Machine Learning & Data Mining — Caltech CS/CNS/EE 155
Miniproject 2: MovieLens Visualization — Released February 16st, 2024

1 Introduction

Team Name: Z Fighters

Team Members: Jonayet Lavin, A. Yusuf Kavranoglu, Dereje Lambert, Ismail Abouamal

Work Division:

- Jonayet worked on Data Preprocessing and Basic Visualizations, as well as the off-the-shelf implementation for Matrix Factorization Visualizations.
- Ismail worked on the non-bias implementation for Matrix Factorization Visualizations and its comparison with the biased version.
- Dereje worked on the bias implementation for Matrix Factorization Visualizations and comparisons between methods.
- Yusuf worked on improving the bias implementation for Matrix Factorization Visualizations and comparison of model performances.

Colab Link:
https://colab.research.google.com/drive/17511Rz9jezlT2Z9oGAWZe0jZExYVgYf4?usp=sharing
Piazza Post:
https://piazza.com/class/lprgcbeelor41p/post/443
Packages Used: scikit-learn, scikit-surprise, matplotlib, seaborn, adjustText, pandas, numpy, requests

2 Data Loading and Preprocessing

The data was obtained from URLs comprising data.csv, movies.csv, train.csv, and test.csv. These file comprised user ratings for movies, movie information including genres, and split versions of the data for training and testing. The Python library requests was used to download these files and they were inputed into pandas DataFrames for manipulation and analysis.

Prior to the analysis, an inspection was performed on the datasets, using Google Sheets' data cleanup features, to detect any inconsistencies. The inspection revealed that there were duplicates in the 'Movie Title' columns of movies.csv, so there were movies with the same title that were associated with different 'Movie ID's. These duplicates had to be removed because if not, it would result in inaccuracies in analysis as ratings for what is essentially the same movie might be fragmented across different identifiers (IDs). The duplicates were removed by identifying the duplicates, grouping these duplicates and creating a mapping from all identified 'Movie ID's to a single representative ID for each group, and applying this mapping for all datasets (data, train, test) to ensure consistency.

3 Data Visualization

1. All ratings in the MovieLens Dataset.

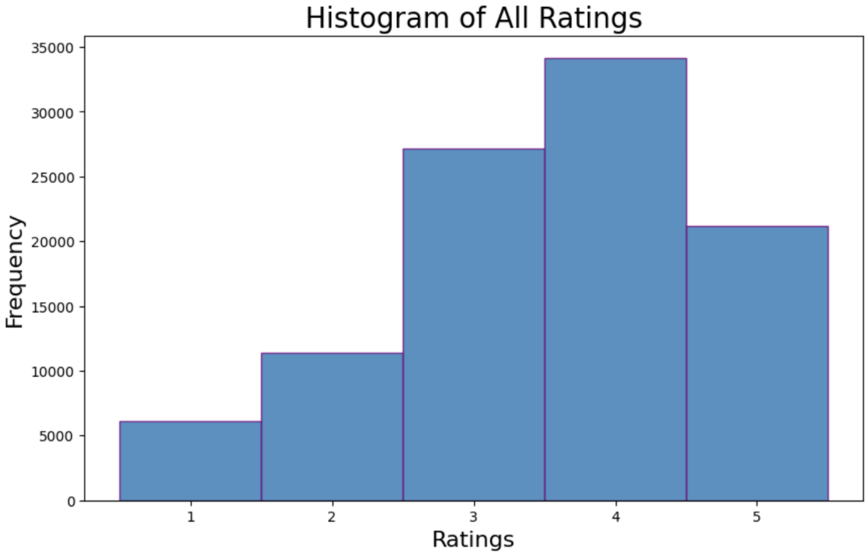

Figure 1: Frequency distribution of user ratings for all movies in the MovieLens dataset, ranging from 1 to 5 stars. The histogram shows bias towards higher ratings, indicating a potential skew in user rating behavior, possibly due to selection bias or systematic bias.

2. All ratings of the ten most popular movies (movies which have received the most ratings).

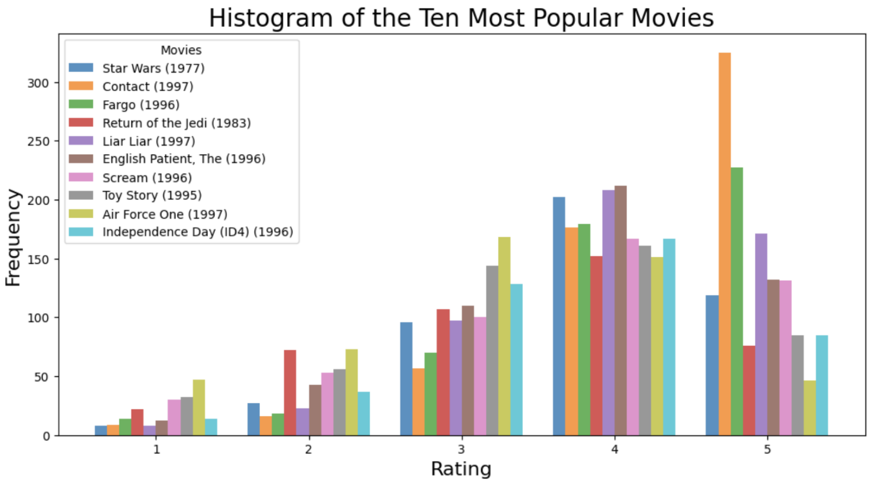

Figure 2: Frequency distribution of user ratings for the ten most popular movies in the MovieLens dataset. Histogram depicts even more bias towards higher rated movies as they have more ratings due to a potential positive feedback loop.

3. All ratings of the ten best movies (movies with the highest average ratings).

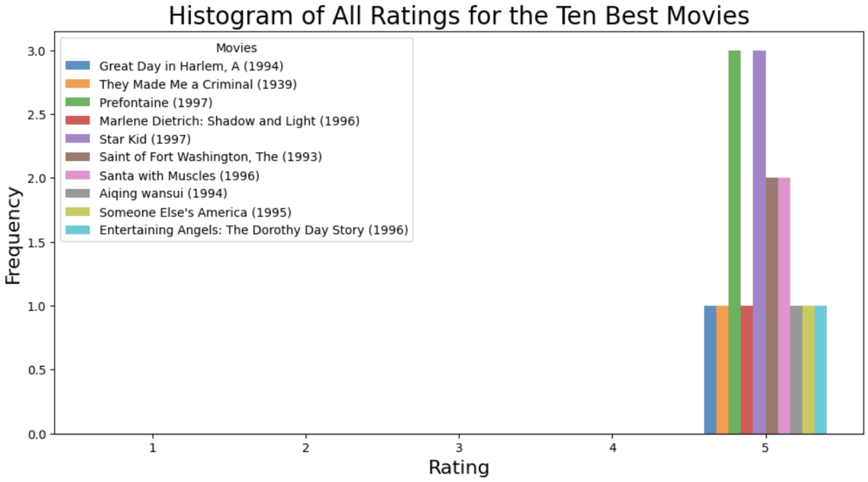

Figure 3: Frequency distribution of user ratings for the ten best movies, determined by the highest average ratings in the MovieLens dataset.

We notice that the dataset is skewed upward, possibly due to case with movies having a small number of very high ratings. Since this is hindering data visualization, we try using weighted ratings, which takes into account both the average rating and the number of ratings. We use IMDb's formula.

The weighted rating W was calculated with the formula:

$$W = \frac{v}{v+m} \cdot R + \frac{m}{v+m} \cdot C$$

where:

- W is the weighted rating.
- R is the average/mean rating for the movie.
- v is the number of votes for the movie, which is equivalent to the number of ratings the movie has received.
- m is the minimum number of votes required for a movie to be considered as a top-rated title.
- C is the mean vote across the dataset (the average rating of all movies).

