TARC
TUNKU ABDUL RAHMAN
UNIVERSITY COLLEGE
BEYOND EDUCATION

BACS3074 ARTIFICIAL INTELLIGENCE

202105 Session, Year 2021/22

Assignment Documentation

Full Name: Tay Xua Hao
Student ID: 21WMR04790
Programme: Bachelor Degree in Data Science
Tutorial Class: 2
Project Title: Recommender System
Module In-Charged:Matrix Factorization

Other team members' data

No	Student Name	Module In Charge
1	Tay Xue Hao(self)	Matrix Factorization
2	Khoo Chyi Ze	Item-based
3	Lee Ming Yi	User-based

Lecturer: Dr. Goh Ching Pang	**Tutor : Ms. Ashikin**

Deadline: 12th Sept 2021 (Week 12, Sunday, turn in to Google Classroom before 11.59pm)

1. Introduction

1.1. Problem Background

The movie recommender system is a system that introduces and recommends movies that a user might like to the user. The movie recommendation system requires a large amount of data to accurately recommend a movie to the users. The dataset should include basic information like userId, movieId, ratings, genres, tags. The problem lies in this study is that there are various types of calculations and algorithms that are used to implement and develop this system. Our studies require us to explore and analyze how those algorithms are used and what is the best to use in the system. Other than that, there are limited datasets that can be used for producing the result we are looking for. Some datasets can be very sparse where they are missing in certain values like ratings when a user to movie utility matrix is formed.

In this assignment, I am in charge of the Matrix Factorization module. Matrix Factorization is a collaborative filtering technique that solves the sparsity problem which is a problem faced by the user and item based technique. I will build a neural network model with Adam optimizer to do Matrix Factorization.

1.2. Objectives/Aims

The objective of this assignment is to improve my understanding and knowledge about the recommender system which is a core part of Artificial Intelligence. By understanding the concept of the recommender system, we will be able to build a model that can accurately predict the movies that a user might like. We aim to learn the concept enough to be able to apply it in various applications other than just for a movie dataset. Besides, we can also learn how to preprocess data to make sure the result for the recommender system is accurate and reliable.

1.3. Motivation

Due to the pandemic, people have been stuck at home for a very long time. As a result, the number of paid netflix subscribers have increased about 60 Million from 2018 to 2020. This situation inspired us to build a movie recommendation system that can improve user satisfaction. Other than that, this work can also be applied in different applications which are similar. For example, a song recommender system or product recommender system. The major motivation for me to complete this project is to master the art of artificial intelligence and develop a system that can change our life in the future with my knowledge.

1.4. Timeline/Milestone

Milestones	**Goals**	**Duration**
Planning	To organize what to do	5 Jul - 11 Jul 2021 (Week 3)
Gather information	Search for reference to use for the following project	12 Jul - 18 Jul 2021 (Week 4)
Dataset Collection	Finding a dataset that is going to used in the project	19 Jul - 25 Jul 2021 (Week 5)
Implementation 1	Search some coding references and implement it into my own project	26 Jul - 1 Aug 2021 (Week 6)
Implementation 2	Continue implementing until everything is working properly	2 Aug - 15 Aug 2021 (Week 7, 8)
Dataset Display Research	Searching for ways on showing the dataset on different display format	16 Aug - 22 Aug 2021 (Week 9)

Implementation 3	Implementation different dataset display formats	23 Aug - 5 Sep 2021 (Week 10, 11)
Finalization	Finalize the overall project and documents	6 Sep - 12 Sep 2021 (Week 12)
Submission	Submit the codes and documents	12 September 2021

2. Research Background

2.1. Background of the applications

Recommender system is a type of AI that is able to filter information and data by using calculations predictions on users information. Recommender systems have been used by many companies to boost their sales. Many industries and applications such as the online shopping applications, online food delivery applications and especially in the social media applications. Most of the companies built and applied a recommender system to help enhance their user's experience One company that rely very heavily on recommender systems is Netflix. Netflix uses dozens of algorithms to make more personalized recommendations. Netlifx believed that their recommender systems had saved $1Billion per year.

