Course Project 1

Devin Romines and Joseph Audras

April 12, 2019

Introduction

Dataset Source: https://www.kaggle.com/lava18/google-play-store-apps (https://www.kaggle.com/lava18/google-play-store-apps)

Description

The Google Play Store Data Set is a data set containing data on over 10,000 apps in the Google Play Store for Android products. The data set looks at things such as rating, number of installations, and genre to produce a plethora of information on each app. For this project, we would like to see if the rating of the app can be accurately predicted by the variables provided in the data set.

The data set was gathered from https://www.kaggle.com/lava18/google-play-store-apps (https://www.kaggle.com/lava18/google-play-store-apps) which was last updated 2 months ago, giving us Version 6, with which we are working. The data was posted by Lavanya Gupta, a software engineer at HSBC Software Development in India. More information on her can be found here https://www.kaggle.com/lava18 (https://www.kaggle.com/lava18). The information in the data set was scraped from the Google Play Store by Lavanya Gupta, in an effort to provide similar information on its apps as is publically available from the Apple App Store so that developers may be more inclined to work in the Android market.

```
library(tidyverse)
library(lubridate)
library(ggplot2)
```

Data Dictionary

- App – Factor, 9660 levels – The name of the app.
- Category – Factor, 34 levels – The general type of app, such as Dating, Cooking, or Art and Design
- Rating – Numerical, range from 0-5 – The rating of the app on a scale from 0-5.
- Reviews – Number, very wide range – The number of reviews that the app received.
- Size (Removed in Cleaning) – Factor, 462 levels – The size in megabytes of the app.
- Installs – Number, given as a factor of 10 – The number of installations an app received.
- Type (Removed in Cleaning) – Factor, 4 levels – Whether or not the app is free.
- Price – Number, mostly 0 but some go very high – The Price of the app.
- Content.Rating – Factor, 4 levels – The rating for the app, Everyone, Everyone 10+, Mature 17+, and Teen.
- Genres – Factor, 120 levels – The genre of the app, Action, Action and Adventure, etc.

- Last.Updated – Factor, 1378 levels – The date on which the app was last updated, given in multiple formats.
- Current.Ver (Removed in Cleaning) – Factor, 2834 levels – The current version of the app, 0.0.0.2, 0.0.1, etc.
- Android.Ver (Removed in Cleaning) – Factor, 35 levels – The version of Android phone required to use the app, given in the form, “version and up”

Data Cleaning

Here we load the data set.

```
RawData <- read.csv("googleplaystore.csv")
```

This next code removes a problematic row that we found to have severe errors when it was inputed.

```
GooglePlayStore <- RawData[-c(10473), ]
```

Some of the values in the data set for the Rating value, which is the one we are looking at, read as NaN, so we will remove all entries that contain that value.

```
GooglePlayStore <- GooglePlayStore[-grep("NaN", GooglePlayStore$Rating), ]
```

The Reviews variable was initially given as a factor variable. Here we change that to be a numeric as it should be.

```
GooglePlayStore$Reviews <- as.numeric(as.character(GooglePlayStore$Reviews))
```

The size variable was deemed unnecessary for our goals for this project so here we remove it.

```
GooglePlayStore <- GooglePlayStore[, -5]
```

The Installs variable was initially given as a factor, with undesirable formatting. Here we remove the characters in the entries we do not want and then convert the Installs variable to a numeric as desired.

```
GooglePlayStore$Installs <- gsub("\\D", "", GooglePlayStore$Installs)
GooglePlayStore$Installs <- as.numeric(as.character(GooglePlayStore$Installs))
```

The Type variable merely indicates whether or not the app is free, which is also given by the better variable, Price, so here we remove it.

```
GooglePlayStore <- GooglePlayStore[, -6]
```

The Price variable is interesting, here we had to remove the dollar sign as we wanted to convert it to a numeric variable; however, when removing the dollar sign we also noticed that the code also removed the decimal point. To recover it, we simply divided every price by 100. This solved the problem.

```
GooglePlayStore$Price <- gsub("[[:punct:]]", "", GooglePlayStore$Price)
GooglePlayStore$Price <- as.numeric(GooglePlayStore$Price)
GooglePlayStore$Price <- GooglePlayStore$Price/100
```

We determined that the Current.Ver variable was irrelevant to our goals, so here we remove it.

```
GooglePlayStore <- GooglePlayStore[, -10]
```

We also determined that the Android.Ver variable was irrelevant to our goals, so here we remove that as well.

```
GooglePlayStore <- GooglePlayStore[, -10]
```

Here we write the cleaned data set to disk.

```
write.csv(GooglePlayStore,"Prepared_Google_Play_Store_App_Data.csv",row.names=FALSE)
```

Because we intend on making a predictive model, here we split the data into a testing and training data set.

```
set.seed(42)

GooglePlayStoreTemp <- GooglePlayStore %>% mutate(id=row_number())
Train <- GooglePlayStoreTemp %>% sample_frac(0.6)
Test <- GooglePlayStoreTemp %>% anti_join(Train,by="id")
Train$id <- NULL
Test$id <- NULL
write.csv(Test,"Test.csv",row.names = FALSE)
rm(Test,GooglePlayStoreTemp)
```

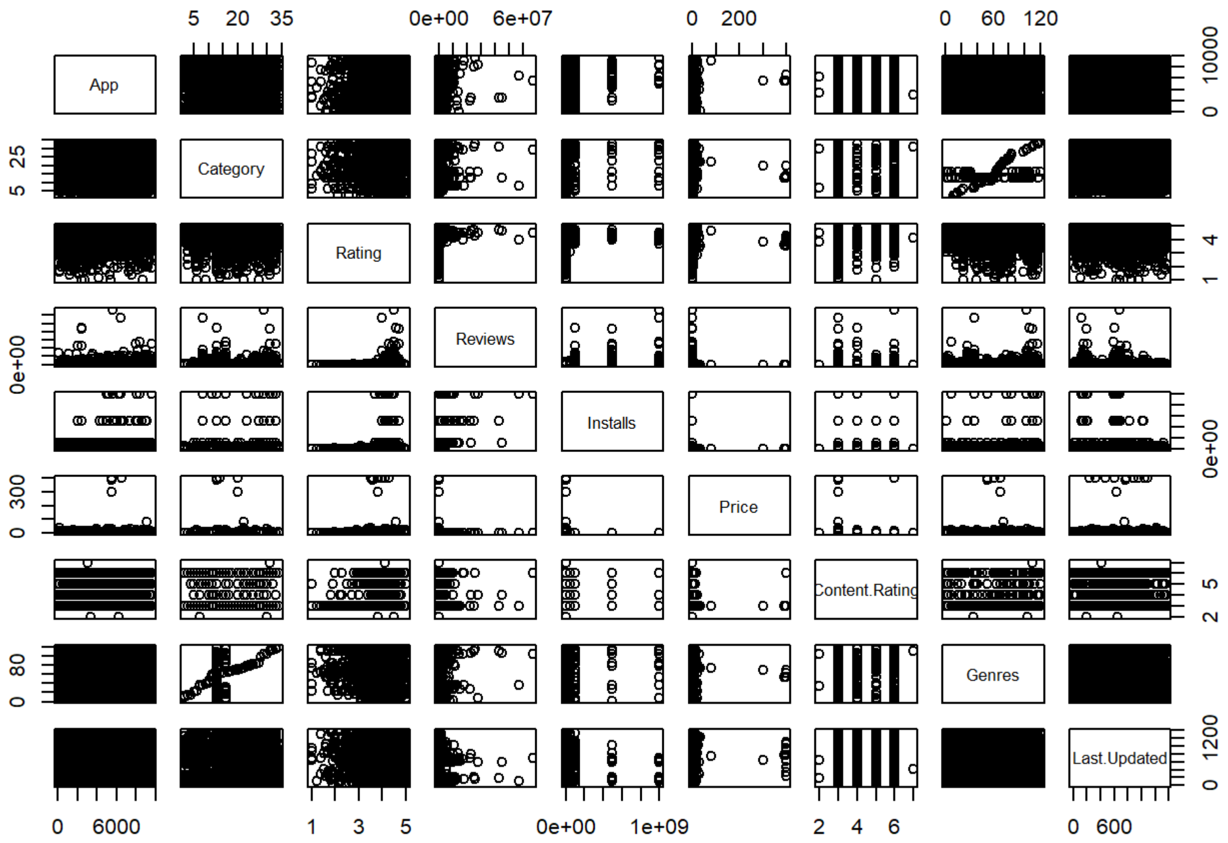
Exploratory Data Analysis

To begin the exploratory data analysis, let’s look at the data set as a whole.

```
summary(Train)
```

