DPA Project

Group Project contributed by all

2023-04-22

```
#Install the pacman package
if(!require(pacman)) install.packages("pacman", repos = "http://cran.us.r-
project.org")

## Loading required package: pacman

## Warning: package 'pacman' was built under R version 4.2.3

#Load the required libraries
#If a package below is missing, p_load will automatically download it from
CRAN
pacman::p_load(tidyverse, ggplot2, ggthemes, data.table, lubridate, caret,
               knitr, scales, treemapify)
pacman::p_load(stringr)
pacman::p_load(dplyr)
```

Data Preparation

```
#Download File
dl <- tempfile()
download.file("http://files.grouplens.org/datasets/movielens/ml-10m.zip", dl)

#Construct a data frame called 'ratings' by utilizing the 'fread' function
from the data.table library
ratings <- fread(text = gsub("::", "\t", readLines(unzip(dl, "ml-
10M100K/ratings.dat"))),
                 col.names = c("userId", "movieId", "rating", "timestamp"))

movies <- str_split_fixed(readLines(unzip(dl, "ml-10M100K/movies.dat")),
"\\:\\:", 3)

colnames(movies) <- c("movieId", "title", "genres")

movies <- as.data.frame(movies) %>%
  mutate(movieId = as.numeric(unique(movieId)),
         title = as.character(title),
         genres = as.character(genres))
movielens <- left_join(ratings, movies, by = "movieId")

#Designate the validation set as 10% of the MovieLens data
set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
sampler
## used
```

```
test_index <-createDataPartition(y = movielens$rating, times = 1, p = 0.1,
list = FALSE)
edx <-movielens[-test_index,]
temp <-movielens[test_index,]
#Check if userId and movieId in validation set are also in edx set
validation <- temp %>%
  semi_join(edx, by = "movieId") %>%
  semi_join(edx, by = "userId")

#Merge the rows that were removed from the validation set back into the edx
set
removed <-anti_join(temp, validation)
```

```
## Joining with `by = join_by(userId, movieId, rating, timestamp, title,
genres)`
```

```
edx <-rbind(edx, removed)
rm(dl, ratings, movies, test_index, temp, movielens, removed)

#divide Training and Test Sets:
set.seed(1, sample.kind = "Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
sampler
## used
```

```
test_index <-createDataPartition(y = edx$rating, times = 1, p = 0.1, list =
F)
edx_train <-edx[-test_index,]
edx_temp <-edx[test_index,]
#Make sure userId and movieId are in the train and test sets
edx_test <-edx_temp %>%
  semi_join(edx_train, by = "movieId") %>%
  semi_join(edx_train, by = "userId")
removed <-anti_join(edx_temp, edx_test)
```

```
## Joining with `by = join_by(userId, movieId, rating, timestamp, title,
genres)`
```

```
edx_train <-rbind(edx_train, removed)
rm(edx_temp, test_index, removed)
```

Analyzing the data

```
edx %>% as_tibble()
```

```
## # A tibble: 9,000,055 × 6
##    userId movieId rating timestamp title
```

```
genres
##     <int>   <dbl>  <dbl>     <int> <chr>
<chr>
##  1      1     122      5 838985046 Boomerang (1992)
Comed…
##  2      1     185      5 838983525 Net, The (1995)
Actio…
##  3      1     292      5 838983421 Outbreak (1995)
Actio…
##  4      1     316      5 838983392 Stargate (1994)
Actio…
##  5      1     329      5 838983392 Star Trek: Generations (1994)
Actio…
##  6      1     355      5 838984474 Flintstones, The (1994)
Child…
##  7      1     356      5 838983653 Forrest Gump (1994)
Comed…
##  8      1     362      5 838984885 Jungle Book, The (1994)
Adven…
##  9      1     364      5 838983707 Lion King, The (1994)
Adven…
## 10      1     370      5 838984596 Naked Gun 33 1/3: The Final Insult (1…
Actio…
## # … with 9,000,045 more rows

#Confirm the dimensions and explore the features and classes of edx.
glimpse(edx)

## Rows: 9,000,055
## Columns: 6
## $ userId    <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
2, …
## $ movieId   <dbl> 122, 185, 292, 316, 329, 355, 356, 362, 364, 370, 377,
420, …
## $ rating    <dbl> 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, …
## $ timestamp <int> 838985046, 838983525, 838983421, 838983392, 838983392,
83898…
## $ title     <chr> "Boomerang (1992)", "Net, The (1995)", "Outbreak
(1995)", "S…
## $ genres    <chr> "Comedy|Romance", "Action|Crime|Thriller",
"Action|Drama|Sci…

#Determine the unique number of userIds, movieIds, and genres

edx %>% summarize(unique_users = length(unique(userId)),
                  unique_movies = length(unique(movieId)),
                  unique_genres = length(unique(genres)))

##   unique_users unique_movies unique_genres
## 1        69878         10677           797
```

```
#Ratings
length(unique(edx$rating))
```

```
## [1] 10
```

```
unique_ratings <-unique(edx$rating)
sort(unique_ratings)
```

```
##  [1] 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
```

```
#View a Tibble of the Ratings Distribution
edx %>% group_by(rating) %>% summarize(ratings_sum = n()) %>%
  arrange(desc(ratings_sum))
```

```
## # A tibble: 10 × 2
##    rating ratings_sum
##     <dbl>       <int>
##  1    4       2588430
##  2    3       2121240
##  3    5       1390114
##  4    3.5      791624
##  5    2        711422
##  6    4.5      526736
##  7    1        345679
##  8    2.5      333010
##  9    1.5      106426
## 10    0.5       85374
```

```
rp <-edx %>% filter(edx$rating >=3)
nrow(rp)/length(edx$rating)
```

```
## [1] 0.8242332
```

