HarvardX Capstone: Movielens

Craig Barnes

7/25/2021

Contents

Intro / Project Overview

Since the early days of the internet, people have talked about movies and given them ratings and reviews. The birth of Netflix (among other streaming services like Hulu) gave rise to one specific kind of data science, Recommendation Systems!

The HarvardX Data Science Capstone: Movielens project is built upon the question:

> "Can I predict what rating any particular user should give to any certain movie?"

The Root Mean Squared Error targets are below:
5 points: >= .9
10 points: 0.86550 <= RMSE <= 0.89999
15 points: 0.86500 <= RMSE <= 0.86549
20 points: 0.86490 <= RMSE <= 0.86499
25 points: RMSE < 0.86490

I built a project from scratch that pulls together all the different levels of complexity when creating a Recommendation System. Recall from our Machine Learning course that some factors to be considered are: each movie is unique in quality and came out during a specific era, each user has their own preferences with respect to movie genres, actors, etc. Also, movies can be brand new or classics (i.e. much older), and generally speaking some genres of movies are simply more agreeable to more people!

In the modeling phase, I attempted a few different options. First, I tried to create a "user profile" and some similarity matrices that would function as relational databases in a way. This option was very accurate, but not flexible and it took a few seconds per prediction to run. This would have taken a week or so to execute a million rows! Secondly, I attempted to fit some simple models based off user/genre preferences. This was not accurate enough to move forward with, but was very fast. Finally, I went with what would serve both an accurate prediction and run quickly enough. The methodology is similar to how we built out predictions in our ML course. Mainly, I didn't want to use a package that did all the analysis for me, like Recosystem for example. Sadly, I tried to beat the Netflix challenge goal of <.8649 and could not… I was able to get in the .87 range though, after a number of different techniques.

The model presented below uses a number of the above components to build out a best guess based on what we know from the training set. In the end, I think that every model can be improved upon… but for the purpose of this project (and given my hardware limitations) this exercise helped to reinforce a ton of skills I learned from the HarvardX Data Science program!

Part 1: The Data

The data itself for the Movielens project is pretty straightforward. Since all students are using the same data, I will not go through the trivial aspects of it, but it is very interesting to note how many users there are compared to movies (~70k users vs 10.6k movies). These dimensions, combined with that fact that not many users rate a lot of movies creates the sparse-matrix that we have seen. In my opinion, this creates the biggest hurdle to the exercise.

The basic dataset has six columns, all of which I use in one way or another.
userId and **movieId** are critical, and of course **rating** is our Y-variable if you consider a formula such as Y~x1+x2+x3.
timestamp is used to calculate how old a movie was when it was reviewed, and year is pulled from the movie **title** and is simply put how old it is now (when compared to this year.)
genres is used to create some one-hot encoding that I fit a small linear model with to test out the effects each genre has to the average rating.

```
[1] "Here's a look at the raw data before any modifications:"
```

```
Rows: 5
Columns: 6
$ userId    <int> 1, 1, 1, 1, 1
$ movieId   <dbl> 122, 185, 231, 292, 316
$ rating    <dbl> 5, 5, 5, 5, 5
$ timestamp <int> 838985046, 838983525, 838983392, 838983421, 838983392
$ title     <chr> "Boomerang (1992)", "Net, The (1995)", "Dumb & Dumber (19...
$ genres    <chr> "Comedy|Romance", "Action|Crime|Thriller", "Comedy", "Act...
```

```
[1] "Number of Unique Users: 69878"
```

```
[1] "Number of Unique Movies: 10677"
```

```
[1] "Number of Movies that fall into each genre: "
```

```
      genres    n
1      Drama 5336
2     Comedy 3703
3   Thriller 1705
4    Romance 1685
5     Action 1473
6      Crime 1117
7  Adventure 1025
8     Horror 1013
9     Sci-Fi  754
10   Fantasy  543
```

The next section shows some feature engineering referenced above. I create the year-reviewed out of the **timestamp**, and then some one-hot encoding columns using str_detect.
Next, I extract the **year** the movie came out from the **title**.
Then, I create some discrete age buckets comprised of 1 year or less old, 3 years or less old, 5 years or less old…all the way to 50 years old or more.
These all serve to give me some more color as I segment out different users and movies based on various traits - for example:

> Should a 5 year old comedy have a better rating than a 30 year old drama on average (all else equal)?

```r
# Feature engineering prior to split get a timestamp, add some one-hot encoding
# for top 10 genres
movielens$timestamp <- as.Date(as.POSIXct(movielens$timestamp, origin = "1970-01-01"))
movielens$Drama1h <- ifelse(str_detect(movielens$genres, "Drama"), 1, 0)
movielens$Comedy1h <- ifelse(str_detect(movielens$genres, "Comedy"), 1, 0)
movielens$Thriller1h <- ifelse(str_detect(movielens$genres, "Thriller"), 1, 0)
movielens$Romance1h <- ifelse(str_detect(movielens$genres, "Romance"), 1, 0)
movielens$Action1h <- ifelse(str_detect(movielens$genres, "Action"), 1, 0)
movielens$Crime1h <- ifelse(str_detect(movielens$genres, "Crime"), 1, 0)
movielens$Adventure1h <- ifelse(str_detect(movielens$genres, "Adventure"), 1, 0)
movielens$Horror1h <- ifelse(str_detect(movielens$genres, "Horror"), 1, 0)
movielens$SciFi1h <- ifelse(str_detect(movielens$genres, "Sci-Fi"), 1, 0)
movielens$Fantasy1h <- ifelse(str_detect(movielens$genres, "Fantasy"), 1, 0)
#
ml1 <- movielens %>% extract(title, c("title_tmp", "year"), regex = "^(.*) \\(([0-9 \\-]*)\\)$",
    remove = F)
ml1$year <- as.Date(paste(ml1$year, "-06-15", sep = ""))
# using june 15th of the year for every movie since it is the middle of the year
ml1$movieAGE <- as.double((ml1$timestamp - ml1$year)/365)
# calculate age of movie relative to each rating
ml1$agegroup <- ifelse(ml1$movieAGE <= 1, "1Y", ifelse(ml1$movieAGE <= 3, "3Y", ifelse(ml1$movieAGE <=
    5, "5Y", ifelse(ml1$movieAGE <= 10, "10Y", ifelse(ml1$movieAGE < 15, "15Y", ifelse(ml1$movieAGE <=
    20, "20Y", ifelse(ml1$movieAGE <= 30, "30Y", ifelse(ml1$movieAGE <= 50, "50Y",
    "50plus"))))))))
#
movielens <- ml1 %>% select(-title) %>% rename(., Title = title_tmp)
#
rm(ml1)
glimpse(movielens[1:3, ])
```

```
> Rows: 3
> Columns: 19
> $ userId      <int> 1, 1, 1
> $ movieId     <dbl> 122, 185, 231
> $ rating      <dbl> 5, 5, 5
> $ timestamp   <date> 1996-08-02, 1996-08-02, 1996-08-02
> $ Title       <chr> "Boomerang", "Net, The", "Dumb & Dumber"
> $ year        <date> 1992-06-15, 1995-06-15, 1994-06-15
> $ genres      <chr> "Comedy|Romance", "Action|Crime|Thriller", "Comedy"
> $ Drama1h     <dbl> 0, 0, 0
> $ Comedy1h    <dbl> 1, 0, 1
> $ Thriller1h  <dbl> 0, 1, 0
> $ Romance1h   <dbl> 1, 0, 0
> $ Action1h    <dbl> 0, 1, 0
> $ Crime1h     <dbl> 0, 1, 0
> $ Adventure1h <dbl> 0, 0, 0
> $ Horror1h    <dbl> 0, 0, 0
> $ SciFi1h     <dbl> 0, 0, 0
> $ Fantasy1h   <dbl> 0, 0, 0
> $ movieAGE    <dbl> 4.134, 1.134, 2.134
> $ agegroup    <chr> "5Y", "3Y", "3Y"
```

```r
dim(movielens)
```