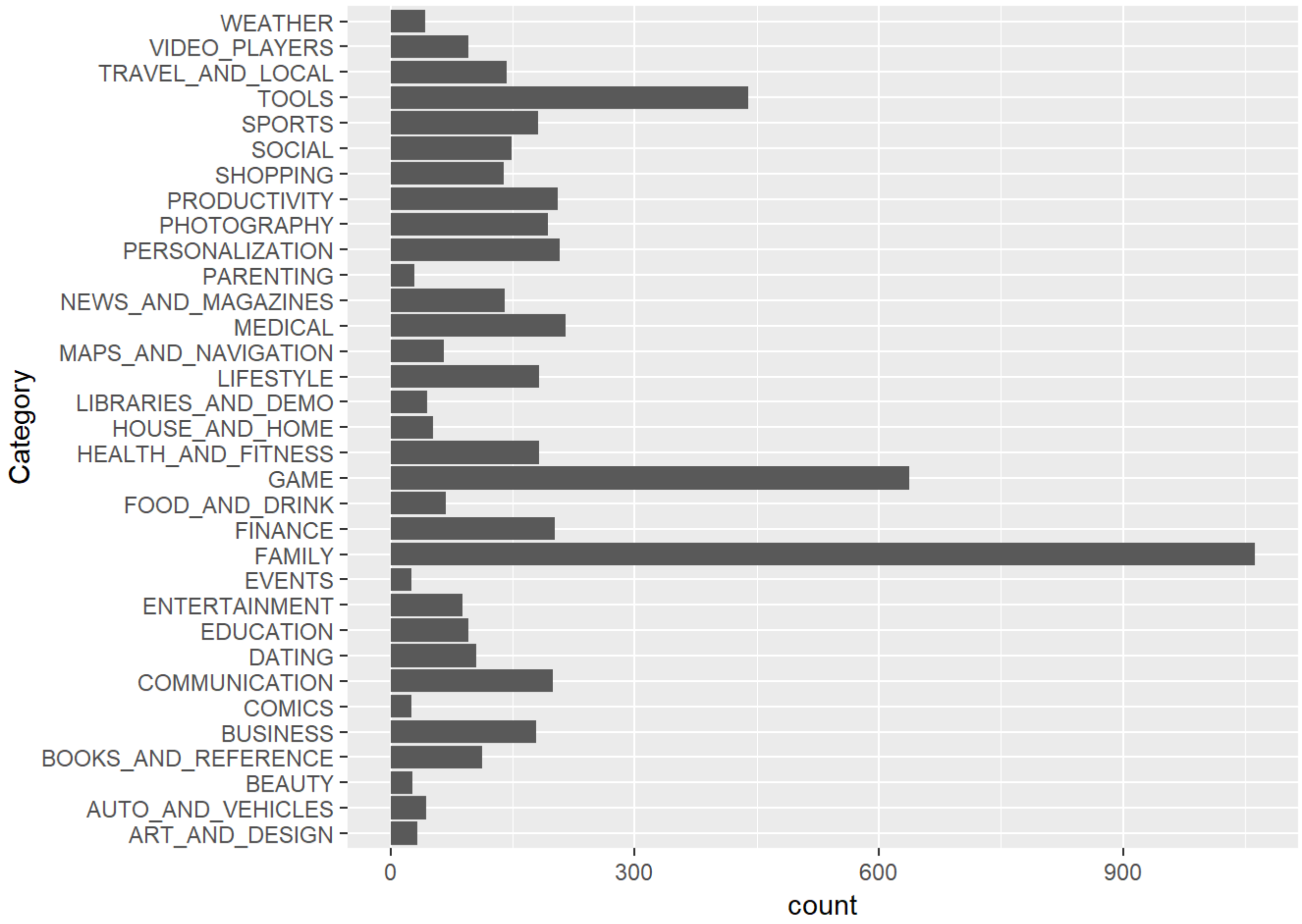
```
##                                                  App
##  CBS Sports App - Scores, News, Stats & Watch Live:   6
##  Nick                                             :   6
##  ROBLOX                                           :   5
##  Subway Surfers                                   :   5
##  8 Ball Pool                                      :   4
##  AliExpress - Smarter Shopping, Better Living     :   4
##  (Other)                                          :5590
##             Category        Rating         Reviews
##  FAMILY         :1062   Min.   :1.000   Min.   :       1
##  GAME           : 638   1st Qu.:4.000   1st Qu.:     192
##  TOOLS          : 440   Median :4.300   Median :    6078
##  MEDICAL        : 215   Mean   :4.191   Mean   :  471539
##  PERSONALIZATION: 208   3rd Qu.:4.500   3rd Qu.:   80327
##  PRODUCTIVITY   : 206   Max.   :5.000   Max.   :66509917
##  (Other)        :2851
##     Installs            Price                Content.Rating
##  Min.   :1.00e+00   Min.   :  0.000                  :   0
##  1st Qu.:1.00e+04   1st Qu.:  0.000   Adults only 18+:   2
##  Median :5.00e+05   Median :  0.000   Everyone       :4490
##  Mean   :1.83e+07   Mean   :  1.084   Everyone 10+   : 225
##  3rd Qu.:5.00e+06   3rd Qu.:  0.000   Mature 17+     : 254
##  Max.   :1.00e+09   Max.   :400.000   Teen           : 648
##                                       Unrated        :   1
##              Genres             Last.Updated
##  Tools          : 440   August 3, 2018: 203
##  Entertainment  : 325   July 31, 2018 : 173
##  Education      : 293   August 2, 2018: 172
##  Medical        : 215   August 1, 2018: 170
##  Action         : 211   July 30, 2018 : 103
##  Personalization: 208   August 6, 2018:  98
##  (Other)        :3928   (Other)       :4701
```

```
pairs(Train)
```