TIMESTAMP

```
#Transform the timestamp column of the edx dataset to a 'RatingYear' format
edx <- edx %>% mutate(timestamp = as.POSIXct(timestamp, origin = "1970-01-
01",
                                             tz = "EST"))
edx$timestamp <- format(edx$timestamp, "%Y")
names(edx)[names(edx) == "timestamp"] <- "RatingYear"
head(edx)
```

```
##    userId movieId rating RatingYear                         title
## 1:      1     122      5       1996              Boomerang (1992)
## 2:      1     185      5       1996               Net, The (1995)
## 3:      1     292      5       1996               Outbreak (1995)
## 4:      1     316      5       1996               Stargate (1994)
## 5:      1     329      5       1996 Star Trek: Generations (1994)
## 6:      1     355      5       1996       Flintstones, The (1994)
##                            genres
## 1:                 Comedy|Romance
## 2:         Action|Crime|Thriller
## 3:  Action|Drama|Sci-Fi|Thriller
## 4:       Action|Adventure|Sci-Fi
## 5: Action|Adventure|Drama|Sci-Fi
## 6:       Children|Comedy|Fantasy
```

```
validation <- validation %>% mutate(timestamp = as.POSIXct(timestamp, origin
= "1970-01-01",
                                                           tz = "EST"))
validation$timestamp <- format(validation$timestamp, "%Y")
names(validation)[names(validation) == "timestamp"] <- "RatingYear"
head(validation)
```

```
##    userId movieId rating RatingYear
## 1:      1     231      5       1996
## 2:      1     480      5       1996
## 3:      1     586      5       1996
## 4:      2     151      3       1997
## 5:      2     858      2       1997
## 6:      2    1544      3       1997
##                                                      title
## 1:                                    Dumb & Dumber (1994)
## 2:                                    Jurassic Park (1993)
## 3:                                       Home Alone (1990)
## 4:                                          Rob Roy (1995)
## 5:                                   Godfather, The (1972)
## 6: Lost World: Jurassic Park, The (Jurassic Park 2) (1997)
##                                     genres
## 1:                                  Comedy
## 2:        Action|Adventure|Sci-Fi|Thriller
## 3:                         Children|Comedy
## 4:                Action|Drama|Romance|War
## 5:                             Crime|Drama
## 6: Action|Adventure|Horror|Sci-Fi|Thriller
```

```
edx_train <- edx_train %>% mutate(timestamp = as.POSIXct(timestamp, origin =
"1970-01-01",
                                                         tz = "EST"))
edx_train$timestamp <- format(edx_train$timestamp, "%Y")
names(edx_train)[names(edx_train) == "timestamp"] <- "RatingYear"
head(edx_train)
```

```
##    userId movieId rating RatingYear                         title
## 1:      1     122      5       1996              Boomerang (1992)
## 2:      1     292      5       1996               Outbreak (1995)
## 3:      1     316      5       1996               Stargate (1994)
## 4:      1     329      5       1996 Star Trek: Generations (1994)
## 5:      1     355      5       1996       Flintstones, The (1994)
## 6:      1     356      5       1996           Forrest Gump (1994)
##                            genres
## 1:                 Comedy|Romance
## 2:  Action|Drama|Sci-Fi|Thriller
## 3:       Action|Adventure|Sci-Fi
## 4: Action|Adventure|Drama|Sci-Fi
## 5:       Children|Comedy|Fantasy
## 6:      Comedy|Drama|Romance|War
```

```
edx_test <-edx_test %>% mutate(timestamp = as.POSIXct(timestamp, origin =
"1970-01-01",
                                                      tz = "EST"))
edx_test$timestamp <- format(edx_test$timestamp, "%Y")
names(edx_test)[names(edx_test) == "timestamp"] <- "RatingYear"
head(edx_test)
```

```
##    userId movieId rating RatingYear
## 1:      1     185      5       1996
## 2:      2     260      5       1997
## 3:      2     590      5       1997
## 4:      2    1049      3       1997
## 5:      2    1210      4       1997
## 6:      3    1148      4       2005
##                                                            title
## 1:                                               Net, The (1995)
## 2: Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)
## 3:                                     Dances with Wolves (1990)
## 4:                            Ghost and the Darkness, The (1996)
## 5:             Star Wars: Episode VI - Return of the Jedi (1983)
## 6:                   Wallace & Gromit: The Wrong Trousers (1993)
##                              genres
## 1:            Action|Crime|Thriller
## 2:          Action|Adventure|Sci-Fi
## 3:          Adventure|Drama|Western
## 4:                 Action|Adventure
## 5:          Action|Adventure|Sci-Fi
## 6: Animation|Children|Comedy|Crime
```

```
range(edx$RatingYear)
```

```
## [1] "1995" "2009"
```

```
#Convert the 'RatingYear' column from character to numeric data type in order
to plot a histogram
edx$RatingYear <-as.numeric(edx$RatingYear)
str(edx)
```

```
## Classes 'data.table' and 'data.frame':   9000055 obs. of  6 variables:
##  $ userId    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ movieId   : num  122 185 292 316 329 355 356 362 364 370 ...
##  $ rating    : num  5 5 5 5 5 5 5 5 5 5 ...
##  $ RatingYear: num  1996 1996 1996 1996 1996 ...
##  $ title     : chr  "Boomerang (1992)" "Net, The (1995)" "Outbreak (1995)"
"Stargate (1994)" ...
```

```
##  $ genres     : chr  "Comedy|Romance" "Action|Crime|Thriller"
"Action|Drama|Sci-Fi|Thriller" "Action|Adventure|Sci-Fi" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
edx %>% group_by(RatingYear, title) %>%
  summarize(Ratings_Sum = n(), Average_Rating = mean(rating)) %>%
  mutate(Average_Rating = sprintf("%0.2f", Average_Rating)) %>%
  arrange(-Ratings_Sum) %>% print(n = 50)
```

```
## `summarise()` has grouped output by 'RatingYear'. You can override using
the
## `.groups` argument.
```

```
## # A tibble: 75,964 × 4
## # Groups:   RatingYear [15]
##    RatingYear title                                                 Ratin…¹
Avera…²
##         <dbl> <chr>                                                   <int>
<chr>
##  1       1996 Batman (1989)                                           12016
3.26
##  2       1996 Dances with Wolves (1990)                               11524
3.79
##  3       1996 Apollo 13 (1995)                                        11393
3.99
##  4       1996 Pulp Fiction (1994)                                     10925
4.01
##  5       1996 Fugitive, The (1993)                                    10901
4.12
##  6       1996 True Lies (1994)                                        10838
3.57
##  7       1996 Forrest Gump (1994)                                      9986
4.12
##  8       1996 Batman Forever (1995)                                    9907
3.13
##  9       1996 Aladdin (1992)                                           9856
3.67
## 10       1996 Jurassic Park (1993)                                     9771
3.84
## 11       1996 Ace Ventura: Pet Detective (1994)                        9724
2.96
## 12       1996 Clear and Present Danger (1994)                          9484
3.71
## 13       1996 Die Hard: With a Vengeance (1995)                        9467
3.48
## 14       1996 Silence of the Lambs, The (1991)                         9341
4.29
## 15       1996 Beauty and the Beast (1991)                              8895
3.68
## 16       1996 Stargate (1994)                                          8845
```

```
3.33
## 17        1996 Shawshank Redemption, The (1994)                         8728
4.48
## 18        1996 Outbreak (1995)                                          8386
3.56
## 19        1996 Star Trek: Generations (1994)                            8284
3.42
## 20        1996 Cliffhanger (1993)                                       8172
3.21
## 21        1996 Braveheart (1995)                                        8106
4.26
## 22        1996 Firm, The (1993)                                         8097
3.54
## 23        1996 Crimson Tide (1995)                                      8039
3.82
## 24        1996 Terminator 2: Judgment Day (1991)                        7994
3.96
## 25        1996 Speed (1994)                                             7949
3.79
## 26        1996 Dumb & Dumber (1994)                                     7938
2.83
## 27        1996 Net, The (1995)                                          7902
3.34
## 28        1996 Lion King, The (1994)                                    7692
3.81
## 29        1996 While You Were Sleeping (1995)                           7674
3.61
## 30        1996 Waterworld (1995)                                        7601
3.07
## 31        1996 Interview with the Vampire: The Vampire Chronicle…       7544
3.41
## 32        1996 GoldenEye (1995)                                         7421
3.44
## 33        1996 Mrs. Doubtfire (1993)                                    7391
3.62
## 34        1996 Seven (a.k.a. Se7en) (1995)                              7022
3.96
## 35        1996 Pretty Woman (1990)                                      6998
3.47
## 36        1996 Mask, The (1994)                                         6945
3.34
## 37        1996 Ghost (1990)                                             6840
3.61
## 38        1996 Natural Born Killers (1994)                              6497
3.15
## 39        1996 Quiz Show (1994)                                         6417
3.65
## 40        1996 Babe (1995)                                              6363
3.87
## 41        1996 Sleepless in Seattle (1993)                              6334
```

```
3.70
## 42       1996 Addams Family Values (1993)                           6072
3.06
## 43       1996 Schindler's List (1993)                               5894
4.52
## 44       1996 Four Weddings and a Funeral (1994)                    5871
3.70
## 45       1996 12 Monkeys (Twelve Monkeys) (1995)                    5861
3.90
## 46       1996 Get Shorty (1995)                                     5817
3.67
## 47       1996 Usual Suspects, The (1995)                            5669
4.30
## 48       1996 Home Alone (1990)                                     5430
3.15
## 49       1996 Disclosure (1994)                                     5373
3.37
## 50       1996 Clueless (1995)                                       5360
3.44
## # … with 75,914 more rows, and abbreviated variable names ¹Ratings_Sum,
## #   ²Average_Rating
```

```
edx_genres <-edx %>% separate_rows(genres, sep = "\\|")
```