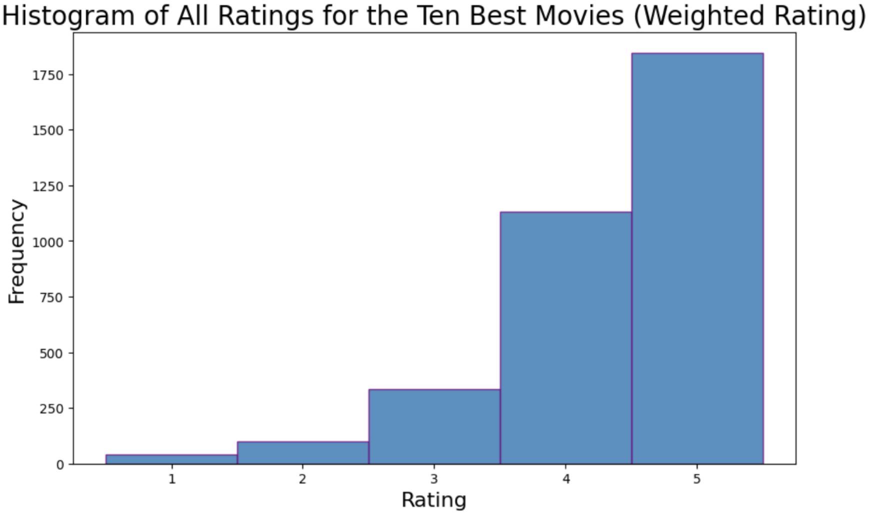

Figure 4: Frequency distribution of user ratings for the ten best movies, weighted by IMDb's formula. Histogram shows a strong preference for high ratings.

4. All ratings of movies from three genres of your choice (create three separate visualizations).

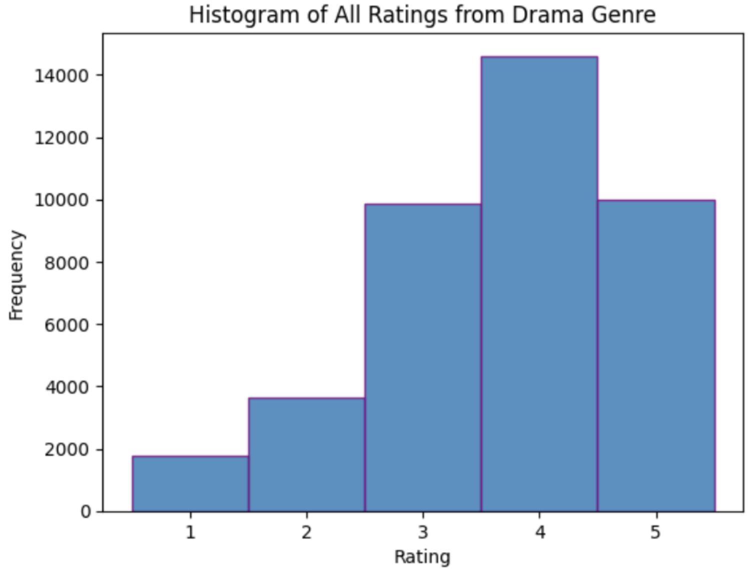

Figure 5: Histogram of ratings frequency for Drama genre, showing a wide range of ratings with a significant leaning towards 4 stars, indicating a diverse but positive reception.

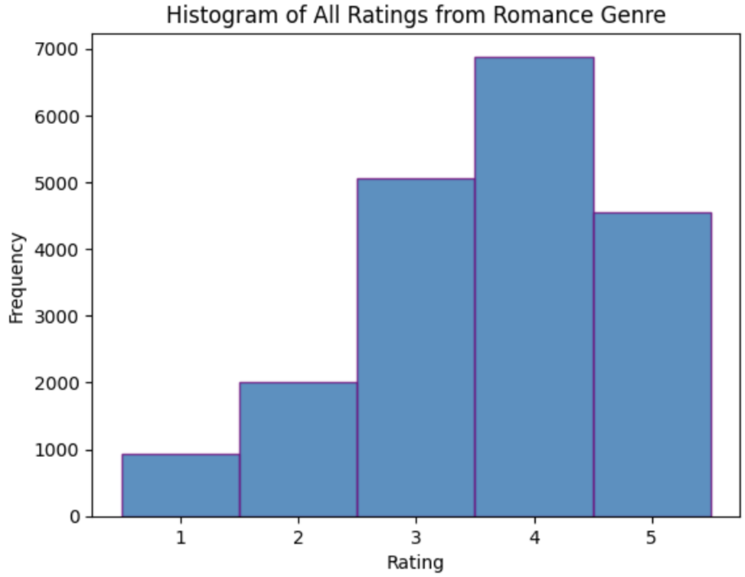

Figure 6: Histogram of ratings frequency for Romance genre, showing a similar distribution as for Drama, indicating generally favorable responses but with less variance.

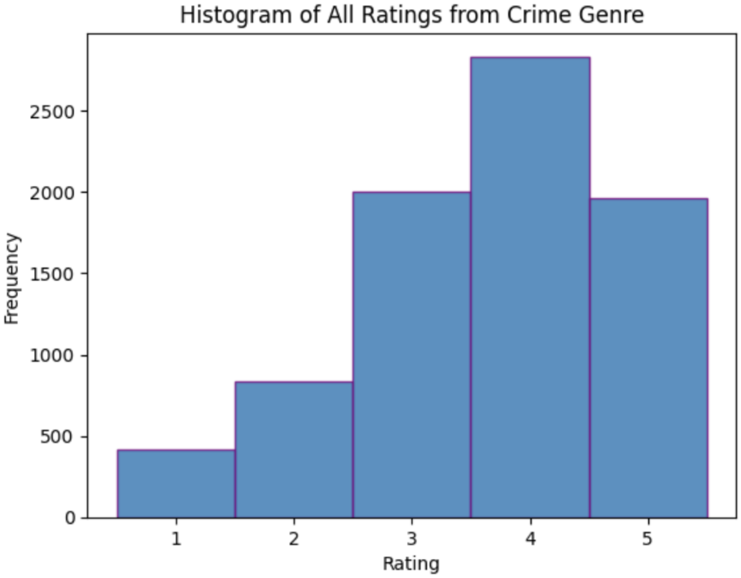

Figure 7: Histogram of ratings frequency for Crime genre, showing a strong preference for 4 star ratings but with fewer high ratings overall, indicating a more critical audience for this genre.

4 Matrix Factorization Visualizations

Method 1: Base version without bias

In this section we will discuss the results of base implementation without bias. Let m and n be the number of users and movies. In our dataset, we have that $m = 943$ and $n = 1682$. Let Y be the $m \times n$ matrix of the movie ratings, where y_{ij} corresponds to user i's rating for the movie j. The goal of a non-biased factorization of Y consists of finding $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{k \times n}$ (where k is the dimension of our latent space)such that y_{ij} is approximated by $u_i^T v_j$ where u_i^T and v_j are the i-th and j-th columns vectors of U and V respectively. In other words, we want

$$Y \simeq U^T V.$$

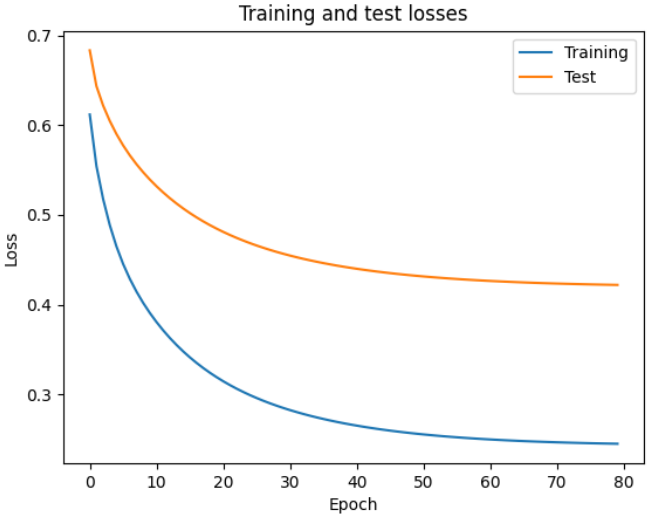

Figure 8: Method 1 train vs test losses for different epochs.