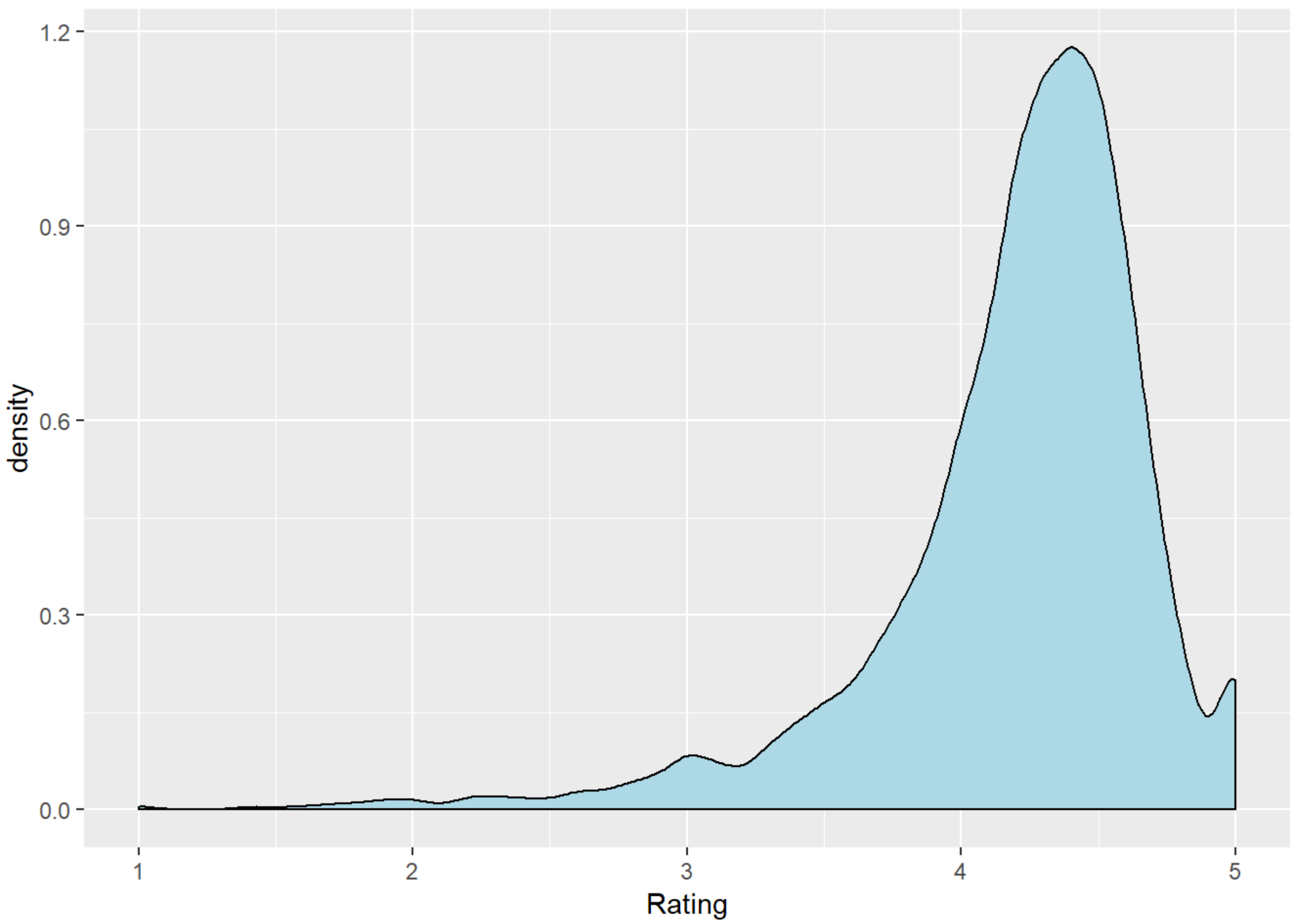


Now, let's look at the individual variables.

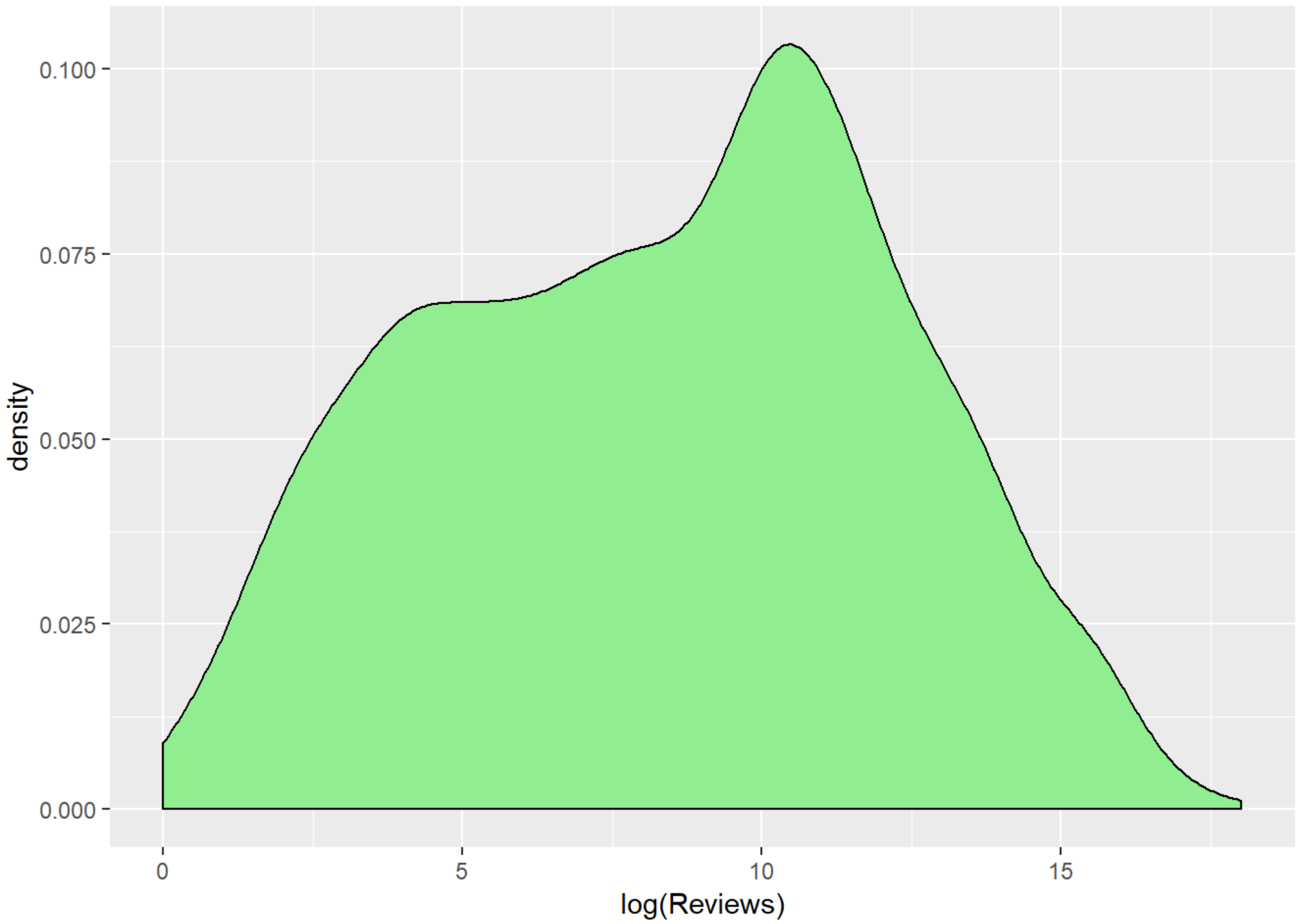
```
ggplot(Train) + geom_bar(aes(x=Category)) + coord_flip()
```