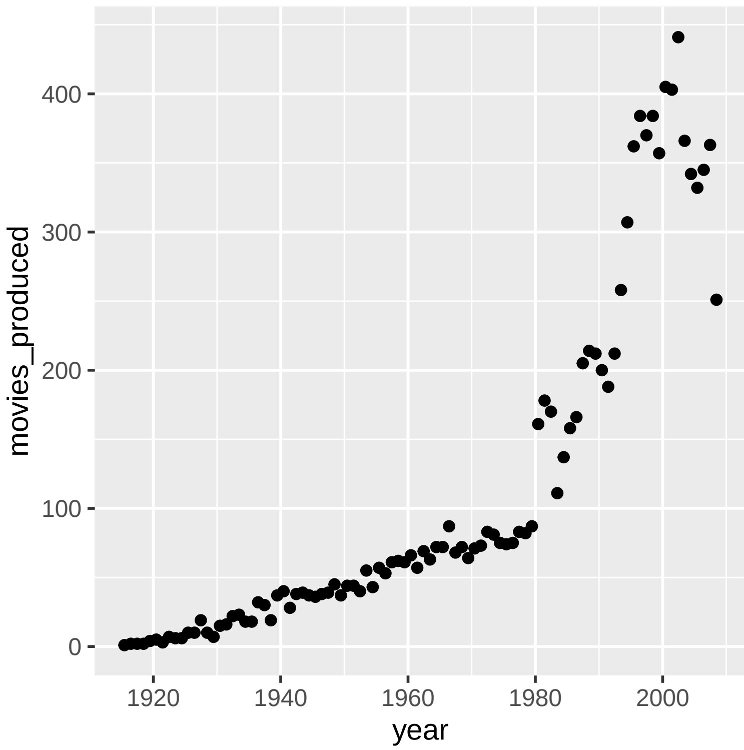
```
> [1] 10000054       19
```

```r
summary(movielens)
```

```
>      userId         movieId          rating       timestamp
>  Min.   :    1   Min.   :    1   Min.   :0.50   Min.   :1995-01-09
>  1st Qu.:18123   1st Qu.:  648   1st Qu.:3.00   1st Qu.:2000-01-01
>  Median :35740   Median : 1834   Median :4.00   Median :2002-10-24
>  Mean   :35870   Mean   : 4120   Mean   :3.51   Mean   :2002-09-20
>  3rd Qu.:53608   3rd Qu.: 3624   3rd Qu.:4.00   3rd Qu.:2005-09-15
>  Max.   :71567   Max.   :65133   Max.   :5.00   Max.   :2009-01-05
>     Title                year                genres             Drama1h
>  Length:10000054    Min.   :1915-06-15   Length:10000054    Min.   :0.000
>  Class :character   1st Qu.:1987-06-15   Class :character   1st Qu.:0.000
>  Mode  :character   Median :1994-06-15   Mode  :character   Median :0.000
>                     Mean   :1990-09-03                      Mean   :0.434
>                     3rd Qu.:1998-06-15                      3rd Qu.:1.000
>                     Max.   :2008-06-15                      Max.   :1.000
>     Comedy1h       Thriller1h      Romance1h       Action1h        Crime1h
>  Min.   :0.000   Min.   :0.000   Min.   :0.00   Min.   :0.000   Min.   :0.000
>  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.00   1st Qu.:0.000   1st Qu.:0.000
>  Median :0.000   Median :0.000   Median :0.00   Median :0.000   Median :0.000
>  Mean   :0.393   Mean   :0.258   Mean   :0.19   Mean   :0.284   Mean   :0.147
>  3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:0.00   3rd Qu.:1.000   3rd Qu.:0.000
>  Max.   :1.000   Max.   :1.000   Max.   :1.00   Max.   :1.000   Max.   :1.000
>   Adventure1h       Horror1h         SciFi1h        Fantasy1h
>  Min.   :0.000   Min.   :0.0000   Min.   :0.000   Min.   :0.000
>  1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:0.000
>  Median :0.000   Median :0.0000   Median :0.000   Median :0.000
>  Mean   :0.212   Mean   :0.0768   Mean   :0.149   Mean   :0.103
>  3rd Qu.:0.000   3rd Qu.:0.0000   3rd Qu.:0.000   3rd Qu.:0.000
>  Max.   :1.000   Max.   :1.0000   Max.   :1.000   Max.   :1.000
>     movieAGE       agegroup
>  Min.   :-1.74   Length:10000054
>  1st Qu.: 2.48   Class :character
>  Median : 7.22   Mode  :character
>  Mean   :12.06
>  3rd Qu.:15.94
>  Max.   :93.58
```

```r
#
```

Take a look at how many movies came out each year:



From here, I create the train/test split, and further split the training data into train/test so that I can rename my original test as "validation set" The math is 10M original -> 9M train, 1M test -> 8.5M train, 500k test, 1M validation

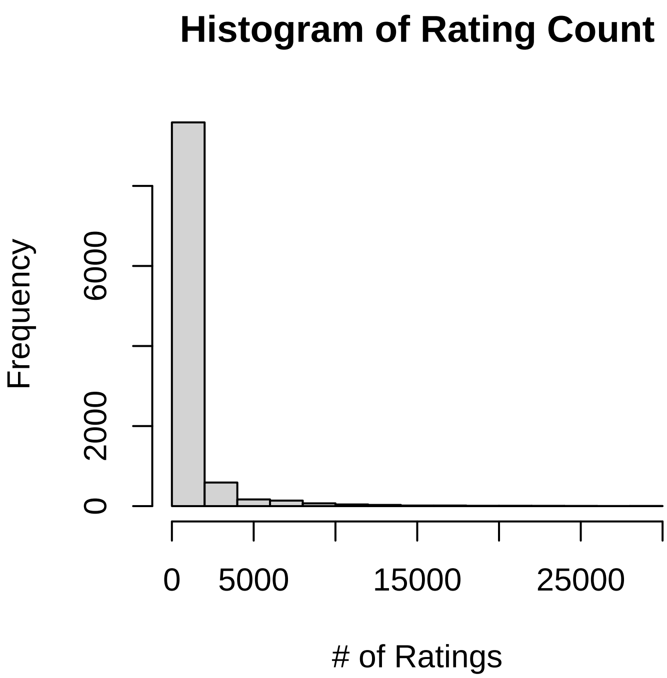
```
> [1] "Here's a look at the training data up to now: "
```

```
> Rows: 3
> Columns: 19
> $ userId      <int> 1, 1, 1
> $ movieId     <dbl> 185, 231, 292
> $ rating      <dbl> 5, 5, 5
> $ timestamp   <date> 1996-08-02, 1996-08-02, 1996-08-02
> $ Title       <chr> "Net, The", "Dumb & Dumber", "Outbreak"
> $ year        <date> 1995-06-15, 1994-06-15, 1995-06-15
> $ genres      <chr> "Action|Crime|Thriller", "Comedy", "Action|Drama|Sci-Fi...
> $ Drama1h     <dbl> 0, 0, 1
> $ Comedy1h    <dbl> 0, 1, 0
> $ Thriller1h  <dbl> 1, 0, 1
> $ Romance1h   <dbl> 0, 0, 0
> $ Action1h    <dbl> 1, 0, 1
> $ Crime1h     <dbl> 1, 0, 0
> $ Adventure1h <dbl> 0, 0, 0
> $ Horror1h    <dbl> 0, 0, 0
> $ SciFi1h     <dbl> 0, 0, 1
> $ Fantasy1h   <dbl> 0, 0, 0
> $ movieAGE    <dbl> 1.134, 2.134, 1.134
> $ agegroup    <chr> "3Y", "3Y", "3Y"
```

Let's move into how I built out my model!

Part 2: Methodology / Motivation

First, lets look into movies. You can see below that lots of movies have a few ratings, a few movies have a lot of ratings!



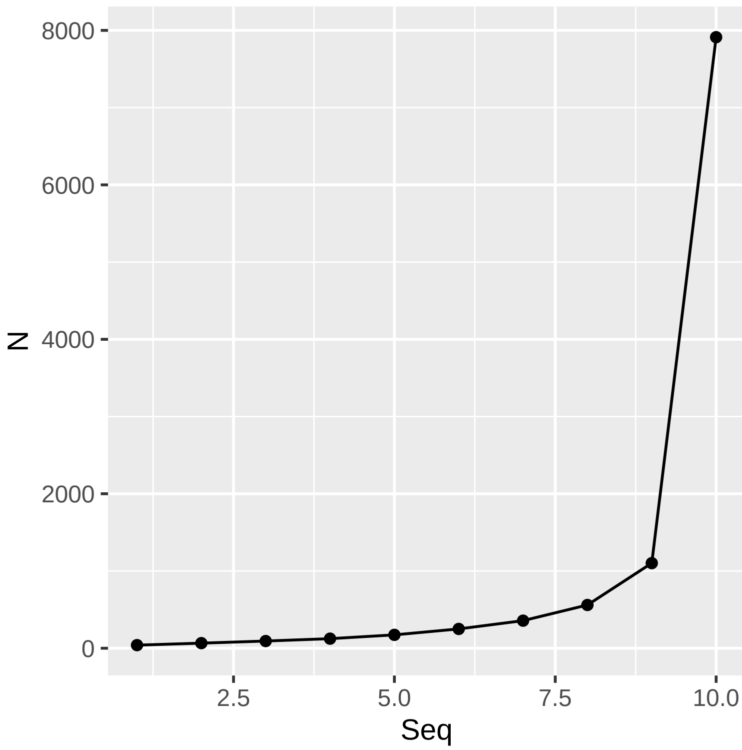
Below I take a count of ratings by movie, arrange in descending order, and run a cumulative sum to total. This will show us how many movies it takes to achieve x% of total ratings in the data, and then I bucket based on how much the movie is "worth" as a % to total.

```
> # A tibble: 10 x 4
>    movieId movie_countofrating percent cumulative
>      <dbl>               <int>   <dbl>      <dbl>
>  1     296               29777 0.00348    0.00348
>  2     356               29542 0.00346    0.00694
>  3     593               28726 0.00336    0.0103
>  4     480               27808 0.00325    0.0135
>  5     318               26742 0.00313    0.0167
>  6     110               24845 0.00291    0.0196
>  7     457               24677 0.00289    0.0225
>  8     589               24646 0.00288    0.0254
>  9     260               24405 0.00285    0.0282
> 10     150               23130 0.00271    0.0309

>
>    a    b    c    d    e    f    g    h    i    j
>   40   66   93  124  173  250  357  560 1102 7912
```

You can see above it takes 40 movies to make up the top decile of ratings, 66 movies make up the next decile, and so on.