In order to implement this factorization, we used our code from homework 5. Note that we had to transpose all the resulting matrices in order to reuse the code. Figure 8 shows the train and test losses as a function of the number of epoches used. We observe that the test error stagnates at values around 0.45 after about 60 epoches. Hence our choice is to train a model for a toral number of 80 epochs. This seems to be a reasonable comprise between getting low train and test errors without risking to overfit the model.

We conducted our qualitative examination of the models trained mainly on a 2D projection of a custom list of 10 movies that we had intuition about how they should be clustered. The list we have constructed includes the first and second movies from Star Wars series, the first and second Godfather movies, two Amityville movies, Once Upon a Time in America, Pulp Fiction, Reservoir Dogs, and Jackie Brown. Here, obviously, it is expected for movies from the same franchise to be clustered together, and naturally we expect clusters of different franchises to be separate from each other, since Godfather, Star Wars, and Amityville (a horror franchise) can have very different settings and audiences from each other. Both Pulp Fiction and Reservoir Dogs are written and directed by Quentin Tarantino, and both saw a lot of success with a very similar audience and general tone of movies. Jackie Brown and Once Upon a Time in America, as well as Pulp Fiction, Reservoir Dogs, and Godfather movies are crime movies and we would expect to see a relation regarding that.

Figure 9 shows the 2D projection for Method 1 implementation described above, for the custom list of movies (deliverable: Any ten movies of your choice from the MovieLens dataset). It is seen that Star Wars movies are closest to each other, and Godfather movies are closest to each other, but they lack strong clusters where they can be distinguished from other movies at a glance. All the crime movies are positioned away from Star Wars movies and it is possible to speak of a broad crime cluster. Pulp Fiction and Reservoir Dogs are not very close to each other. Amityville movies are also closest to each other and the three franchises of

Star Wars, Godfather, and Amityville are distantly placed from each other.

Figure 10 shows the 2D projection for ten movies that have the best average rating (deliverable: The ten best movies). When compared to other figures with different movie sets, one thing that is noticeable is that the scale of y-axis being smaller. This can be attributed to all movies on this set having very few reviews, all of which are ratings of 5, so their representations could be lacking diversity. Clusters of different genres can't be directly seen. There are six drama movies, two documentaries, one fantasy, and one comedy movie.

Figure 11 shows the 2D projection for ten most popular movies (deliverable: The ten most popular movies). For this set of movies, it is seen that, in fact, Star Wars movies are not clustered together.

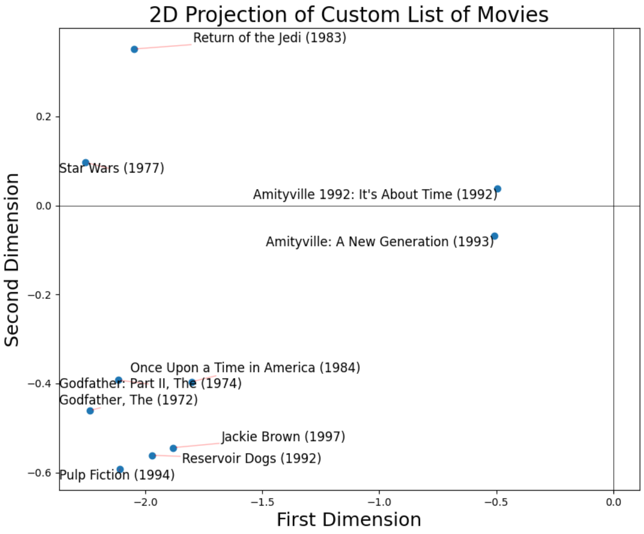

Figure 9: Method 1: 2D projections for ten movies from the MovieLens dataset.

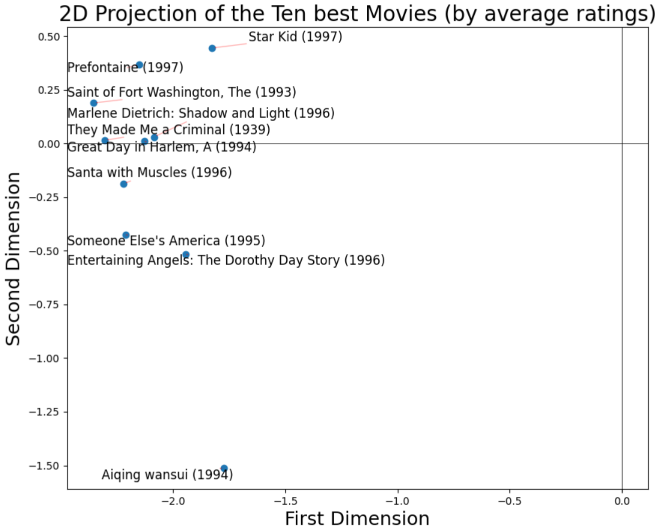

Figure 10: Method 1: 2D projections for the ten best movies (movies with the highest average ratings).

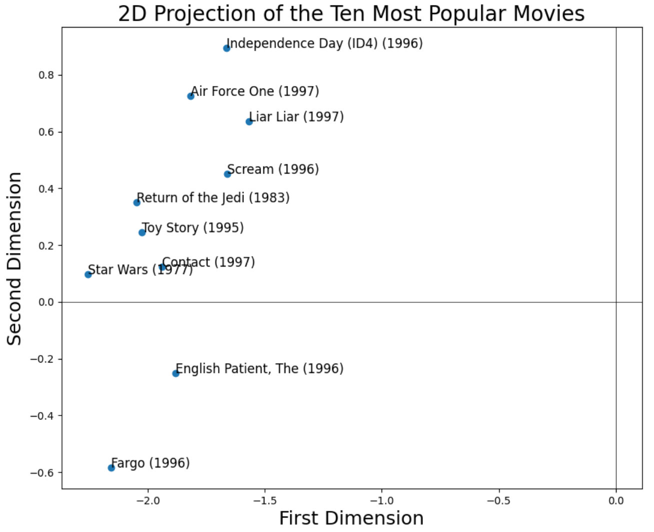

Figure 11: Method 1: 2D projections for the ten most popular movies (movies which have received the most ratings).

Figures 12, 13 and 14 show the 2D projection for three different genres: Crime, Drama and Romance. Here, we note an important observation for the Drama and Romance genres. In fact, our algorithm performs a sub-clustering of these movies in many instances. As an example, consider the case of Desperado (1995) and Legends of the Fall (1994). While both of these movies belong to the Romance category, it turns out that their plots happen in a Western setting, which suggests that matrix factorization can possibly isolate movies with multiple genres at the same time (in this case, Drama and Western). Another example of this happens with the two Drama movies Richard II (1995) and Dead Man Walking (1995). In this case, both of their plots involve a murder case, which means that both of these movies also belong to the Thriller category. Finally, in the case of Crime movies, this phenomenon occurs with the movies Rumble in the Bronx(1995) and Strange(1995) (both of these movies belong to the Action category as well)

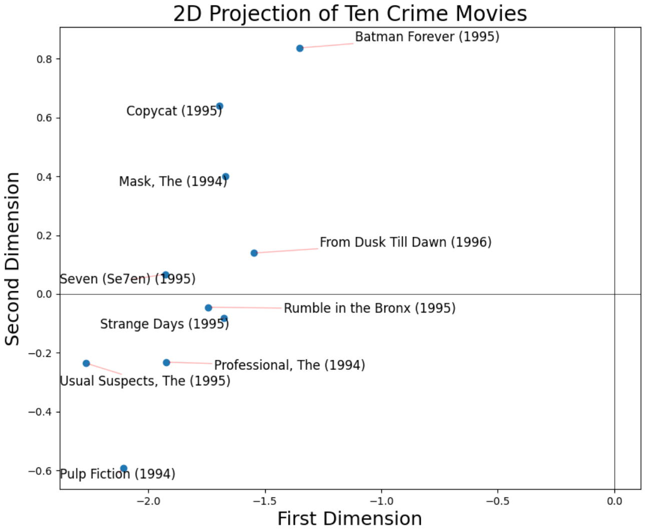

Figure 12: Method 1: 2D projections for ten Crime movies.