Sum of Movie Ratings per Genre

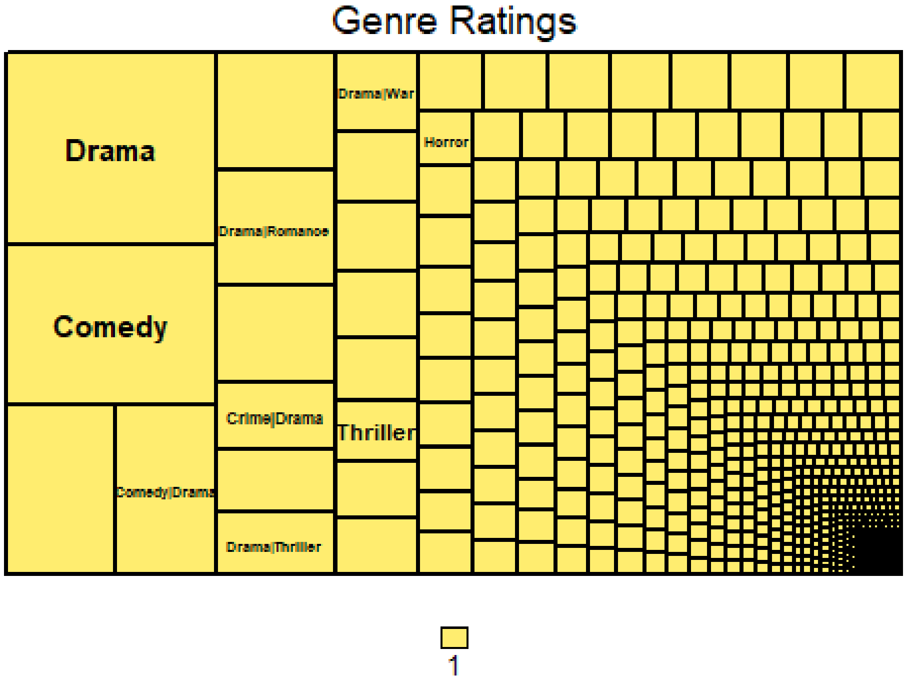
```
edx_genres %>%
  group_by(genres) %>% summarize(Ratings_Sum = n(), Average_Rating =
mean(rating)) %>%
    arrange(-Ratings_Sum)
```

```
## # A tibble: 20 × 3
##    genres             Ratings_Sum Average_Rating
##    <chr>                    <int>          <dbl>
##  1 Drama                  3910127           3.67
##  2 Comedy                 3540930           3.44
##  3 Action                 2560545           3.42
##  4 Thriller               2325899           3.51
##  5 Adventure              1908892           3.49
##  6 Romance                1712100           3.55
##  7 Sci-Fi                 1341183           3.40
##  8 Crime                  1327715           3.67
##  9 Fantasy                 925637           3.50
## 10 Children                737994           3.42
## 11 Horror                  691485           3.27
## 12 Mystery                 568332           3.68
## 13 War                     511147           3.78
## 14 Animation               467168           3.60
## 15 Musical                 433080           3.56
## 16 Western                 189394           3.56
## 17 Film-Noir               118541           4.01
## 18 Documentary                 93066          3.78
## 19 IMAX                         8181          3.77
## 20 (no genres listed)              7          3.64
```

```
library(treemap)
```

```
## Warning: package 'treemap' was built under R version 4.2.3
```

```
# sum of  ratings by genre
genre_ratings <- aggregate(rating ~ genres, edx, sum)
# construct treemap
treemap(genre_ratings, index = "genres", vSize = "rating",
        type = "value", palette = "Set3",
        title = "Genre Ratings")
```



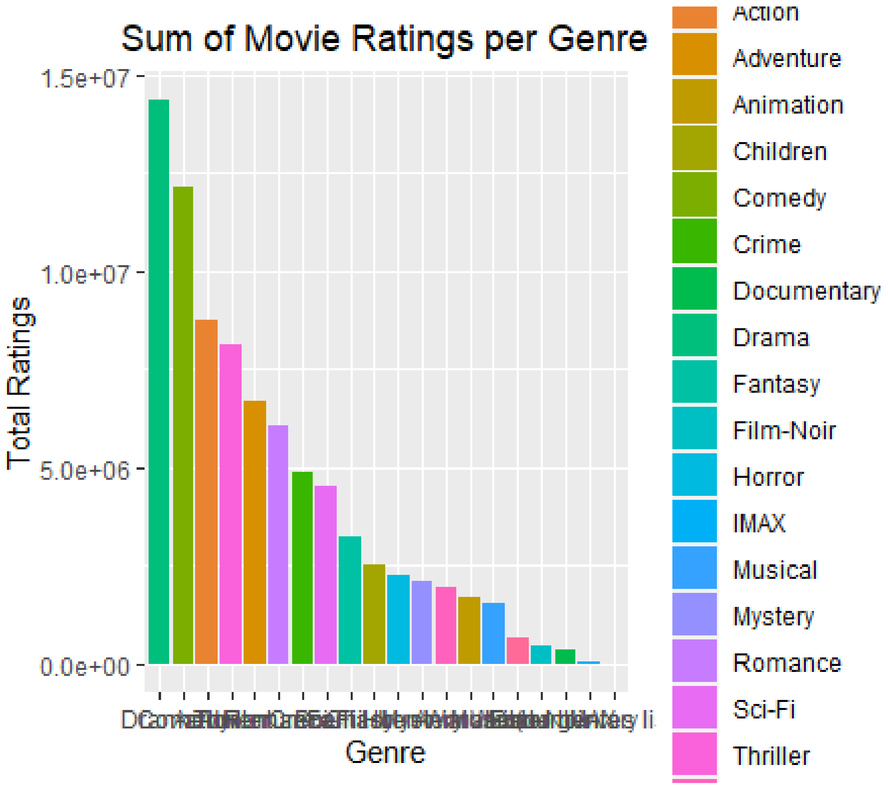
```
#Arrange the Genres by Mean Rating
edx_genres %>%
  group_by(genres) %>% summarize(Ratings_Sum = n(), Average_Rating =
mean(rating)) %>%
  arrange(-Average_Rating)
```

```
## # A tibble: 20 × 3
##    genres             Ratings_Sum Average_Rating
##    <chr>                    <int>          <dbl>
##  1 Film-Noir               118541           4.01
##  2 Documentary              93066           3.78
##  3 War                     511147           3.78
##  4 IMAX                      8181           3.77
##   5 Mystery                 568332           3.68
##   6 Drama                  3910127           3.67
##   7 Crime                  1327715           3.67
##   8 (no genres listed)           7           3.64
##   9 Animation               467168           3.60
## 10 Musical                  433080           3.56
## 11 Western                  189394           3.56
## 12 Romance                 1712100           3.55
## 13 Thriller                2325899           3.51
## 14 Fantasy                  925637           3.50
## 15 Adventure               1908892           3.49
## 16 Comedy                  3540930           3.44
## 17 Action                  2560545           3.42
## 18 Children                 737994           3.42
## 19 Sci-Fi                  1341183           3.40
## 20 Horror                   691485           3.27
```

```
#Coerce the 'genres' column from character data type to factor data type
edx$genres <-as.factor(edx$genres)
edx_genres$genres <-as.factor(edx_genres$genres)
class(edx_genres$genres)
```