Below, you'll see graphically how large of a tail there is with respect to movies that have very few ratings.



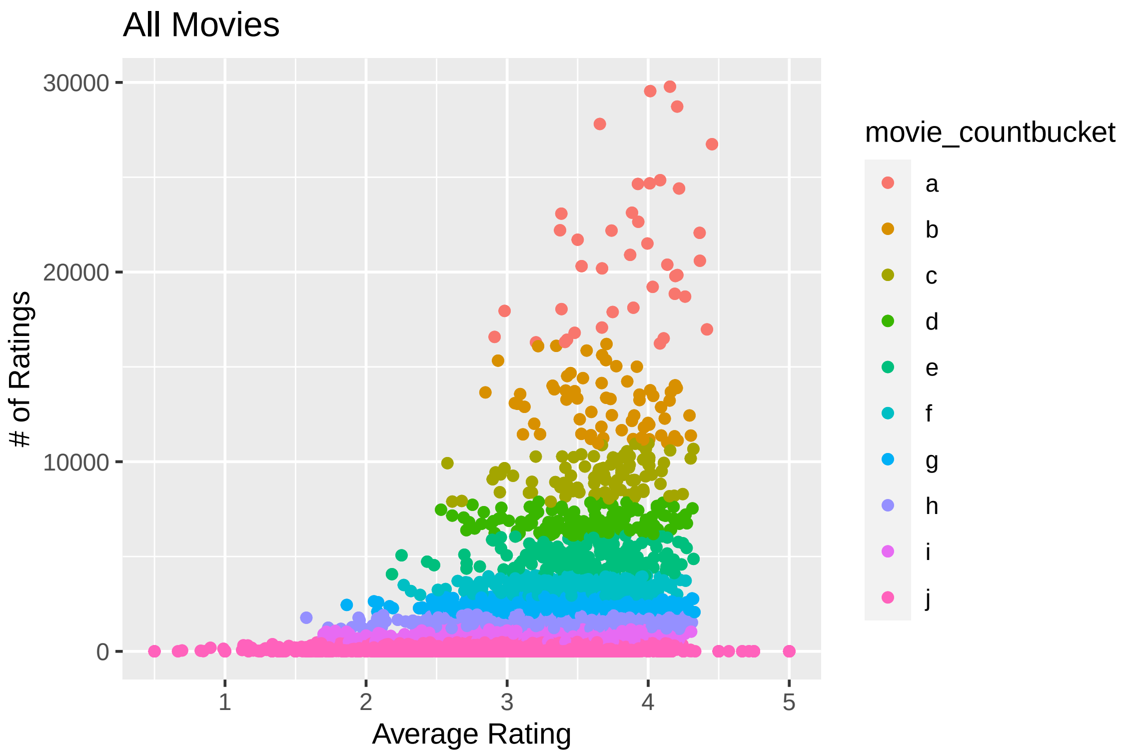
Not surprisingly, the buckets towards the beginning are rated higher. See below table:

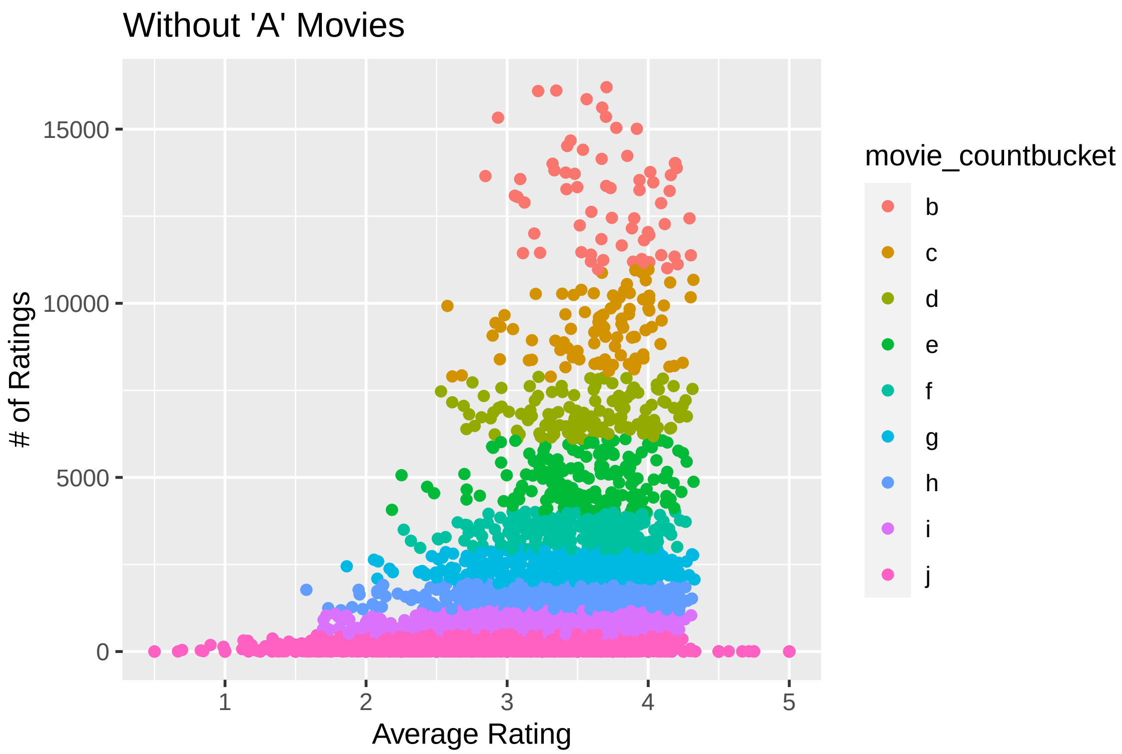
```
> # A tibble: 10 x 2
>    movie_countbucket     x
>    <chr>             <dbl>
>  1 a                  3.85
>  2 b                  3.71
>  3 c                  3.67
>  4 d                  3.56
>  5 e                  3.56
>  6 f                  3.51
>  7 g                  3.41
>  8 h                  3.38
>  9 i                  3.25
> 10 j                  3.12
```

But, how strong of a correlation are these buckets to the average rating?

```
> [1] 0.2101
```

Below, I look into the table above but with some color. First I show Average Rating vs # of Ratings for all movies. Second, I show Average Rating vs # of Ratings for all movies except movies labeled as bucket A.



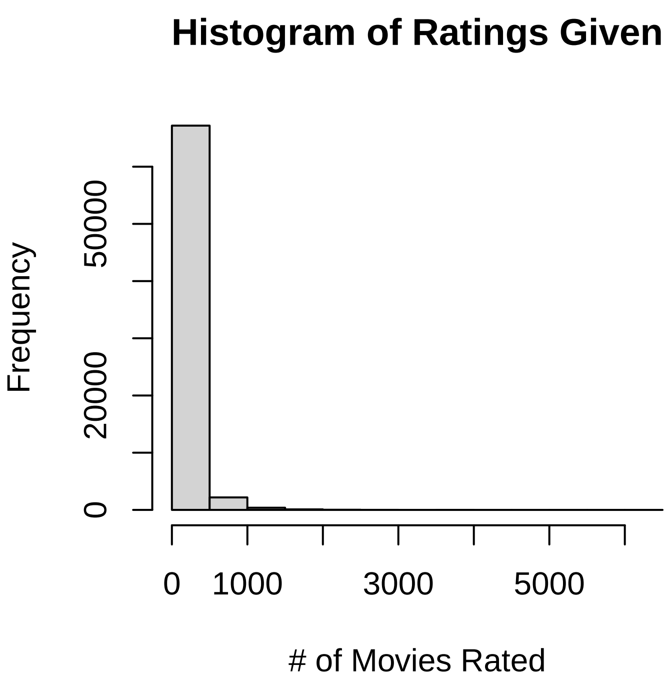


You can certainly see a shape (perhaps non-linear) in the above plots. Generally speaking, the more ratings a movie has. . . the less variance there is in average rating (or at least the distance between min and max rating for that bucket is smaller!).
Here's what the training set looks like so far:

```
> Rows: 3
```

```
> Columns: 22
> $ userId              <int> 1, 1, 1
> $ movieId             <dbl> 185, 231, 292
> $ rating              <dbl> 5, 5, 5
> $ timestamp           <date> 1996-08-02, 1996-08-02, 1996-08-02
> $ Title               <chr> "Net, The", "Dumb & Dumber", "Outbreak"
> $ year                <date> 1995-06-15, 1994-06-15, 1995-06-15
> $ genres              <chr> "Action|Crime|Thriller", "Comedy", "Action|Dram...
> $ Drama1h             <dbl> 0, 0, 1
> $ Comedy1h            <dbl> 0, 1, 0
> $ Thriller1h          <dbl> 1, 0, 1
> $ Romance1h           <dbl> 0, 0, 0
> $ Action1h            <dbl> 1, 0, 1
> $ Crime1h             <dbl> 1, 0, 0
> $ Adventure1h         <dbl> 0, 0, 0
> $ Horror1h            <dbl> 0, 0, 0
> $ SciFi1h             <dbl> 0, 0, 1
> $ Fantasy1h           <dbl> 0, 0, 0
> $ movieAGE            <dbl> 1.134, 2.134, 1.134
> $ agegroup            <chr> "3Y", "3Y", "3Y"
> $ movie_avgrating     <dbl> 3.123, 2.936, 3.415
> $ movie_countofrating <int> 12893, 15329, 13753
> $ movie_countbucket   <chr> "b", "b", "b"
```

Next, let's look into users. Just like the movies above, a few people rate a ton of movies and a ton of people rate a few movies.



