Movie Recommendation System

Bing Syuan Wang
Department of ECE
University of Washington
wben@uw.edu

Michelle Liu
Department of ECE
University of Washington
mcliu@uw.edu

Abstract

We create a movie recommendation system based on personal habits. We proposed serval methods, including ratings, overview, features, logistic regression, and MLP. The result is positive, and the recommended lists are credible.

GitHub url: https://github.com/SyXuan/EE-596-A-Su-22-Final-Project

1 Short introduction

Many OTT (Over-the-top) media services provide recommendation system services, including Netflix, HBOMAX, Disney+, Hulu, etc. The recommendation is usually based on several factors, such as release date, view, popularity genre, or personal habits. Sometimes, the system recommendation is also affected by the manufacturer's sponsorship or the company's other factors. Therefore, we want to create a system based on personal behavior, such as watching history.

2 Technical descriptions, including algorithm

In this project, we choose the TMDB (The Movie Database, https://www.themoviedb.org/) as our database. Also, we use the field of Title, Genre, Actor, Director and Overview.

2.1 Recommend by Rating

We want to use IMDB's formula to calculate the data result of TMDB. IMDB's rating is based on this formula as follows,

$$IMDb\ Rating = \frac{v}{v+m} \times R + \frac{m}{v+m} \times C$$

R = average for the movie (mean); v = number of votes for the movie; m = minimum votes required to be listed; C = the mean vote across the whole report

2.2 Recommend by Overview

First, we use the overview from the TMDB database. Second, using the TFI-IDF to get a vector matrix of 4803*20987. Third, we do cosine similarity to compare the score between each movie. After we get the score, the highest one would be our recommendation.

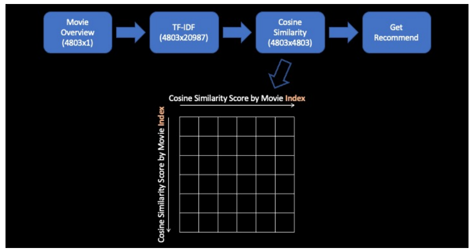

Figure 1: Recommend by Overview

2.3 Recommend by Title, Cast, Director, Keywords, Genres, Overview

We turn Title, Cast, Director, Keywords, Genres, and Overview to an array list, do the CountVectorizer, Cosine Similarity, and recommend.

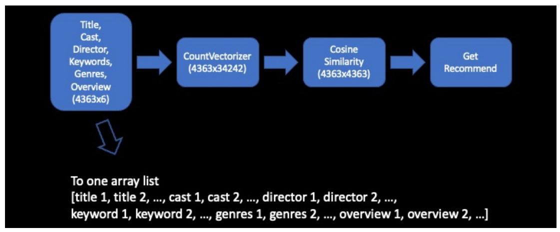

Figure 2: Recommend by Title, Cast, Director, Keywords, Genres, Overview

2.4 Recommend by Logistic Regression

We take CountVectorizer as our X. Since The data do not have a label. Therefore, we create Y by ourselves. Then, we take the second highest cosine similarity score as our Y.

The result of accuracy is only 0.13. Although the accuracy is not high, this method is feasible.

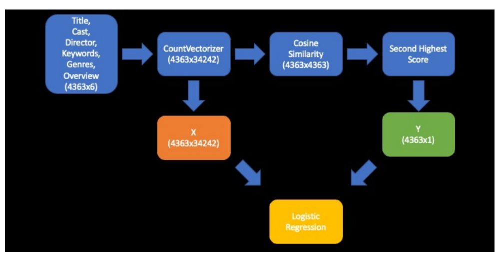

Figure 3: Recommend by Logistic Regression

2.5 Recommend by MLP (Multi-layer perception)

MLP (Multi-layer perception) uses neural networks to do deep learning.

The source of the first layer is CountVectorizer. It has 34242 features. There are four hidden layers in the middle, and all their activation functions are ReLU. The final layer is the output layer. Its activation function is Softmax.

The output layer has 4363 output classes from our 4363 movies, totaling 19840907 parameters.

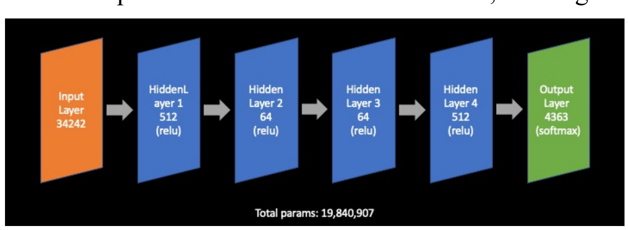

Figure 4: Recommend by MLP (Multi-layer perception)

3 Experimental results

3.1 Recommend by Rating

The result of the top ten TMDB databases which use the IMDB's formula, and at the column on the right side, which is the actual ranking of IMDB.