2.2. Analysis of selected tool with any other relevant tools

Tools comparison	**Remark**	**Selected tool : Python Libraries(scikit-learn, numpy, pandas, tensorflow, keras)**	**Recombee**	**Apache Mahout**
Type of license and open source license	State all types of license	- BSD 3 clause license - New BSD licence - Liberal BSD license - Apache License 2.0 (tensorflow) MIT license(keras)	MIT Licence 3.1.0	Apache license 2.0
Year founded	When is this tool being introduced?	Scikit-learn : 2007 Numpy : 2006 Pandas : 2008 Tensorflow : 2015 Keras : 2015	2015	2009
Founding company	Owner	Tensorflow: Google brain Team	Recombee	Apache Software Foundation
License Pricing	Compare the prices if the license is used for development and business/commercialization	Free	Free	Free
Supported features	What features that it offers?	Scikit : Provides splitting of data into training and testing sets, calculates MSE, and provides ways	Provide recommendation engine, personalized	Provides a framework for tasks such as data mining on large

		to normalize your data. Numpy : Provides many mathematical operations, interactive and intuitive features.NumPy arrays provide an effective way to store similar data sets. Pandas : Provides better visualization of data, forming pivot tables, reshaping datasets. Tensorflow : Provides optimizers for neural networks such as Adam and SGD. Keras : Provides API to define deep learning models and support evaluation of such models.(Dense, Flatten, Embedding, models, etc)	content, real-time analysis, and quick and easy integration	amounts of data. It also allows users to quickly and effectively analyze large amounts of data.
Common applications	In what areas this tool is usually used?	All these packages and libraries are commonly used for machine learning and deep learning models	Product recommendation system	Product recommendation system
Customer support	How the customer support is given, e.g.	Through online communities, chatrooms, phone calls, email.	Phone, email support, proprietary	Online communities, phone, and email services

	proprietary, online community, etc.			
Limitations	The drawbacks of the software	Scikit : Cannot do automatic Machine Learning and Deep Learning pipelines reliably. Numpy : Very hard for a beginner to learn and insertion as well as deletion process is very costly as data is stored in contiguous memory. Pandas : Have very complex syntaxes and it is only useful for 2D matrices not 3D. Tensorflow :Lacking in terms of speed and usage and requires GPU to run faster. Keras : Quite difficult to detect errors and the root cause of it.	Have limited support for specific applications	It doesn't provide clear visualization and little support for scientific features.

2.3. Justify why the selected tool is suitable

In this project, I used Python libraries which are pandas, numpy, scikit-learn, and libraries, tensorflow, and keras. Pandas, numpy, and scikit-learn do not require any installation but they provide many useful features for me to build my model. Compared to other tools, they are much easier to use as well. To use these packages I just need to import them. For the libraries, tensorflow and keras, they are the best tools for deep learning purposes. They are easy to use and come with a lot of different customization for your model. For example, the dropout, flatten, embedding layers. To use these libraries, I only needed to install them using the code “pip install keras”, “pip install tensorflow” , which is very convenient. Other than that, there are many online tutorials and guides on how to implement these libraries in my recommender system as they are very popular and widely used libraries.