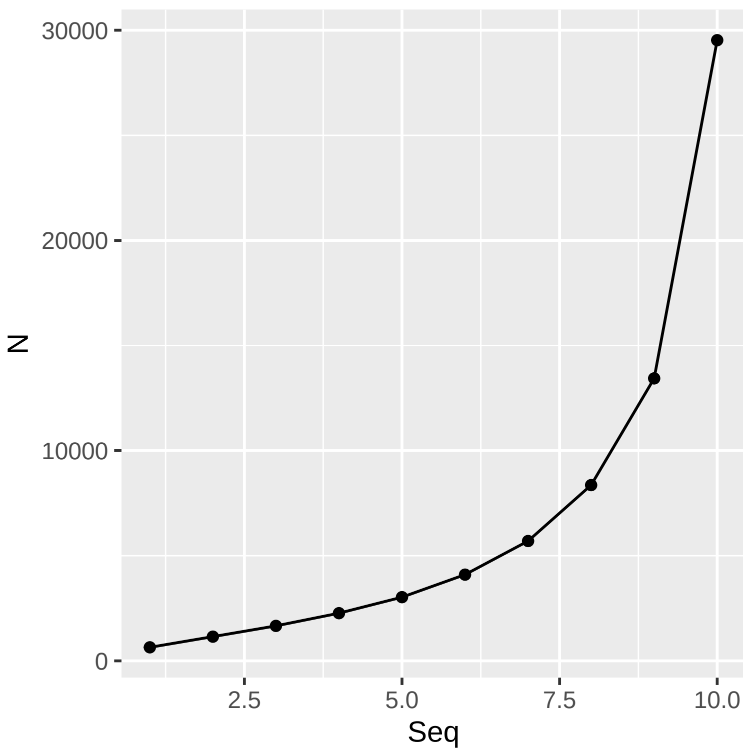

```
>      userId      user_avgrating  usermovcount
>  Min.   :    1   Min.   :0.50   Min.   :  10
>  1st Qu.:17943   1st Qu.:3.36   1st Qu.:  30
>  Median :35798   Median :3.63   Median :  59
>  Mean   :35782   Mean   :3.61   Mean   : 122
>  3rd Qu.:53620   3rd Qu.:3.90   3rd Qu.: 134
```

```
>  Max.   :71567   Max.   :5.00   Max.   :6271
```

Above, you can see that there are users with 10 ratings given, there are users with 6,271 ratings given, the mean is 122 and median is 59. Talk about an enormous variance!

```
>
>     a     b     c     d     e     f     g     h     i     j
>   643  1151  1663  2267  3029  4101  5700  8360 13437 29527
```

Just like the movie section, I bucket out the users into deciles as well. You can see that 643 users make up 10% of ratings. That means that ~1% of users make up 10% of ratings.
The figure below can illustrate this. The last two buckets account for ~43k users of the 69k total… but only 20% of ratings come from those users making up 2/3 of the user population!



A look at the training set through the above:

```
> Rows: 3
> Columns: 25
> $ userId              <int> 1, 1, 1
> $ movieId             <dbl> 185, 231, 292
> $ rating              <dbl> 5, 5, 5
> $ timestamp           <date> 1996-08-02, 1996-08-02, 1996-08-02
> $ Title               <chr> "Net, The", "Dumb & Dumber", "Outbreak"
> $ year                <date> 1995-06-15, 1994-06-15, 1995-06-15
> $ genres              <chr> "Action|Crime|Thriller", "Comedy", "Action|Dram...
> $ Drama1h             <dbl> 0, 0, 1
> $ Comedy1h            <dbl> 0, 1, 0
> $ Thriller1h          <dbl> 1, 0, 1
> $ Romance1h           <dbl> 0, 0, 0
> $ Action1h            <dbl> 1, 0, 1
> $ Crime1h             <dbl> 1, 0, 0
> $ Adventure1h         <dbl> 0, 0, 0
```

```
> $ Horror1h             <dbl> 0, 0, 0
> $ SciFi1h              <dbl> 0, 0, 1
> $ Fantasy1h            <dbl> 0, 0, 0
> $ movieAGE             <dbl> 1.134, 2.134, 1.134
> $ agegroup             <chr> "3Y", "3Y", "3Y"
> $ movie_avgrating      <dbl> 3.123, 2.936, 3.415
> $ movie_countofrating <int> 12893, 15329, 13753
> $ movie_countbucket    <chr> "b", "b", "b"
> $ user_avgrating       <dbl> 5, 5, 5
> $ usermovcount         <int> 18, 18, 18
> $ user_countbucket     <chr> "j", "j", "j"
```

Genre plays a big role as well. Below I show a glm fit on the one-hot encoding with echo=TRUE.

```
# fitting Genre based on one-hot
gfit <- train %>% select(rating, 8:17) %>% glm(formula = rating ~ ., family = "gaussian")
genre_coefficients <- gfit$coefficients
#
train$genrefit <- genre_coefficients[1] + (train[, 8] * genre_coefficients[2]) +
    (train[, 9] * genre_coefficients[3]) + (train[, 10] * genre_coefficients[4]) +
    (train[, 11] * genre_coefficients[5]) + (train[, 12] * genre_coefficients[6]) +
    (train[, 13] * genre_coefficients[7]) + (train[, 14] * genre_coefficients[8]) +
    (train[, 15] * genre_coefficients[9]) + (train[, 16] * genre_coefficients[10]) +
    (train[, 17] * genre_coefficients[11])
#
```

Here is the training set with the genre fit:

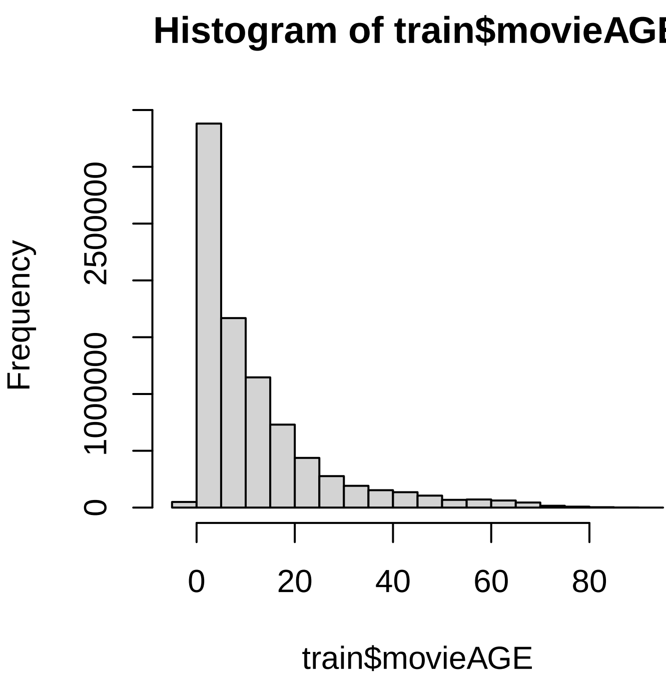
```
glimpse(train[1:3, ])
```

```
> Rows: 3
> Columns: 26
> $ userId               <int> 1, 1, 1
> $ movieId              <dbl> 185, 231, 292
> $ rating               <dbl> 5, 5, 5
> $ timestamp            <date> 1996-08-02, 1996-08-02, 1996-08-02
> $ Title                <chr> "Net, The", "Dumb & Dumber", "Outbreak"
> $ year                 <date> 1995-06-15, 1994-06-15, 1995-06-15
> $ genres               <chr> "Action|Crime|Thriller", "Comedy", "Action|Dram...
> $ Drama1h              <dbl> 0, 0, 1
> $ Comedy1h             <dbl> 0, 1, 0
> $ Thriller1h           <dbl> 1, 0, 1
> $ Romance1h            <dbl> 0, 0, 0
> $ Action1h             <dbl> 1, 0, 1
> $ Crime1h              <dbl> 1, 0, 0
> $ Adventure1h          <dbl> 0, 0, 0
> $ Horror1h             <dbl> 0, 0, 0
> $ SciFi1h              <dbl> 0, 0, 1
> $ Fantasy1h            <dbl> 0, 0, 0
> $ movieAGE             <dbl> 1.134, 2.134, 1.134
> $ agegroup             <chr> "3Y", "3Y", "3Y"
> $ movie_avgrating      <dbl> 3.123, 2.936, 3.415
> $ movie_countofrating <int> 12893, 15329, 13753
> $ movie_countbucket    <chr> "b", "b", "b"
> $ user_avgrating       <dbl> 5, 5, 5
> $ usermovcount         <int> 18, 18, 18
```

```
> $ user_countbucket    <chr> "j", "j", "j"
> $ genrefit            <dbl> 3.510, 3.349, 3.510
```

```
rm(gfit)
#
```