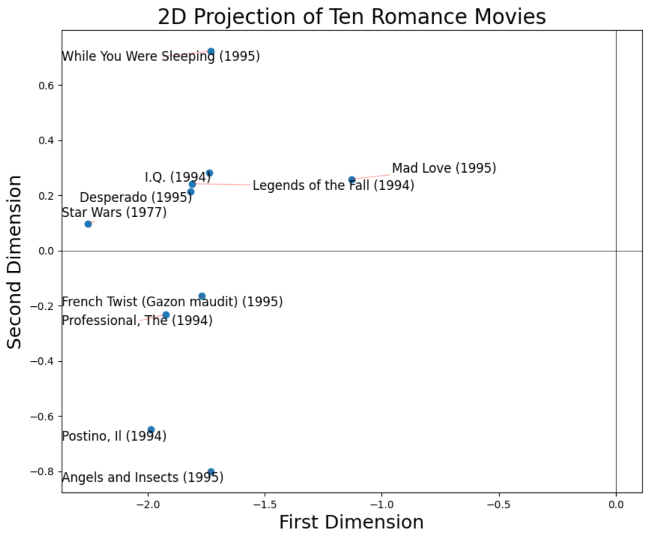

Figure 14: Method 1: 2D projections for ten Romance movies.

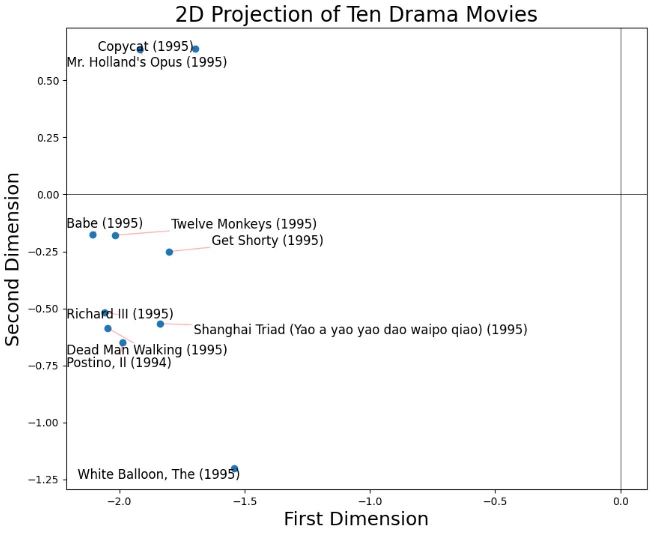

Figure 13: Method 1: 2D projections for ten Drama movies.

Method 2: Bias terms

For the bias term method, we incorporated bias terms a and b for each user and movie, respectively, to model global tendencies of the various users and movies. Following method 1 and ”Matrix Factorization Techniques for Recommender Systems” by Koren et al. [1], we included the bias terms in the loss function as follows:

$$\min_{U,V,a,b} \sum_{(i,j)\in\kappa} (y_{ij} - \mu - a_i - b_j - u_i^T v_j)^2 + \lambda(||U_i||^2 + ||V_j||^2 + a_i^2 + b_j^2) \tag{1}$$

where μ is the global average rating, a_i is the bias term for user i, b_j is the bias term for movie j, and λ is the regularization parameter. The stopping criterion was not used since the training would stop too early. Instead, we used a fixed number of iterations (80) due to the loss function not decreasing after that point. For the bias term method, we implemented 3 different methods: the first method used no regularization, the second method used a regularization parameter of 0.1, and the third method used a regularization parameter of 0.1 with the addition of Learning rate decay. The learning rate decay was implemented by decreasing the learning rate by a factor of 0.96 every epoch.

Bias with no regularization

Method 2, when applied without regularization, resulted in the model starting to overfit as early as the fifth epoch as it can be seen from Figure 15. The model was trained for 80 epochs with a learning rate of 0.01. The 2D projections for the movies had some of the expected clusters but lacked some of the expected properties. For the custom movies set, Star Wars movies and Amityville movies could still be considered as clustered together, however there was still a considerable distance between the movies. Pulp Fiction and Reservoir Dogs were very far apart and only close in terms of the x-axis (Figure 16). This led us to methods that would improve the learning curves to avoid overfitting and possibly get better clusters in the 2D projections.

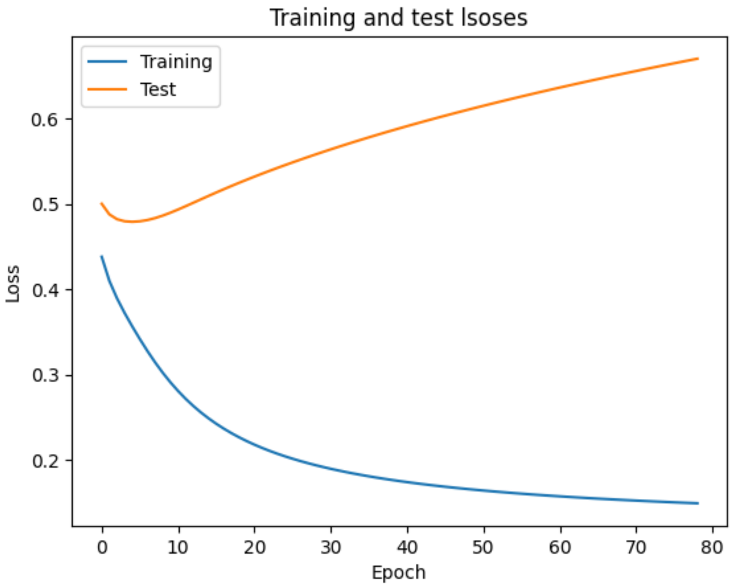

Figure 15: Training and test losses over 80 epochs for Method 2 without regularization

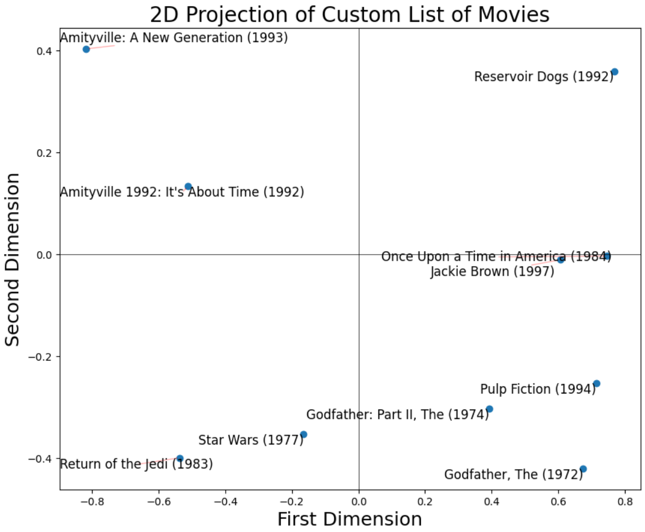

Figure 16: 2D projections for 10 custom movies, for Method 2 without regularization

Bias with regularization

We tried different values for the regularization parameter λ including 0.1 and different values in the perimeter of 0.1. Values larger than 0.1 ended up dominating the loss gradients and values lower than 0.1 proved ineffective in overcoming overfitting, so we chose λ as 0.1.