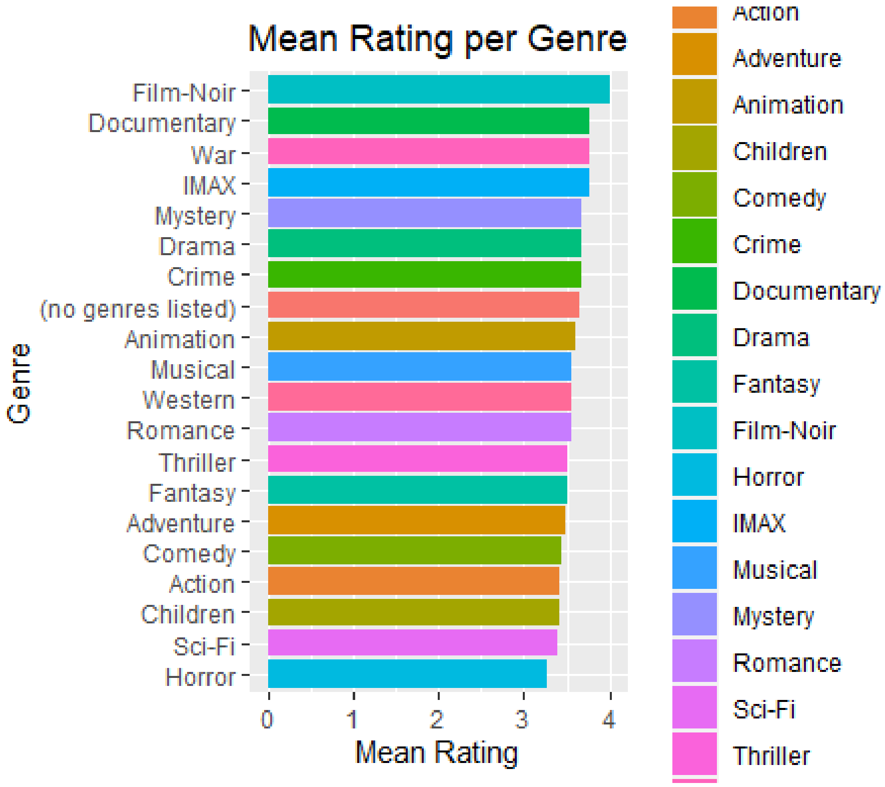
```
## [1] "factor"
```

```
library(ggplot2)

# Aggregate of ratings per genre
genre_ratings <- edx %>%
  separate_rows(genres, sep = "\\|") %>%
  group_by(genres) %>%
  summarize(total_ratings = sum(rating))

ggplot(genre_ratings, aes(x = reorder(genres, -total_ratings), y =
total_ratings, fill = genres)) +
  geom_bar(stat = "identity") +
  ggtitle("Sum of Movie Ratings per Genre") +
  xlab("Genre") +
  ylab("Total Ratings") +
  theme(plot.title = element_text(hjust = 0.5))
```



Mean Rating per Genre

```r
library(ggplot2)

mean_ratings <- edx %>%
  separate_rows(genres, sep = "\\|") %>%
  group_by(genres) %>%
  summarize(mean_rating = mean(rating), .groups = 'drop')

ggplot(mean_ratings, aes(x = reorder(genres, mean_rating), y = mean_rating,
fill = genres)) +
  geom_bar(stat = 'identity') +
  coord_flip() +
  ggtitle("Mean Rating per Genre") +
  xlab("Genre") +
  ylab("Mean Rating") +
  theme(plot.title = element_text(hjust = 0.5))
```



```
yearreleaseda <-as.numeric(str_sub(edx$title, start = -5, end = -2))
edx <- edx %>% mutate(yearReleased = yearreleaseda)
head(edx)
```

```
##    userId movieId rating RatingYear                          title
## 1:      1     122      5       1996               Boomerang (1992)
## 2:      1     185      5       1996                Net, The (1995)
## 3:      1     292      5       1996                Outbreak (1995)
## 4:      1     316      5       1996                Stargate (1994)
## 5:      1     329      5       1996 Star Trek: Generations (1994)
## 6:      1     355      5       1996       Flintstones, The (1994)
##                           genres yearReleased
## 1:                Comedy|Romance         1992
## 2:         Action|Crime|Thriller         1995
## 3:  Action|Drama|Sci-Fi|Thriller         1995
## 4:       Action|Adventure|Sci-Fi         1994
## 5: Action|Adventure|Drama|Sci-Fi         1994
## 6:       Children|Comedy|Fantasy         1994
```

```
#Do the same for the validation set
yearreleasedb <-as.numeric(str_sub(validation$title, start = -5, end = -2))
validation <- validation %>% mutate(yearReleased = yearreleasedb)
head(validation)
```

```
##    userId movieId rating RatingYear
## 1:      1     231      5       1996
## 2:      1     480      5       1996
## 3:      1     586      5       1996
## 4:      2     151      3       1997
## 5:      2     858      2       1997
## 6:      2    1544      3       1997
##                                                       title
## 1:                                     Dumb & Dumber (1994)
## 2:                                     Jurassic Park (1993)
## 3:                                        Home Alone (1990)
## 4:                                           Rob Roy (1995)
## 5:                                    Godfather, The (1972)
## 6: Lost World: Jurassic Park, The (Jurassic Park 2) (1997)
##                                      genres yearReleased
## 1:                                  Comedy         1994
## 2:        Action|Adventure|Sci-Fi|Thriller         1993
## 3:                         Children|Comedy         1990
## 4:                Action|Drama|Romance|War         1995
## 5:                             Crime|Drama         1972
## 6: Action|Adventure|Horror|Sci-Fi|Thriller         1997
```

```
#This is also applied to edx_train & edx_test for later modeling purposes
yearreleasedc <-as.numeric(str_sub(edx_train$title, start = -5, end = -2))
edx_train <- edx_train %>% mutate(yearReleased = yearreleasedc)
head(edx_train)
```

```
##    userId movieId rating RatingYear                         title
## 1:      1     122      5       1996              Boomerang (1992)
## 2:      1     292      5       1996               Outbreak (1995)
## 3:      1     316      5       1996               Stargate (1994)
## 4:      1     329      5       1996 Star Trek: Generations (1994)
## 5:      1     355      5       1996       Flintstones, The (1994)
## 6:      1     356      5       1996           Forrest Gump (1994)
##                            genres yearReleased
## 1:                Comedy|Romance         1992
## 2:  Action|Drama|Sci-Fi|Thriller         1995
## 3:       Action|Adventure|Sci-Fi         1994
## 4: Action|Adventure|Drama|Sci-Fi         1994
## 5:       Children|Comedy|Fantasy         1994
## 6:      Comedy|Drama|Romance|War         1994
```

```
yearreleasedd <-as.numeric(str_sub(edx_test$title, start = -5, end = -2))
edx_test <- edx_test %>% mutate(yearReleased = yearreleasedd)
head(edx_test)
```

```
##    userId movieId rating RatingYear
## 1:      1     185      5       1996
## 2:      2     260      5       1997
## 3:      2     590      5       1997
## 4:      2    1049      3       1997
## 5:      2    1210      4       1997
## 6:      3    1148      4       2005
##                                                           title
## 1:                                              Net, The (1995)
## 2: Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)
## 3:                                    Dances with Wolves (1990)
## 4:                           Ghost and the Darkness, The (1996)
## 5:            Star Wars: Episode VI - Return of the Jedi (1983)
## 6:                  Wallace & Gromit: The Wrong Trousers (1993)
##                              genres yearReleased
## 1:            Action|Crime|Thriller         1995
## 2:          Action|Adventure|Sci-Fi         1977
## 3:          Adventure|Drama|Western         1990
## 4:                 Action|Adventure         1996
## 5:          Action|Adventure|Sci-Fi         1983
## 6: Animation|Children|Comedy|Crime         1993
```

Use the newly defined “yearReleased” column to add a “MovieAge” column

```
edx <-edx %>% mutate(MovieAge = 2020 - yearReleased)
validation <-validation %>% mutate(MovieAge = 2020 - yearReleased)
edx_train <-edx_train %>% mutate(MovieAge = 2020 - yearReleased)
edx_test <-edx_test %>% mutate(MovieAge = 2020 - yearReleased)
```

Movie Age

```
summary(edx$MovieAge)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   12.00   22.00   26.00   29.78   33.00  105.00
```

Modeling The formula for RMSE can be defined as follows with y¯¯u,i the prediction of movie i by user u, and yu,the rating of movie i, by user u. N is then defined as the number of user/movie combinations and the sum of these different combinations.

```
RMSE <- function(true_ratings, predicted_ratings){
  sqrt(mean((true_ratings - predicted_ratings)^2))
}
```

