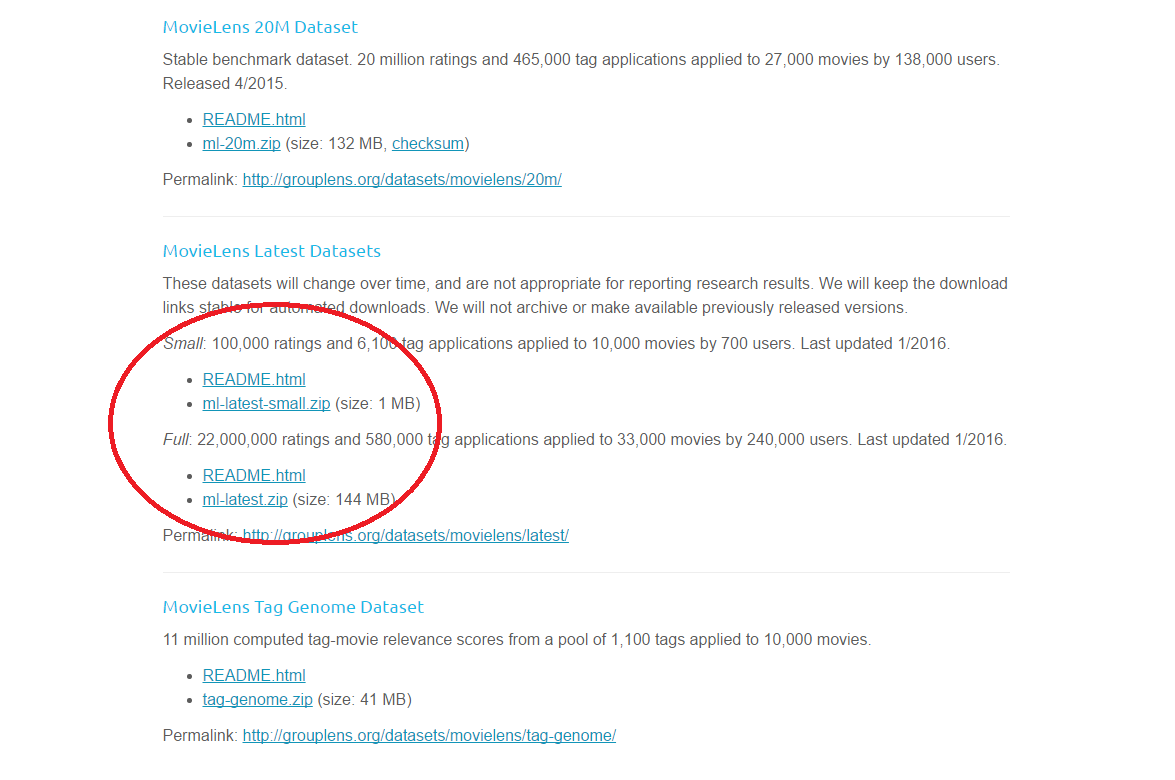
# Data Science Intensive Capstone Project: A Movie Recommender System

## Project Purpose:

Using a dataset of movie ratings across hundreds of movies and hundreds of users, the goal of this project is to develop a robust movie recommender system. Though this is not a novel problem, having been well researched thanks in large part to the Netflix Prize competition, it remains highly relevant due to its potential impact across a variety of industries. Though the current dataset can only provide movie recommendations, it is a methodology that can be applied to online retail, other entertainment (books, television, video games), and travel destinations to name just a few.

## About the Data:

MovieLens.org is a website that allows users to provide ratings and tags to movies. It is operated by GroupLens, a research group from the University of Minnesota. For this project, I chose to use the Full and Small versions of the Latest Dataset available at <http://grouplens.org/datasets/movielens/>, updated as of January 2016.

