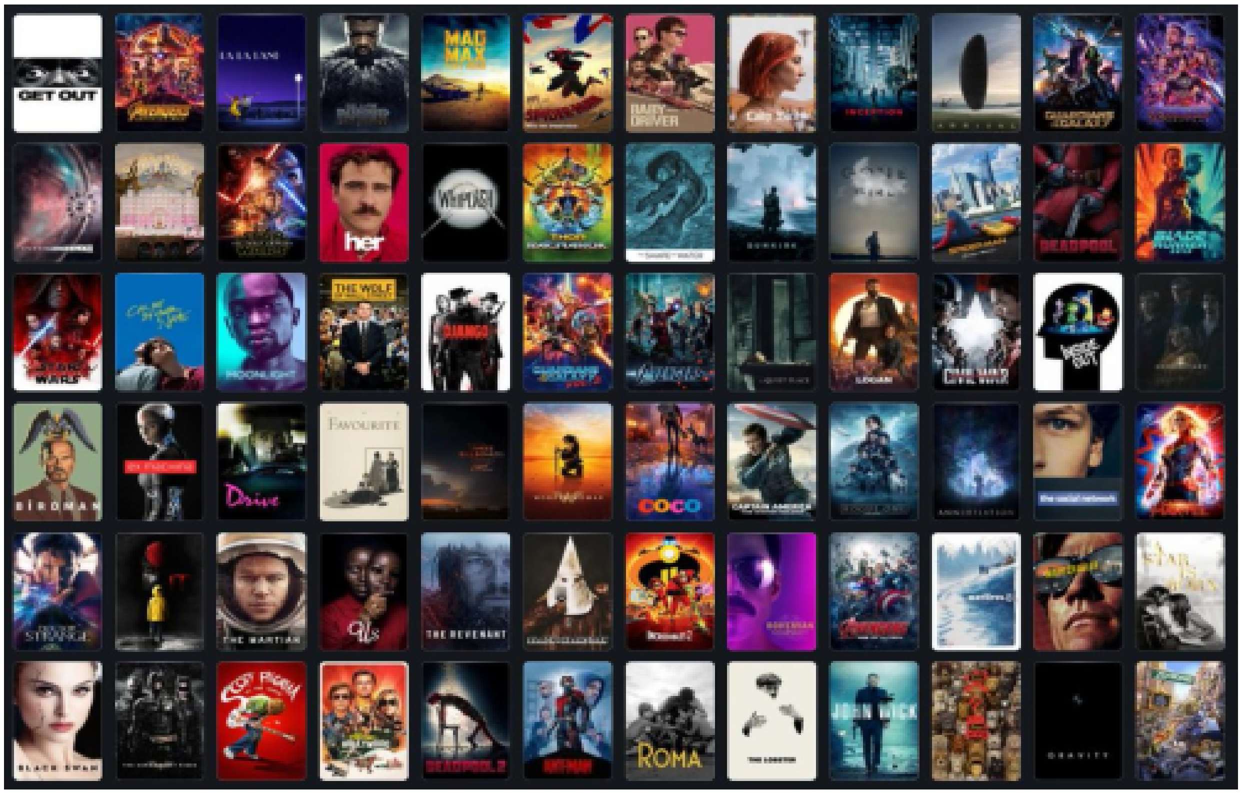
CP421 Project Paper

By: Cole Hanniwell (180320780), Einstein Oyewole (180517070), Jack Pentesco (180191760) and Jacob Orrico (180414490)



Introduction:

Deciding on what movie to watch is always a time consuming task where everyone always seems undecided on what they prefer. People are stuck on what genres they might like and go back and forth contemplating movies. Our movie recommendation aims to solve this problem so that users can discover new movies they might like on those indecisive days. The approach we have decided to take is combining Singular Value Decomposition (SVD) and Reinforcement Learning. The objective of this project is to predict ratings a user would give to a movie, thus increasing the likelihood of them finding the movie appealing. How we will go about this project is through a system that will recommend movies that have higher ratings to each of the respective users. The movies listed with a higher predicted rating are the top choices for the user, so they feel more at ease about their movie and decide quicker. Overall, our well-thought out approach to designing the movie recommendation system will help users choose more desirable movies, to match their preferences.