Begin Modeling: Benchmarking Model

```
edx_train_mu <-mean(edx_train$rating)
NRMSE_M1 <- RMSE(edx_test$rating, edx_train_mu)
#Table the Results
results_table <-tibble(Model_Type = "NRMSE", RMSE = NRMSE_M1) %>%
                  mutate(RMSE = sprintf("%0.4f", RMSE))
results_table

## # A tibble: 1 × 2
##   Model_Type RMSE
##   <chr>      <chr>
## 1 NRMSE      1.0601
```

Median Table

```
edx_train_median <-median(edx_train$rating)
MM_M2 <-RMSE(edx_test$rating, edx_train_median)
#Table the Results
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model"),
                       RMSE = c(NRMSE_M1, MM_M2)) %>%
                        mutate(RMSE = sprintf("%0.4f", RMSE))
results_table
```

```
## # A tibble: 2 × 2
##   Model_Type   RMSE
##   <chr>        <chr>
## 1 NRMSE        1.0601
## 2 Median_Model 1.1668
```

Movie Effects Model

```
bi <- edx_train %>% group_by(movieId) %>%
  summarize(b_i = mean(rating - edx_train_mu))
```

create the prediction

```
prediction_bi <-edx_train_mu + edx_test %>%
  left_join(bi, by = "movieId") %>% .$b_i
MEM_M3 <-RMSE(edx_test$rating, prediction_bi)
#Table the Results
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model", "Movie
Effects"),
                       RMSE = c(NRMSE_M1, MM_M2, MEM_M3)) %>%
                        mutate(RMSE = sprintf("%0.4f", RMSE))
results_table
```

```
## # A tibble: 3 × 2
##   Model_Type    RMSE
##   <chr>         <chr>
## 1 NRMSE         1.0601
## 2 Median_Model  1.1668
## 3 Movie Effects 0.9430
```

Adding User Effects to the Movie Effects Model:

```
bu <-edx_train %>% left_join(bi, by = "movieId") %>% group_by(userId) %>%
  summarize(b_u = mean(rating - edx_train_mu - b_i))
```

Create the Prediction Then check the prediction against the test set to determine the RMSE and table the results.

```
prediction_bu <-edx_test %>% left_join(bi, by = "movieId") %>%
  left_join(bu, by = "userId") %>%
  mutate(predictions = edx_train_mu + b_i + b_u) %>% .$predictions
UEM_M4 <-RMSE(edx_test$rating, prediction_bu)
#Table the Results
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model", "Movie
Effects", "Movie & User Effects"),
                       RMSE = c(NRMSE_M1, MM_M2, MEM_M3, UEM_M4)) %>%
                        mutate(RMSE = sprintf("%0.4f", RMSE))
results_table

## # A tibble: 4 × 2
##   Model_Type           RMSE
##   <chr>                <chr>
## 1 NRMSE                1.0601
## 2 Median_Model         1.1668
## 3 Movie Effects        0.9430
## 4 Movie & User Effects 0.8647
```

Adding Movie Age Effects: (Movie, User & Movie Age Effects Model)

```
ba <- edx_train %>%
  left_join(bi, by="movieId") %>% left_join(bu, by ="userId") %>%
  group_by(MovieAge) %>% summarize(b_a = mean(rating - b_i - b_u -
edx_train_mu))
```

Create the Prediction Check the prediction against the test set to determine the RMSE and table the results.

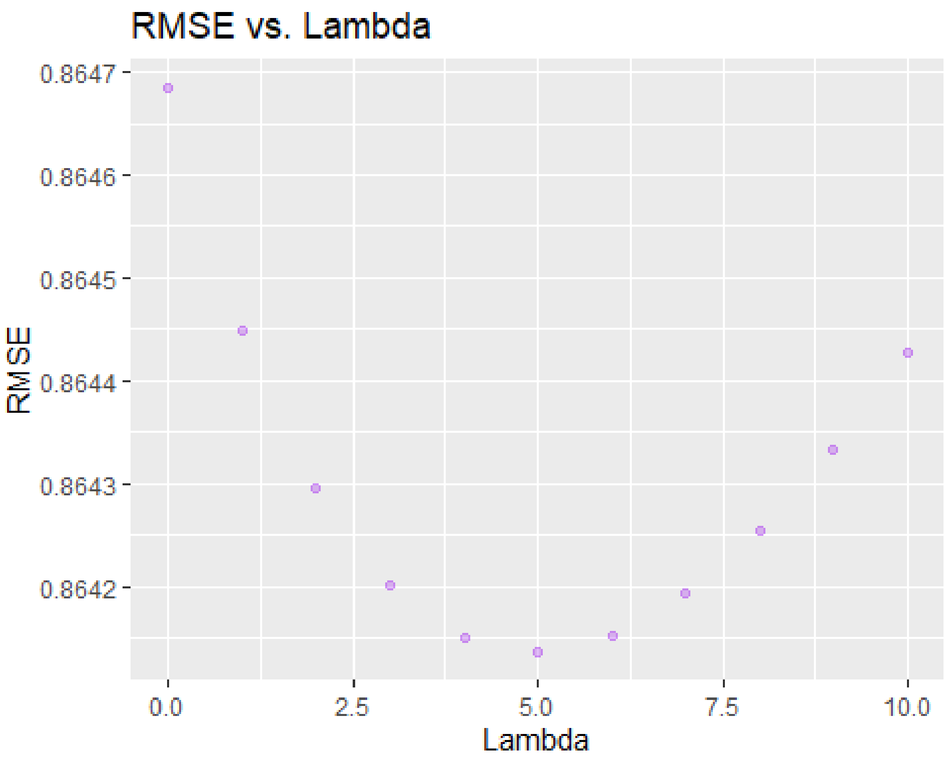
```
predictions_ma <- edx_test %>%
  left_join(bi, by = "movieId") %>% left_join(bu, by = "userId") %>%
  left_join(ba, by = "MovieAge") %>%  mutate(predictions = edx_train_mu + b_i
+ b_u + b_a) %>%
  .$predictions
UMMAE_M5 <-RMSE(edx_test$rating, predictions_ma)
#Table the results
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model", "Movie
Effects",
                                      "Movie & User Effects",
                                      "User, Movie & Movie Age Effects"),
                       RMSE = c(NRMSE_M1, MM_M2, MEM_M3, UEM_M4, UMMAE_M5))
%>%
                        mutate(RMSE = sprintf("%0.4f", RMSE))
results_table

## # A tibble: 5 × 2
##   Model_Type                      RMSE
##   <chr>                           <chr>
## 1 NRMSE                           1.0601
## 2 Median_Model                    1.1668
## 3 Movie Effects                   0.9430
## 4 Movie & User Effects            0.8647
## 5 User, Movie & Movie Age Effects 0.8643
```

Movie & User Effects Model with Regularization:

```
lambdasR <-seq(0, 10, 1)
RMSES <- sapply(lambdasR, function(l){
  edx_train_mu <- mean(edx_train$rating)

  b_i <- edx_train %>%
    group_by(movieId) %>%
    summarize(b_i = sum(rating - edx_train_mu)/(n() + l))

  b_u <- edx_train %>%
    left_join(b_i, by='movieId') %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_i - edx_train_mu)/(n() +l))

  predicted_ratings <- edx_test %>%
    left_join(b_i, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    mutate(pred = edx_train_mu + b_i +  b_u) %>% .$pred

return(RMSE(predicted_ratings, edx_test$rating))
})
#Determine which lambda minimizes the RMSE
lambda <- lambdasR[which.min(RMSES)]
lambda
```

```
## [1] 5
```

```
library(ggplot2)

# Create a data frame with lambdasR and RMSES
data <- data.frame(lambdasR = lambdasR, RMSES = RMSES)

# Create the scatter plot
ggplot(data, aes(x = lambdasR, y = RMSES)) +
  geom_point(color = "purple", alpha = 0.3) +
  ggtitle("RMSE vs. Lambda") +
  xlab("Lambda") + ylab("RMSE")
```