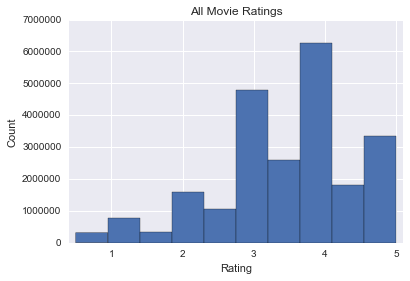


# Insights from Data Exploration

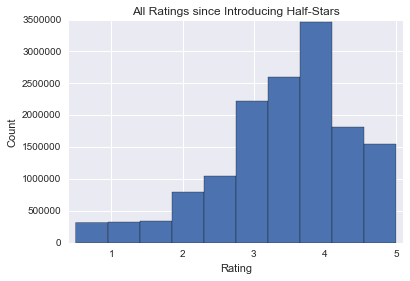
Prior to building the recommendation system, an exploratory survey of the Full data set revealed a few interesting and impactful insights which are highlighted below.

## Movie Ratings:

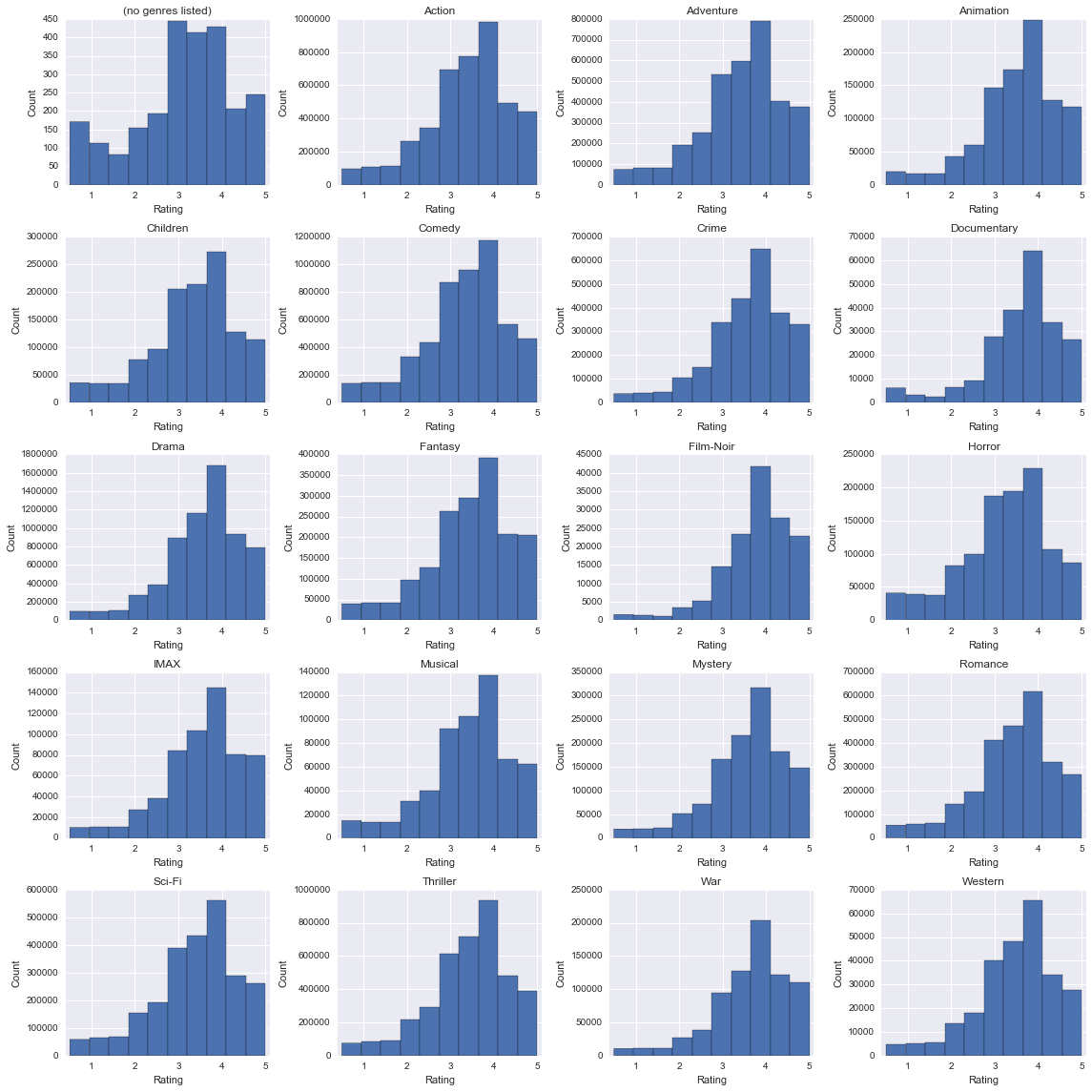
One of the first things that jumps out about this dataset is the odd distribution of the ratings across all movies:



The easily noticeable bias towards whole-number ratings does have an explanation: the half-star ratings were not implemented until a much later date. In fact, the earliest half-star rating occurs in February 2003, whereas the dataset’s earliest ratings start in January 1995. Because this has such a large effect on the distribution of ratings, when training the recommendation system, it will be necessary to correct for this bias. Therefore, prior to training any model, we must filters out data prior to August 2003, allowing for a six month adoption period. Correcting for this discrepancy, we arrive at the following distribution:



The result, a left skewed distribution, more closely matches what was expected. Considering that people will tend to watch something they already have an interest in, there should be an inherent bias towards higher ratings. Few people will intentionally take the time to watch a movie that they have no interest in and then rate it. This distribution remains prevalent across an examination of the movie genres as depicted in the histograms below:



## Regarding Genres:

Moviegoers often identify highly with specific genres that match their interests. As such, genres are very important to a user’s decision to watch a movie or not. The MovieLens dataset classifies all movies into a combination of the following genres:

|  |  |  |
| --- | --- | --- |
| '(no genres listed)' | 'Action' | 'Adventure' |
| 'Animation' | 'Children' | 'Comedy' |
| 'Crime' | 'Documentary' | 'Drama' |
| 'Fantasy' | 'Film-Noir' | 'Horror' |
| 'IMAX' | 'Musical' | 'Mystery' |
| 'Romance' | 'Sci-Fi' | 'Thriller' |
| 'War' | 'Western' |  |

Only movies without a genre will go in to the ‘(no genres listed)’ category. This isn’t a true genre as it doesn’t have anything to do with the contents of the movie. By that same logic, ‘IMAX’ isn’t truly a genre either as it has more to do with the equipment used to record and/or project the movie. Furthermore, not all people watching a movie will watch it in IMAX. Therefore, both of these ‘genres’ are ignored in further exploratory work.

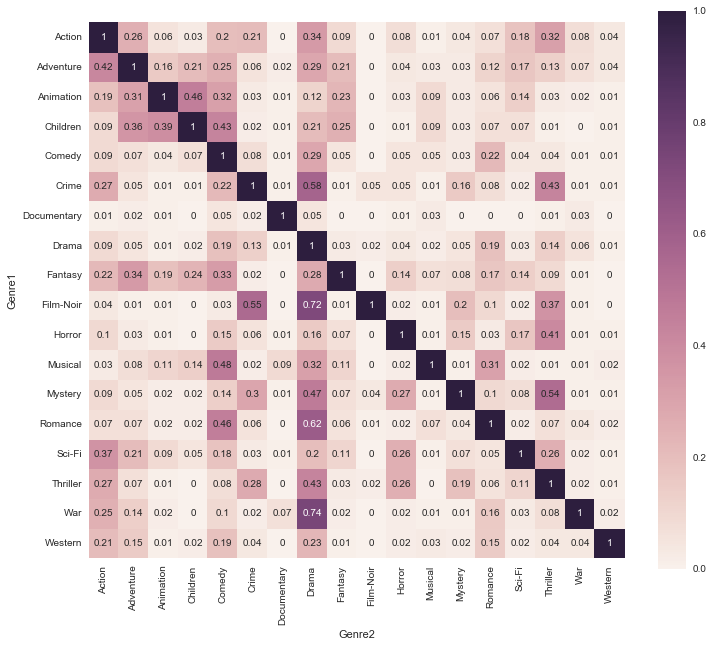
Genres pose an interesting challenge because they are essentially classifying the movies into different categories with one exception: movies can belong to multiple categories at once. This goes against the very definition of a categorical variable. Instead of using them as a categorical variables, it is possible to convert this information into a ‘genre similarity’ score by measuring the overlap between two specific genres as seen in the figure below. It essentially determines what percentage of movies that are identified as belonging to Genre1 also belong to Genre2.