The two databases have repetitive ranking when using the same formula, which means the IMDB's formula is credible.

index	title	release_date	score_IMDB	vote_average	IMDB_rating
1881	The Shawshank Redemption	1994/9/23	8.05	8.5	1
662	Fight Club	1999/10/15	7.93	8.3	12
65	The Dark Knight	2008/7/16	7.92	8.2	3
3232	Pulp Fiction	1994/10/8	7.9	8.3	8
96	Inception	2010/7/14	7.86	8.1	13
3337	The Godfather	1972/3/14	7.85	8.4	2
95	Interstellar	2014/11/5	7.8	8.1	27
809	Forrest Gump	1994/7/6	7.8	8.2	11
329	The Lord of the Rings: The Return of the King	2003/12/1	7.72	8.1	7
1990	The Empire Strikes Back	1980/5/17	7.69	8.2	15

Figure 5: The result of the rating

3.2 Recommend by Overview

We use “The Lord of the Rings: The Fellowship of the Ring” as the original movie. The following picture shows the top ten recommendations.

The top five are “The Lord of the Rings” series, and the rest is recommended based on some keywords.

The Lord of the Rings: The Fellowship of the Ring

index	title	cosine_score	note
262	The Lord of the Rings: The Fellowship of the Ring	1	
98	The Hobbit: An Unexpected Journey	0.195158	Series
22	The Hobbit: The Desolation of Smaug	0.180959	Series
329	The Lord of the Rings: The Return of the King	0.166745	Series
330	The Lord of the Rings: The Two Towers	0.147305	Series
1933	Underclassman	0.124586	"young", "ring" in overview
4185	Sublime	0.119095	"ring" in overview
2344	Raging Bull	0.117544	"ring" in overview
1616	What's the Worst That Could Happen?	0.117095	"ring" in overview
3714	Exiled	0.112993	"ring" in overview
1438	Krull	0.094786	"protect" in overview

Figure 6: The result of the overview

3.3 Recommend by Title, Cast, Director, Keywords, Genres, Overview

The input movie is “The Lord of the Rings: The Fellowship of the Ring”. We can know the relationship from the note part that shows why it is recommended.

The Lord of the Rings: The Fellowship of the Ring

index	title	cosine_score	note
259	The Lord of the Rings: The Fellowship of the Ring	1	titles x 4, cast x 5, director x 1, keywords x 5, genres x 3, overview x 33
326	The Lord of the Rings: The Two Towers	0.42956	titles x 2, cast x 3, director x 1, keywords x 4, genres x 3, overview x 4
325	The Lord of the Rings: The Return of the King	0.396438	titles x 2, cast x 3, director x 1, keywords x 3, genres x 3, overview x 3
95	The Hobbit: An Unexpected Journey	0.395130	cast x 2, director x 1, keywords x 4, genres x 3, overview x 4
22	The Hobbit: The Desolation of Smaug	0.339932	cast x 1, director x 1, keywords x 4, genres x 2, overview x 2
19	The Hobbit: The Battle of the Five Armies	0.339343	cast x 1, director x 1, keywords x 4, genres x 3, overview x 3
1580	What's the Worst That Could Happen?	0.195311	genres x 1, overview x 2
1407	Krull	0.147844	genres x 3, overview x 2
2264	Raging Bull	0.133352	overview x 3
1885	Underclassman	0.130677	genres x 1, overview x 2
61	Jupiter Ascending	0.126295	cast x 1, genres x 3, overview x 1

Figure 7: The result of Title, Cast, Director, Keywords, Genres, Overview

3.4 Logistic Regression

We input two of “The Lord of the Rings” movies, and it recommends “The Lord of the Rings: The Two Towers” movie.

```
getRecBySklearnModel([
    'The Lord of the Rings: The Fellowship of the Ring’,
    'The Hobbit: An Unexpected Journey’,
    ], model_lr)
```

Figure 8: The code of logistic regression first example

```
Predict index: 326
Predict movie title: The Lord of the Rings: The Two Towers

Diff with input movie "The Lord of the Rings: The Fellowship of the Ring":
titles x 2 cast x 3 director x 1 keywords x 4 genres x 3 overview x 4

Diff with input movie "The Hobbit: An Unexpected Journey":
cast x 1 director x 1 keywords x 3 genres x 3
```

Figure 9: The result of logistic regression first example

We input three random movies, and the predicted movie is based on Genres. Here, the genre is drama.

```
getRecBySklearnModel(
    df_new.title.sample(n=3, random_state=10).tolist(),
    model_lr)
```

Figure 10: The code of logistic regression second example

```
Predict index: 4035
Predict movie title: Love Me Tender

Diff with input movie "The Passion of the Christ":
genres x 1 overview x 1

Diff with input movie "A Madea Christmas":
genres x 1

Diff with input movie "Coal Miner's Daughter":
genres x 2
```

Figure 11: The result of logistic regression second example

We input “The Matrix”, “The Matrix Reloaded”, “Speed”, and “Point Break” as four movies. It recommends “The Matrix Revolutions” because they are all performed by 'Keanu Reeves.'

```
getRecBySklearnModel([
    'The Matrix’,
    'The Matrix Reloaded’,
    'Speed’,
    'Point Break’
    ], model=model_lr)
```

Figure 10: The code of logistic regression third example

```
Predict index: 120
Predict movie title: The Matrix Revolutions

Diff with input movie "The Matrix":
titles x 1 cast x 4 director x 2 keywords x 4 genres x 2

Diff with input movie "The Matrix Reloaded":
titles x 1 cast x 4 director x 2 keywords x 3 genres x 4 overview x 6

Diff with input movie "Speed":
cast x 1 genres x 2 overview x 1

Diff with input movie "Point Break":
genres x 2 overview x 1
```

Figure 10: The code of logistic regression third example

3.4 MLP (Multi-layer perception)

The Loss and accuracy are shown in Figure 11. Training is getting better, but validation is not.

