed Symbol because it was just character; removed category because it was the same as genre, removed Current Version because it isn’t as necessary as when the app was last updated. Kept Android version because exploring app compatibility with OS might be interesting.

Converting character values to numeric

The convert parameter in seperate didn’t work so here we are manually converting Installs and Size to numeric variables

```
apps2$Installs = as.numeric(gsub(",","",apps2$Installs))
apps2$Size = as.numeric(as.character(apps2$Size))
glimpse(apps2)
```

```
## Observations: 7,728
## Variables: 11
## $ App              <chr> "Photo Editor & Candy Camera & Grid & ScrapBo...
## $ Rating           <dbl> 4.1, 3.9, 4.7, 4.5, 4.3, 4.4, 3.8, 4.1, 4.4, ...
## $ Reviews          <int> 159, 967, 87510, 215644, 967, 167, 178, 36815...
## $ Size             <dbl> 19.0, 14.0, 8.7, 25.0, 2.8, 5.6, 19.0, 29.0, ...
## $ Type             <chr> "M", "M", "M", "M", "M", "M", "M", "M", "M", ...
## $ Installs         <dbl> 1e+04, 5e+05, 5e+06, 5e+07, 1e+05, 5e+04, 5e+...
## $ Price            <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ `Content Rating` <chr> "Everyone", "Everyone", "Everyone", "Teen", "...
## $ Genres           <chr> "Art & Design", "Art & Design;Pretend Play", ...
## $ `Last Updated`   <chr> "January 7, 2018", "January 15, 2018", "Augus...
## $ `Android Ver`    <chr> "4.0.3 and up", "4.0.3 and up", "4.0.3 and up...
```

Converting dates

```
apps2$`Last Updated` = gsub(",","",apps2$`Last Updated`)
apps2 %>% mutate(`Last Update` = mdy(`Last Updated`)) -> apps2
```

Attempting to factor prices as paid or not:

```
apps2 %>% mutate(Price = case_when(
                              Price == 0 ~ Price,
                              Price > 0 ~ Price/Price,
                              TRUE ~ NA_real_)) -> apps2
```

Converting kilobyte app size

```
apps2 %>% mutate(SIZE = case_when(
                              Type == "M" ~ Size,
                              Type == "k" ~ Size/1024,
                              TRUE ~ NA_real_)) -> apps2

apps2 %>% separate_rows(Genres, sep = ";", convert = FALSE) -> apps2

apps2$Size = NULL
apps2$Type = NULL
```

The case_when function helped out here because values in the columns to be changed aren’t the same.

Some Descriptive Statistics

```
apps2 %>% group_by(Genres) %>% summarise(m=mean(Rating))
```

```
## # A tibble: 53 x 2
##    Genres                 m
##    <chr>              <dbl>
##  1 Action              4.27
##  2 Action & Adventure  4.31
##  3 Adventure           4.22
##  4 Arcade              4.30
##  5 Art & Design        4.36
##  6 Auto & Vehicles     4.15
##  7 Beauty              4.29
##  8 Board               4.30
##  9 Books & Reference   4.32
## 10 Brain Games         4.36
## # ... with 43 more rows
```

```
(apps_aov = aov(Rating~SIZE+Installs+Price+Reviews+Genres+`Content Rating`+`Last Update`, data =
apps2))
```

```
## Call:
##    aov(formula = Rating ~ SIZE + Installs + Price + Reviews + Genres +
##     `Content Rating` + `Last Update`, data = apps2)
##
## Terms:
##                      SIZE  Installs     Price   Reviews    Genres
## Sum of Squares    18.3974    3.5788    5.9096    5.1909   79.5519
## Deg. of Freedom         1         1         1         1        52
##                 `Content Rating` `Last Update` Residuals
## Sum of Squares            1.1561       41.7246 2182.3085
## Deg. of Freedom                5             1      8082
##
## Residual standard error: 0.5196353
## Estimated effects may be unbalanced
```

```
summary(apps_aov)
```

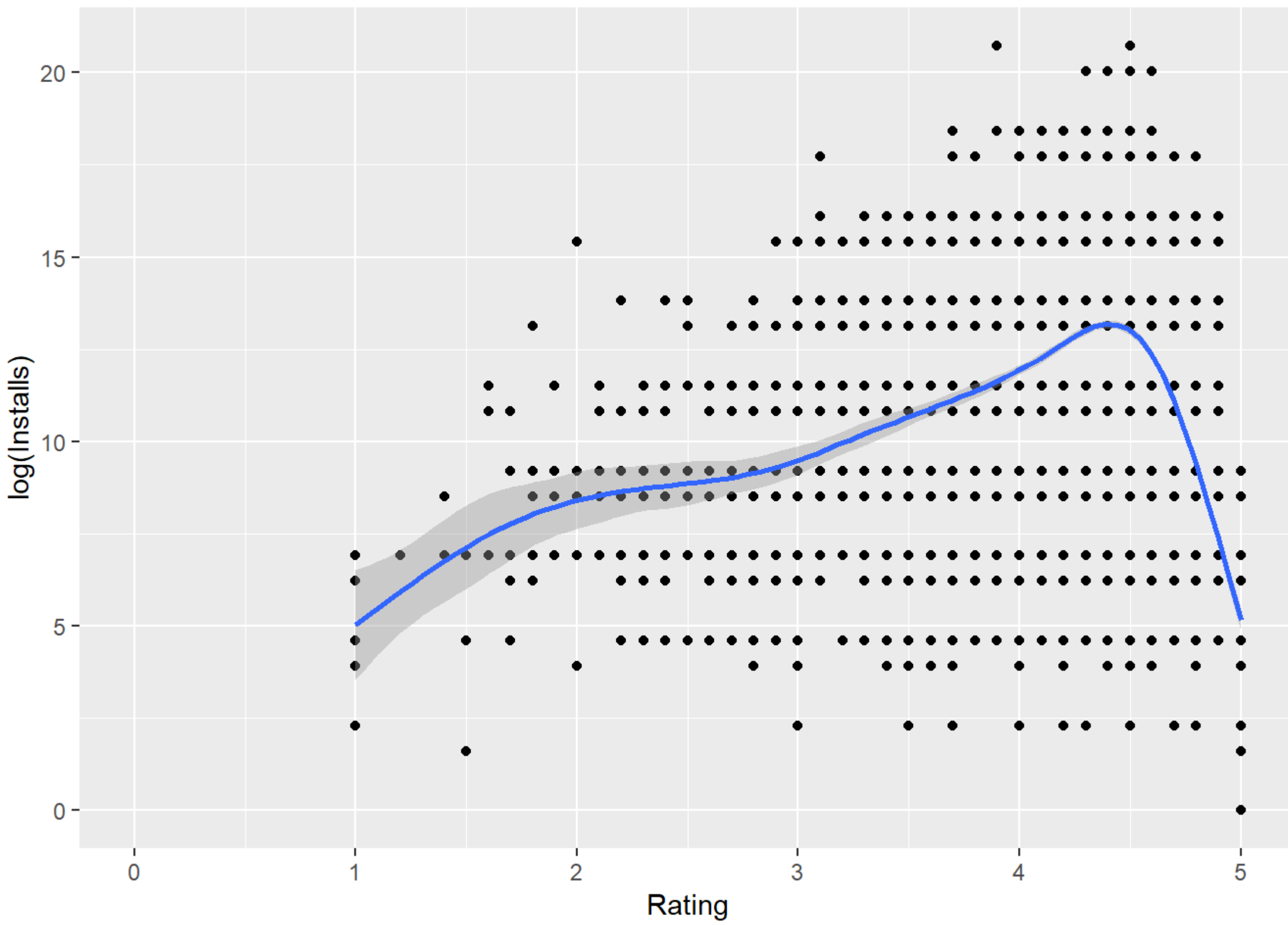
```
##                    Df Sum Sq Mean Sq F value   Pr(>F)
## SIZE                1   18.4   18.40  68.133  < 2e-16 ***
## Installs            1    3.6    3.58  13.254 0.000274 ***
## Price               1    5.9    5.91  21.886 2.94e-06 ***
## Reviews             1    5.2    5.19  19.224 1.18e-05 ***
## Genres             52   79.6    1.53   5.666  < 2e-16 ***
## `Content Rating`    5    1.2    0.23   0.856 0.509658
## `Last Update`       1   41.7   41.72 154.524  < 2e-16 ***
## Residuals        8082 2182.3    0.27
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the results of the analysis of Variance we observe there is a significant difference in the means in all the categories except for Content Rating, given their p-values are far less than 0.05.

Graphing the relationship between the rating and other numerical factors

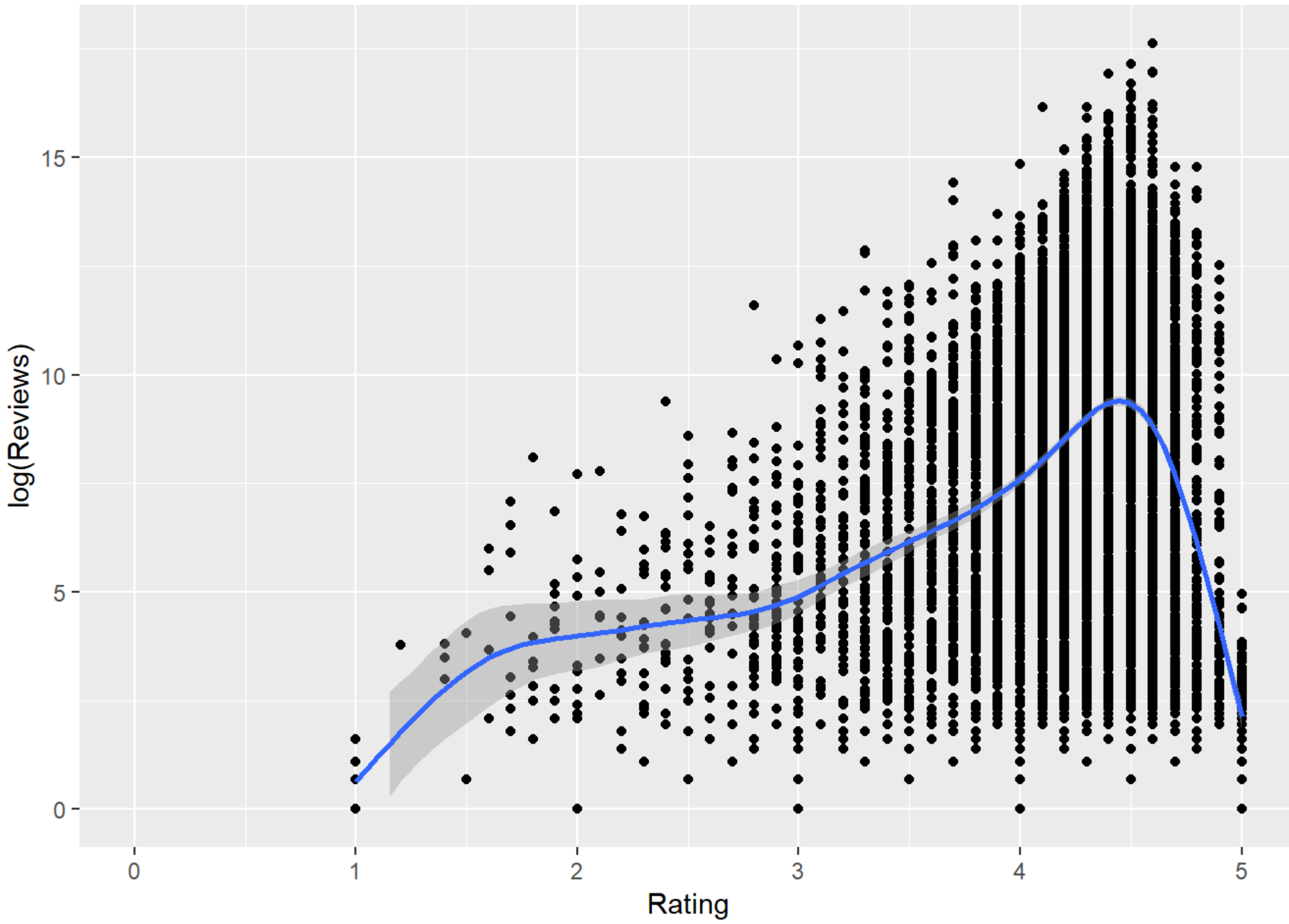
```
ggplot(apps2, aes(Rating,log(Installs)))+geom_point()+xlim(0,5)+ylim(min(log(apps2$Installs)),ma
x(log(apps2$Installs)))+geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



