MOVIELENS RECOMMENDATION

The overarching goal of this assignment is to create a model that predicts users' ratings given a movie. This is all based on the movielens dataset, a dataset with over *9 million* rows. More specifically, after running the starting code given, we are using the edx dataset to predict user ratings in the validation set.

The movielens dataset consists of the following columns:

```
## [1] "userId"    "movieId"   "rating"    "timestamp" "title"     "genres"
```

userId:

There are 69878 different people giving their ratings. Each person is given a unique user ID and each user rates a few movies.

movieId:

There are 10677 different movies to be rated. Each movie is given a unique movie Id and each movie is rated by a few people.

rating:

It is the rating given by the user for the movie. Ratings are discrete values, having steps of 0.5 and ranging from 0.5 to 5. A higher rating means the movie was better received by the user.

timestamp:

Timestamp is a specific encoding of time. With certain conversions, they represent when the users made their ratings.

title:

These are simply the titles of the movies represented by the movie IDs.

genres:

There are many genres of movies and this column records the genre(s) of each movie. If a movie has multiple genres, they are separated with the delimiter '|'.

Methods

I was unable to produce any graphs of this as any attempt will just kill my program. So I just hypothesized. A person is likely to rate a movie higher if other people also rate it as high. The measure I use for this is the movie average rating. The way to check this is by going through all the movies and calculating the average rating per movie. I used the following code:

```
train_set$movie_avg <- ave(train_set$rating,train_set$movieId)
```

Also, a person is likely to rate a movie higher if he/she rates other movies as high. However, it should be noted that a person may also rate a movie higher if the movie itself is good. So I used the metric 'tendency to overscore', which measures how likely a person is to rate a movie higher than the average. I feel that this is a better gauge. I used the following code:

```
train_set <- train_set%>%mutate(overscore=rating-movie_avg)
train_set$tendency_overscore <- ave(train_set$overscore,train_set$userId)
```

I also used the users' and movies' individual rankings as a parameter. So instead of using the unrelated, chaotic nature of userId and movieId as predictors, I decided to use their corresponding rankings as such. I used the following code:

```
train_set$movie_rank<-rank(train_set$movie_avg,train_set$movieId)
train_set$user_rank<-rank(train_set$tendency_overscore,train_set$userId)
```

Originally, I was under impression that a user's rating of a movie was dependent of the age of the movie, as well as the genre of the movie. However, since the data set is very large, I am unable to do the necessary data manipulation to get the parameters I needed.
As for the age of the movie, I needed to find the year the movie was released, as well as the year the movie was rated.
Under the column 'title', the movies come attached with the release date, in brackets. So, using this code:

```
titles<-edx$title
years<-pbsapply(titles,function(x){
  str_extract(x,regex('\\(\\d{4}\\)'))%>%substr(2,5)
})
edx$movie_year<-as.numeric(years)
train_set <- train_set%>%mutate(rating_year=as.numeric(year(as_datetime(timestamp))),years_since_watche
```

I was able to extract the age of the movie. However, it takes about 30 minutes each time to do this process, so in the end I decided not to use this as a predictor. Besides, the first model I did on this only gave it a coefficient of around 0.0008.

Results

I used the caret package 'train' model to create my predictive model.

```
TrainData <- train_set%>%select(movie_avg,tendency_overscore,timestamp,movie_rank,user_rank)
TrainClasses <- train_set$rating
model<-train(TrainData,TrainClasses,method='lm',tuneLength=10,trControl=trainControl(method='boot'))
```

Testing the results, I got a pretty good result of RMSE = 0.8652998. For fears of crashing my RStudio again, I didn't want to look into the specifics. I didn't want to see which entries gave the highest errors.

Conclusion

Hopefully when I upgrade my computer, I can conduct a more in-depth analysis of my results, but for now, this is all I have.