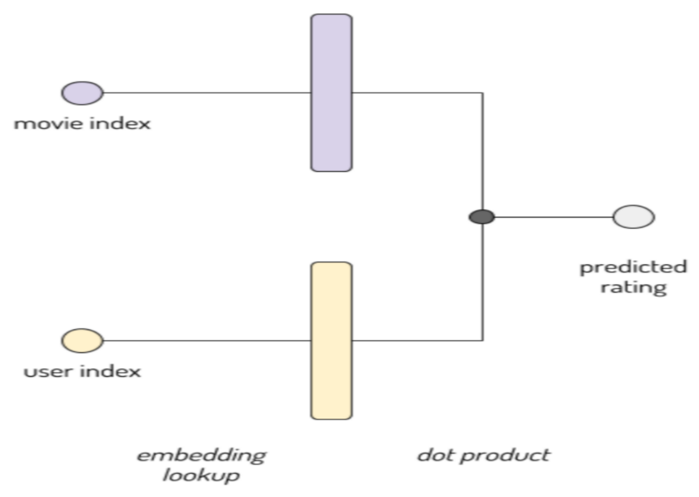
3. Methodology

3.1. Description of dataset

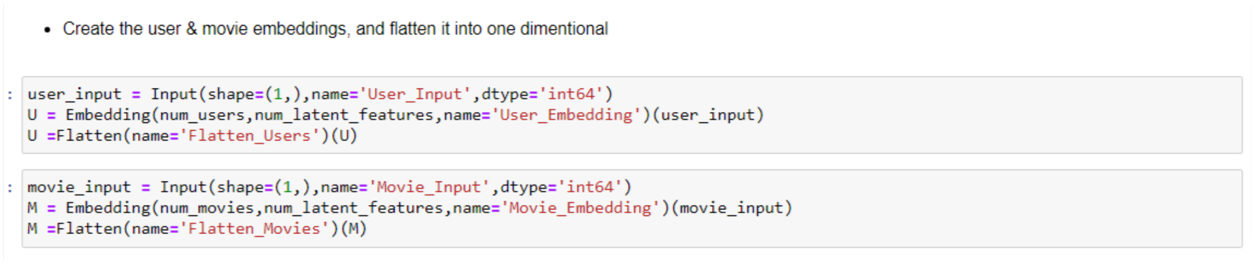
We are using the MovieLens dataset provided by GroupLens. GroupLens have provided us with 3 important datasets in the form of csv files, which are ratings.csv, movies.csv, tags.csv. The ratings dataset contains data about the userId and the ratings given to which movieId. The movies dataset contains data about the movieId, the movie title and the genre of the movies. Finally, the tags dataset contains data about userId, movieId, and the tag given. However I will only use the ratings and movies dataset.

3.2. Applications of the algorithm(s)

The way Matrix Factorization works is that we will use the user to movie matrix and split the big matrix into two smaller matrices, which are the user and movie matrixes. One dimension of the matrix will be the user or movie index, and the other will be the latent features that the Embedding layer from keras will try to find. The features are usually not human interpretable, but we sometimes classify them as how much a user likes a certain genre of movie, and how much a movie contains elements of certain genres. Using the dot product of these two matrices after converting them into one dimensional, we can get the predicted ratings. Below is an image representation of how it works.



The way I implemented my Matrix Factorization is by using a neural network model to predict the user's ratings. I will be using Keras Functional API instead of Sequential API as I have multiple inputs. First, I will initialize the input layers which are the movie index and user index. Then I will pass the inputs to two Embedding layers. The movie embedding layer will accept 2 arguments, the number of movies (9724), number of latent factors. The user embedding layer will accept 2 arguments, the number of users(610), number of latent factors. The purpose of this layer is to reduce the user to movie matrix into a matrix of smaller dimensions which is (number of movie/users * number of latent factors). The number of latent factors I used is 100, which means the users and movies will have 100 underlying features that this layer will try to find. After getting both the embeddings, we will flatten both of them using the Flatten layer. This means that I am converting the multidimensional matrix, which are the user and movie embeddings into a 1-dimensional array so that we can use it as an input for the next layer, and it makes training faster.

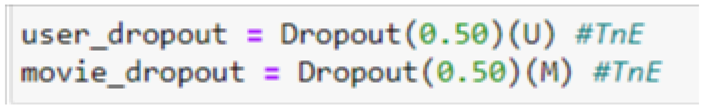


- Create the user & movie embeddings, and flatten it into one dimentional

```
user_input = Input(shape=(1,),name='User_Input',dtype='int64')
U = Embedding(num_users,num_latent_features,name='User_Embedding')(user_input)
U =Flatten(name='Flatten_Users')(U)
```

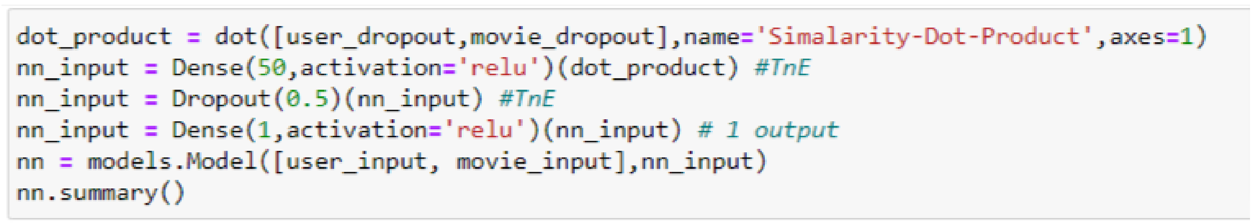
```
movie_input = Input(shape=(1,),name='Movie_Input',dtype='int64')
M = Embedding(num_movies,num_latent_features,name='Movie_Embedding')(movie_input)
M =Flatten(name='Flatten_Movies')(M)
```

Next, the input of the Flatten layers will be passed into two dropout layers, user_dropout, and movie_dropout. Dropout is a regularization technique which is commonly used for avoiding overfitting issues. The way it works is that it will drop certain neurons of the input before passing it to the next layer so that no certain weights are way too large as it cannot rely on one feature. The keep probability I will be using is 0.5 for all the dropout layers. This means the layer will keep 50% of the inputs. A good deep_prob is usually around 0.5 to 0.8.