We initially started off with a reinforcement learning approach which involves trial and error from previous actions. The aim of the method is to maximize the immediate reward. The agent can reach the goal through a sequence of actions, but the actions and rewards must be trained so that the model can generate new actions. Exploitation vs. Exploration was also something we took into consideration. Exploitation refers to the optimal actions and uses existing knowledge to find the best reward. It fails to recognize a potentially more promising reward is out there through other unknown actions. On the other hand, exploration makes a recommendation without any underlying bias. This is more of a long-term approach because the agent can learn more about undetected actions that lead to more benefits. Exploration can be risky because the result could be something totally undesirable, however it may lead to new realizations that we might find useful. It has the ability to recommend actions and popular movies which in turn helps us learn more about the user.

SVD is the popular algorithm that won Netflix's million dollar competition in 2012. It is still one of the best algorithms for predicting movie ratings by users. The algorithm is extremely efficient and accurate but suffers from cold starts. It takes advantage of dimension reduction properties which is one of the reasons it is very fast.

Related Work:

We compared our method to a total of five baseline methods consisting of the global mean, user mean, movie mean, non-negative matrix factorization (NMF) and K-nearest neighbors. The global-mean takes the mean of the existing ratings we have as the prediction, which essentially means the same value will be given to every prediction. The user-mean takes the average rating of each respective user and uses that as the prediction. Thirdly, the movie-mean applies the same idea as the user-mean, but movies are replaced with users for this method. Non-negative matrix factorization involves factorizing the original matrix into two matrices, which only have positive elements. It is able to obtain significant information from these non-negative matrices. Lastly, K-nearest neighbors groups K data points together that share common characteristics as it aims to measure the distance between the test and training points. Whatever the class is of the point with the minimum distance to the new point, is the class given to the new point.

Solution/Method:

Our final model is a combination of Reinforced Learning and SVD. We decided on this combination because we wanted to remove the cold-start that SVD has. Using just Singular Value Decomposition to predict ratings of new users and new movies leads to a 11-21% increase in error. The reinforced learning model allows us to recommend movies to the user and build up enough data to the point SVD will not face a cold start challenge. Once a threshold of movies the user has rated is reached through the Reinforced Learning suggestions we use SVD to predict the users ratings on the remaining unwatched movies. The SVD model is much more efficient than the Reinforced Learning allowing us to create predicted user ratings faster and suggesting the highest rated predicted movies.

The Reinforced Learning model is made with a Reinforcement Learning Process that maximizes immediate reward (users movie rating). We used model-free value-based reinforcement learning to learn the value of suggesting a movie after a user has rated a set of movies. In this model, we define a state S as the set of movies rated, actions A as the movies to recommend, and the rewards as the ratings. The algorithm has a function $Q\colon S * A \to \mathbb{R}$ that calculates the quality of state-action combination (suggesting a movie). Q is initialized as an empty matrix of

0’s and is updated when users make a rating using the following function

$Q[S_i , A_i] = \sum_{j=1}^{|S|} (\frac{rating-2.5}{7.5}) * similarity(A_i, A_j)$. The similarity metric used in the Q-value was Euclidean Distance which performed better than Minkowski and Manhattan. The function used to update the Q-values gives a positive reward for recommending well rated movies and negative rewards (punishment) for recommending poorly rated movies. The agent selects the actions (movies) with the highest Q-value for a user as the recommendations. This model, as stated, recommends movies and does not predict ratings so we did not use it for our experiments later.