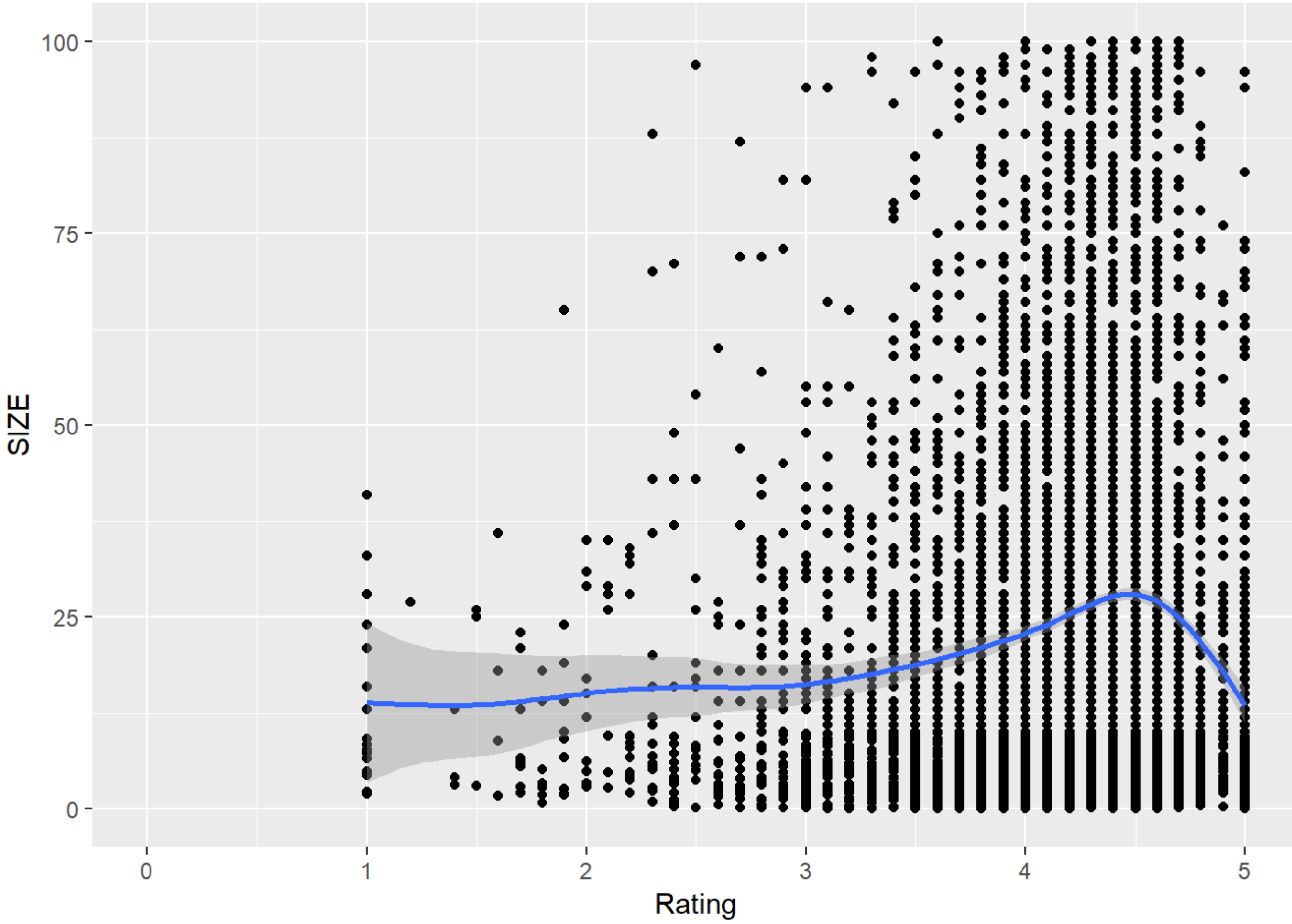

```
ggplot(apps2, aes(Rating,log(Reviews)))+geom_point()+xlim(0,5)+ylim(min(log(apps2$Reviews)),max
(log(apps2$Reviews)))+geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



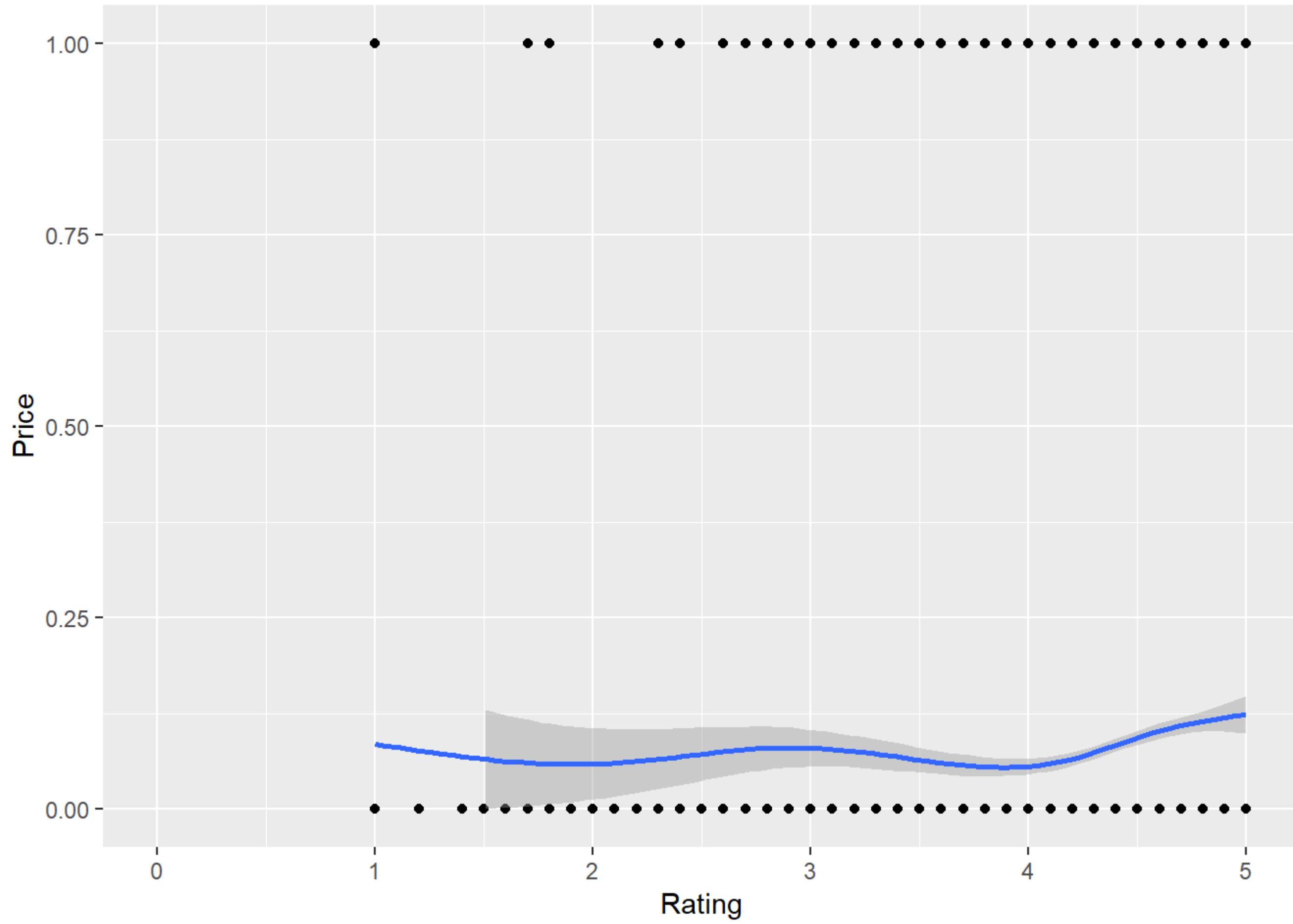