The age of a movie definitely plays a part as well. Below I take average rating by year the movie came out, and then average rating by the age group bucket I created earlier.




```
> # A tibble: 10 x 2
>    year       avg_byyear
>    <date>          <dbl>
>  1 1915-06-15       3.25
>  2 1916-06-15       3.69
>  3 1917-06-15       3.79
>  4 1918-06-15       3.65
>  5 1919-06-15       3.33
>  6 1920-06-15       3.95
>  7 1921-06-15       3.87
>  8 1922-06-15       3.91
>  9 1923-06-15       3.75
> 10 1924-06-15       3.93
```

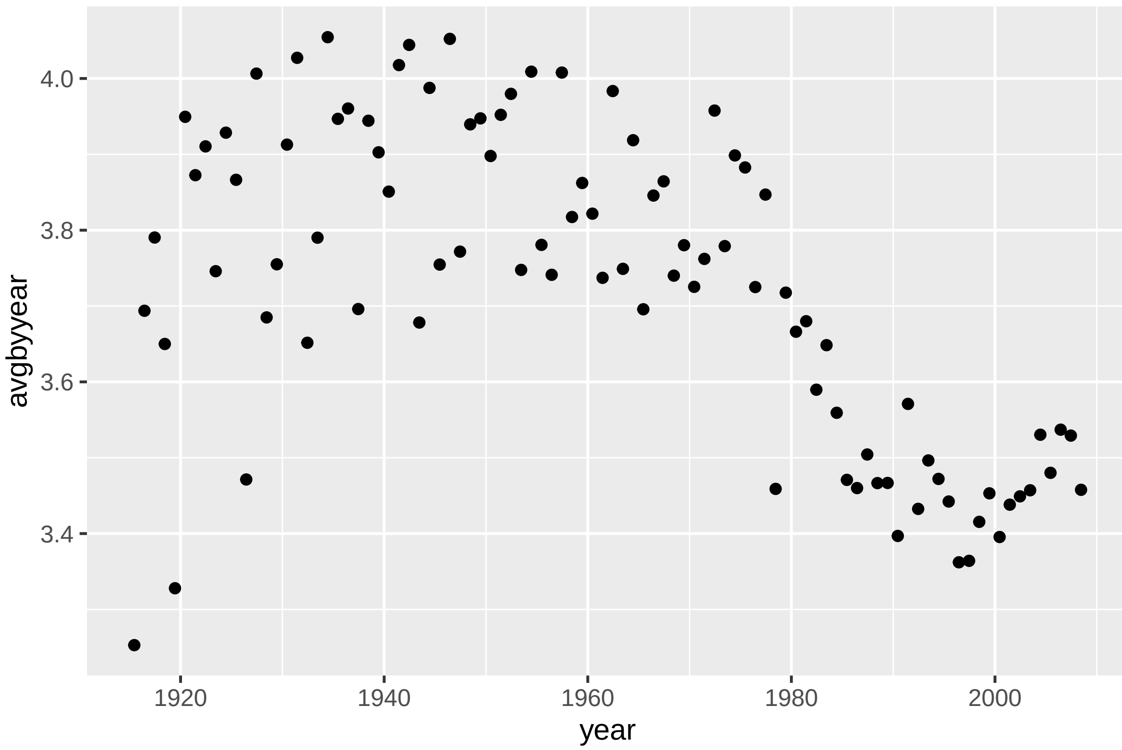
```
> # A tibble: 9 x 2
>   agegroup avg_byagegroup
>   <chr>             <dbl>
> 1 10Y                3.43
> 2 15Y                3.41
> 3 1Y                 3.46
> 4 20Y                3.51
> 5 30Y                3.67
> 6 3Y                 3.48
> 7 50plus             3.90
> 8 50Y                3.83
```

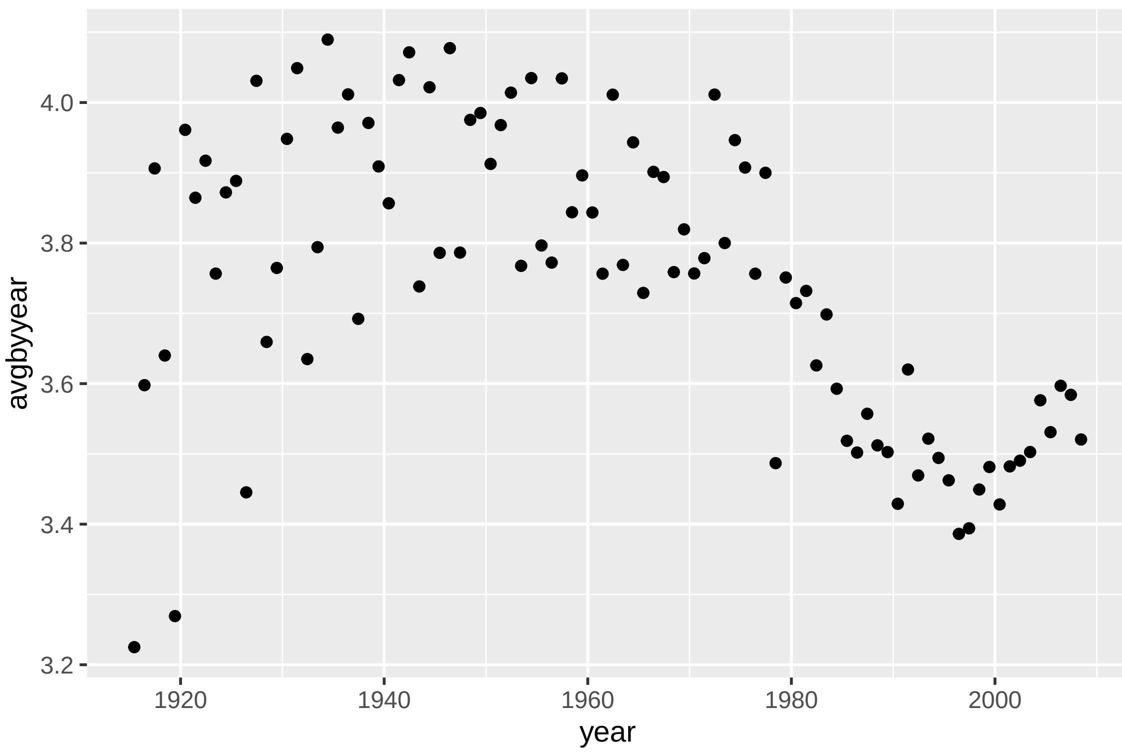
```
> 9 5Y                3.46
```

Here is the training set up to now:

```
> Rows: 3
> Columns: 28
> $ userId               <int> 1, 1, 1
> $ movieId              <dbl> 185, 231, 292
> $ rating               <dbl> 5, 5, 5
> $ timestamp            <date> 1996-08-02, 1996-08-02, 1996-08-02
> $ Title                <chr> "Net, The", "Dumb & Dumber", "Outbreak"
> $ year                 <date> 1995-06-15, 1994-06-15, 1995-06-15
> $ genres               <chr> "Action|Crime|Thriller", "Comedy", "Action|Dram...
> $ Drama1h              <dbl> 0, 0, 1
> $ Comedy1h             <dbl> 0, 1, 0
> $ Thriller1h           <dbl> 1, 0, 1
> $ Romance1h            <dbl> 0, 0, 0
> $ Action1h             <dbl> 1, 0, 1
> $ Crime1h              <dbl> 1, 0, 0
> $ Adventure1h          <dbl> 0, 0, 0
> $ Horror1h             <dbl> 0, 0, 0
> $ SciFi1h              <dbl> 0, 0, 1
> $ Fantasy1h            <dbl> 0, 0, 0
> $ movieAGE             <dbl> 1.134, 2.134, 1.134
> $ agegroup             <chr> "3Y", "3Y", "3Y"
> $ movie_avgrating      <dbl> 3.123, 2.936, 3.415
> $ movie_countofrating <int> 12893, 15329, 13753
> $ movie_countbucket    <chr> "b", "b", "b"
> $ user_avgrating       <dbl> 5, 5, 5
> $ usermovcount         <int> 18, 18, 18
> $ user_countbucket     <chr> "j", "j", "j"
> $ genrefit             <dbl> 3.510, 3.349, 3.510
> $ avg_byagegroup       <dbl> 3.478, 3.478, 3.478
> $ avg_byyear           <dbl> 3.442, 3.472, 3.442
```

In the next two tables, you can see that older movies (in general) have a better average rating.