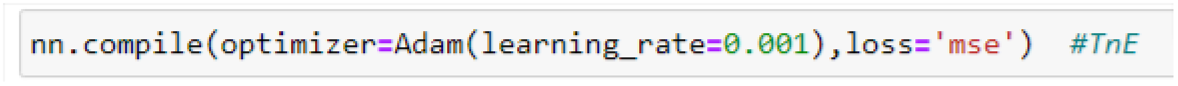


```
user_dropout = Dropout(0.50)(U) #TnE
movie_dropout = Dropout(0.50)(M) #TnE
```

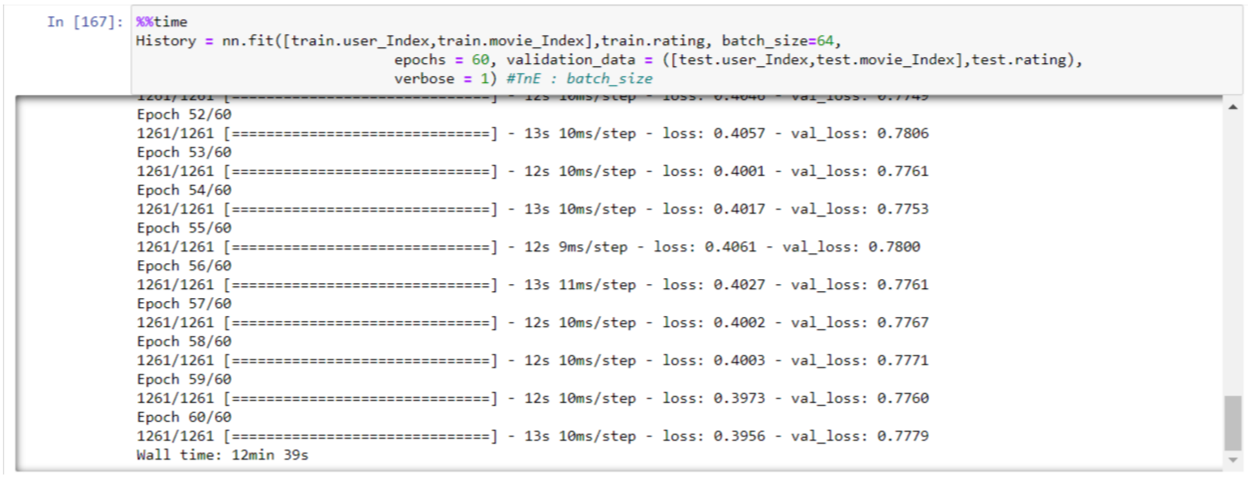
Finally, I will pass the outputs of the dropout layers to a Dot layer. This layer will accept the movie_dropout, and user_dropout as inputs and calculate the dot products of the user and movie. Then the output of the dot layer will be passed into a dense layer with 50 neurons (50 is the value decided based on trial and error), with the ‘relu’ activation function. ReLU is a type of piecewise activation function that directly gives the inputs as output if the input is positive, and output zero if it is not. Then the outputs will be passed into another dropout layer and finally be passed into a final dense layer with ‘relu’ activation function and one neuron, which is the final output, predicted rating.


```
dot_product = dot([user_dropout,movie_dropout],name='Simalarity-Dot-Product',axes=1)
nn_input = Dense(50,activation='relu')(dot_product) #TnE
nn_input = Dropout(0.5)(nn_input) #TnE
nn_input = Dense(1,activation='relu')(nn_input) # 1 output
nn = models.Model([user_input, movie_input],nn_input)
nn.summary()
```