```
ggplot(apps2, aes(Rating,SIZE))+geom_point()+xlim(0,5)+ylim(min(apps2$SIZE),max(apps2$SIZE))+geo
m_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
ggplot(apps2, aes(Rating,Price))+geom_point()+xlim(0,5)+ylim(min(apps2$Price),max(apps2$Price))+
geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
#appplot + ylim(0,5) + xlim(1,1.0e+09)
#appplot + ylim(0,5)
```

Effective rating goes down after reaching its peak in proportion to the categories above as expected. This highlights the fact that good apps exist but doesn't get much attention or the ratings and reviews are fabricated.

Regression Models

Fixing duplicate Genres under similar terms

```
apps2 %>% mutate(Genres = case_when(
                              Genres == "Educational" ~ "Education",
                              Genres == "Music & Audio" ~ "Music",
                              Genres == "Music & Video" ~ "Music",
                              TRUE ~ Genres)) -> apps2
```

```
appreg = lm(Rating~SIZE+I(log(Installs))+I(log(Reviews))+`Last Update`+Genres+Price, data = apps
2, weights = Installs)
summary(appreg)
```

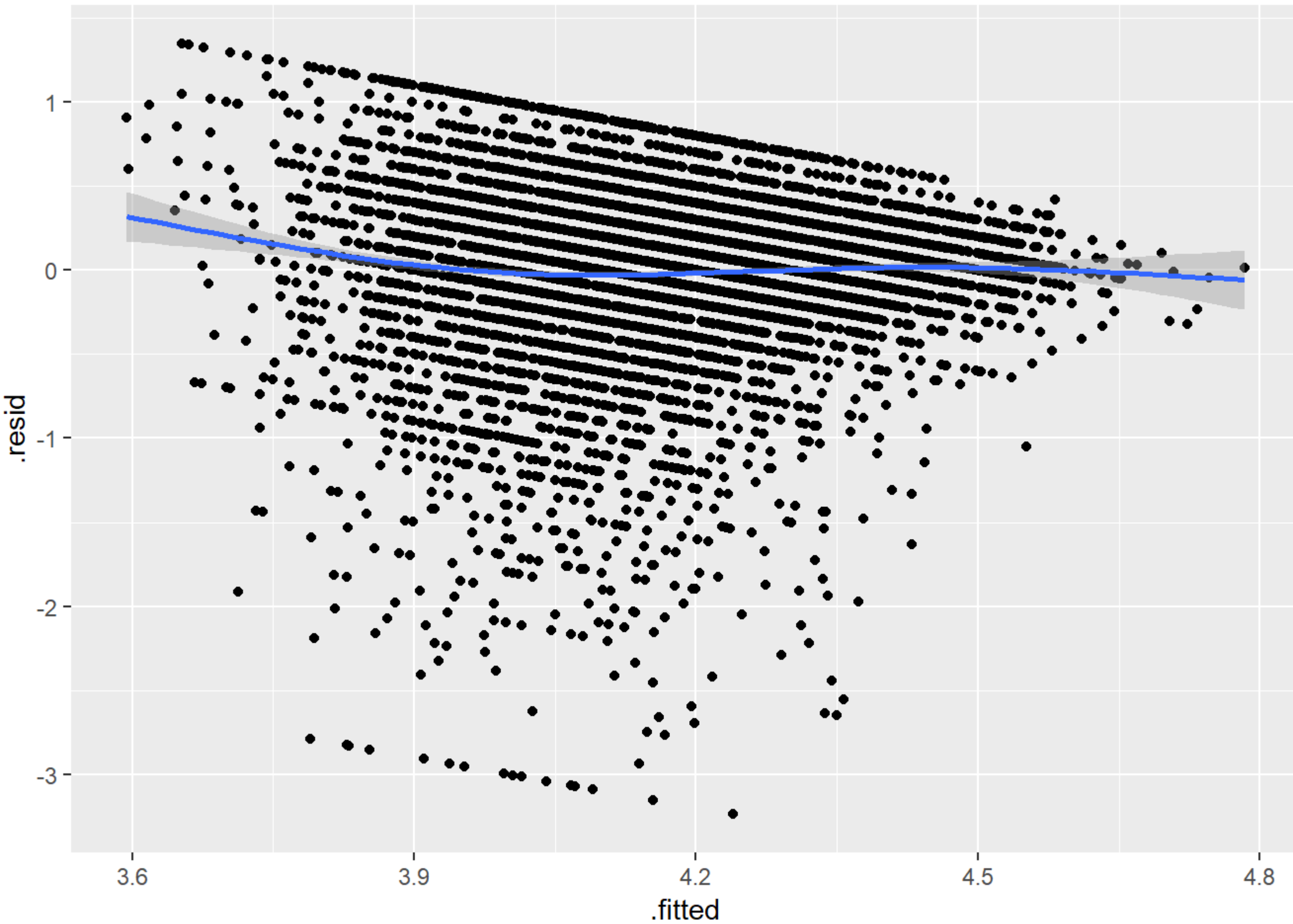
```
##
## Call:
## lm(formula = Rating ~ SIZE + I(log(Installs)) + I(log(Reviews)) +
##     `Last Update` + Genres + Price, data = apps2, weights = Installs)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -7136.9   -70.0     6.9    91.4  5108.2
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   2.371e+00  1.745e-01  13.581  < 2e-16 ***
## SIZE                          2.997e-04  9.493e-05   3.157 0.001601 **
## I(log(Installs))             -7.739e-02  2.335e-03 -33.143  < 2e-16 ***
## I(log(Reviews))               1.026e-01  2.164e-03  47.403  < 2e-16 ***
## `Last Update`                 1.053e-04  1.008e-05  10.444  < 2e-16 ***
## GenresAction & Adventure      6.491e-02  1.192e-02   5.443 5.39e-08 ***
## GenresAdventure               5.194e-03  1.352e-02   0.384 0.700844
## GenresArcade                  7.527e-02  6.908e-03  10.896  < 2e-16 ***
## GenresArt & Design            3.268e-01  4.430e-02   7.376 1.79e-13 ***
## GenresAuto & Vehicles         1.330e-01  6.668e-02   1.995 0.046077 *
## GenresBeauty                  2.454e-01  1.200e-01   2.045 0.040916 *
## GenresBoard                   1.577e-01  3.319e-02   4.750 2.07e-06 ***
## GenresBooks & Reference       1.110e-01  3.775e-02   2.939 0.003298 **
## GenresBrain Games             1.973e-01  2.781e-02   7.096 1.39e-12 ***
## GenresBusiness                9.796e-02  2.000e-02   4.899 9.81e-07 ***
## GenresCard                    1.540e-01  4.203e-02   3.665 0.000249 ***
## GenresCasino                  1.133e-01  4.263e-02   2.658 0.007872 **
## GenresCasual                 -2.944e-02  7.164e-03  -4.109 4.00e-05 ***
## GenresComics                 -2.848e-01  1.050e-01  -2.711 0.006726 **
## GenresCommunication           5.099e-02  8.886e-03   5.738 9.94e-09 ***
## GenresCreativity              1.214e-01  4.782e-02   2.538 0.011155 *
## GenresDating                 -5.856e-02  3.754e-02  -1.560 0.118813
## GenresEducation               1.286e-01  1.551e-02   8.293  < 2e-16 ***
## GenresEntertainment          -6.859e-02  1.143e-02  -6.002 2.03e-09 ***
## GenresEvents                  6.810e-02  1.346e-01   0.506 0.612975
## GenresFinance                 5.069e-02  2.541e-02   1.995 0.046045 *
## GenresFood & Drink           -5.122e-02  3.361e-02  -1.524 0.127546
## GenresHealth & Fitness        2.652e-01  1.634e-02  16.226  < 2e-16 ***
## GenresHouse & Home            1.550e-01  5.114e-02   3.031 0.002445 **
## GenresLibraries & Demo       -6.289e-02  5.759e-02  -1.092 0.274817
## GenresLifestyle              -4.125e-02  2.187e-02  -1.886 0.059350 .
## GenresMaps & Navigation       3.340e-02  3.377e-02   0.989 0.322617
## GenresMedical                 1.936e-01  6.486e-02   2.985 0.002845 **
## GenresMusic                   5.156e-02  3.685e-02   1.399 0.161718
## GenresNews & Magazines       -1.338e-01  1.128e-02 -11.869  < 2e-16 ***
## GenresParenting               1.834e-01  9.063e-02   2.024 0.043037 *
## GenresPersonalization         1.728e-01  1.537e-02  11.240  < 2e-16 ***
## GenresPhotography             6.508e-02  1.026e-02   6.345 2.34e-10 ***
## GenresPretend Play            8.442e-02  2.130e-02   3.963 7.46e-05 ***
## GenresProductivity            1.713e-01  1.009e-02  16.967  < 2e-16 ***
## GenresPuzzle                  8.543e-02  1.178e-02   7.254 4.42e-13 ***
## GenresRacing                  1.554e-02  1.149e-02   1.353 0.176067
```

```
## GenresRole Playing            6.498e-02  2.244e-02   2.895 0.003796 **
## GenresShopping                7.321e-02  1.286e-02   5.695 1.28e-08 ***
## GenresSimulation              4.162e-02  1.687e-02   2.468 0.013625 *
## GenresSocial                 -2.892e-02  1.543e-02  -1.874 0.060996 .
## GenresSports                  2.217e-02  1.072e-02   2.068 0.038684 *
## GenresStrategy                2.016e-03  1.169e-02   0.172 0.863151
## GenresTools                   1.373e-01  9.775e-03  14.046  < 2e-16 ***
## GenresTravel & Local          2.971e-02  2.420e-02   1.228 0.219578
## GenresTrivia                 -1.518e-02  3.987e-02  -0.381 0.703505
## GenresVideo Players & Editors  6.832e-02  1.579e-02   4.327 1.53e-05 ***
## GenresWeather                 8.027e-02  3.773e-02   2.127 0.033434 *
## GenresWord                    2.059e-01  3.580e-02   5.752 9.15e-09 ***
## Price                         1.149e-01  6.038e-02   1.903 0.057037 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 438.5 on 8090 degrees of freedom
## Multiple R-squared:  0.5307, Adjusted R-squared:  0.5276
## F-statistic: 169.4 on 54 and 8090 DF,  p-value: < 2.2e-16
```

Taking the log of Installs and Reviews and weighing it with the Installs helped increase the R squared to about 53% from under 1%

```
ggplot(appreg, aes(x=.fitted, y=.resid))+geom_point()+geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



This is a peculiar residual plot due to the fact there is a specific range for the ratings (0-5), there seems to be a certain pattern in its overall shape but it's difficult to judge the overall randomness.