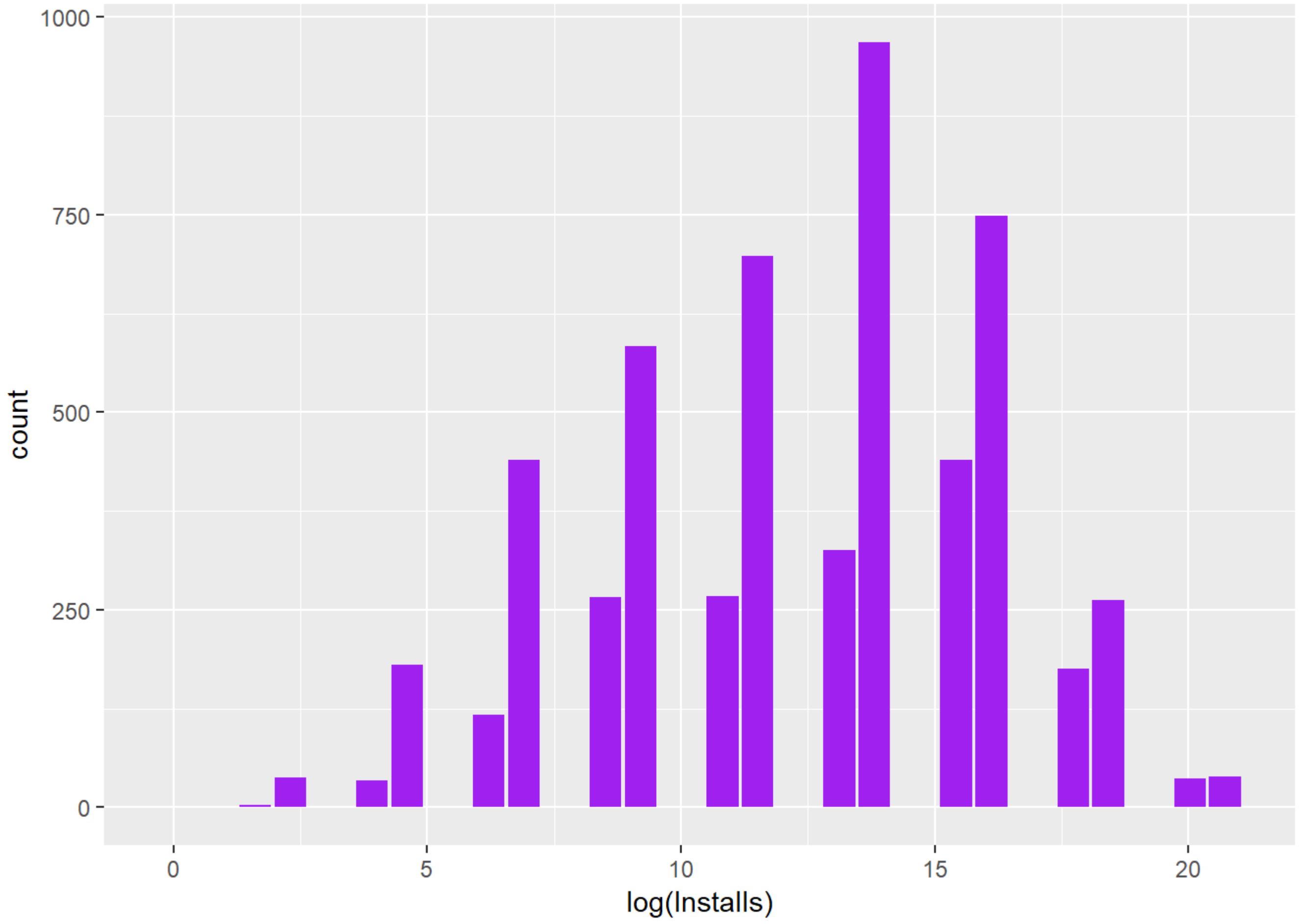
```
ggplot(Train) + geom_density(aes(x=Rating),fill="light blue")
```



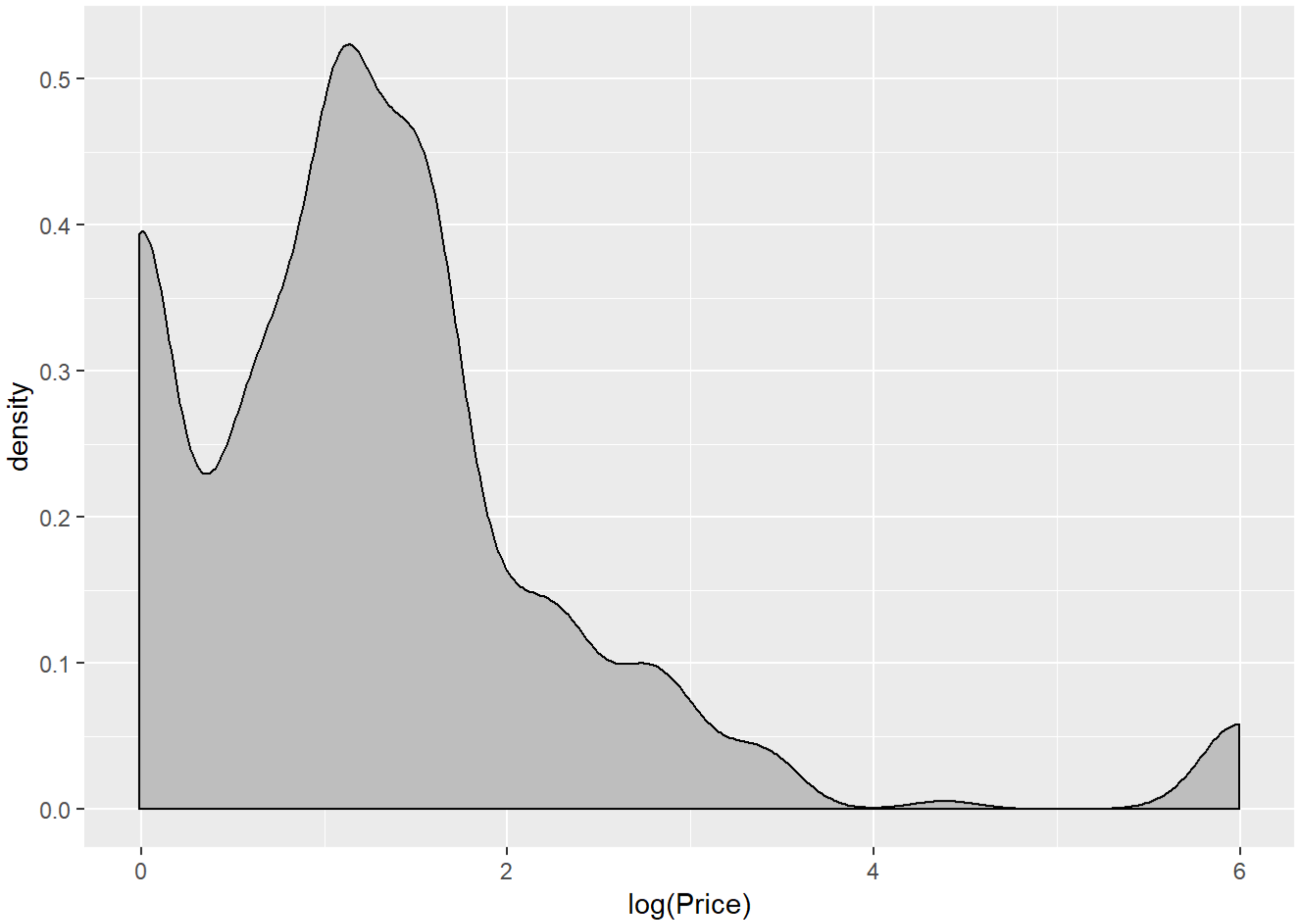
```
ggplot(Train) + geom_density(aes(x=log(Reviews)),fill="light green")
```