As expected, the model ended up getting much better test errors, but larger yet more realistic training errors, as it can be seen from Figure 17. The test loss converged around 0.41. The 2D projections were also much better. On Figure 18, it can be seen that Amityville movies and Star Wars movies got closer to each other when compared to the projections for without regularization on Figure 16. Godfather movies also ended up with their own cluster, whereas before they were clustered together with Pulp Fiction. Pulp Fiction and Reservoir Dogs were very closely located this time. It's also notable that Jackie Brown and Once Upon a Time in America have their cluster, both of which are crime movies and feature Robert De Niro, which can be a big factor on people's choice on which movie to watch.

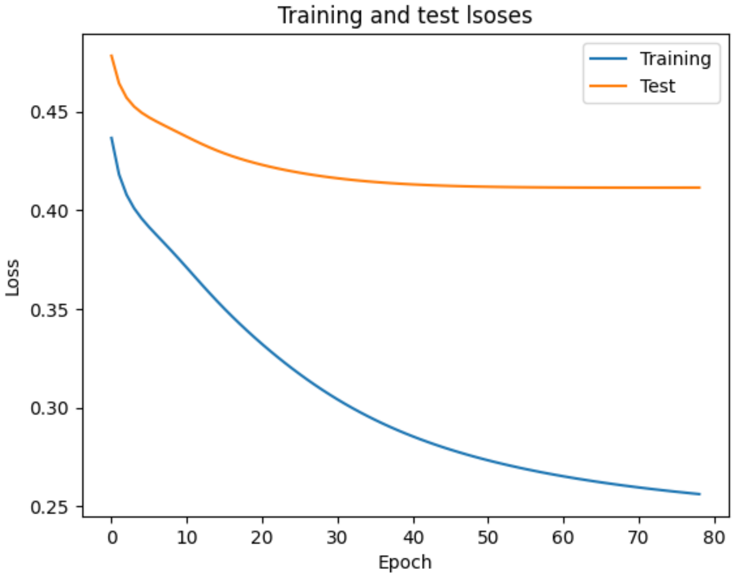

Figure 17: Training and test losses over 80 epochs for Method 2 with regularization, $\lambda = 0.1$

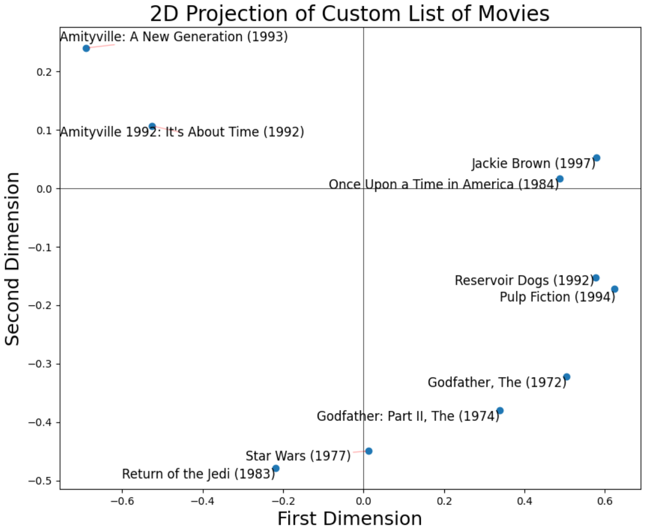

Figure 18: 2D projections for 10 custom movies, for Method 2 with regularization, $\lambda = 0.1$

Bias with regularization and learning rate decay

We tried having a diminishing learning rate because of the fact that the test errors were converging very early on the training when regularization was used (Figure 17), thinking that smaller learning rates after the initial convergence could better fine-tune the model and lead to better local minimums. For this, we used the decay strategy visualized on Figure 19, with a starting value of 0.1 and getting multiplied with 0.96 every epoch.

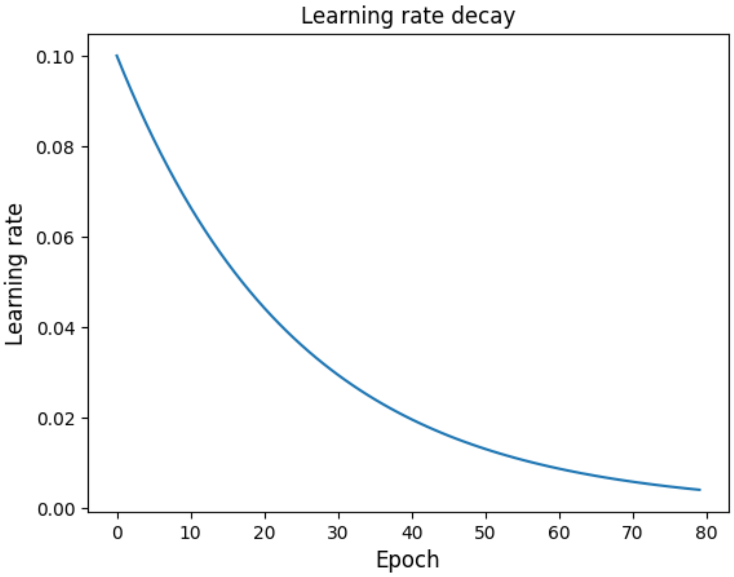

Figure 19: Learning rate decay starting from 0.1

On Figure 20, it can be seen that the test loss still converged near 0.41 with only being slightly better for different random seeds. However, the result on 2D projections were promising. Figure 21 shows that while retaining the relations between Star Wars, God Father and Tarantino movies, Amityville movies got much closer to each other, which are the movies from the series that are only one year apart.

We consider both of the training improvement steps over the non-regularized bias method as a success seeing how the expected clusters became more pronounced at every iteration, while not suffering in terms of test error.

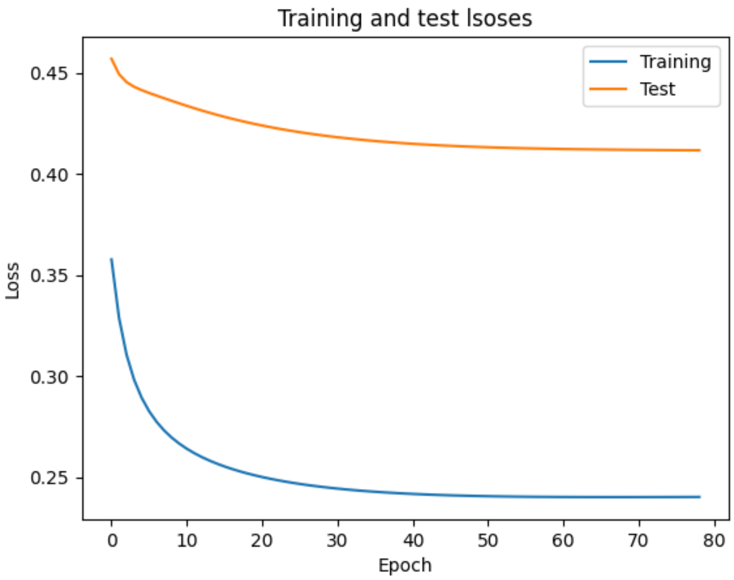

Figure 20: Training and test losses over 80 epochs for Method 2 with regularization, $\lambda = 0.1$ and learning rate decay

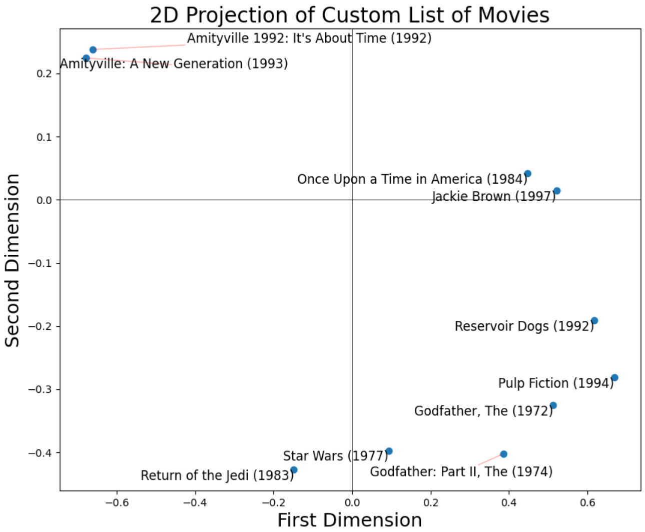

Figure 21: 2D projections for 10 custom movies, for Method 2 with regularization, $\lambda = 0.1$ and learning rate decay