Below, you can see that movies that get reviewed more often are better… bucket A has most reviews, bucket J has the least… conversely, in the second table you can see that users that review more movies rate movies lower. I have two theories: 1) they are more critical and review with more thoughtfulness 2) they are more likely to run into a bad movie if they watch a ton of movies.

```
> # A tibble: 10 x 2
>    movie_countbucket avg_bymovbucket
```

```
>    <chr>                        <dbl>
>  1 a                             3.88
>  2 b                             3.69
>  3 c                             3.67
>  4 d                             3.56
>  5 e                             3.56
>  6 f                             3.51
>  7 g                             3.41
>  8 h                             3.38
>  9 i                             3.26
> 10 j                             3.20
```

```
> # A tibble: 10 x 2
>    user_countbucket avg_byuserbucket
>    <chr>                        <dbl>
>  1 a                             3.24
>  2 b                             3.33
>  3 c                             3.42
>  4 d                             3.47
>  5 e                             3.53
>  6 f                             3.58
>  7 g                             3.61
>  8 h                             3.64
>  9 i                             3.65
> 10 j                             3.65
```

Below is a glimpse of the training set, the test set, and the validation set through now…

```
> Rows: 3
> Columns: 30
> $ userId              <int> 1, 1, 1
> $ movieId             <dbl> 185, 231, 292
> $ rating              <dbl> 5, 5, 5
> $ timestamp           <date> 1996-08-02, 1996-08-02, 1996-08-02
> $ Title               <chr> "Net, The", "Dumb & Dumber", "Outbreak"
> $ year                <date> 1995-06-15, 1994-06-15, 1995-06-15
> $ genres              <chr> "Action|Crime|Thriller", "Comedy", "Action|Dram...
> $ Drama1h             <dbl> 0, 0, 1
> $ Comedy1h            <dbl> 0, 1, 0
> $ Thriller1h          <dbl> 1, 0, 1
> $ Romance1h           <dbl> 0, 0, 0
> $ Action1h            <dbl> 1, 0, 1
> $ Crime1h             <dbl> 1, 0, 0
> $ Adventure1h         <dbl> 0, 0, 0
> $ Horror1h            <dbl> 0, 0, 0
> $ SciFi1h             <dbl> 0, 0, 1
> $ Fantasy1h           <dbl> 0, 0, 0
> $ movieAGE            <dbl> 1.134, 2.134, 1.134
> $ agegroup            <chr> "3Y", "3Y", "3Y"
> $ movie_avgrating     <dbl> 3.123, 2.936, 3.415
> $ movie_countofrating <int> 12893, 15329, 13753
> $ movie_countbucket   <chr> "b", "b", "b"
> $ user_avgrating      <dbl> 5, 5, 5
> $ usermovcount        <int> 18, 18, 18
> $ user_countbucket    <chr> "j", "j", "j"
```

```
> $ genrefit            <dbl> 3.510, 3.349, 3.510
> $ avg_byagegroup      <dbl> 3.478, 3.478, 3.478
> $ avg_byyear          <dbl> 3.442, 3.472, 3.442
> $ avg_bymovbucket     <dbl> 3.695, 3.695, 3.695
> $ avg_byuserbucket    <dbl> 3.648, 3.648, 3.648
```

```
> Rows: 3
> Columns: 19
> $ userId      <int> 1, 2, 3
> $ movieId     <dbl> 316, 648, 590
> $ rating      <dbl> 5.0, 2.0, 3.5
> $ timestamp   <date> 1996-08-02, 1997-07-07, 2006-01-01
> $ Title       <chr> "Stargate", "Mission: Impossible", "Dances with Wolves"
> $ year        <date> 1994-06-15, 1996-06-15, 1990-06-15
> $ genres      <chr> "Action|Adventure|Sci-Fi", "Action|Adventure|Mystery|Th...
> $ Drama1h     <dbl> 0, 0, 1
> $ Comedy1h    <dbl> 0, 0, 0
> $ Thriller1h  <dbl> 0, 1, 0
> $ Romance1h   <dbl> 0, 0, 0
> $ Action1h    <dbl> 1, 1, 0
> $ Crime1h     <dbl> 0, 0, 0
> $ Adventure1h <dbl> 1, 1, 1
> $ Horror1h    <dbl> 0, 0, 0
> $ SciFi1h     <dbl> 1, 0, 0
> $ Fantasy1h   <dbl> 0, 0, 0
> $ movieAGE    <dbl> 2.134, 1.060, 15.559
> $ agegroup    <chr> "3Y", "3Y", "20Y"
```

```
> Rows: 3
> Columns: 19
> $ userId      <int> 1, 1, 1
> $ movieId     <dbl> 122, 362, 586
> $ rating      <dbl> 5, 5, 5
> $ timestamp   <date> 1996-08-02, 1996-08-02, 1996-08-02
> $ Title       <chr> "Boomerang", "Jungle Book, The", "Home Alone"
> $ year        <date> 1992-06-15, 1994-06-15, 1990-06-15
> $ genres      <chr> "Comedy|Romance", "Adventure|Children|Romance", "Childr...
> $ Drama1h     <dbl> 0, 0, 0
> $ Comedy1h    <dbl> 1, 0, 1
> $ Thriller1h  <dbl> 0, 0, 0
> $ Romance1h   <dbl> 1, 1, 0
> $ Action1h    <dbl> 0, 0, 0
> $ Crime1h     <dbl> 0, 0, 0
> $ Adventure1h <dbl> 0, 1, 0
> $ Horror1h    <dbl> 0, 0, 0
> $ SciFi1h     <dbl> 0, 0, 0
> $ Fantasy1h   <dbl> 0, 0, 0
> $ movieAGE    <dbl> 4.134, 2.134, 6.137
> $ agegroup    <chr> "5Y", "3Y", "10Y"
```

```
> [1] 9.238
```

Part 3: Modeling / Results

Note that this section will include substantially more on screen R-code, as I readers to follow as "close to the code" as possible. First I fit the training set to the model I have built.

```
# y is rating, x's are 1movie_avgrating, 2user_avgrating, 3genrefit, 4avg by
# year, 5avg_byage, 6avg_bymovbucket, 7avg_byuserbucket
lookup <- train %>% select(rating, movie_avgrating, user_avgrating, genrefit, avg_byyear,
    avg_byagegroup, avg_bymovbucket, avg_byuserbucket)
fit <- lookup %>% glm(formula = rating ~ ., family = "gaussian")
fit_coefficients <- fit$coefficients
fit_coefficients
```

```
>      (Intercept)  movie_avgrating   user_avgrating         genrefit
>         -1.30221          0.91067          0.87520         -0.01679
>       avg_byyear   avg_byagegroup  avg_bymovbucket avg_byuserbucket
>         -0.09543          0.04816         -0.06404         -0.28702
```

```
#
train$yhat <- fit_coefficients[1] + (fit_coefficients[2] * train$movie_avgrating) +
    (fit_coefficients[3] * train$user_avgrating) + (fit_coefficients[4] * train$genrefit) +
    (fit_coefficients[5] * train$avg_byyear) + (fit_coefficients[6] * train$avg_byagegroup) +
    (fit_coefficients[7] * train$avg_bymovbucket) + (fit_coefficients[8] * train$avg_byuserbucket)
#
```

Below you will see a glimpse of the train set in its final form, with the Yhat prediction. The RMSE on the training set is reported below.

```
#
dim(train)
```

```
> [1] 8550061      31
```

```
glimpse(train[1:3, ])
```

```
> Rows: 3
> Columns: 31
> $ userId              <int> 1, 1, 1
> $ movieId             <dbl> 185, 231, 292
> $ rating              <dbl> 5, 5, 5
> $ timestamp           <date> 1996-08-02, 1996-08-02, 1996-08-02
> $ Title               <chr> "Net, The", "Dumb & Dumber", "Outbreak"
> $ year                <date> 1995-06-15, 1994-06-15, 1995-06-15
> $ genres              <chr> "Action|Crime|Thriller", "Comedy", "Action|Dram...
> $ Drama1h             <dbl> 0, 0, 1
> $ Comedy1h            <dbl> 0, 1, 0
> $ Thriller1h          <dbl> 1, 0, 1
> $ Romance1h           <dbl> 0, 0, 0
> $ Action1h            <dbl> 1, 0, 1
> $ Crime1h             <dbl> 1, 0, 0
> $ Adventure1h         <dbl> 0, 0, 0
> $ Horror1h            <dbl> 0, 0, 0
> $ SciFi1h             <dbl> 0, 0, 1
> $ Fantasy1h           <dbl> 0, 0, 0
> $ movieAGE            <dbl> 1.134, 2.134, 1.134
> $ agegroup            <chr> "3Y", "3Y", "3Y"
> $ movie_avgrating     <dbl> 3.123, 2.936, 3.415
> $ movie_countofrating <int> 12893, 15329, 13753
```

```
> $ movie_countbucket   <chr> "b", "b", "b"
> $ user_avgrating      <dbl> 5, 5, 5
> $ usermovcount        <int> 18, 18, 18
> $ user_countbucket    <chr> "j", "j", "j"
> $ genrefit            <dbl> 3.510, 3.349, 3.510
> $ avg_byagegroup      <dbl> 3.478, 3.478, 3.478
> $ avg_byyear          <dbl> 3.442, 3.472, 3.442
> $ avg_bymovbucket     <dbl> 3.695, 3.695, 3.695
> $ avg_byuserbucket    <dbl> 3.648, 3.648, 3.648
> $ yhat                <dbl> 4.414, 4.243, 4.680
```

```
RMSE(train$rating, train$yhat)
```

```
> [1] 0.8708
```