Kmeans for Genres

Here we attempt K-means clustering to create our mega genres to reduce the overwhelming number of genres we have. We are gonna cluster them into 10 specific genres as we feel it's a good amount of variety.

```
#Here we use the kmodes function from the kLaR package that is suitable for categorical variable clusterting like our genres
# genres = kmodes(apps2[,c(3:7,11)], modes = 10, iter.max = 10, weighted = FALSE)
#
# #Each
# plot(jitter(apps2$Rating),col = genres$cluster)
# points(genres$modes, col = 1:5, pch = 8)
# genres$modes
# genres$cluster

#I commented this part out because it kept crashing due to memory issues
```

```
genres = readRDS("genres.rds")
```

However I'm loading the cluster that had the preferable generated genres to be used for the rest of the analysis.

```
apps2 %>% mutate(cluster=genres$cluster) -> apps2
```

```
clustapp = genres$modes
```

```
apps2 %>% group_by(cluster, Genres) %>% count() %>% summarise(m=max(n)) %>% summarise(mm=max(m))
```

```
## # A tibble: 10 x 2
##    cluster    mm
##      <int> <dbl>
##  1       1   289
##  2       2   629
##  3       3   572
##  4       4   186
##  5       5   160
##  6       6   113
##  7       7   157
##  8       8   169
##  9       9   117
## 10      10   202
```

```
#Adding the new mega genres to the dataset
apps2 %>% mutate(True_Genre = case_when(
                                         cluster == 1 & Genres != clustapp$Genres[1] ~ clustapp
$Genres[1],
                                         cluster == 2 & Genres != clustapp$Genres[2] ~ clustapp
$Genres[2],
                                         cluster == 3 & Genres != clustapp$Genres[3] ~ clustapp
$Genres[3],
                                         cluster == 4 & Genres != clustapp$Genres[4] ~ clustapp
$Genres[4],
                                         cluster == 5 & Genres != clustapp$Genres[5] ~ clustapp
$Genres[5],
                                         cluster == 6 & Genres != clustapp$Genres[6] ~ clustapp
$Genres[6],
                                         cluster == 7 & Genres != clustapp$Genres[7] ~ clustapp
$Genres[7],
                                         cluster == 8 & Genres != clustapp$Genres[8] ~ clustapp
$Genres[8],
                                         cluster == 9 & Genres != clustapp$Genres[9] ~ clustapp
$Genres[9],
                                         cluster == 10 & Genres != clustapp$Genres[10] ~ clustap
p$Genres[10],
                                         TRUE ~ Genres)) -> apps3
```

```
apps3 %>% group_by(True_Genre) %>% count()
```

```
## # A tibble: 10 x 2
## # Groups:   True_Genre [10]
##    True_Genre           n
##    <chr>            <int>
##  1 Action            1398
##  2 Business           169
##  3 Dating             245
##  4 Education         1531
##  5 Finance            186
##  6 Health & Fitness   376
##  7 Lifestyle          202
##  8 Productivity       160
##  9 Shopping           705
## 10 Tools             3173
```

Regression Model with the new Genres:

```
newreg = lm(Rating~SIZE+I(log(Installs))+I(log(Reviews))+`Last Update`+True_Genre+Price, data =
apps3, weights = Installs)
summary(newreg)
```

```
##
## Call:
## lm(formula = Rating ~ SIZE + I(log(Installs)) + I(log(Reviews)) +
##     `Last Update` + True_Genre + Price, data = apps3, weights = Installs)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -6224.1   -77.3     6.6    85.5  7214.3
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                2.862e+00  1.810e-01  15.810  < 2e-16 ***
## SIZE                       7.136e-05  8.183e-05   0.872  0.38318
## I(log(Installs))          -9.856e-02  1.938e-03 -50.863  < 2e-16 ***
## I(log(Reviews))            1.219e-01  1.913e-03  63.702  < 2e-16 ***
## `Last Update`              8.578e-05  1.046e-05   8.200 2.78e-16 ***
## True_GenreBusiness         4.505e-02  2.405e-02   1.873  0.06108 .
## True_GenreDating          -8.916e-02  1.825e-02  -4.884 1.06e-06 ***
## True_GenreEducation        5.838e-02  1.247e-02   4.684 2.86e-06 ***
## True_GenreFinance          2.075e-02  2.917e-02   0.711  0.47692
## True_GenreHealth & Fitness  2.350e-01  1.823e-02  12.891  < 2e-16 ***
## True_GenreLifestyle        1.151e-01  3.784e-02   3.042  0.00236 **
## True_GenreProductivity     1.657e-01  9.669e-03  17.136  < 2e-16 ***
## True_GenreShopping         3.896e-02  7.292e-03   5.343 9.41e-08 ***
## True_GenreTools           -5.669e-03  4.284e-03  -1.324  0.18570
## Price                      6.897e-02  6.546e-02   1.054  0.29208
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 475.6 on 8130 degrees of freedom
## Multiple R-squared:  0.4453, Adjusted R-squared:  0.4444
## F-statistic: 466.2 on 14 and 8130 DF,  p-value: < 2.2e-16
```

R-squared was lowered and for a good reason since we had an over abundance of genres previously that was inflating the value.

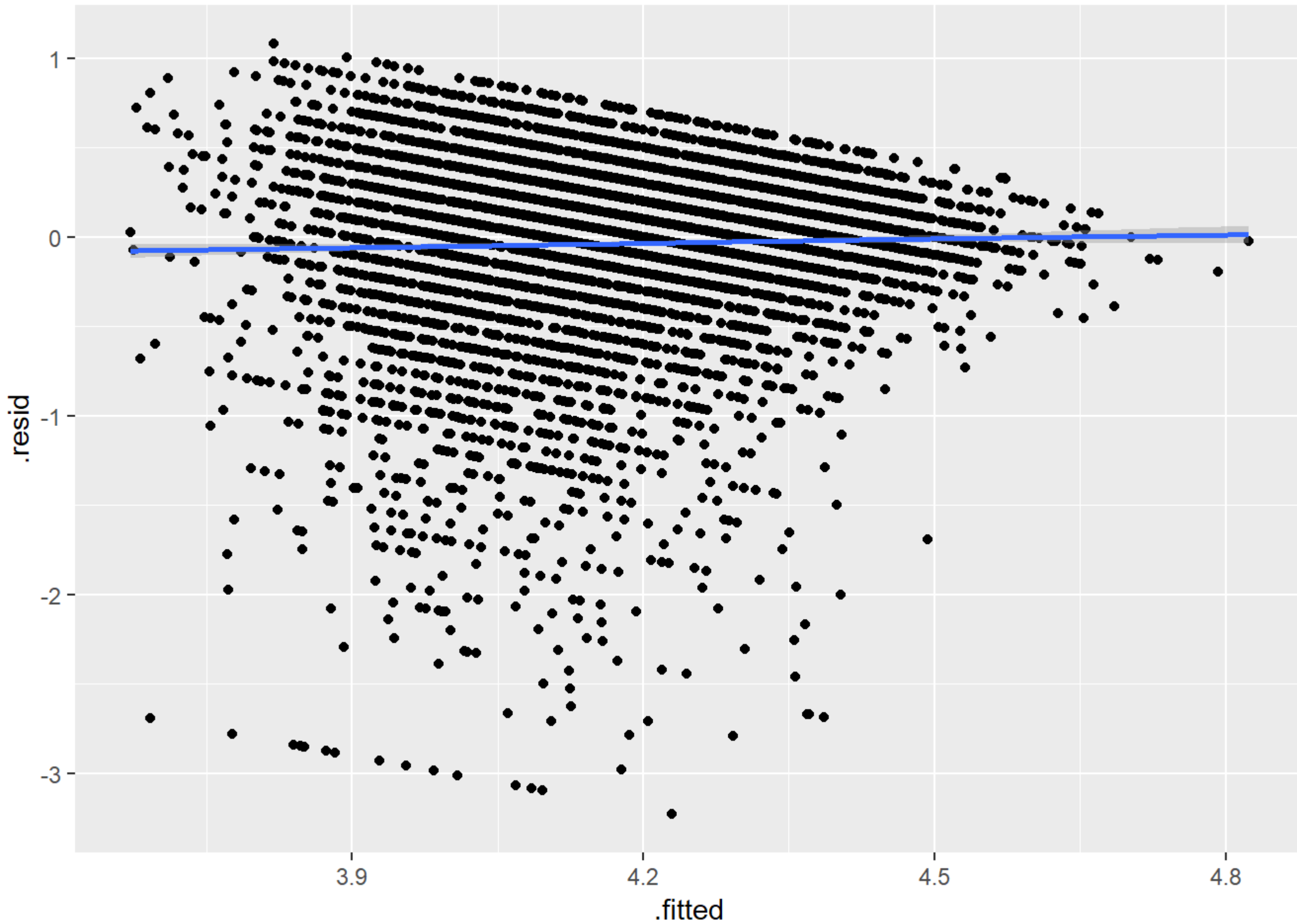
```
apps3 %>% filter(Rating >= 1 & Rating < 5) -> apps4
reg1 = lm(Rating~SIZE+I(log(Installs))+I(log(Reviews))+`Last Update`+True_Genre-1+Price, data =
 apps4, weights = Installs)
summary(reg1)
```

```
##
## Call:
## lm(formula = Rating ~ SIZE + I(log(Installs)) + I(log(Reviews)) +
##     `Last Update` + True_Genre - 1 + Price, data = apps4, weights = Installs)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -6224.0   -84.9     6.0    92.9  7214.5
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## SIZE                       7.128e-05  8.322e-05   0.857    0.392
## I(log(Installs))          -9.856e-02  1.971e-03 -50.010  < 2e-16 ***
## I(log(Reviews))            1.219e-01  1.946e-03  62.638  < 2e-16 ***
## `Last Update`              8.578e-05  1.064e-05   8.062 8.63e-16 ***
## True_GenreAction           2.862e+00  1.841e-01  15.546  < 2e-16 ***
## True_GenreBusiness         2.907e+00  1.863e-01  15.606  < 2e-16 ***
## True_GenreDating           2.773e+00  1.857e-01  14.937  < 2e-16 ***
## True_GenreEducation        2.921e+00  1.838e-01  15.889  < 2e-16 ***
## True_GenreFinance          2.883e+00  1.872e-01  15.400  < 2e-16 ***
## True_GenreHealth & Fitness  3.097e+00  1.855e-01  16.695  < 2e-16 ***
## True_GenreLifestyle        2.977e+00  1.881e-01  15.833  < 2e-16 ***
## True_GenreProductivity     3.028e+00  1.847e-01  16.395  < 2e-16 ***
## True_GenreShopping         2.901e+00  1.839e-01  15.779  < 2e-16 ***
## True_GenreTools            2.857e+00  1.837e-01  15.547  < 2e-16 ***
## Price                      6.897e-02  6.658e-02   1.036    0.300
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 483.6 on 7860 degrees of freedom
## Multiple R-squared:  0.9986, Adjusted R-squared:  0.9986
## F-statistic: 3.653e+05 on 15 and 7860 DF,  p-value: < 2.2e-16
```