On Figure 22, it is seen that Star Wars and Return of the Jedi are the closest to each other, which was the first sanity check of 2D projections of this set. Independence Day and Air Force One are both highly action oriented movies and can be considered as less serious. This less seriousness is seen to vaguely diminish over the x axis as the value goes from negative to positive, since Liar Liar is a comedy movie, Scream is a horror movie which are both not very well regarded when compared to movies like Star Wars, The English Patient and Fargo. Fargo, and The English Patient have more intense plot elements and have received a better reviews when compared to movies on the left-most movies. Toy Story, while being very popular, also doesn't have much serious elements in it due to its targeted audience. When checked against our figure for 10 custom movie selections, this seriousness across the x-axis vaguely holds, while Once Upon a Time in America or The Godfather being more serious movies than Reservoir Dogs, Reservoir Dogs could still be considered a serious movie, and again, horror movies like Amityville lack seriousness in plot while aiming to effect the viewer with jump-scares.

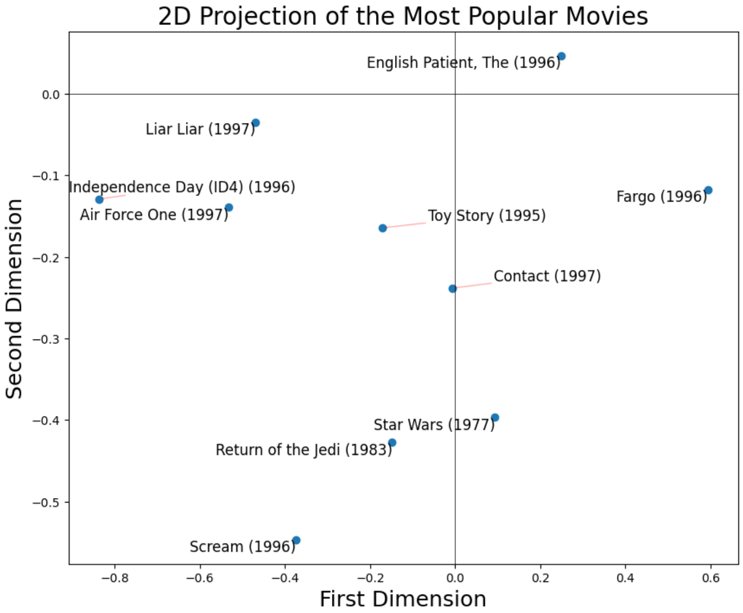

Figure 22: 2D projections for 10 most rated movies, for Method 2 with regularization, $\lambda = 0.1$ and learning rate decay

On Figure 23, the seriousness deduction over x-axis from Figures 21 and 22 again mostly holds, with the right-most movies like Aiqing Wansui, Someone Else's America and Entertaining Angels: The Dorothy Day Story all having weighty subjects. Overall, the orientation of the figure is right-leaning unlike the other figures, however the information of serious movies being liked better can't be deduced because of the lack of numbers for this movies, as stated before for method 1.

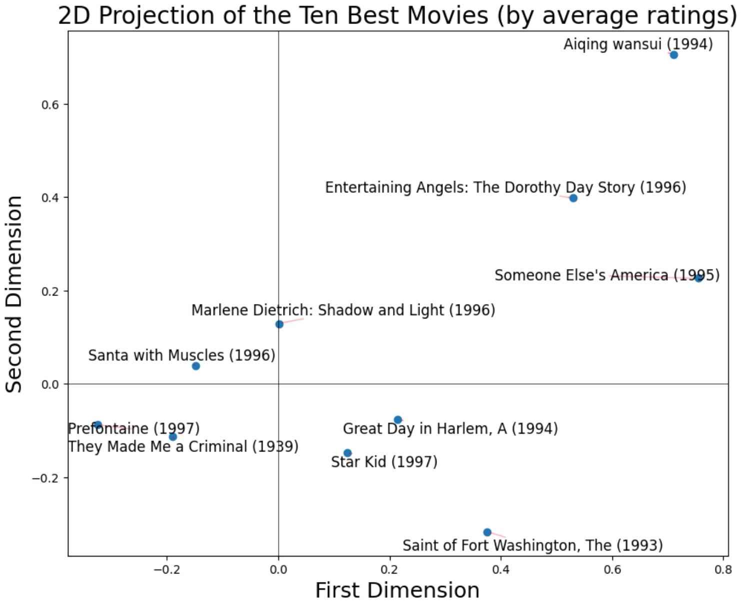

Figure 23: 2D projections for 10 best rated movies, for Method 2 with regularization, $\lambda = 0.1$ and learning rate decay

Figures 24, 25 and 26 show the 2D projection for three different genres: Crime, Drama and Romance. It is seen that out of the three figures, the drama figure's movies are right leaning while for the other figures, the mass-center of the distribution of movies in terms of x-axis is almost central. A relation between this fact, and the drama genre inherently having seriousness could exist. Also, when the romance and crime figures are examined, it can be seen that drama movies mostly fall on positive values of x.

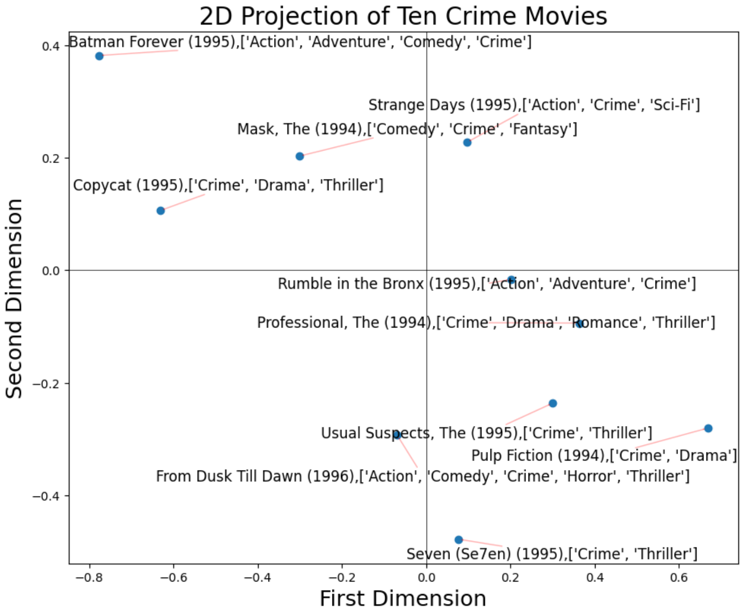

Figure 24: 2D projections for 10 movies of the crime genre, for Method 2 with regularization, $\lambda = 0.1$ and learning rate decay