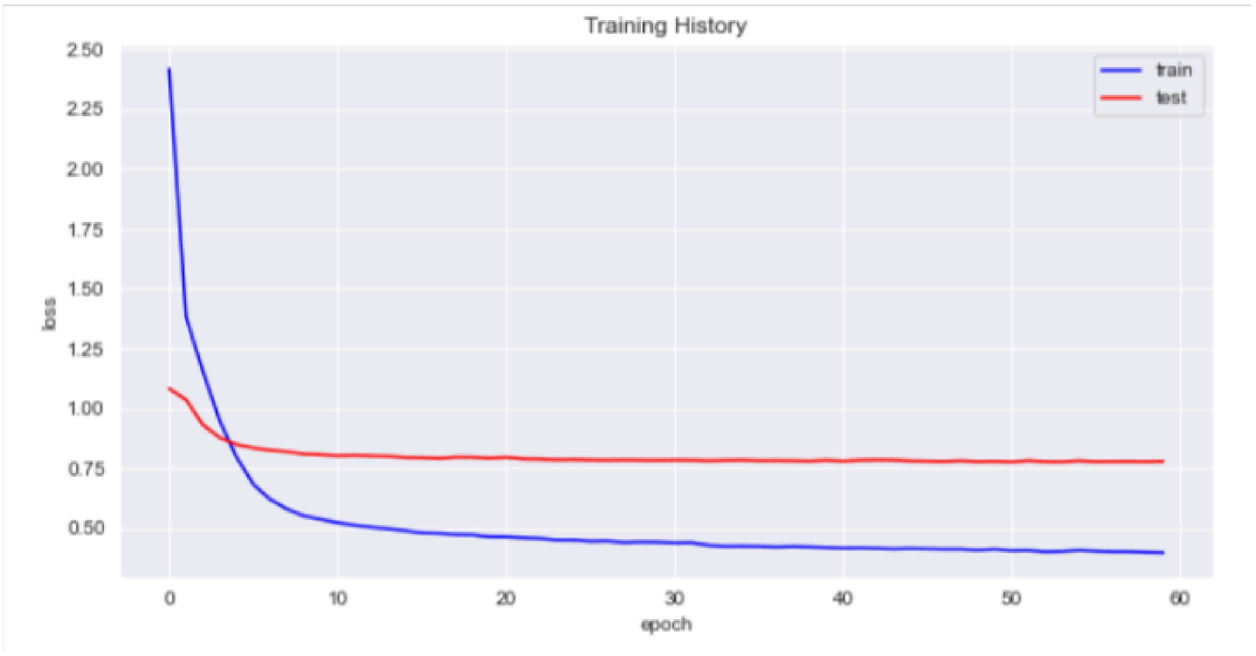
I chose Adam (Adaptive Moment Estimation) as my optimization algorithm with learning rate 0.001 and MSE as my loss function. Adam is a gradient descent optimization algorithm. According to my research, Adam optimizer is better than Stochastic Gradient Descent algorithm at dealing with deep learning models. After trying with both SGD and Adam, I found out that Adam does indeed produce lower MSE and RMSE, which means it is better at adjusting the weights.

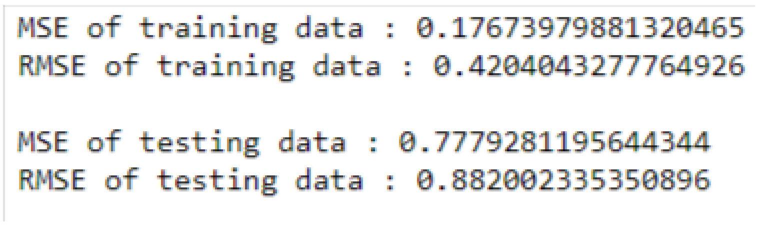

```
nn.compile(optimizer=Adam(learning_rate=0.001),loss='mse')   #TnE
```

Finally, I will start training my model with 60 epochs, and batch size 64.


```
In [167]: %%time
          History = nn.fit([train.user_Index,train.movie_Index],train.rating, batch_size=64,
                                          epochs = 60, validation_data = ([test.user_Index,test.movie_Index],test.rating),
                                          verbose = 1) #TnE : batch_size

          Epoch 52/60
          1261/1261 [==============================] - 13s 10ms/step - loss: 0.4057 - val_loss: 0.7806
          Epoch 53/60
          1261/1261 [==============================] - 12s 10ms/step - loss: 0.4001 - val_loss: 0.7761
          Epoch 54/60
          1261/1261 [==============================] - 13s 10ms/step - loss: 0.4017 - val_loss: 0.7753
          Epoch 55/60
          1261/1261 [==============================] - 12s 9ms/step - loss: 0.4061 - val_loss: 0.7800
          Epoch 56/60
          1261/1261 [==============================] - 13s 11ms/step - loss: 0.4027 - val_loss: 0.7761
          Epoch 57/60
          1261/1261 [==============================] - 12s 10ms/step - loss: 0.4002 - val_loss: 0.7767
          Epoch 58/60
          1261/1261 [==============================] - 12s 10ms/step - loss: 0.4003 - val_loss: 0.7771
          Epoch 59/60
          1261/1261 [==============================] - 12s 10ms/step - loss: 0.3973 - val_loss: 0.7760
          Epoch 60/60
          1261/1261 [==============================] - 13s 10ms/step - loss: 0.3956 - val_loss: 0.7779
          Wall time: 12min 39s
```

After plotting the training history of the model in terms of MSE, we can see that the model performs amazingly well with the training data with 0.1767 MSEand performs quite well with testing data as well with 0.7779 MSE.