Here we opted to model app ratings that are in the range of [1,5) since it is a more realistic range. The extremely high R-squared is misleading since here we are looking at the mean rating for each genre. We also observe there is a negative slope for installs and positive slope for the number of reviews which confirms some of our hypotheses in the introduction where a decent app has a lot of installs but not enough reviews to have the score it truly deserves. As expected, the “Last Update” is also significant because it affects the overall rating of the app if the developers don’t update to fix the bugs, so the more updated an app is the better the rating hence the positive slope.

```
ggplot(reg1, aes(x=.fitted, y=.resid))+geom_point()+geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



The residual plot doesn't look much different from before except it's less condensed at the top since we are not considering 5 star ratings.

Making regression models for each genre of apps

```
apps4 %>% filter(True_Genre == "Action") %>% lm(Rating~SIZE+I(log(Installs))+I(log(Reviews))+`La
st Update`+Price, data = ., weights = Installs) %>% tidy()
```

```
## # A tibble: 6 x 5
##   term              estimate std.error statistic   p.value
##   <chr>                <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)       3.14      0.396         7.94 4.04e- 15
## 2 SIZE              0.000228  0.000184      1.24 2.16e-  1
## 3 I(log(Installs)) -0.120     0.00328     -36.6  4.00e-205
## 4 I(log(Reviews))   0.147     0.00376      39.0  1.56e-224
## 5 `Last Update`     0.0000716 0.0000229     3.12 1.83e-  3
## 6 Price             0.139     0.138         1.01 3.14e-  1
```

```
apps4 %>% filter(True_Genre == "Business") %>% lm(Rating~SIZE+I(log(Installs))+I(log(Reviews))+`
Last Update`+Price, data = ., weights = Installs) %>% tidy()
```

```
## # A tibble: 6 x 5
##   term              estimate std.error statistic  p.value
##   <chr>                <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)       7.43      4.84         1.54  1.27e- 1
## 2 SIZE             -0.00300   0.00101     -2.97  3.49e- 3
## 3 I(log(Installs)) -0.126     0.0166      -7.56  4.17e-12
## 4 I(log(Reviews))   0.168     0.0168      10.0   2.76e-18
## 5 `Last Update`    -0.000171  0.000273    -0.628 5.31e- 1
## 6 Price            -0.432     1.91        -0.226 8.22e- 1
```

```
apps4 %>% filter(True_Genre == "Dating") %>% lm(Rating~SIZE+I(log(Installs))+I(log(Reviews))+`La
st Update`+Price, data = ., weights = Installs) %>% tidy()
```

```
## # A tibble: 6 x 5
##   term              estimate std.error statistic   p.value
##   <chr>                <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)      -4.96      3.93        -1.26  0.208
## 2 SIZE             -0.00321   0.000804    -3.99  0.0000886
## 3 I(log(Installs))  0.0715    0.0280       2.55  0.0113
## 4 I(log(Reviews))  -0.0214    0.0308      -0.694 0.489
## 5 `Last Update`     0.000474  0.000224     2.12  0.0353
## 6 Price             0.606     0.469        1.29  0.197
```

```
apps4 %>% filter(True_Genre == "Education") %>% lm(Rating~SIZE+I(log(Installs))+I(log(Reviews))+
`Last Update`+Price, data = ., weights = Installs) %>% tidy()
```

```
## # A tibble: 6 x 5
##   term              estimate std.error statistic  p.value
##   <chr>                <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)       1.20     0.419         2.87  4.22e- 3
## 2 SIZE             -0.00165  0.000300     -5.50  4.44e- 8
## 3 I(log(Installs)) -0.0581   0.00932      -6.24  5.67e-10
## 4 I(log(Reviews))   0.102    0.00791      12.9   4.51e-36
## 5 `Last Update`     0.000165 0.0000235     7.03  3.13e-12
## 6 Price             0.0686   0.0701        0.979 3.28e- 1
```

```
apps4 %>% filter(True_Genre == "Finance") %>% lm(Rating~SIZE+I(log(Installs))+I(log(Reviews))+`L
ast Update`+Price, data = ., weights = Installs) %>% tidy()
```

```
## # A tibble: 6 x 5
##   term              estimate std.error statistic  p.value
##   <chr>                <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)       6.99      6.26         1.12  2.65e- 1
## 2 SIZE              0.000808  0.000765     1.06  2.92e- 1
## 3 I(log(Installs)) -0.122     0.0171      -7.09  3.29e-11
## 4 I(log(Reviews))   0.154     0.0168       9.21  1.10e-16
## 5 `Last Update`    -0.000150  0.000354    -0.422 6.74e- 1
## 6 Price            -0.0845    0.733       -0.115 9.08e- 1
```

```
apps4 %>% filter(True_Genre == "Health & Fitness") %>% lm(Rating~SIZE+I(log(Installs))+I(log(Rev
iews))+`Last Update`+Price, data = ., weights = Installs) %>% tidy()
```

```
## # A tibble: 6 x 5
##   term              estimate std.error statistic  p.value
##   <chr>                <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)      -1.10     1.10         -1.00  3.17e- 1
## 2 SIZE             -0.000697 0.000884     -0.788 4.31e- 1
## 3 I(log(Installs)) -0.0669   0.0103       -6.51  2.52e-10
## 4 I(log(Reviews))   0.114    0.0148        7.71  1.25e-13
## 5 `Last Update`     0.000298 0.0000633     4.71  3.51e- 6
## 6 Price             0.0661   0.158         0.418 6.76e- 1
```

```
apps4 %>% filter(True_Genre == "Lifestyle") %>% lm(Rating~SIZE+I(log(Installs))+I(log(Reviews))+
`Last Update`+Price, data = ., weights = Installs) %>% tidy()
```

```
## # A tibble: 6 x 5
##   term              estimate std.error statistic     p.value
##   <chr>                <dbl>     <dbl>     <dbl>       <dbl>
## 1 (Intercept)      -1.52      2.08        -0.731 0.466
## 2 SIZE              0.00152   0.00164      0.927 0.355
## 3 I(log(Installs))  0.00940   0.0217       0.432 0.666
## 4 I(log(Reviews))   0.161     0.0293       5.50  0.000000140
## 5 `Last Update`     0.000219  0.000121     1.82  0.0709
## 6 Price             0.390     0.321        1.21  0.226
```

```
apps4 %>% filter(True_Genre == "Productivity") %>% lm(Rating~SIZE+I(log(Installs))+I(log(Review
s))+`Last Update`+Price, data = ., weights = Installs) %>% tidy()
```

```
## # A tibble: 6 x 5
##   term              estimate std.error statistic     p.value
##   <chr>                <dbl>     <dbl>     <dbl>       <dbl>
## 1 (Intercept)       1.33     1.45          0.916 0.361
## 2 SIZE             -0.00315  0.000793     -3.97  0.000114
## 3 I(log(Installs))  0.0170   0.0144        1.19  0.237
## 4 I(log(Reviews))   0.0591   0.0109        5.45  0.000000212
## 5 `Last Update`     0.000116 0.0000813     1.43  0.155
## 6 Price             0.420    0.409         1.03  0.305
```

```
apps4 %>% filter(True_Genre == "Shopping") %>% lm(Rating~SIZE+I(log(Installs))+I(log(Reviews))+`
Last Update`+Price, data = ., weights = Installs) %>% tidy()
```

```
## # A tibble: 6 x 5
##   term              estimate std.error statistic  p.value
##   <chr>                <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)       0.916    0.662        1.38   1.67e- 1
## 2 SIZE              0.000374 0.000368     1.01   3.11e- 1
## 3 I(log(Installs)) -0.105    0.0215      -4.86   1.45e- 6
## 4 I(log(Reviews))   0.116    0.0102      11.3    2.18e-27
## 5 `Last Update`     0.000208 0.0000363    5.73   1.51e- 8
## 6 Price            -0.146    7.53        -0.0194 9.85e- 1
```

```
apps4 %>% filter(True_Genre == "Tools") %>% lm(Rating~SIZE+I(log(Installs))+I(log(Reviews))+`Las
t Update`+Price, data = ., weights = Installs) %>% tidy()
```