Building the Movie & User Effects Model with Regularization

```
b_i <- edx_train %>%
  group_by(movieId) %>%
  summarize(b_i = sum(rating - edx_train_mu)/(n()+lambda))
b_u <-edx_train %>%
  left_join(b_i, by="movieId") %>%
  group_by(userId) %>%
  summarize(b_u = sum(rating - b_i - edx_train_mu)/(n()+lambda))
reg_prediction <- edx_test %>%
  left_join(b_i, by = "movieId") %>%
  left_join(b_u, by = "userId") %>%
  mutate(predictions = edx_train_mu + b_i + b_u) %>% .$predictions

UMEM_REG_M6 <-RMSE(edx_test$rating, reg_prediction)
#Table the Results
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model", "Movie 
Effects",
                                      "Movie & User Effects",
                                      "Movie, User & Movie Age Effects",
                                      "Movie & User Effects 
w/Regularization"),
                       RMSE = c(NRMSE_M1, MM_M2, MEM_M3, UEM_M4,
                                UMMAE_M5, UMEM_REG_M6)) %>%
                        mutate(RMSE = sprintf("%0.6f", RMSE))
results_table
```

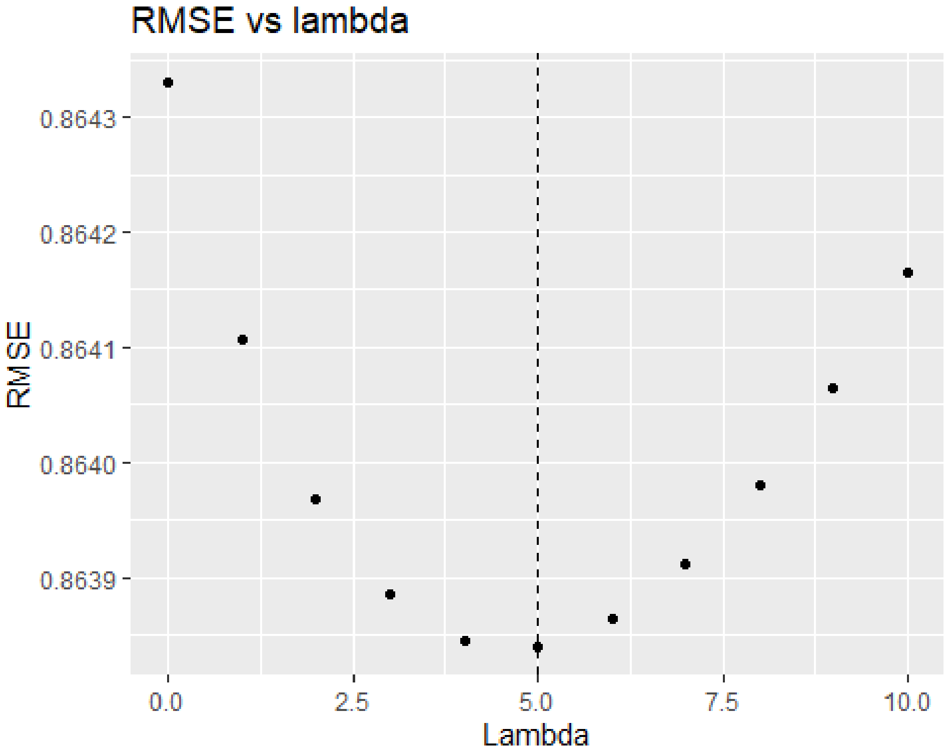
```
## # A tibble: 6 × 2
##   Model_Type                                RMSE
##   <chr>                                     <chr>
## 1 NRMSE                                     1.060054
## 2 Median_Model                              1.166756
## 3 Movie Effects                             0.942961
## 4 Movie & User Effects                      0.864684
## 5 Movie, User & Movie Age Effects           0.864330
## 6 Movie & User Effects w/Regularization 0.864136
```

Movie, User & Movie Age Effects Model with Regularization:

```
lambdasM <-seq(0, 10, 1)
RMSES2 <-sapply(lambdasM, function(l){
  edx_train_mu <-mean(edx_train$rating)

  b_i <-edx_train %>%
    group_by(movieId) %>%
    summarize(b_i = sum(rating - edx_train_mu)/(n() + l))

  b_u <-edx_train %>%
    left_join(b_i, by='movieId') %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_i - edx_train_mu)/(n() +l))

  b_a <-edx_train %>%
    left_join(b_i, by = "movieId") %>% left_join(b_u, by = "userId") %>%
    group_by(MovieAge) %>%
    summarize(b_a = sum(rating - b_i - b_u - edx_train_mu)/(n()+l))

  predicted_ratings <-edx_test %>%
    left_join(b_i, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    left_join(b_a, by = "MovieAge") %>%
    mutate(predictions = edx_train_mu + b_i + b_u + b_a) %>% .$predictions

  return(RMSE(predicted_ratings, edx_test$rating))
})
lambda2 <- lambdasM[which.min(RMSES2)]
lambda2
```

```
## [1] 5
```

```
library(ggplot2)

# Create a data frame with lambdasM and RMSES2
df <- data.frame(lambda = lambdasM, RMSE = RMSES2)

# Create a scatterplot of RMSEs vs lambdasM
ggplot(df, aes(x = lambda, y = RMSE)) +
  geom_point() +
  geom_vline(xintercept = lambda2, linetype = "dashed") +
  ggtitle("RMSE vs lambda") +
  xlab("Lambda") + ylab("RMSE")
```



Building the User, Movie & Movie Age Effects Model with Regularization

```
b_i <- edx_train %>%
  group_by(movieId) %>%
  summarize(b_i = sum(rating - edx_train_mu)/(n()+lambda2))
b_u <-edx_train %>%
  left_join(b_i, by = "movieId") %>%
  group_by(userId) %>%
  summarize(b_u = sum(rating - b_i - edx_train_mu)/(n()+lambda2))
b_a <-edx_train %>%
  left_join(b_i, by = "movieId") %>% left_join(b_u, by = "userId") %>%
  group_by(MovieAge) %>%
  summarize(b_a = sum(rating - b_i - b_u - edx_train_mu)/(n()+lambda2))
reg_prediction2 <- edx_test %>%
  left_join(b_i, by = "movieId") %>%
  left_join(b_u, by = "userId") %>%
  left_join(b_a, by = "MovieAge") %>%
  mutate(pred = edx_train_mu + b_i + b_u + b_a) %>%
  pull(pred)
UMMAE_REG_M7 <-RMSE(edx_test$rating, reg_prediction2)
#Table the Results
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model", "Movie
Effects",
                                      "Movie & User Effects",
                                      "User, Movie & Movie Age Effects",
                                      "Movie & User Effects 
w/Regularization",
                                      "User, Movie & Movie Age Effects 
w/Regularization"),
                       RMSE = c(NRMSE_M1, MM_M2, MEM_M3, UEM_M4,
                                UMMAE_M5, UMEM_REG_M6, UMMAE_REG_M7)) %>%
                        mutate(RMSE = sprintf("%0.5f", RMSE))
results_table
```

```
## # A tibble: 7 × 2
##   Model_Type                                       RMSE   
##   <chr>                                            <chr>  
## 1 NRMSE                                            1.06005
## 2 Median_Model                                     1.16676
## 3 Movie Effects                                    0.94296
## 4 Movie & User Effects                             0.86468
## 5 User, Movie & Movie Age Effects                  0.86433
## 6 Movie & User Effects w/Regularization            0.86414
## 7 User, Movie & Movie Age Effects w/Regularization 0.86384
```

Using Validation: Now we will move on to using the edx & validation sets to confirm our Final Model achieves an RMSE less than .8649.