Next I will fit the test set with the model I have generated. I'll need to join the new columns in, as the aggregations and averaging were only done on the training set (obviously!).

```
#
testing <- left_join(test, movielookup)
testing <- left_join(testing, userlookup)
testing <- left_join(testing, movie_count_bucket_df)
testing <- left_join(testing, user_count_bucket_df)
testing <- left_join(testing, avg_rating_by_year)
testing <- left_join(testing, avg_rating_byagegroup)
glimpse(testing[1:3, ])
```

```
> Rows: 3
> Columns: 29
> $ userId              <int> 1, 2, 3
> $ movieId             <dbl> 316, 648, 590
> $ rating              <dbl> 5.0, 2.0, 3.5
> $ timestamp           <date> 1996-08-02, 1997-07-07, 2006-01-01
> $ Title               <chr> "Stargate", "Mission: Impossible", "Dances with...
> $ year                <date> 1994-06-15, 1996-06-15, 1990-06-15
> $ genres              <chr> "Action|Adventure|Sci-Fi", "Action|Adventure|My...
> $ Drama1h             <dbl> 0, 0, 1
> $ Comedy1h            <dbl> 0, 0, 0
> $ Thriller1h          <dbl> 0, 1, 0
> $ Romance1h           <dbl> 0, 0, 0
> $ Action1h            <dbl> 1, 1, 0
> $ Crime1h             <dbl> 0, 0, 0
> $ Adventure1h         <dbl> 1, 1, 1
> $ Horror1h            <dbl> 0, 0, 0
> $ SciFi1h             <dbl> 1, 0, 0
> $ Fantasy1h           <dbl> 0, 0, 0
> $ movieAGE            <dbl> 2.134, 1.060, 15.559
> $ agegroup            <chr> "3Y", "3Y", "20Y"
> $ movie_avgrating     <dbl> 3.348, 3.385, 3.740
> $ movie_countofrating <int> 16109, 18047, 22188
> $ movie_countbucket   <chr> "b", "a", "a"
> $ user_avgrating      <dbl> 5.000, 3.278, 3.981
> $ usermovcount        <int> 18, 18, 27
> $ user_countbucket    <chr> "j", "j", "j"
> $ avg_bymovbucket     <dbl> 3.695, 3.877, 3.877
> $ avg_byuserbucket    <dbl> 3.648, 3.648, 3.648
```

```
> $ avg_byyear          <dbl> 3.472, 3.362, 3.397
> $ avg_byagegroup      <dbl> 3.478, 3.478, 3.507
```

```
testing$genrefit <- genre_coefficients[1] + (testing[, 8] * genre_coefficients[2]) +
    (testing[, 9] * genre_coefficients[3]) + (testing[, 10] * genre_coefficients[4]) +
    (testing[, 11] * genre_coefficients[5]) + (testing[, 12] * genre_coefficients[6]) +
    (testing[, 13] * genre_coefficients[7]) + (testing[, 14] * genre_coefficients[8]) +
    (testing[, 15] * genre_coefficients[9]) + (testing[, 16] * genre_coefficients[10]) +
    (testing[, 17] * genre_coefficients[11])
#
glimpse(testing[1:3, ])
```

```
> Rows: 3
> Columns: 30
> $ userId              <int> 1, 2, 3
> $ movieId             <dbl> 316, 648, 590
> $ rating              <dbl> 5.0, 2.0, 3.5
> $ timestamp           <date> 1996-08-02, 1997-07-07, 2006-01-01
> $ Title               <chr> "Stargate", "Mission: Impossible", "Dances with...
> $ year                <date> 1994-06-15, 1996-06-15, 1990-06-15
> $ genres              <chr> "Action|Adventure|Sci-Fi", "Action|Adventure|My...
> $ Drama1h             <dbl> 0, 0, 1
> $ Comedy1h            <dbl> 0, 0, 0
> $ Thriller1h          <dbl> 0, 1, 0
> $ Romance1h           <dbl> 0, 0, 0
> $ Action1h            <dbl> 1, 1, 0
> $ Crime1h             <dbl> 0, 0, 0
> $ Adventure1h         <dbl> 1, 1, 1
> $ Horror1h            <dbl> 0, 0, 0
> $ SciFi1h             <dbl> 1, 0, 0
> $ Fantasy1h           <dbl> 0, 0, 0
> $ movieAGE            <dbl> 2.134, 1.060, 15.559
> $ agegroup            <chr> "3Y", "3Y", "20Y"
> $ movie_avgrating     <dbl> 3.348, 3.385, 3.740
> $ movie_countofrating <int> 16109, 18047, 22188
> $ movie_countbucket   <chr> "b", "a", "a"
> $ user_avgrating      <dbl> 5.000, 3.278, 3.981
> $ usermovcount        <int> 18, 18, 27
> $ user_countbucket    <chr> "j", "j", "j"
> $ avg_bymovbucket     <dbl> 3.695, 3.877, 3.877
> $ avg_byuserbucket    <dbl> 3.648, 3.648, 3.648
> $ avg_byyear          <dbl> 3.472, 3.362, 3.397
> $ avg_byagegroup      <dbl> 3.478, 3.478, 3.507
> $ genrefit            <dbl> 3.391, 3.431, 3.782
```

```
#
testing$yhat <- fit_coefficients[1] + (fit_coefficients[2] * testing$movie_avgrating) +
    (fit_coefficients[3] * testing$user_avgrating) + (fit_coefficients[4] * testing$genrefit) +
    (fit_coefficients[5] * testing$avg_byyear) + (fit_coefficients[6] * testing$avg_byagegroup) +
    (fit_coefficients[7] * testing$avg_bymovbucket) + (fit_coefficients[8] * testing$avg_byuserbucket)
#
```

Below is a glimpse at the test set in its final form, with yhat predictions. The RMSE on the test set is reported below.

```
glimpse(testing[1:3, ])
```

```
> Rows: 3
> Columns: 31
> $ userId                <int> 1, 2, 3
> $ movieId               <dbl> 316, 648, 590
> $ rating                <dbl> 5.0, 2.0, 3.5
> $ timestamp             <date> 1996-08-02, 1997-07-07, 2006-01-01
> $ Title                 <chr> "Stargate", "Mission: Impossible", "Dances with...
> $ year                  <date> 1994-06-15, 1996-06-15, 1990-06-15
> $ genres                <chr> "Action|Adventure|Sci-Fi", "Action|Adventure|My...
> $ Drama1h               <dbl> 0, 0, 1
> $ Comedy1h              <dbl> 0, 0, 0
> $ Thriller1h            <dbl> 0, 1, 0
> $ Romance1h             <dbl> 0, 0, 0
> $ Action1h              <dbl> 1, 1, 0
> $ Crime1h               <dbl> 0, 0, 0
> $ Adventure1h           <dbl> 1, 1, 1
> $ Horror1h              <dbl> 0, 0, 0
> $ SciFi1h               <dbl> 1, 0, 0
> $ Fantasy1h             <dbl> 0, 0, 0
> $ movieAGE              <dbl> 2.134, 1.060, 15.559
> $ agegroup              <chr> "3Y", "3Y", "20Y"
> $ movie_avgrating       <dbl> 3.348, 3.385, 3.740
> $ movie_countofrating <int> 16109, 18047, 22188
> $ movie_countbucket     <chr> "b", "a", "a"
> $ user_avgrating        <dbl> 5.000, 3.278, 3.981
> $ usermovcount          <int> 18, 18, 27
> $ user_countbucket      <chr> "j", "j", "j"
> $ avg_bymovbucket       <dbl> 3.695, 3.877, 3.877
> $ avg_byuserbucket      <dbl> 3.648, 3.648, 3.648
> $ avg_byyear            <dbl> 3.472, 3.362, 3.397
> $ avg_byagegroup        <dbl> 3.478, 3.478, 3.507
> $ genrefit              <dbl> 3.391, 3.431, 3.782
> $ yhat                  <dbl> 4.619, 3.143, 4.074
```

```
RMSE(testing$rating, testing$yhat)
```

```
> [1] 0.8783
```

```
#
```