```
## # A tibble: 6 x 5
##   term               estimate std.error statistic   p.value
##   <chr>                 <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)       2.98      0.287         10.4  7.90e- 25
## 2 SIZE             -0.000371  0.000120      -3.08 2.10e-  3
## 3 I(log(Installs)) -0.0603    0.00426      -14.1  4.63e- 44
## 4 I(log(Reviews))   0.0822    0.00365       22.5  6.31e-104
## 5 `Last Update`     0.0000730 0.0000164      4.45 9.05e-  6
## 6 Price             0.145     0.144          1.00 3.16e-  1
```

Here we can observe some of the factors aren’t significant at all for some genres while some p-values are borderline zero like in the Action games genre where the p-values for Installs and Reviews are to the power of 200+ while Price and Size aren’t significant. Most apps are free so the p-value for price is understandable and size isn’t significant to the ratings because every kind of game has its own required size. Last update for action games is significant because if the game isn’t patched from bugs soonthen ratings plummet fast.

Last update isn’t significant on fiance apps because they are usually made a for a select set of functions and as long as they are performing right, updates aren’t need as frequently other than to sort out major nugs or OS optimization.

Tools app has a highly significant p-value with reviews and that’s because tools apps are concerned with managing the phone as a whole like boosting battery life and file management so if they aren’t working as intended consumers will report their issues which significantly affect the ratings.

Sentiment Analysis

Here we explore user reviews and visualize their positive and negative reviews and attempt to connect them to our previous modelss

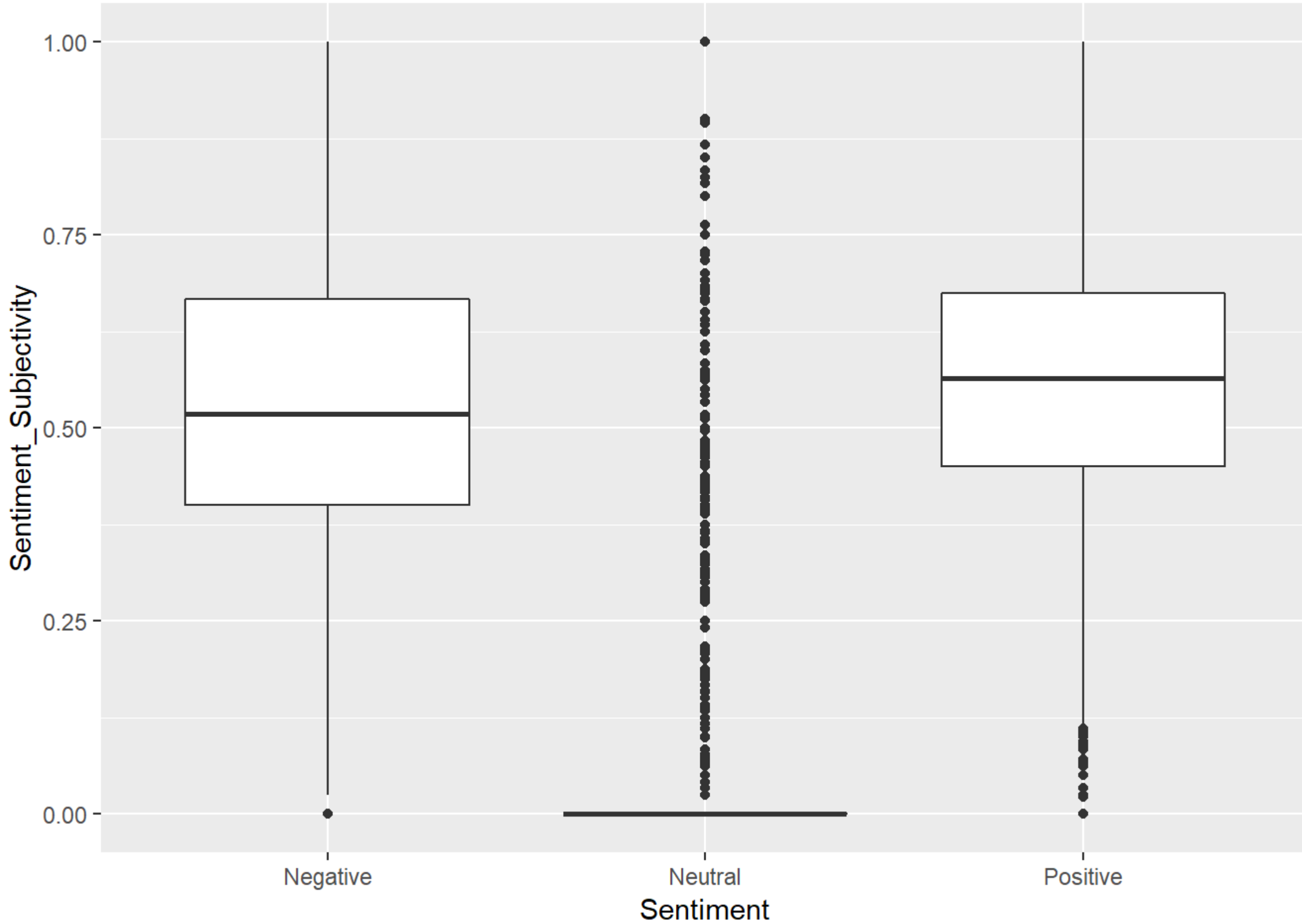
```
#Reading and cleaning the data
reviews = read_csv("googleplaystore_user_reviews.csv")
```

```
## Parsed with column specification:
## cols(
##   App = col_character(),
##   Translated_Review = col_character(),
##   Sentiment = col_character(),
##   Sentiment_Polarity = col_double(),
##   Sentiment_Subjectivity = col_double()
## )
```

```
(reviews = reviews %>% drop_na())
```

```
## # A tibble: 37,427 x 5
##    App    Translated_Review    Sentiment Sentiment_Polar~ Sentiment_Subje~
##    <chr>  <chr>                <chr>                <dbl>            <dbl>
##  1 10 Be~ "I like eat delicio~ Positive              1               0.533
##  2 10 Be~ This help eating he~ Positive              0.25            0.288
##  3 10 Be~ Works great especia~ Positive              0.4             0.875
##  4 10 Be~ Best idea us         Positive              1               0.3
##  5 10 Be~ Best way             Positive              1               0.3
##  6 10 Be~ Amazing              Positive              0.6             0.9
##  7 10 Be~ Looking forward app, Neutral               0               0
##  8 10 Be~ It helpful site ! I~ Neutral               0               0
##  9 10 Be~ good you.            Positive              0.7             0.6
## 10 10 Be~ Useful information ~ Positive              0.2             0.1
## # ... with 37,417 more rows
```

```
reviews %>% ggplot(aes(x=Sentiment,y=Sentiment_Subjectivity))+geom_boxplot()
```



Seperating every review into Individual Words

```
(tidy_reviews <- reviews %>% filter(Sentiment != "Neutral") %>% group_by(Sentiment) %>% unnest_t
okens(word,Translated_Review))
```

```
## # A tibble: 643,935 x 5
## # Groups:   Sentiment [2]
##    App             Sentiment Sentiment_Polari~ Sentiment_Subjecti~ word
##    <chr>           <chr>                 <dbl>               <dbl> <chr>
##  1 10 Best Foods ~ Positive                  1               0.533 i
##  2 10 Best Foods ~ Positive                  1               0.533 like
##  3 10 Best Foods ~ Positive                  1               0.533 eat
##  4 10 Best Foods ~ Positive                  1               0.533 delici~
##  5 10 Best Foods ~ Positive                  1               0.533 food
##  6 10 Best Foods ~ Positive                  1               0.533 that's
##  7 10 Best Foods ~ Positive                  1               0.533 i'm
##  8 10 Best Foods ~ Positive                  1               0.533 cooking
##  9 10 Best Foods ~ Positive                  1               0.533 food
## 10 10 Best Foods ~ Positive                  1               0.533 myself
## # ... with 643,925 more rows
```

```
#Counting the most common terms after filtering out common words
data("stop_words")
tidy_reviews = tidy_reviews %>% anti_join(stop_words)
```

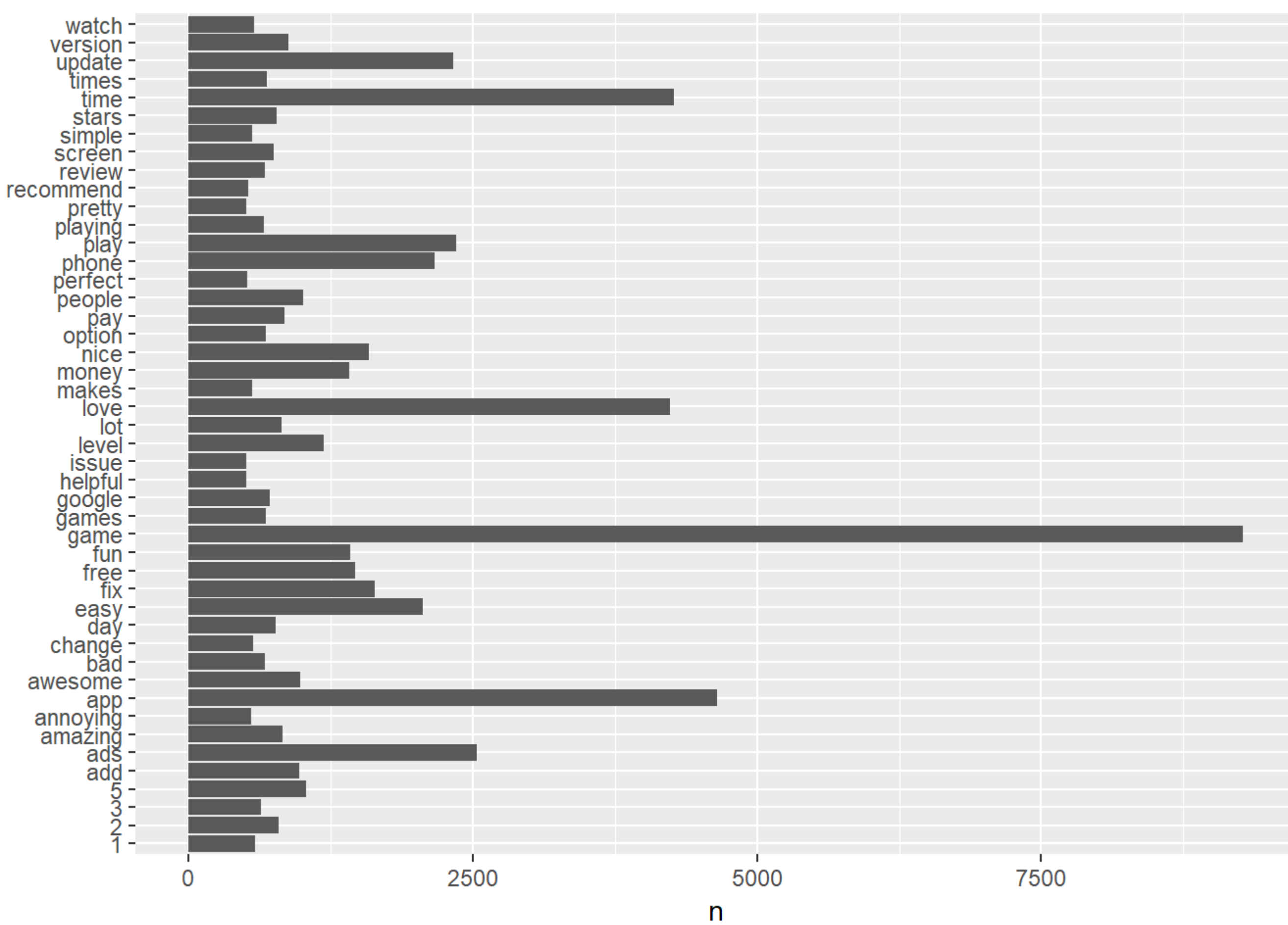
```
## Joining, by = "word"
```