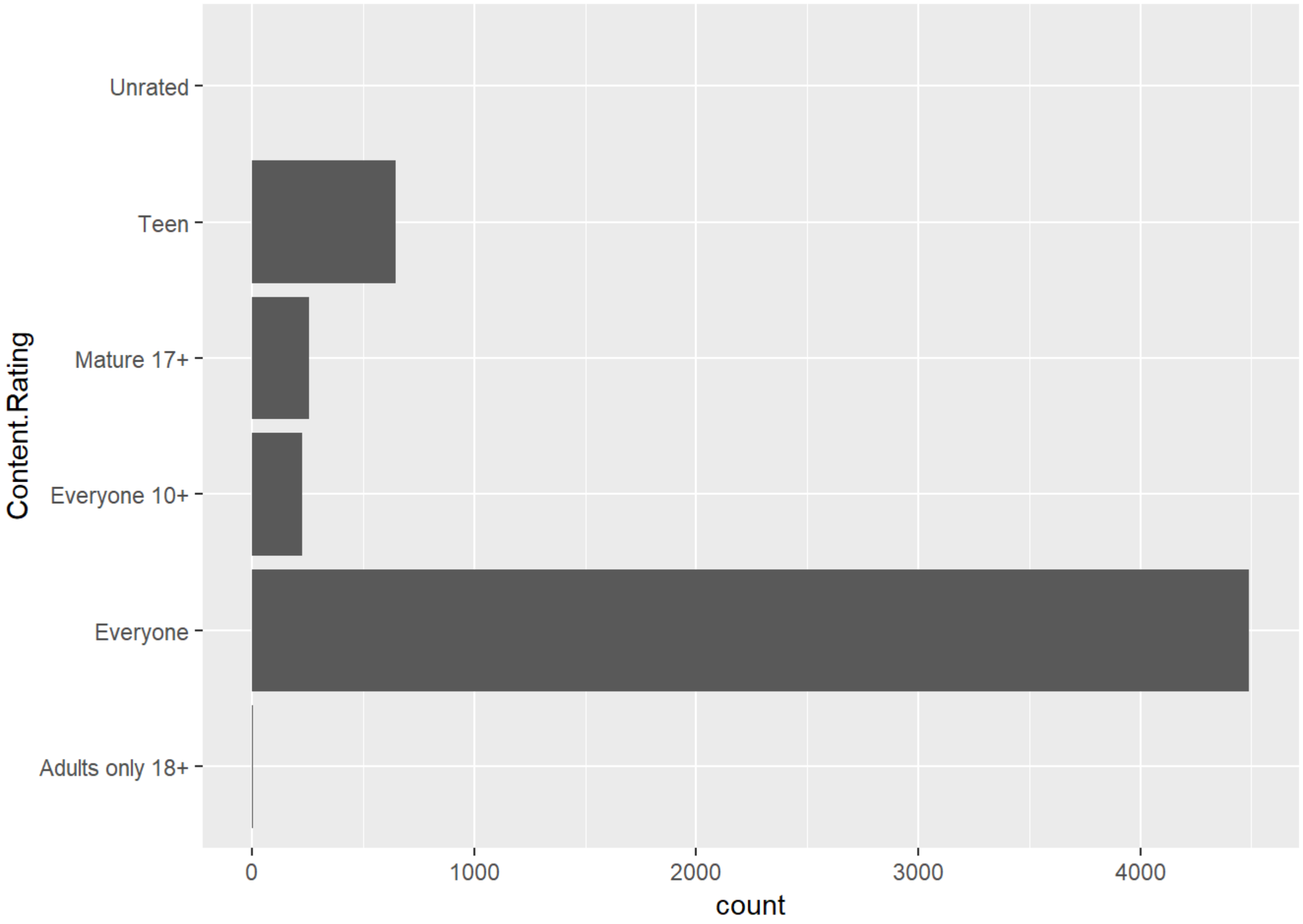
```
ggplot(Train) + geom_bar(aes(x=log(Installs)),fill="purple")
```



```
ggplot(Train) + geom_density(aes(x=log(Price)),fill="gray")
```

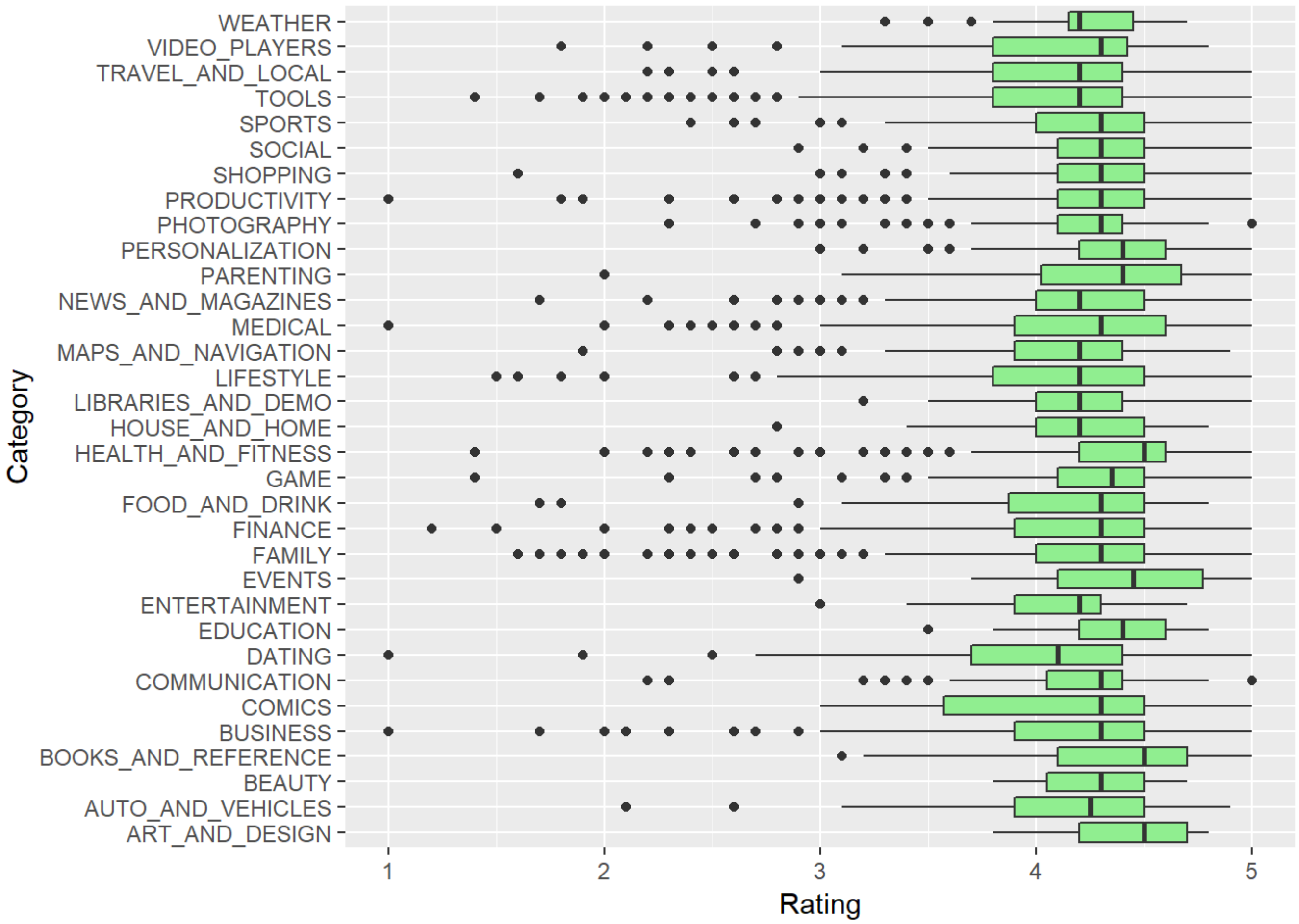


```
ggplot(Train) + geom_bar(aes(x=Content.Rating)) + coord_flip()
```