Finally, I will fit the validation set with the model I have generated. I'll need to join the new columns in just like the test set, as the aggregations and averaging were only done on the training set (obviously!! that is the most important part!).

```
#
testing <- left_join(validation, movielookup)
testing <- left_join(testing, userlookup)
testing <- left_join(testing, movie_count_bucket_df)
testing <- left_join(testing, user_count_bucket_df)
testing <- left_join(testing, avg_rating_by_year)
testing <- left_join(testing, avg_rating_byagegroup)
glimpse(testing[1:3, ])
```

```
> Rows: 3
> Columns: 29
> $ userId                <int> 1, 1, 1
> $ movieId               <dbl> 122, 362, 586
```

```
> $ rating                <dbl> 5, 5, 5
> $ timestamp             <date> 1996-08-02, 1996-08-02, 1996-08-02
> $ Title                 <chr> "Boomerang", "Jungle Book, The", "Home Alone"
> $ year                  <date> 1992-06-15, 1994-06-15, 1990-06-15
> $ genres                <chr> "Comedy|Romance", "Adventure|Children|Romance",...
> $ Drama1h               <dbl> 0, 0, 0
> $ Comedy1h              <dbl> 1, 0, 1
> $ Thriller1h            <dbl> 0, 0, 0
> $ Romance1h             <dbl> 1, 1, 0
> $ Action1h              <dbl> 0, 0, 0
> $ Crime1h               <dbl> 0, 0, 0
> $ Adventure1h           <dbl> 0, 1, 0
> $ Horror1h              <dbl> 0, 0, 0
> $ SciFi1h               <dbl> 0, 0, 0
> $ Fantasy1h             <dbl> 0, 0, 0
> $ movieAGE              <dbl> 4.134, 2.134, 6.137
> $ agegroup              <chr> "5Y", "3Y", "10Y"
> $ movie_avgrating       <dbl> 2.866, 3.448, 3.055
> $ movie_countofrating <int> 2061, 3423, 13089
> $ movie_countbucket     <chr> "g", "f", "b"
> $ user_avgrating        <dbl> 5, 5, 5
> $ usermovcount          <int> 18, 18, 18
> $ user_countbucket      <chr> "j", "j", "j"
> $ avg_bymovbucket       <dbl> 3.413, 3.512, 3.695
> $ avg_byuserbucket      <dbl> 3.648, 3.648, 3.648
> $ avg_byyear            <dbl> 3.433, 3.472, 3.397
> $ avg_byagegroup        <dbl> 3.464, 3.478, 3.433
```

```
testing$genrefit <- genre_coefficients[1] + (testing[, 8] * genre_coefficients[2]) +
    (testing[, 9] * genre_coefficients[3]) + (testing[, 10] * genre_coefficients[4]) +
    (testing[, 11] * genre_coefficients[5]) + (testing[, 12] * genre_coefficients[6]) +
    (testing[, 13] * genre_coefficients[7]) + (testing[, 14] * genre_coefficients[8]) +
    (testing[, 15] * genre_coefficients[9]) + (testing[, 16] * genre_coefficients[10]) +
    (testing[, 17] * genre_coefficients[11])
#
glimpse(testing[1:3, ])
```

```
> Rows: 3
> Columns: 30
> $ userId                <int> 1, 1, 1
> $ movieId               <dbl> 122, 362, 586
> $ rating                <dbl> 5, 5, 5
> $ timestamp             <date> 1996-08-02, 1996-08-02, 1996-08-02
> $ Title                 <chr> "Boomerang", "Jungle Book, The", "Home Alone"
> $ year                  <date> 1992-06-15, 1994-06-15, 1990-06-15
> $ genres                <chr> "Comedy|Romance", "Adventure|Children|Romance",...
> $ Drama1h               <dbl> 0, 0, 0
> $ Comedy1h              <dbl> 1, 0, 1
> $ Thriller1h            <dbl> 0, 0, 0
> $ Romance1h             <dbl> 1, 1, 0
> $ Action1h              <dbl> 0, 0, 0
> $ Crime1h               <dbl> 0, 0, 0
> $ Adventure1h           <dbl> 0, 1, 0
> $ Horror1h              <dbl> 0, 0, 0
> $ SciFi1h               <dbl> 0, 0, 0
```

```
> $ Fantasy1h           <dbl> 0, 0, 0
> $ movieAGE            <dbl> 4.134, 2.134, 6.137
> $ agegroup            <chr> "5Y", "3Y", "10Y"
> $ movie_avgrating     <dbl> 2.866, 3.448, 3.055
> $ movie_countofrating <int> 2061, 3423, 13089
> $ movie_countbucket   <chr> "g", "f", "b"
> $ user_avgrating      <dbl> 5, 5, 5
> $ usermovcount        <int> 18, 18, 18
> $ user_countbucket    <chr> "j", "j", "j"
> $ avg_bymovbucket     <dbl> 3.413, 3.512, 3.695
> $ avg_byuserbucket    <dbl> 3.648, 3.648, 3.648
> $ avg_byyear          <dbl> 3.433, 3.472, 3.397
> $ avg_byagegroup      <dbl> 3.464, 3.478, 3.433
> $ genrefit            <dbl> 3.388, 3.602, 3.349
```

```r
#
testing$yhat <- fit_coefficients[1] + (fit_coefficients[2] * testing$movie_avgrating) +
    (fit_coefficients[3] * testing$user_avgrating) + (fit_coefficients[4] * testing$genrefit) +
    (fit_coefficients[5] * testing$avg_byyear) + (fit_coefficients[6] * testing$avg_byagegroup) +
    (fit_coefficients[7] * testing$avg_bymovbucket) + (fit_coefficients[8] * testing$avg_byuserbucket)
#
```

Below is a glimpse at the validation set in its final form, with yhat predictions. The RMSE on the validation set is reported below.

```r
glimpse(testing[1:3, ])
```

```
> Rows: 3
> Columns: 31
> $ userId              <int> 1, 1, 1
> $ movieId             <dbl> 122, 362, 586
> $ rating              <dbl> 5, 5, 5
> $ timestamp           <date> 1996-08-02, 1996-08-02, 1996-08-02
> $ Title               <chr> "Boomerang", "Jungle Book, The", "Home Alone"
> $ year                <date> 1992-06-15, 1994-06-15, 1990-06-15
> $ genres              <chr> "Comedy|Romance", "Adventure|Children|Romance",...
> $ Drama1h             <dbl> 0, 0, 0
> $ Comedy1h            <dbl> 1, 0, 1
> $ Thriller1h          <dbl> 0, 0, 0
> $ Romance1h           <dbl> 1, 1, 0
> $ Action1h            <dbl> 0, 0, 0
> $ Crime1h             <dbl> 0, 0, 0
> $ Adventure1h         <dbl> 0, 1, 0
> $ Horror1h            <dbl> 0, 0, 0
> $ SciFi1h             <dbl> 0, 0, 0
> $ Fantasy1h           <dbl> 0, 0, 0
> $ movieAGE            <dbl> 4.134, 2.134, 6.137
> $ agegroup            <chr> "5Y", "3Y", "10Y"
> $ movie_avgrating     <dbl> 2.866, 3.448, 3.055
> $ movie_countofrating <int> 2061, 3423, 13089
> $ movie_countbucket   <chr> "g", "f", "b"
> $ user_avgrating      <dbl> 5, 5, 5
> $ usermovcount        <int> 18, 18, 18
> $ user_countbucket    <chr> "j", "j", "j"
> $ avg_bymovbucket     <dbl> 3.413, 3.512, 3.695
> $ avg_byuserbucket    <dbl> 3.648, 3.648, 3.648
```

```
> $ avg_byyear          <dbl> 3.433, 3.472, 3.397
> $ avg_byagegroup      <dbl> 3.464, 3.478, 3.433
> $ genrefit            <dbl> 3.388, 3.602, 3.349
> $ yhat                <dbl> 4.201, 4.718, 4.357
```

```
RMSE(testing$rating, testing$yhat)
```

```
> [1] 0.8772
```

```
#
```

Part 4: Conclusion / Remarks

The RMSE that I was able to achieve did not meet my expectations. When I started this project, I had quite a few ideas about how to tackle it…but one thing I can say with certainty is that I underestimated the difficult task of making 1 million predictions! It's not only hard to do this accurately, but it takes time to run through 1 million rows too.

Overall, this exercise was extremely fun and reinforced so many of the skills I learned in this certificate program. One consideration I want to bring up is that I would want to see how the modeling from this research would progress if I had access to significantly more computing power. While a 16gb RAM/i5 machine is satisfactory, there are computers out there that may be able to use certain clustering algorithms to create user profiles, and then model out predictions based on those profiles with a linear model or regression tree. As we covered in the Machine Learning course, this dataset is far too large to fit a true model with a standard laptop (which makes me jealous).

Thank you for following along, and I welcome any feedback that you have.

Craig Barnes | craig.michael.barnes@gmail.com