```
tidy_reviews %>% group_by(Sentiment) %>% count(word,sort = TRUE) %>% filter(n > 500) %>%
  mutate(word=reorder(word,n)) %>%
  ggplot(aes(word,n))+geom_col()+xlab(NULL)+coord_flip()
```

```
## Warning in mutate_impl(.data, dots): Unequal factor levels: coercing to
## character
```

```
## Warning in mutate_impl(.data, dots): binding character and factor vector,
## coercing into character vector

## Warning in mutate_impl(.data, dots): binding character and factor vector,
## coercing into character vector
```



Majority of the frequently occuring words are general terms used for apps and genres like “app” and “games”

WordCloud

```
library(wordcloud)
```

```
## Warning: package 'wordcloud' was built under R version 3.5.3
```

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

```
tidy_reviews %>% filter(Sentiment=="Positive") %>% count(word) %>% with(wordcloud(word,n,max.words = 175))
```


```
tidy_reviews %>% filter(Sentiment=="Negative") %>% count(word) %>% with(wordcloud(word,n,max.words = 175))
```

As expected both word clouds contains common terms that describes apps in general but also their own terms that correspond to their sentiment.

```
review_apps = apps4 %>% inner_join(reviews)
```

```
## Joining, by = "App"
```

```
review_apps %>% filter(Sentiment != "Neutral") -> review_apps
```

WordClouds for each genre

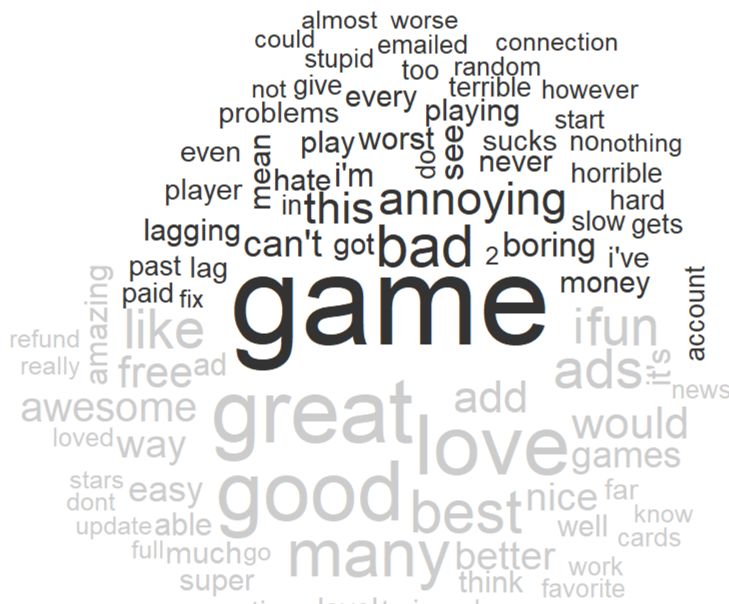
```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
review_apps %>% filter(True_Genre=="Education") %>% unnest_tokens(word,Translated_Review) %>% co
unt(word,Sentiment,sort = TRUE) %>%
  acast(word ~ Sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```


```
review_apps %>% filter(True_Genre=="Action") %>% unnest_tokens(word,Translated_Review) %>% count
(word,Sentiment,sort = TRUE) %>%
  acast(word ~ Sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```



```
review_apps %>% filter(True_Genre=="Business") %>% unnest_tokens(word,Translated_Review) %>% cou
nt(word,Sentiment,sort = TRUE) %>%
  acast(word ~ Sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): hipchat could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): notifications could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): business could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): can could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): constantly could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): new could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): calls could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): disconnects could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): open could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): annoying could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): ppl could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): rooms could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): communication could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): everything could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): extremely could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): though could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): accounts could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): better could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): connected could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): missing could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): page could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): email could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): why could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): every could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): messages could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): says could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): automatically could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): completely could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): enjoy could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): exists could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): grab could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): horribly could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): laggy could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): recreating could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): slack could not be fit on page. It will not be plotted.
```

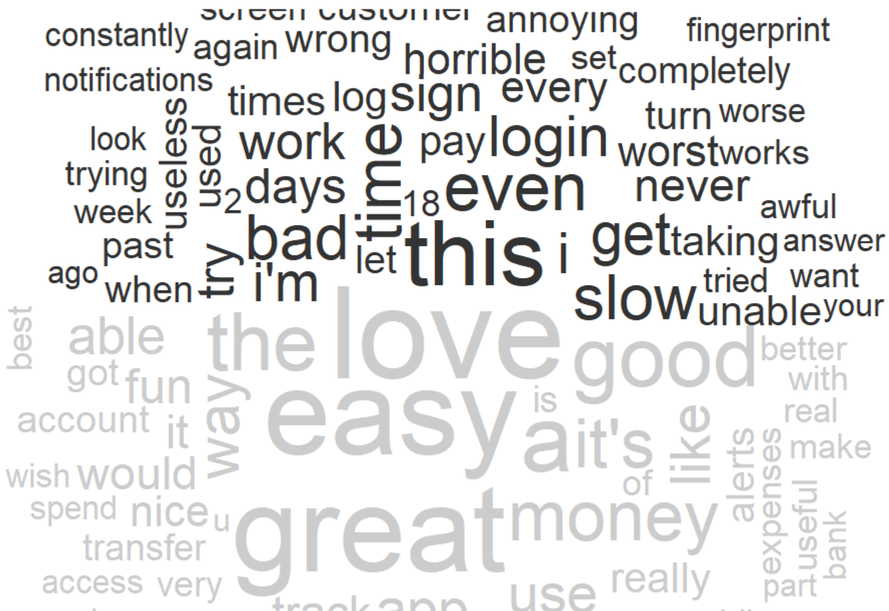
```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): tedious could not be fit on page. It will not be plotted.
```


```
review_apps %>% filter(True_Genre=="Dating") %>% unnest_tokens(word,Translated_Review) %>% count
(word,Sentiment,sort = TRUE) %>%
  acast(word ~ Sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): anyone could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): rating could not be fit on page. It will not be plotted.
```


```
review_apps %>% filter(True_Genre=="Finance") %>% unnest_tokens(word,Translated_Review) %>% coun
t(word,Sentiment,sort = TRUE) %>%
  acast(word ~ Sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```




```
review_apps %>% filter(True_Genre=="Health & Fitness") %>% unnest_tokens(word,Translated_Review)
%>% count(word,Sentiment,sort = TRUE) %>%
  acast(word ~ Sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```


```
review_apps %>% filter(True_Genre=="Lifestyle") %>% unnest_tokens(word,Translated_Review) %>% co
unt(word,Sentiment,sort = TRUE) %>%
  acast(word ~ Sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```


```
review_apps %>% filter(True_Genre=="Productivity") %>% unnest_tokens(word,Translated_Review) %>%
count(word,Sentiment,sort = TRUE) %>%
  acast(word ~ Sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): eventually could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): images could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): within could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): slow could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): please could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): function could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): takes could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): using could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): sharing could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): photos could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): update could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): want could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): disappointed could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): what's could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): version could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): especially could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): properly could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): restarts could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): sucks could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): waste could not be fit on page. It will not be plotted.
```

```
## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words =
## 100): upgrade could not be fit on page. It will not be plotted.
```


```
review_apps %>% filter(True_Genre=="Shopping") %>% unnest_tokens(word,Translated_Review) %>% cou
nt(word,Sentiment,sort = TRUE) %>%
  acast(word ~ Sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```


```
review_apps %>% filter(True_Genre=="Tools") %>% unnest_tokens(word,Translated_Review) %>% count
(word,Sentiment,sort = TRUE) %>%
  acast(word ~ Sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                   max.words = 100)
```

Unfortunately some of the word clouds couldn't be properly generated however we again see common terms for apps appear across all the word clouds but this time we have terms that specific to each genre in both positive and negative light. For example for Action games (just games in general actually) we have positivie terms like "good" and "graphics" while negative terms like "connection" which mainly corresponds to games that require an online connection to play and server connection issues are plenty (Looking at you EA).

My favorite one is for Dating genre where you see a nice big "fake" from the negative side which represents all dating apps because they keep their platform alive by generating fake profiles which most users don't take kindly to after finding out.