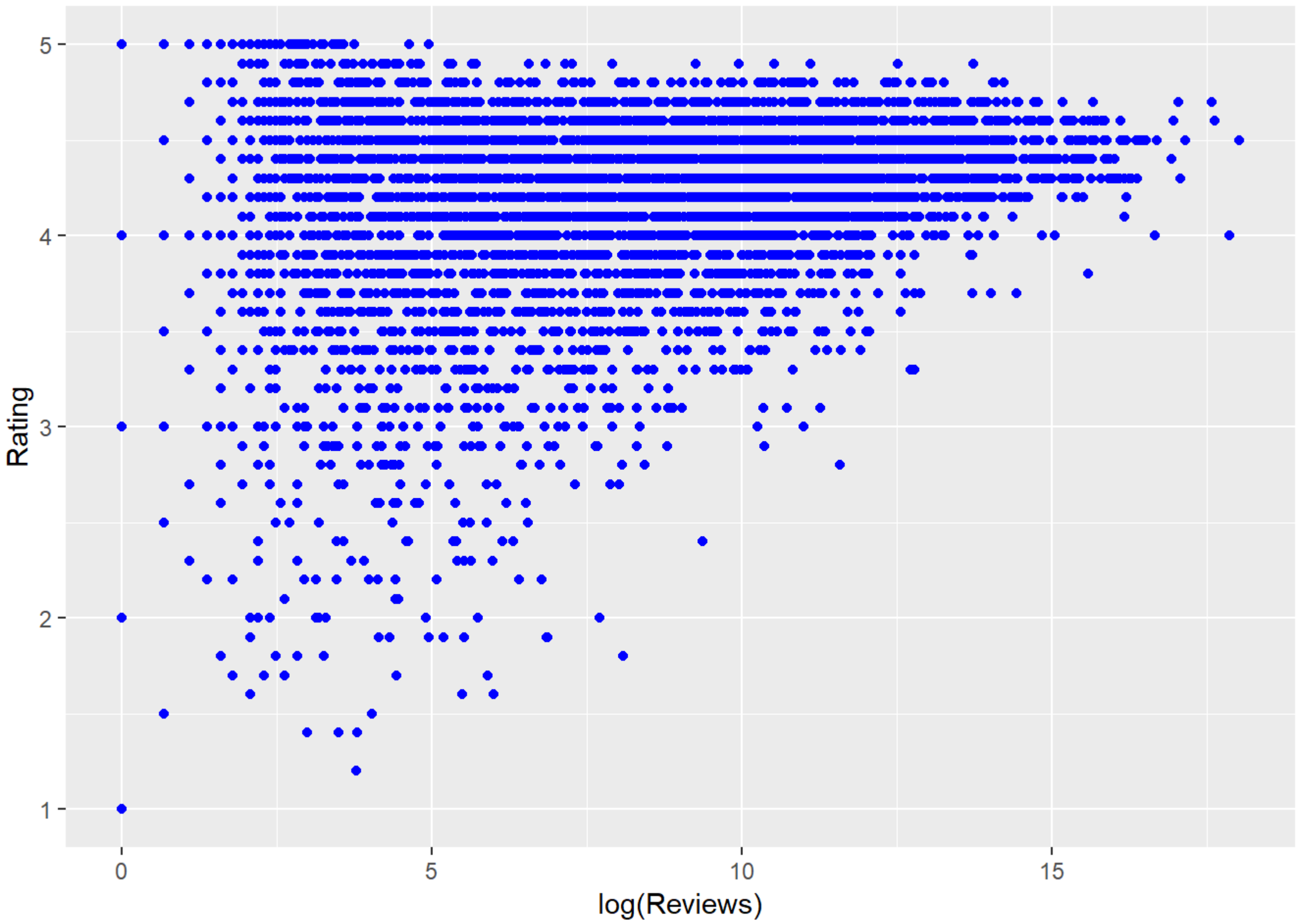


Let’s look at some of the relationships between the Rating variable, our response, and some of the possible predictors.

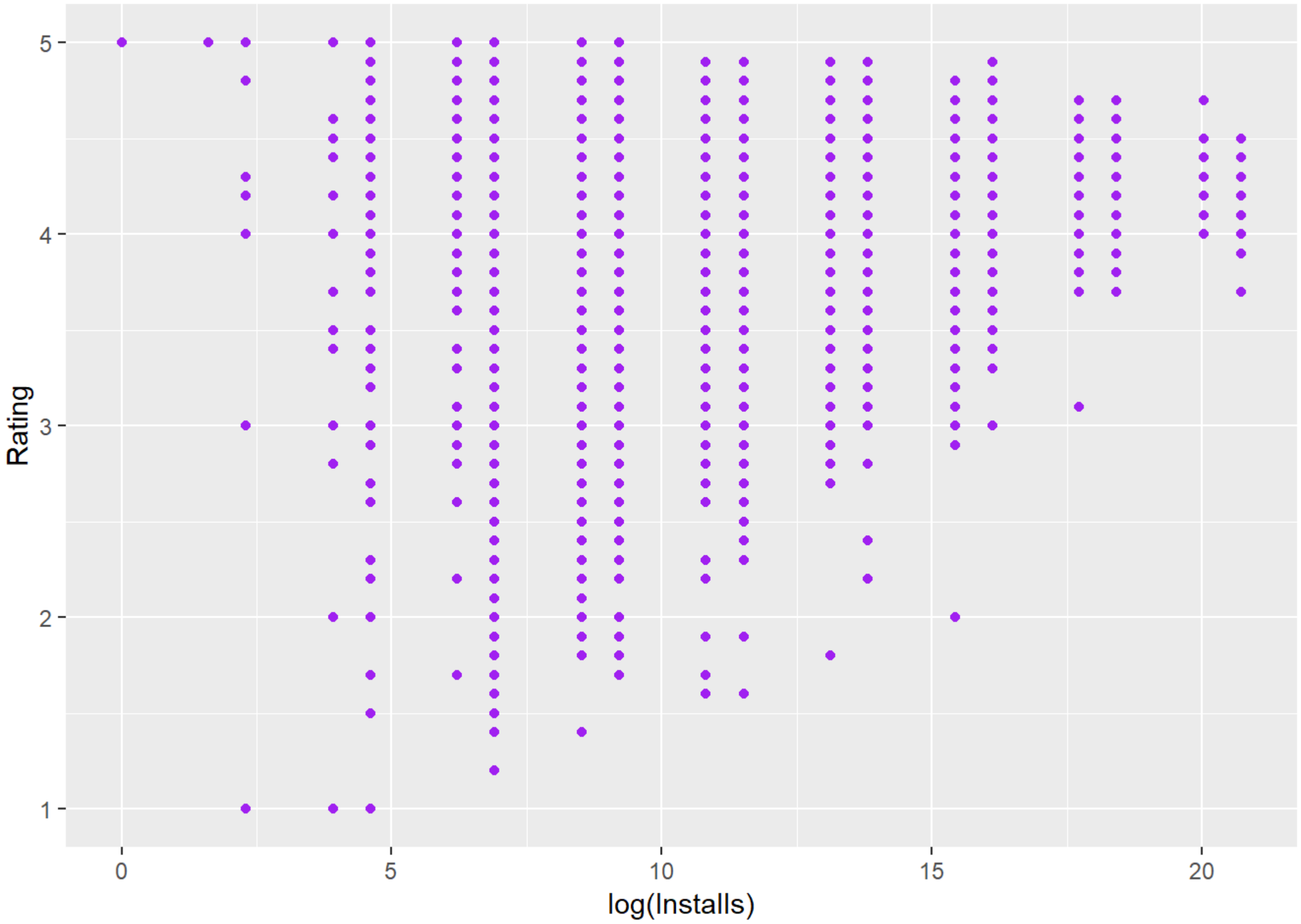
```
ggplot(Train) + geom_boxplot(aes(x=Category,y=Rating), fill="light green") + coord_flip()
```