The Benchmarking Model with Validation:

```
edx_mu <-mean(edx$rating)
FRMSE_M1 <-RMSE(validation$rating, edx_mu)
#Table the Results
results_table <-tibble(Model_Type = ("NRMSE"),
                       Final_RMSE_Validation = (NRMSE_M1)) %>%
                      mutate(Final_RMSE_Validation = sprintf("%0.5f", 
Final_RMSE_Validation))
results_table
```

```
## # A tibble: 1 × 2
##   Model_Type Final_RMSE_Validation
##   <chr>      <chr>                
## 1 NRMSE      1.06005
```

Median Model with validation:

```
edx_med <-median(edx$rating)
FRMSE_M2 <-RMSE(validation$rating, edx_med)
#Table the Results
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model"),
                       Final_RMSE_Validation = c(FRMSE_M1, FRMSE_M2)) %>%
  mutate(Final_RMSE_Validation = sprintf("%0.5f", Final_RMSE_Validation))
results_table
```

```
## # A tibble: 2 × 2
##   Model_Type   Final_RMSE_Validation
##   <chr>        <chr>
## 1 NRMSE        1.06120
## 2 Median_Model 1.16802
```

```
bi <- edx %>% group_by(movieId) %>%
  summarize(b_i = mean(rating - edx_mu))
#Prediction
prediction_bi <-edx_mu + validation %>%
  left_join(bi, by = "movieId") %>% .$b_i
FRMSE_M3 <-RMSE(validation$rating, prediction_bi)
#Table the Results
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model", "Movie
Effects"),
                       Final_RMSE_Validation = c(FRMSE_M1, FRMSE_M2,
FRMSE_M3)) %>%
  mutate(Final_RMSE_Validation = sprintf("%0.5f", Final_RMSE_Validation))
results_table
```

```
## # A tibble: 3 × 2
##   Model_Type    Final_RMSE_Validation
##   <chr>         <chr>
## 1 NRMSE         1.06120
## 2 Median_Model  1.16802
## 3 Movie Effects 0.94391
```

Movie & User Effects Model with Validation

```
bu <-edx %>% left_join(bi, by = "movieId") %>% group_by(userId) %>%
  summarize(b_u = mean(rating - edx_mu - b_i))
#Prediction
prediction_bu <-validation %>% left_join(bi, by = "movieId") %>%
  left_join(bu, by = "userId") %>%
  mutate(predictions = edx_mu + b_i + b_u) %>% .$predictions
FRMSE_M4 <-RMSE(validation$rating, prediction_bu)
#Table the Results
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model", "Movie
Effects",
                                      "Movie & User Effects"),
                       Final_RMSE_Validation = c(FRMSE_M1, FRMSE_M2,
FRMSE_M3,
                                                 FRMSE_M4)) %>%
  mutate(Final_RMSE_Validation = sprintf("%0.5f", Final_RMSE_Validation))
results_table
```

```
## # A tibble: 4 × 2
##   Model_Type           Final_RMSE_Validation
##   <chr>                <chr>
## 1 NRMSE                1.06120
## 2 Median_Model         1.16802
## 3 Movie Effects        0.94391
## 4 Movie & User Effects 0.86535
```

Movie, User & Movie Age Effects with Validation:

```
ba <- edx %>%
  left_join(bi, by = "movieId") %>% left_join(bu, by = "userId") %>%
  group_by(MovieAge) %>% summarize(b_a = mean(rating - b_i - b_u - edx_mu))
#Prediction
predictions_ma <- validation %>%
  left_join(bi, by = "movieId") %>% left_join(bu, by = "userId") %>%
  left_join(ba, by = "MovieAge") %>%  mutate(predictions = edx_mu + b_i + b_u
+ b_a) %>%
  .$predictions
FRMSE_M5 <-RMSE(validation$rating, predictions_ma)
#Table the Results
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model", "Movie
Effects",
                                      "Movie & User Effects",
                                      "Movie, User, & Movie Age Effects"),
                       Final_RMSE_Validation = c(FRMSE_M1, FRMSE_M2,
FRMSE_M3,
                                                 FRMSE_M4, FRMSE_M5)) %>%
  mutate(Final_RMSE_Validation = sprintf("%0.5f", Final_RMSE_Validation))
results_table

## # A tibble: 5 × 2
##   Model_Type                       Final_RMSE_Validation
##   <chr>                            <chr>
## 1 NRMSE                            1.06120
## 2 Median_Model                     1.16802
## 3 Movie Effects                    0.94391
## 4 Movie & User Effects             0.86535
## 5 Movie, User, & Movie Age Effects 0.86500
```

Movie & User Effects with Regularization (Validation):

```
lambda

## [1] 5

#Movie & User Effects Model with Regularization using the validation set

b_i <-edx %>%
  group_by(movieId) %>%
  summarize(b_i = sum(rating - edx_mu)/(n()+lambda))
b_u <-edx %>%
  left_join(b_i, by="movieId") %>%
  group_by(userId) %>%
  summarize(b_u = sum(rating - b_i - edx_mu)/(n()+lambda))
reg_prediction <-validation %>%
  left_join(b_i, by = "movieId") %>%
  left_join(b_u, by = "userId") %>%
  mutate(predictions = edx_mu + b_i + b_u) %>% .$predictions

FRMSE_M6 <-RMSE(validation$rating, reg_prediction)
#Table the Results
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model", "Movie
Effects",
                                      "Movie & User Effects",
                                      "Movie, User, & Movie Age Effects",
                                      "Movie & User Effects
w/Regularization"),
                       Final_RMSE_Validation = c(FRMSE_M1, FRMSE_M2,
FRMSE_M3,
                                                 FRMSE_M4, FRMSE_M5,
                                                 FRMSE_M6)) %>%
                        mutate(Final_RMSE_Validation = sprintf("%0.5f",
                                               Final_RMSE_Validation))
results_table
```

```
## # A tibble: 6 × 2
##   Model_Type                             Final_RMSE_Validation
##   <chr>                                  <chr>
## 1 NRMSE                                  1.06120
## 2 Median_Model                           1.16802
## 3 Movie Effects                          0.94391
## 4 Movie & User Effects                   0.86535
## 5 Movie, User, & Movie Age Effects       0.86500
## 6 Movie & User Effects w/Regularization 0.86482
```

Final Model with Validation: This Model features Movie, User, & Movie Age Effects with Regularization

```
lambda2
```

```
## [1] 5
```