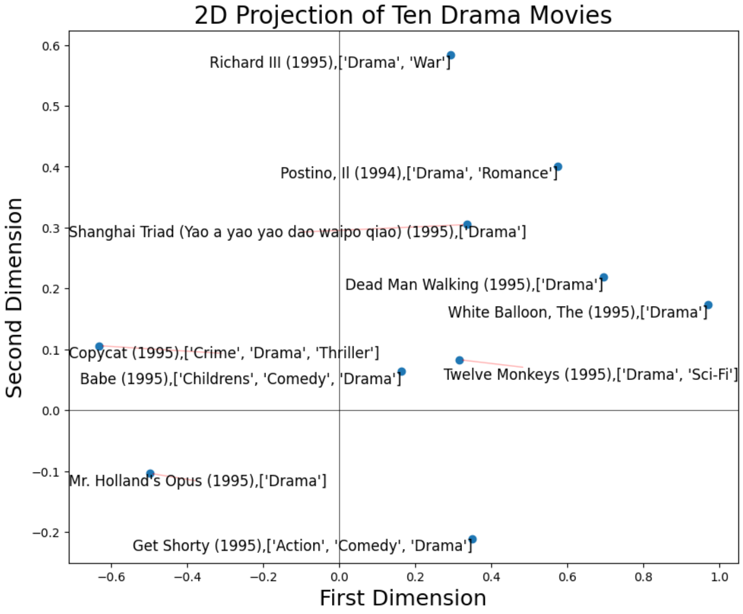

Figure 25: 2D projections for 10 movies of the drama genre, for Method 2 with regularization, $\lambda = 0.1$ and learning rate decay

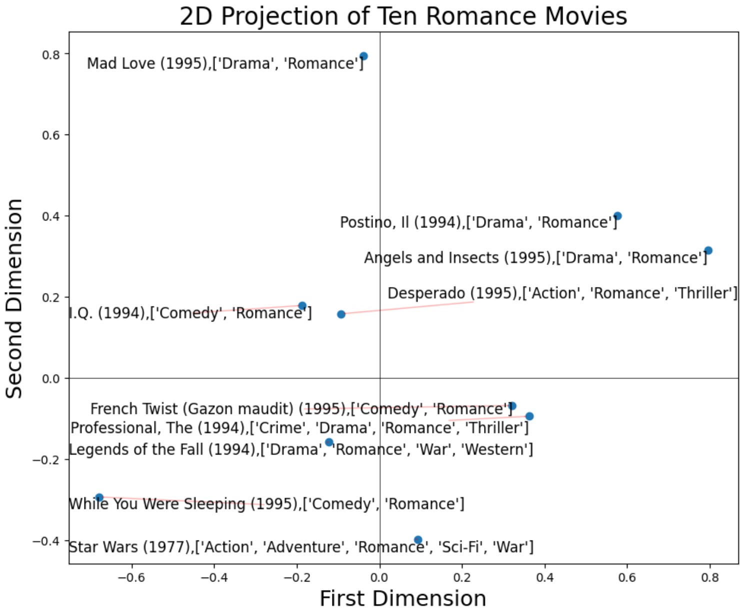

Figure 26: 2D projections for 10 movies of the romance genre, for Method 2 with regularization, $\lambda = 0.1$ and learning rate decay

Method 3: Off-the-shelf implementation

For the off-the-shelf implementation of matrix factorization, the Singular Value Decomposition (SVD) algorithm from the Surprise library was used. The Surprise library's matrix factorization via SVD reduced a user-item rating matrix into two lower-dimensional matrices representing latent user and item factors. The factorization process minimizes reconstruction error between the original matrix and the product of the two latent matrices. The data was prepared by loading the training data into a trainset object and converting the test data into a list of tuples to ensure compatibility with the Surprise framework.

The model's performance was optimized by conducting hyperparameter tuning with grid search, which contained various parameters: the number of epochs (fixed at 300 as in Set 5), the number of latent factors (50, 100), the learning rate (0.01, 0.03, 0.1), and the regularization term (0.01, 0.1). Since a learning rate of 0.3 was used in Set 5, a slightly lower and one magnitude higher learning rate was examined to see the affect. From Set 5, we also know that regularization term of 1 suppresses the model's ability to learn and 0.1 was the best regularization term so that and a magnitude lower was used in the grid search for this parameter. The combination of parameters that led to the lowest test error was epochs=300, latent factors=100, lr=0.01, reg=0.1 (Figure 27).

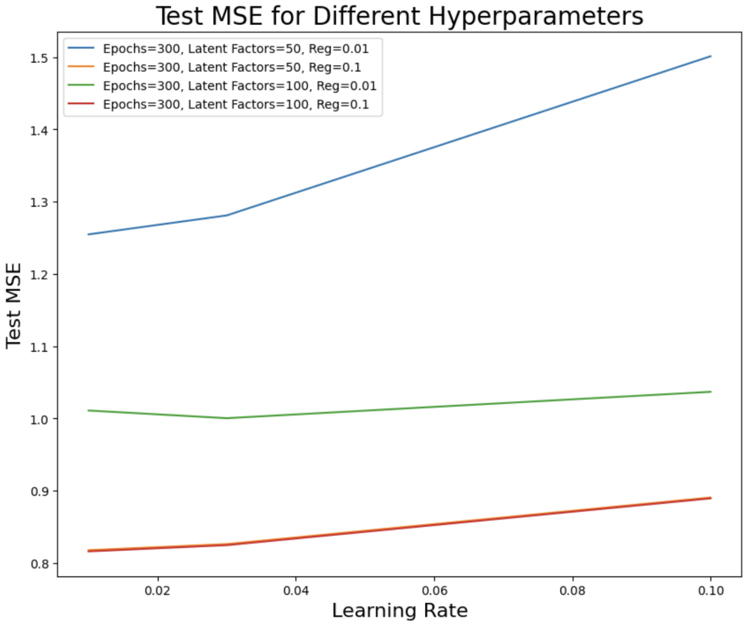

Figure 27: Plot of learning rate versus mean squared error for test set. Shows that training for 300 epochs using 100 latent factors and a regularization term of 0.1 produces the best results.

Below are the matrix factorization visualizations obtained with the Surprise SVD implementation:

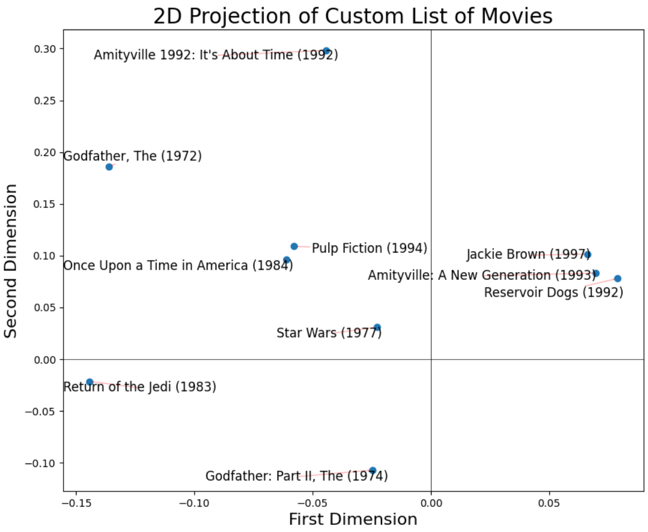

Figure 28: Method 3: Any ten movies of your choice from the MovieLens dataset.

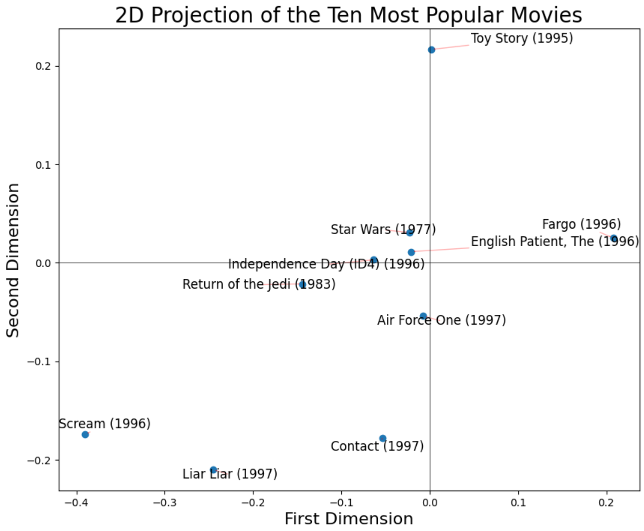

Figure 29: Method 3: The ten most popular movies (movies which have received the most ratings).