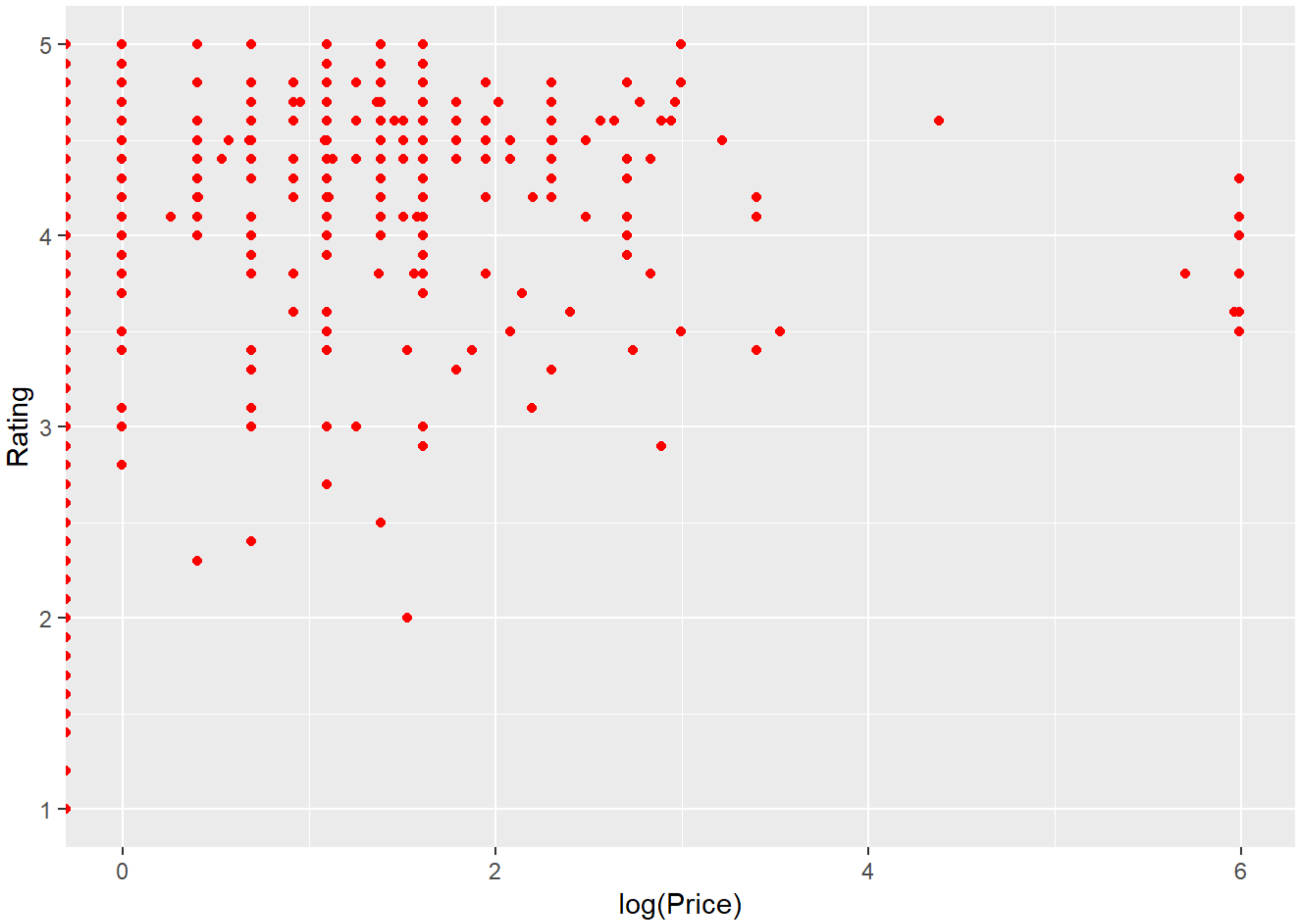
```
ggplot(Train) + geom_point(aes(x=log(Reviews),y=Rating), color="blue")
```



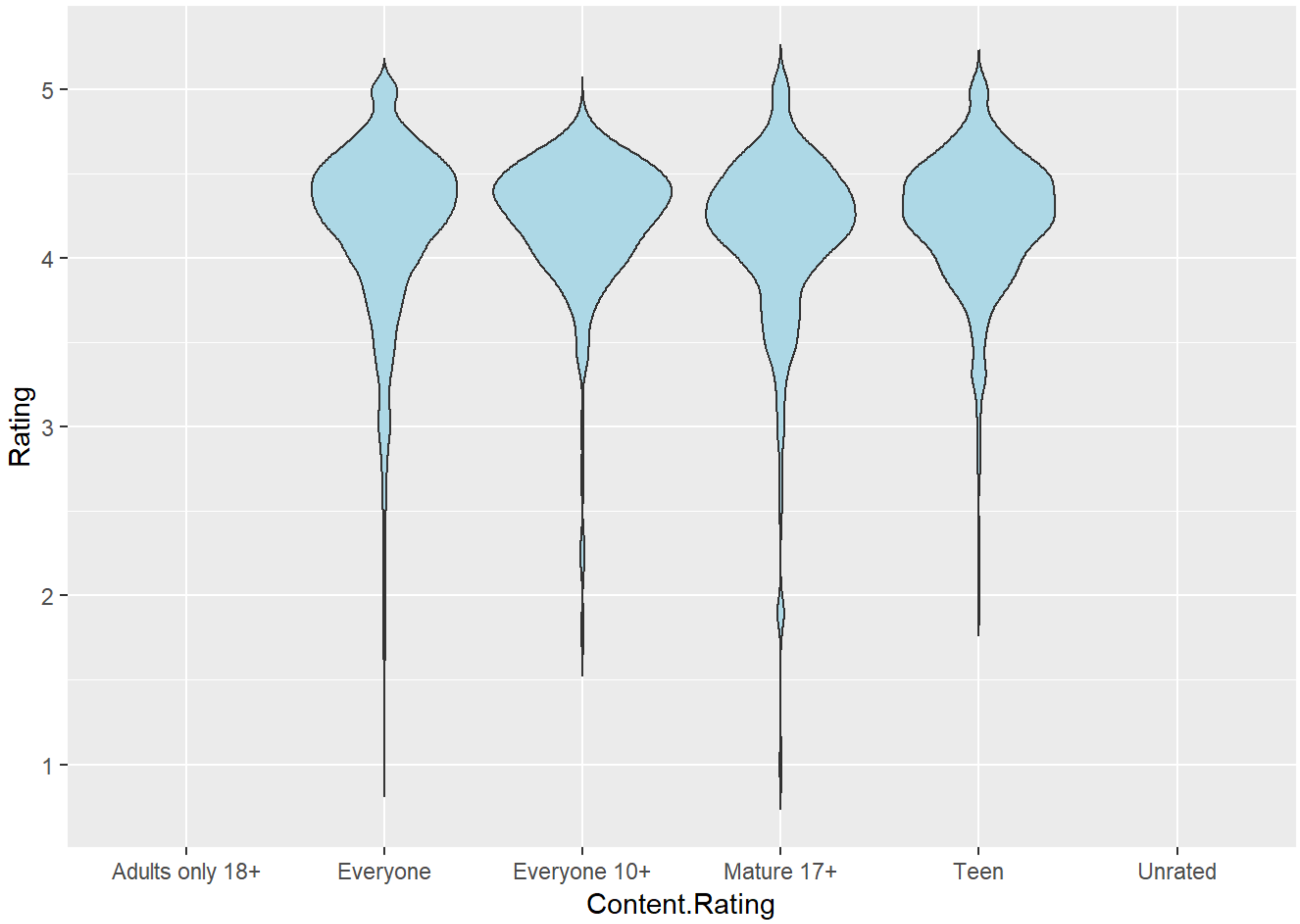
```
ggplot(Train) + geom_point(aes(x=log(Installs),y=Rating), color="purple")
```



```
ggplot(Train) + geom_point(aes(x=log(Price),y=Rating), color="red")
```