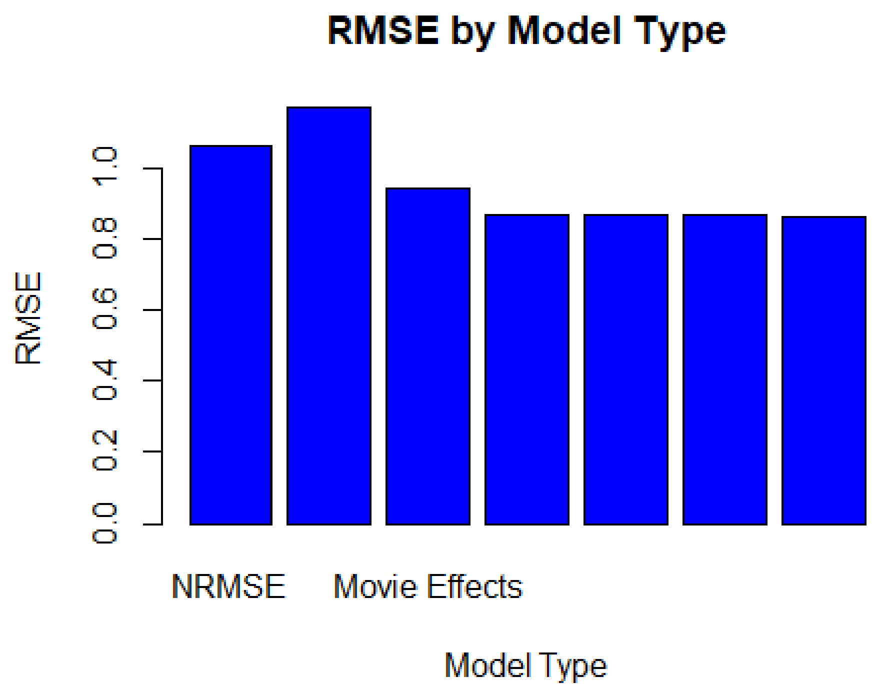
```
b_i <- edx %>%
  group_by(movieId) %>%
  summarize(b_i = sum(rating - edx_mu)/(n()+lambda2))
b_u <-edx %>%
  left_join(b_i, by="movieId") %>%
  group_by(userId) %>%
  summarize(b_u = sum(rating - b_i - edx_mu)/(n()+lambda2))
b_a <-edx %>%
  left_join(b_i, by="movieId") %>% left_join(b_u, by= "userId") %>%
  group_by(MovieAge) %>%
  summarize(b_a = sum(rating - b_i - b_u - edx_mu)/(n()+lambda2))
reg_prediction2 <-validation %>%
  left_join(b_i, by = "movieId") %>%
  left_join(b_u, by = "userId") %>%
  left_join(b_a, by = "MovieAge") %>%
  mutate(predictions = edx_mu + b_i + b_u + b_a) %>% .$predictions

FRMSE_M7 <-RMSE(validation$rating, reg_prediction2)
#Table the Results
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model", "Movie
Effects",
                                      "Movie & User Effects",
                                      "Movie, User, & Movie Age Effects",
                                      "Movie & User Effects
w/Regularization",
                                      "Movie, User & Movie Age Effects
w/Regularization"),
                       Final_RMSE_Validation = c(FRMSE_M1, FRMSE_M2,
FRMSE_M3,
                                                 FRMSE_M4, FRMSE_M5,
                                                 FRMSE_M6, FRMSE_M7)) %>%
  mutate(Final_RMSE_Validation = sprintf("%0.5f",
                                         Final_RMSE_Validation))
results_table

## # A tibble: 7 × 2
##   Model_Type                                       Final_RMSE_Validation
##   <chr>                                            <chr>
## 1 NRMSE                                            1.06120
## 2 Median_Model                                     1.16802
## 3 Movie Effects                                    0.94391
## 4 Movie & User Effects                             0.86535
## 5 Movie, User, & Movie Age Effects                 0.86500
## 6 Movie & User Effects w/Regularization            0.86482
## 7 Movie, User & Movie Age Effects w/Regularization 0.86452

#Building the User, Movie & Movie Age Effects Model with Regularization
```

Table the training & test set results against those of the validation set

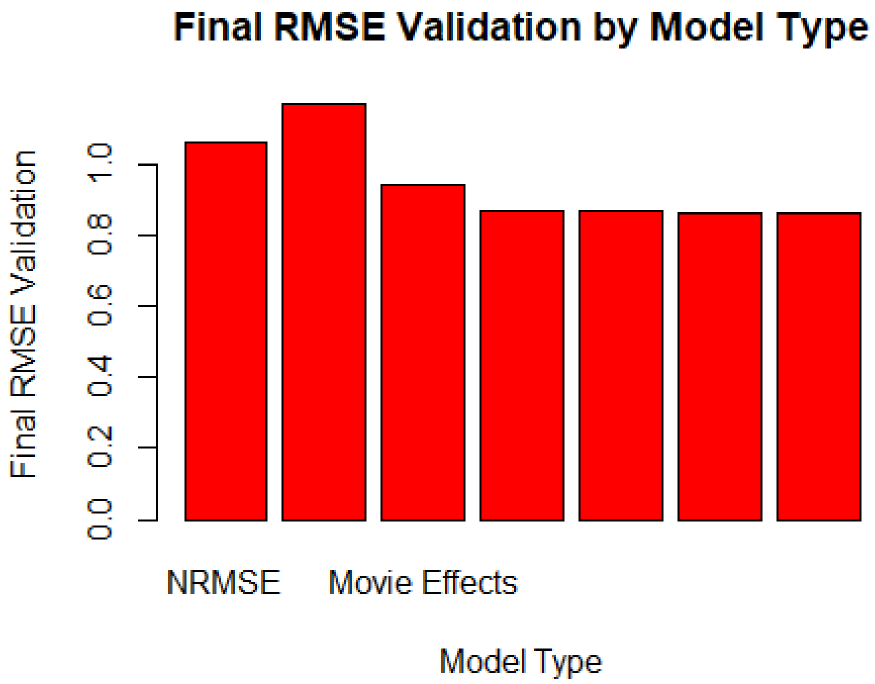
```
results_table <-tibble(Model_Type = c("NRMSE", "Median_Model", "Movie
Effects",
                                      "Movie & User Effects",
                                      "Movie, User & Movie Age Effects",
                                      "Movie & User Effects
w/Regularization",
                                      "User, Movie & Movie Age Effects
w/Regularization"),
                       RMSE = c(NRMSE_M1, MM_M2, MEM_M3, UEM_M4,
                                UMMAE_M5, UMEM_REG_M6, UMMAE_REG_M7),
                       Final_RMSE_Validation = c(FRMSE_M1, FRMSE_M2,
                                                 FRMSE_M3, FRMSE_M4,
                                                 FRMSE_M5, FRMSE_M6,
                                                 FRMSE_M7)) %>%
  mutate(Final_RMSE_Validation = sprintf("%0.5f",
                                         Final_RMSE_Validation)) %>%
  mutate(RMSE = sprintf("%0.5f", RMSE))


results_table

## # A tibble: 7 × 3
##   Model_Type                                        RMSE
Final_RMSE_Validation
##   <chr>                                             <chr>   <chr>
## 1 NRMSE                                             1.06005 1.06120
## 2 Median_Model                                      1.16676 1.16802
## 3 Movie Effects                                     0.94296 0.94391
## 4 Movie & User Effects                              0.86468 0.86535
## 5 Movie, User & Movie Age Effects                   0.86433 0.86500
## 6 Movie & User Effects w/Regularization             0.86414 0.86482
## 7 User, Movie & Movie Age Effects w/Regularization 0.86384 0.86452

#The kable function in knitr table of the final results
results_table %>% knitr::kable()
```

Model_Type	RMSE	Final_RMSE_Validation
NRMSE	1.06005	1.06120
Median_Model	1.16676	1.16802
Movie Effects	0.94296	0.94391
Movie & User Effects	0.86468	0.86535
Movie, User & Movie Age Effects	0.86433	0.86500
Movie & User Effects w/Regularization	0.86414	0.86482
User, Movie & Movie Age Effects w/Regularization	0.86384	0.86452

```
# Create a data frame with the given data
model_data <- data.frame(
  Model_Type = c("NRMSE", "Median_Model", "Movie Effects", "Movie & User
Effects",
                 "Movie, User & Movie Age Effects", "Movie & User Effects
w/Regularization",
                 "User, Movie & Movie Age Effects w/Regularization"),
  RMSE = c(1.06005, 1.16676, 0.94296, 0.86468, 0.86433, 0.86414, 0.86384),
  Final_RMSE_Validation = c(1.06120, 1.16802, 0.94391, 0.86535, 0.86500,
0.86482, 0.86452)
)

# Create a bar plot of RMSE
barplot(
  model_data$RMSE,
  names.arg = model_data$Model_Type,
  xlab = "Model Type",
  ylab = "RMSE",
  main = "RMSE by Model Type",
  col = "blue"
)
```



```
# Create a bar plot of Final_RMSE_Validation
barplot(
  model_data$Final_RMSE_Validation,
  names.arg = model_data$Model_Type,
  xlab = "Model Type",
  ylab = "Final RMSE Validation",
  main = "Final RMSE Validation by Model Type",
  col = "red"
)
```



This Final Model achieves an RMSE of .86452 The lowest RMSE using the validation set is the Final Validation Model featuring Regularized User, Movie & Movie Age Effects.