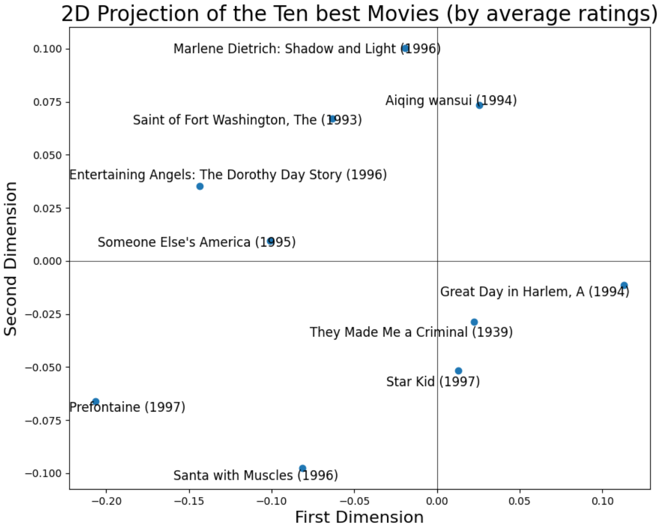

Figure 30: Method 3: The ten best movies (movies with the highest average ratings).

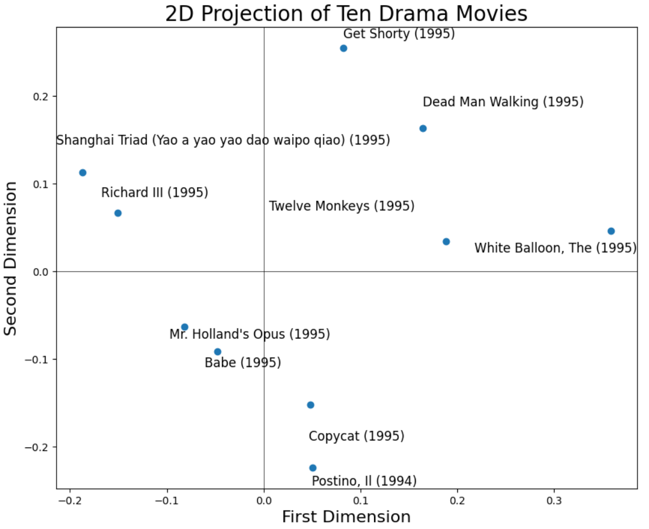

Figure 31: Method 3: Ten movies from Drama genre.

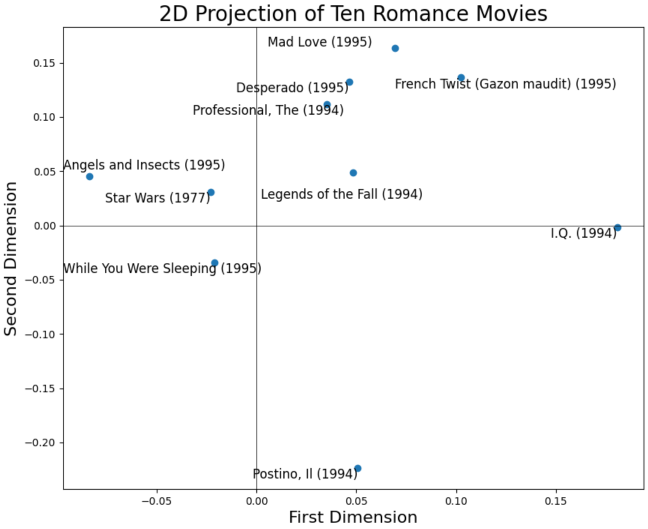

Figure 32: Method 3: Ten movies from Romance genre.

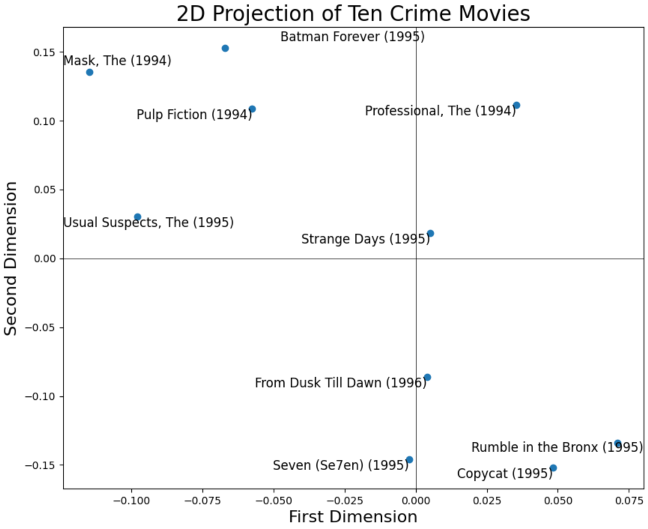

Figure 33: Method 3: Ten movies from crime genre.

Observation of Results:

In general, a few clusters are noticeable in the different Figures 17 -21, but it mostly seems random. This might be due to the possibility that latent factors derived by the matrix factorization may be capturing abstract and complex attributes that do not translate well into only two dimensions. Perhaps the structure of the data would be more interpretable with more dimensions. For instance, in Figure 28, Jackie Brown and Reservoire Dogs are clustered together, which makes sense as they have similar themes like crime and morality and share the same director, so they have similar narrative techniques. However, it wouldn't make much sense for Amityville: A New Generation to be clustered with those two as it is a completely different genre. Since they appear in a cluster in the 2D visualization, this would mean that the same users rated these movies similarly or they have underlying qualities discerned by the algorithm.

Comparing the visualization of the most popular movies (Figure 29) and the best movies (Figure 30), we see that the most popular movies seem to cluster near the origin, whereas the ten best movies are spread away from the origin. Regarding the visualizations of ten Drama (Figure 31), ten Romance (Figure 32), and ten Crime genres (Figure 33), the ten Drama and ten Crime movies appear spread apart from each other, and the ten Romance movies seem to cluster closer to the axis of the first dimension, above second dimension > 0.

5 Comparison of Methods

To facilitate the comparison, let's assess the sub-clustering efficacy of the three distinct methods within the Drama genre. Recall that under Method 1, we observed a Drama-Crime cluster consisting of Richard II, Shanghai Triad and "Dead Man Walking". This cluster was not "pure" in the sense that Postino, II (which is not a crime movie) has also been grouped around these three Drama-Crime movies. On the other hand, it is reasonable to expect the algorithm to create a sub-cluster consisting of Babe and Mr. Holland's Opus from other films. However, we do not observe this particular sub-clustering under Method 1.

Adding a bias to our model in Method 1 undo some of the clusters we observed earlier. The undoing of some of the clusters could be attributed to Method 2 taking into account the user's interactions with the global average rating and the movie's interactions with the global average rating. Both interactions are taken individually, unlike Method 1. Thus the respective relationships between the movies shift for the 2 dimensions. Additionally, under Method 2 we observe a new phenomenon: Drama-Crime like movies all share a higher y-coordinate (e.g Richard III and Dead Man Walking) while Family friendly movies are mapped into lower y-coordinates points (e.g Babe and Mr. Hollands's Opus). However, we still have some outliers that do not match this pattern (e.g instance Postino, II). This seems to be fixed by Method 3 since there is a full y-axis transition from Drama-Crime movies (Get Shorty, Dead Man Walking, Shaghai Triad, Richard III) to Drama-Thriller movies (Twelve Monkeys and The White Balloon) and finishing with Drama-Family movies (Mr. Hollands Opus and Babe). The last two movies Copycat and Postino, II belong to two different sub-genres (Mystery and Comedy, respectively). Method 3 also uses biases like Method 2. The improvements that we see from Method 2 to 3 could be attributed to the increased number of latent factors. Method 2 has $k = 20$ latent factors while Method 3 has $k = 100$ latent factors. Thus Method 3 is able to capture more relationships derived from the relationship between the users, movies, and global average ratings.