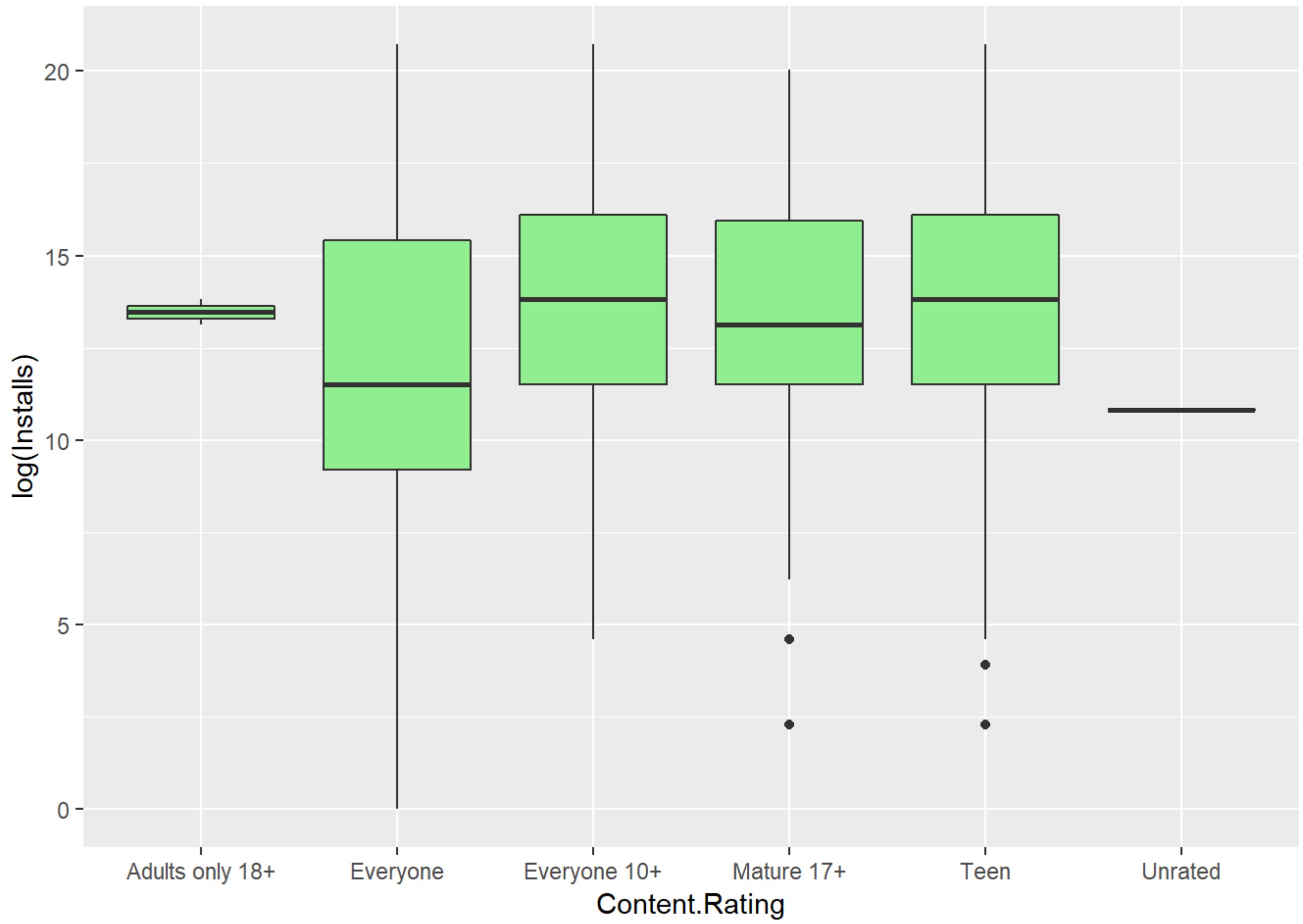


```
ggplot(Train) + geom_violin(aes(x=Content.Rating,y=Rating), trim=FALSE, fill="light blue")
```

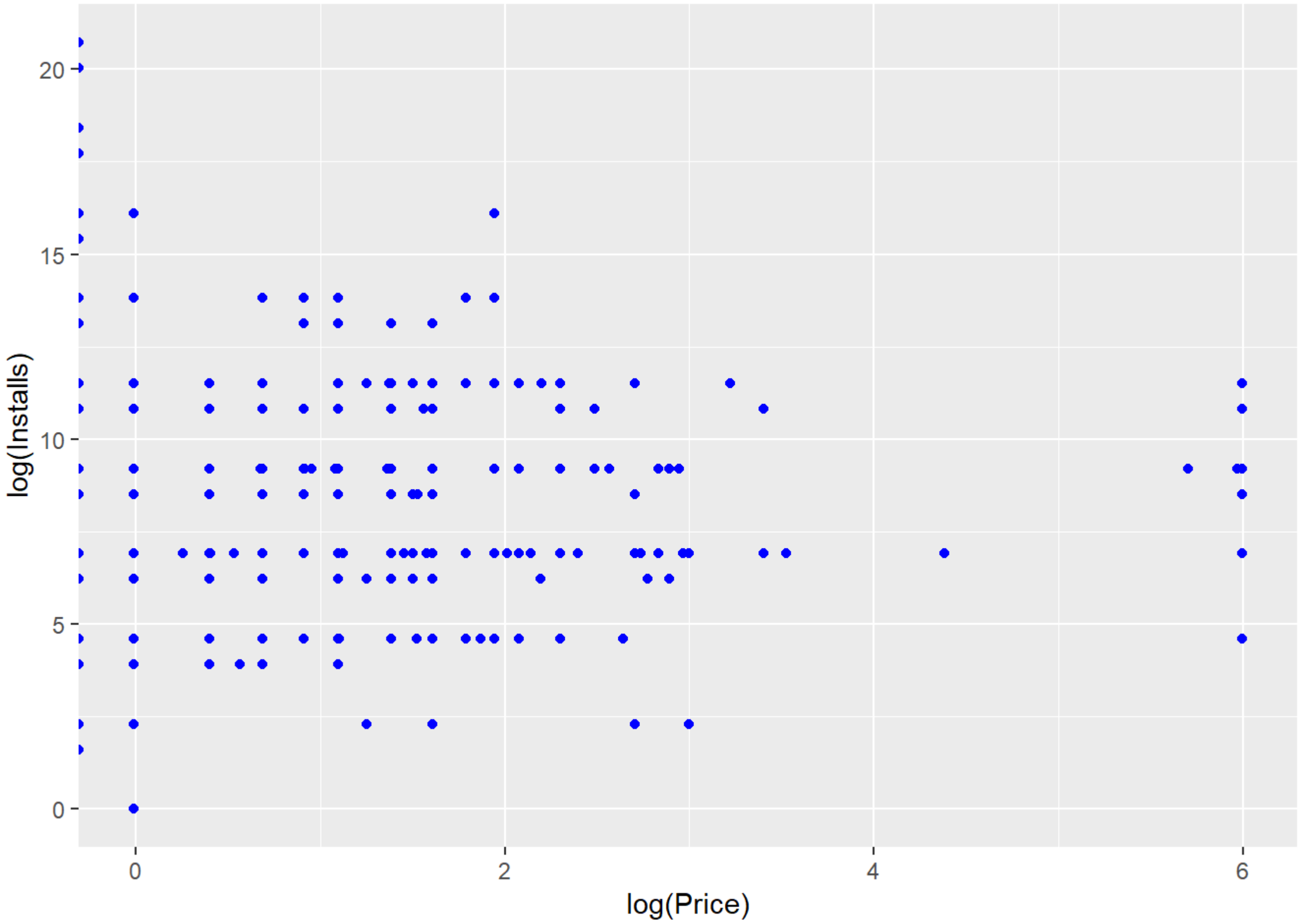


Now, let’s look at some of the relationships between the other variables.

```
ggplot(Train) + geom_boxplot(aes(x=Content.Rating,y=log(Installs)), fill="light green")
```



```
ggplot(Train) + geom_point(aes(x=log(Price),y=log(Installs)), color="blue")
```



Conclusion

It appears from the data that the Reviews variable seems quite positively related to the Ratings variable. While ratings of 5 can be achieved across all Review amounts, they seem to be more common with higher Review amounts. The same can be said for the Installs variable, in that the more installations a particular app receives could correlate to the rating it is given. The Price appears to have essentially the opposite effect. With the exception of free apps, lower priced apps appear to earn higher rating scores thant more expensive apps. Content rating does not appear to affect an app’s rating.

With regard to the other relationships explored, Content Rating seemd very equal across app installations, with apps rated as Everyone had the most variance, probably due to the fact that most of the apps in the data set are rated Everyone, as seen by the Content Rating bar plot earlier. When price was compared to installations, an almost bubble appeared in the bottom right portion of the plot, indicating that less expensive apps might warrent more installations from users.

Moving forward, we feel that the model we are aiming to create should include the Review, Installs, and Price variables, possibly Genres and Category as well.