A common technique for recommender systems is Collaborating Filtering which relies on the user history and items in the dataset. SVD is a mathematical model for dimension reduction and can be applied to Collaborative Filtering. The algorithm projects the data to a lower dimension while attempting keeping the vast majority of details. SVD extracts features and correlation from the user-item matrix. We predict a users (user x) rating (r_{xi}) for a movie (movie i) using the formula:

$\hat{r}_{xi} = \mu + b_x + b_i + \sum_j q_{ij} * p_{xj}$. Where μ represents the mean of the normal distribution for factor vectors (default at 0), b_x and b_i represent the user bias and movie bias respectively, and q and p represent the movies factor and user factor respectively. To find some optimal values of the aforementioned weights, the algorithm minimizes the regularized sum of squares error using stochastic gradient descent. Regularized Sum of Squares Error:

$$\sum_{r_{ui} \in R_{train}} \left(r_{ui} - \hat{r}_{ui}\right)^2 + \lambda \left(b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2\right)$$

Experiments:

To choose the optimal parameters for the SVD algorithm we ran a grid search on different ranges for each parameter. The optimal parameters for SVD in the surprise package were n_factors = 150, n_epochs = 75, lr_all = 0.01, reg_all = 0.1. Where n_factors is the number of factors, n_epochs is the number of iterations of stochastic gradient descent, lr_all is the learning rate for the bias terms and the factor representations and reg_all is the regularization term for all weights, λ.

Data:

The dataset consisted of 610 users with 100386 ratings across 9742 movies, both the User IDs and Movie IDs were consistent through the files. There was an average overall rating of 3.5 with a standard deviation of 1.04. The first movie was rated in 1996 and the last movie was rated in 2018, with 1475 unique tags that users gave to describe the movies. Tags are metadata that are generated by the user and they can consist of short phrases or even a single word. Most common tag is “In Netflix Queue” and the most common genres are “no genre listed” and “Film-Noir”.

For the reinforced learning method we scraped some new data from the web to test on the recommender. We joined the new data with the movies we had in our system and used the recommender to only recommend movies in the system the user had seen. We wanted to suggest only movies the user had seen to be able to see if the recommender was suggesting highly rated movies. Also, when creating the Q matrix for the recommender we exploded the genres for each movie. We also used the bag-of-words (count vectorizer) technique for the tags of each movie when creating the Q matrix. A movie similarity matrix was also computed one time and imported for creation of the Q matrix.

For the collaborative filtering SVD it only requires a dataset containing the user's Id, movie Ids and the user assigned ratings to corresponding movies. Using the ratings table, we removed the timestamp feature and created a training and testing set for our experiment.

Evaluation and Results:

A test set that consisted of 20% of the data was taken out of the original data to evaluate the effectiveness of our model. The test set was fed into the SVD model and predicted ratings were produced for the test set to compare to the ratings in the test set that already exist. The Root-Mean Square error (RMSE) was the metric used. The formula for this is

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

Where, ŷ is a predicted rating and y is the actual rating. To further evaluate our method we compared our RMSE to RMSE' produced by five other models. These five prediction models were, use the global-mean of the ratings as the prediction, use user-means as the prediction, use movie-means as the prediction, non-negative matrix factorization and K-nearest neighbors.

The first three prediction models where the mean is used are good basic baseline methods to use to compare how a very simple model will compare against one that is more complex, to see if a complex model is even necessary. The last two prediction methods that we compared our model against are very common approaches to solve the movie recommendation problem, which is why they were used.

Below are the RMSE for each method:

Method	**Root-Mean Square Error (RMSE)**
Global Mean	150.60

User Mean	132.37
Movie Mean	112.06
Non-negative matrix factorization	0.9220
K-nearest neighbors	0.9477
SVD (Our Model)	0.8489

As seen above, our model produces the best RMSE out of all the methods that were compared against it. This means the ratings that the SVD model predicted were the closest overall to the actual ratings.

RMSE was the metric used because it gives a relatively high weight to large errors and in this case larger errors are undesirable. When attempting to recommend a movie to a user, the recommender system will recommend the movies with the highest predicted ratings. Large errors are undesirable because the system should not recommend movies the users will not like (low actual rating but high predicted rating). The goal is to keep users interested in the movies the system is recommending for them so the worse a prediction is, the more the model should be punished for it.