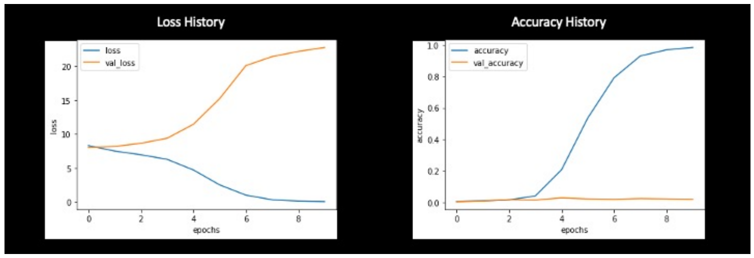

Figure 11: Loss history and accuracy history

We input two of “The Lord of the Rings” movies. The result is “The Hobbit”. However, the result is different from logic regression, but they come from the same series.

```
getRecByMLP([
    'The Lord of the Rings: The Fellowship of the Ring',
    'The Hobbit: An Unexpected Journey',
    ], model_mlp)
```

Figure 12: The code of MLP first example

```
Predict index: 22
Predict movie title: The Hobbit: The Desolation of Smaug

Diff with input movie "The Lord of the Rings: The Fellowship of the Ring":
cast x 1 director x 1 keywords x 4 genres x 2 overview x 2

Diff with input movie "The Hobbit: An Unexpected Journey":
titles x 1 cast x 3 director x 1 keywords x 3 genres x 2 overview x 5
```

Figure 13: The result of MLP's first example

We input three random movies for the MLP model, which considers not only genres but also uses the keywords of overview.

```
getRecByMLP(
    df_new.title.sample(n=3, random_state=10).tolist(),
    model_mlp)
```

Figure 14: The code of MLP second example

```
Predict index: 3851
Predict movie title: Sherrybaby

Diff with input movie "The Passion of the Christ":
genres x 1 overview x 1

Diff with input movie "A Madea Christmas":
genres x 1 overview x 1

Diff with input movie "Coal Miner's Daughter":
genres x 1
```

Figure 12: The result of MLP's second example

We input “The Matrix”, “The Matrix Reloaded”, “Speed”, and “Point Break” as four movies. It recommends “The Matrix Revolutions”, and the same as Logistic Regression because they are all performed by 'Keanu Reeves. '

```
getRecByMLP([
    'The Matrix',
    'The Matrix Reloaded',
    'Speed',
    'Point Break'
    ], model=model_mlp)
```

Figure 12: The code of MLP's third example

```
Predict index: 120
Predict movie title: The Matrix Revolutions

Diff with input movie "The Matrix":
titles x 1 cast x 4 director x 2 keywords x 4 genres x 2

Diff with input movie "The Matrix Reloaded":
titles x 1 cast x 4 director x 2 keywords x 3 genres x 4 overview x 6

Diff with input movie "Speed":
cast x 1 genres x 2 overview x 1

Diff with input movie "Point Break":
genres x 2 overview x 1
```

Figure 12: The result of MLP's third example

4 Discussion of results, strengths/weaknesses, what worked, what didn't

The advantage is that this model does not need to be labeled in advance. Our target is generated by the program. Also, others recommendation systems in the reference data are recommended by the title of the movie as an index, but our system can use all features. It can generate predicted features by itself, so it can be used more flexibly for films that are not limited to the database

The disadvantage is that because there are no labeled targets, it is difficult to judge whether our model is good or bad.

5 Future work

We have several expectations of our future work. First, we want to add history weight to the model, and the standard would be the closing time, the higher weight. Secondly, adding weight based on the feature affects the final score. Third, the results output of MLP are all probabilities; traditionally, we would only get the highest scores, the predicted movie, of the classification results. Therefore, we would want to add multiple recommendations in the future. Fourth, the accuracies of logistic regression and MLP are not ideal because we take cosine similarity as y, but the result would not be the same as Y. Thus, we would like to find a better solution to solve this problem.

References

[1] Sajal Halder & A.M. Jehad Sarkar & Young-Koo Lee. (2012) Movie Recommendation System Based on Movie Swarm. *2012 Second International Conference on Cloud and Green Computing*

[2] P. Kirubanantham & A. Saranya & D. Senthil Kumar. (2021) Convolutional Recommended Neural Network system based on user reviews for movies. *2021 4th International Conference on Computing and Communications Technologies (ICCCT)*

[3] Yeo Chan Yoon & Jun Woo Lee. (2018) Movie Recommendation Using Metadata Based Word2Vec Algorithm. *2018 International Conference on Platform Technology and Service (PlatCon)*