A quick glance at the Figure below allows us to draw some quick conclusions:

1. Some genres have very little overlap with others. In particular, documentaries stand well apart from the other genres. This is to be expected considering that the tone and purpose behind a documentary is generally to educate and inform the audience rather than to entertain with a story.
2. Most genres have a sizeable overlap with Drama (particularly Crime, Film-Noir, Romance, and War), but the reverse is not true. The drama genre, as its defined, contains stories about realistic characters facing realistic struggles, a rather general theme which appears across most movies.
3. There are sizeable overlaps in some of the expected genre pairings, for example:
   * Adventure and Action
   * War and Drama
   * Romance and Comedy
   * Mystery and Thriller

This only really serves to indicate that some genres function well together, while other pairings have either not been thoroughly explored or are just not meant to be combined (Children and War).

While these genre overlaps are interesting to study on their own, they could also be used to enhance the recommender system by guiding it to movies with a higher genre similarity.



# Building a Recommender System

## Data Filters:

Prior to building the model, a review of the raw data shows that some of the data should not be used when training and testing the recommender system:

1. As mentioned above, the raw data has a bias towards whole-star ratings due to the late implementation of the half-star option. All ratings prior to August 2003 are filtered out to correct the discrepancy and allow for a 6 month adoption period.
2. There are a few users who seem to have rated upwards of 2000 movies making them significant outliers by nearly doubling, or in a few instances, over doubling the number of ratings when compared to the next most active user. Due to the rather suspicious amount of activity from these users, they were filtered out of the dataset.
3. In order to properly train, validate, and test the data, all partitions must contain many ratings per movie and per user. The model uses 80% of the data for training and 4-fold cross-validation, while the remaining 20% is held back for testing. To stand a chance of meeting the requirements, users with fewer than 40 ratings were filtered out. Then, movies with fewer than 35 ratings were also removed. The second filter is applied on the results of the first meaning that some users may have lost some of their ratings, hence the higher threshold (40 versus 35). These threshold values were picked in an effort to ensure that all partitions had over five ratings per movie and per user. Each 20% of the data should, in theory, should have at least 7 entries per movie and user, but random partitioning may make it less which is why there is a small margin for error in this step.

The combination of all these adjustments to the raw data have significantly lowered the size of the Small dataset from 105,339 to 20,816 ratings.

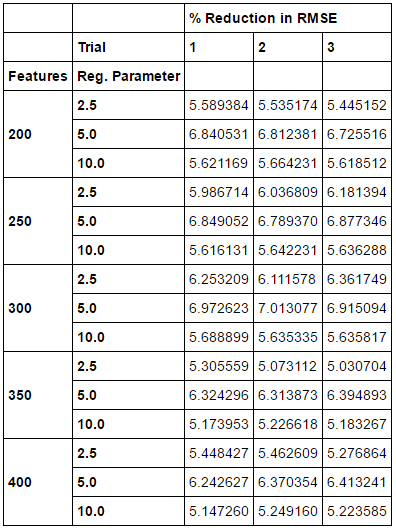
## Performance Metrics:

Similar to the Netflix Prize Competition, this model uses Root-Mean-Square-Error as a metric to evaluate model performance. The measure of success is the extent to which the recommender system can reduce the RMSE in comparison to the average movie rating, the simplest reasonable recommendation possible.

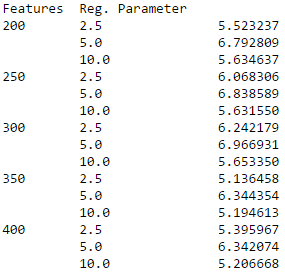
## Optimizing and Training the Recommender System:

The model is being trained from a completely random initialization of both movie and user features. An estimate of what a single user will rate a single movie can be obtained by the sum of products between that user’s and that movie’s features. This permits for the use of gradient descent in a collaborative filtering algorithm, adjusting both the user and movie features every iteration.