```
#Some experimentation with fitting sentiment variables to a model
review_apps %>% group_by(Sentiment) %>% count()
```

```
## # A tibble: 2 x 2
## # Groups:   Sentiment [2]
##   Sentiment     n
##   <chr>     <int>
## 1 Negative  12641
## 2 Positive  30336
```

```
review_apps %>% mutate(Sentiment3=cut(Sentiment_Polarity, breaks=c(-2,0,2))) %>%
  glm(Sentiment3~Sentiment_Subjectivity,family = "binomial",data = .) %>% tidy()
```

```
## # A tibble: 2 x 5
##   term                    estimate std.error statistic  p.value
##   <chr>                      <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)                0.316    0.0327      9.67 4.15e-22
## 2 Sentiment_Subjectivity    1.03     0.0577     17.8  5.64e-71
```

```
reg2 = lm(Rating~SIZE+I(log(Installs))+I(log(Reviews))+`Last Update`+Price+Sentiment, data = rev
iew_apps, weights = Installs)
summary(reg2)
```

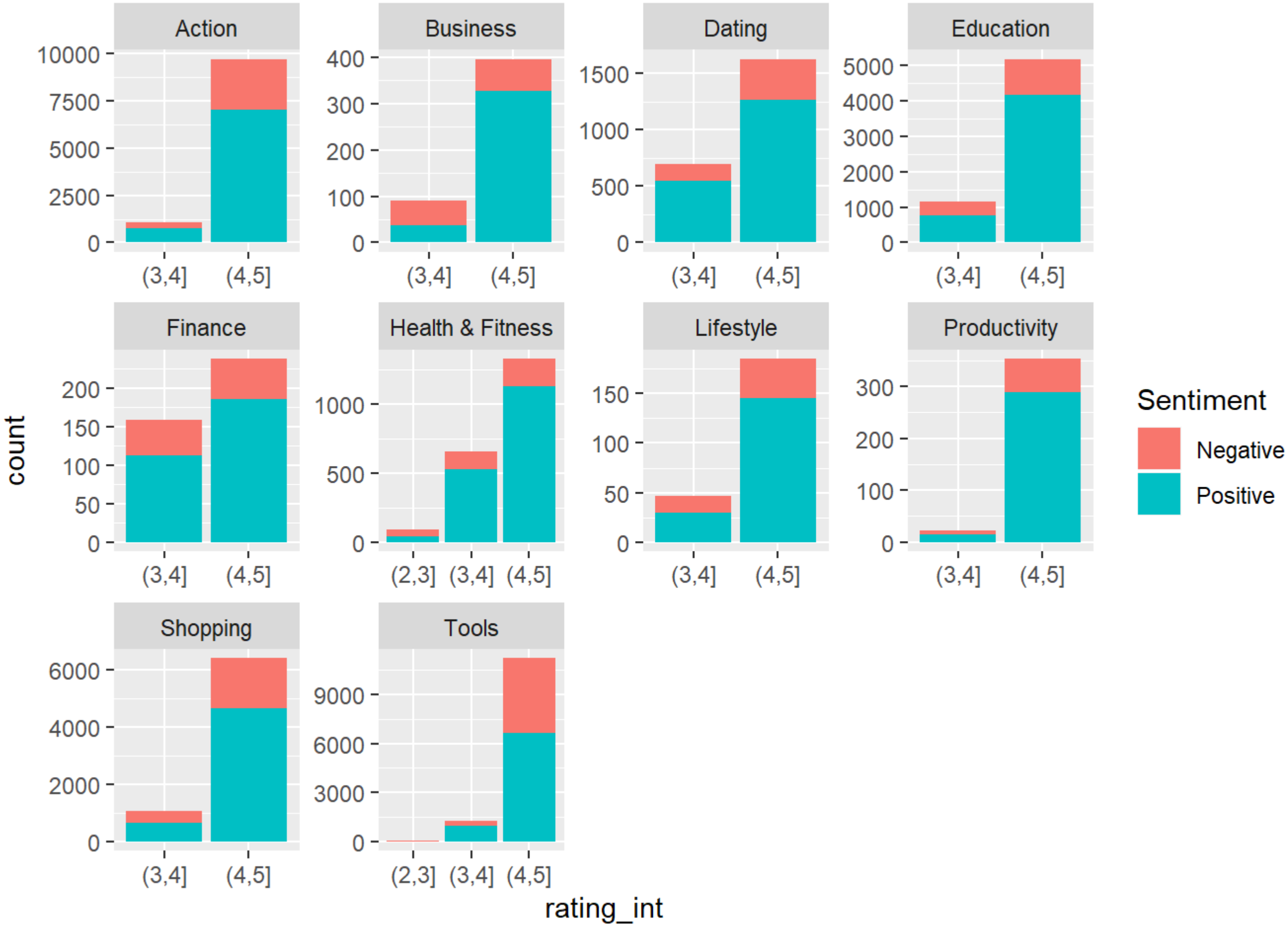
```
##
## Call:
## lm(formula = Rating ~ SIZE + I(log(Installs)) + I(log(Reviews)) +
##     `Last Update` + Price + Sentiment, data = review_apps, weights = Installs)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -3157.4  -330.0   -36.5   216.2  5629.0
##
## Coefficients:
##                     Estimate Std. Error  t value Pr(>|t|)
## (Intercept)        9.515e-01  1.032e-01    9.222   <2e-16 ***
## SIZE               1.565e-03  2.644e-05   59.198   <2e-16 ***
## I(log(Installs))  -1.129e-01  4.308e-04 -262.169   <2e-16 ***
## I(log(Reviews))    9.072e-02  4.965e-04  182.726   <2e-16 ***
## `Last Update`      2.285e-04  5.957e-06   38.350   <2e-16 ***
## Price             -8.107e-02  5.090e-02   -1.593    0.111
## SentimentPositive  1.249e-02  8.716e-04   14.325   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 661.9 on 42970 degrees of freedom
## Multiple R-squared:  0.8409, Adjusted R-squared:  0.8409
## F-statistic: 3.785e+04 on 6 and 42970 DF,  p-value: < 2.2e-16
```

From this model we can see all the factors are significant from before except for price due to most apps being free. The R-squared is at a decent 84% as well. Here we have our Sentiment variable applied and we can observe from its estimate that app ratings increase by about 0.0125 for every positive review compared to negative reviews. The standard error for it is pretty low so we can trust this estimate.

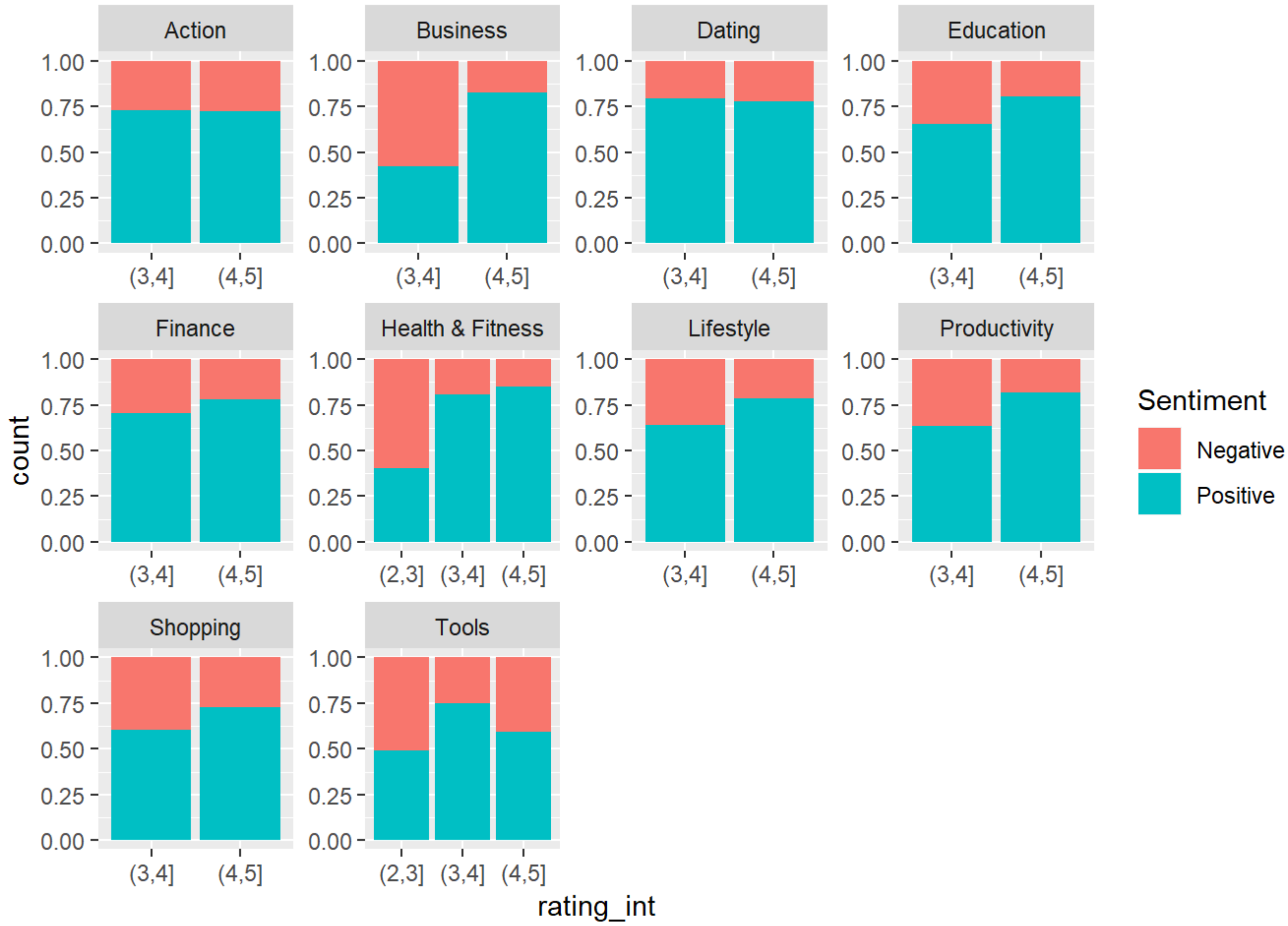
```
#Categorized the ratings into specific ranges
review_apps %>% mutate(rating_int=cut(Rating,breaks = c(0,1,2,3,4,5))) -> review_apps
```

Conclusion

```
review_apps %>% filter(Sentiment != "Neutral") %>% ggplot(aes(rating_int, fill=Sentiment))+geom_
bar()+facet_wrap(~True_Genre,scales = "free")
```




```
review_apps %>% filter(Sentiment != "Neutral") %>% ggplot(aes(rating_int, fill=Sentiment))+geom_
bar(position = "fill")+facet_wrap(~True_Genre,scales = "free")
```



Here is a simple visualization that ties our sentiment analysis with our response variable rating from before. There are several interesting things to note here with the most eye catching one being the presence of negative reviews in (4,5] star ratings. While there are significantly more positive reviews in all categories, we have to remember we are talking about the “Sentiment” of the review here which means someone can give a good rating to the app but their review in discussing the flaws of the app make it a negative review.

Looking at the proportions, the distrubtuions vary between ratings and each genre with some of them being roughly equal while others increase in positive reviews the higher the rating is.

In conclusion, my original aim for this analysis was to model a formula that calculates the effective rating of an app based on factors discussed here but that proved to be too ambitious. Instead I ended up exploring the relationships the app ratings have with variables like the number of installs and reviews along with the sentiment of reviews to explain how these variables statistically affect the app’s overall rating so users aren’t misled by what they see at first glance.