There are two key parameters that needed to be optimized, the number of features (per user and movie) and the regularization parameter. As mentioned above, 4-fold cross-validation was used on an 80% partition across a variety of parameter values. The 4-fold cross-validation essentially sets a quarter of the data aside for scoring and trains the model on the remaining three quarters. After storing the results (reduction in RMSE compared to providing the average rating), it sets aside the second quarter of the data and trains a new version of the model on the remaining data. This is repeated four times so that each quarter of the 80% partition acts as the cross-validation data once. The results are then averaged to determine the model’s performance. This 4-fold validation was run for every pairing of regularization parameter (2.5, 5, or 10) and number of features (200, 250, 300, 350, 400). This entire process was repeated two more times so that each pairing was 4-fold cross-validated in three separate trials. The three trials are a precautionary measure due to the fact that every model trained begins from a fresh random initialization and that all of the solutions the model determined are likely to be local optima. A summary of the results is as follows:



By averaging across the three trials per parameter pairing and selecting the highest % Reduction in RMSE, we get the optimal parameter values of 300 features and 5 as a regularization parameter:



These parameters were then used to train five more models using the full 80% partition. These five models were scored against the 20% testing partition and their results averaged to report a 7.41% reduction in RMSE when compared to just providing a user with the movie’s average rating.

The Full dataset, due to processing limitations could not use the same batch gradient descent algorithm as the Small dataset. Although a separate mini-batch gradient descent algorithm has also been developed, there are no results to report as it could take weeks to fully optimize and train the model on such a large dataset.

## Generating a List of Similar Movies

Given a movie name, the algorithm is capable of generating a list of similar movies. To enable this recommendation, the model must calculate ‘distances’ between every possible movie pairing by using the respective movie’s learned features. Because the features may be highly correlated, it is best to start the process by performing Principal Component Analysis (PCA). Apart from compressing the features and making them easier to visualize, PCA also maps the correlated movie features onto orthogonal co-ordinates making them linearly uncorrelated. Now, the newly mapped features can be used to calculate a ‘distance’ between two movies. The smaller the distance, the more similar the movies.

However, before calculating the distance, there is one other concern to address: the varying scale of the principal components. This is easily addressed by normalizing the values on a scale of 0 to 1. Finally, we can determine the Euclidean distance between the normalized principal components to assign a numeric measure to similarity between movies. After applying the distance to every movie pairing and providing a target movie, the model is able to quickly generate a list of movies similar to said target:



## Making a Personalized Movie Recommendation

If a user takes the time to rate several movies, the model is also capable of providing a personalized list of recommendations by predicting what that user’s rating across all movies included in the data. This is achieved by performing linear regression to learn the new user’s features based on the movie ratings they have provided. The linear regression does require its regularization parameter to be optimized. For this optimization, there is not really any need for a cross-validation or testing partition. The user features will always converge towards a global minima that varies solely with the regularization parameter. Having learned the user’s features, predictions of that user’s ratings are readily available by taking the dot product of the each movie’s features and the new user’s features. The most time intensive part of this process is getting the user to rate their watched movies, but once that is done, the recommendation system can quickly generate a list of personalized movie recommendations:



# Ideas for Improvement

As with all learning algorithms, this recommender system is limited by its inputs. In this case, the data used to actually build the model only spans 337 movies and 255 users, a tiny fraction of the available data from the Full dataset. Though I have built functioning stochastic and mini-batch gradient descent algorithms that could tackle the Full dataset, they both require a lot more time to optimize and debug in comparison to the current implementation. Due to the time and computational constraints, the Small dataset had to suffice for now. Making the switch to the Full Dataset, however, is just one way to improve upon this model.

As mentioned above, the overlap in genres could likely be used to improve upon the recommendations. Currently, the list the model generates is equally open to all movies, but rewarding a movie with a ‘genre similarity’ score, depending on how it’s implemented, could give more direction to the recommendations. For example, in the list above, “L.A. Confidential” does not seem like a movie that would be similar to “Harry Potter and the Philosopher’s Stone”. Another recommendation belonging to the Adventure and Fantasy genres would be more fitting.