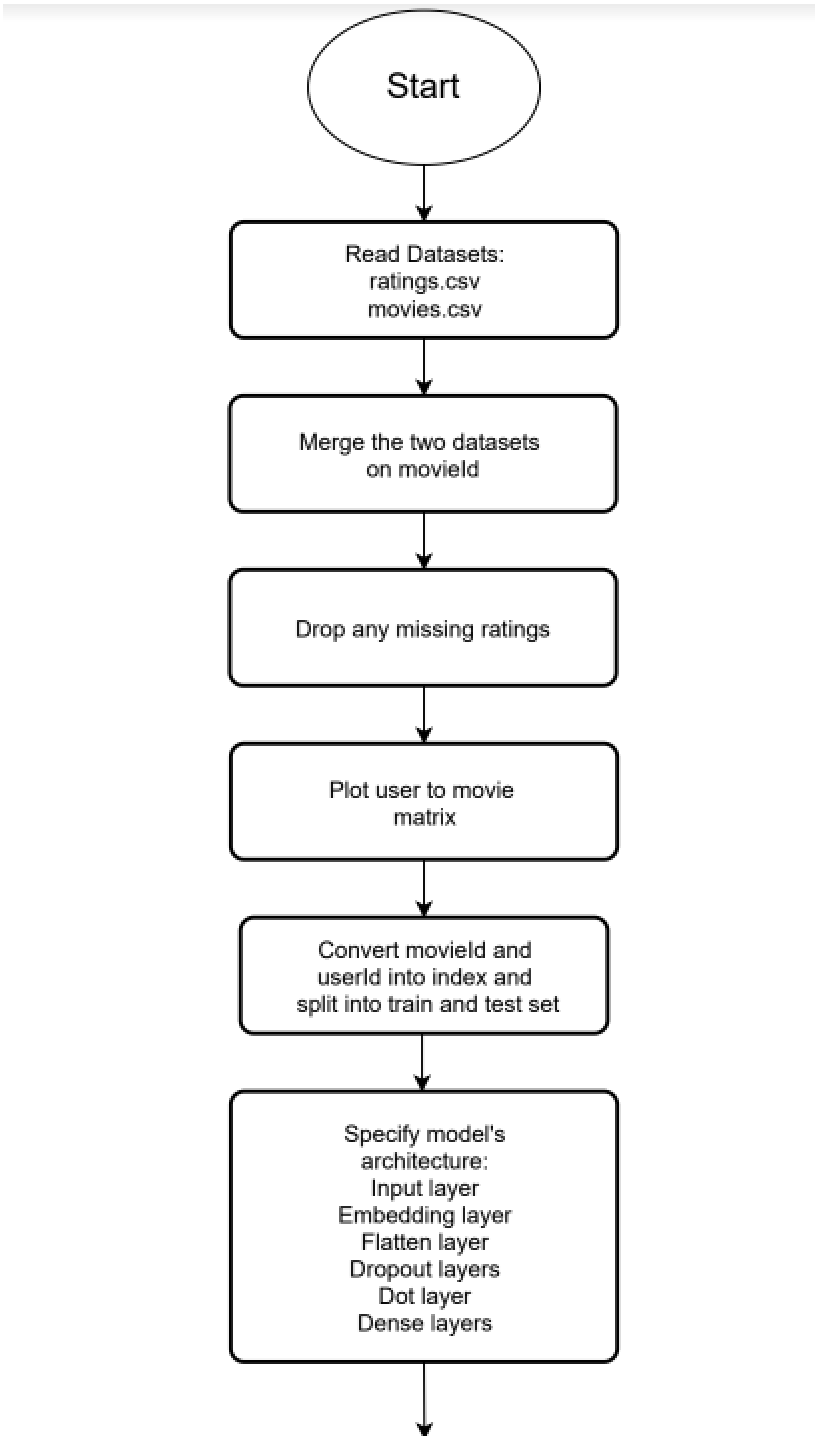


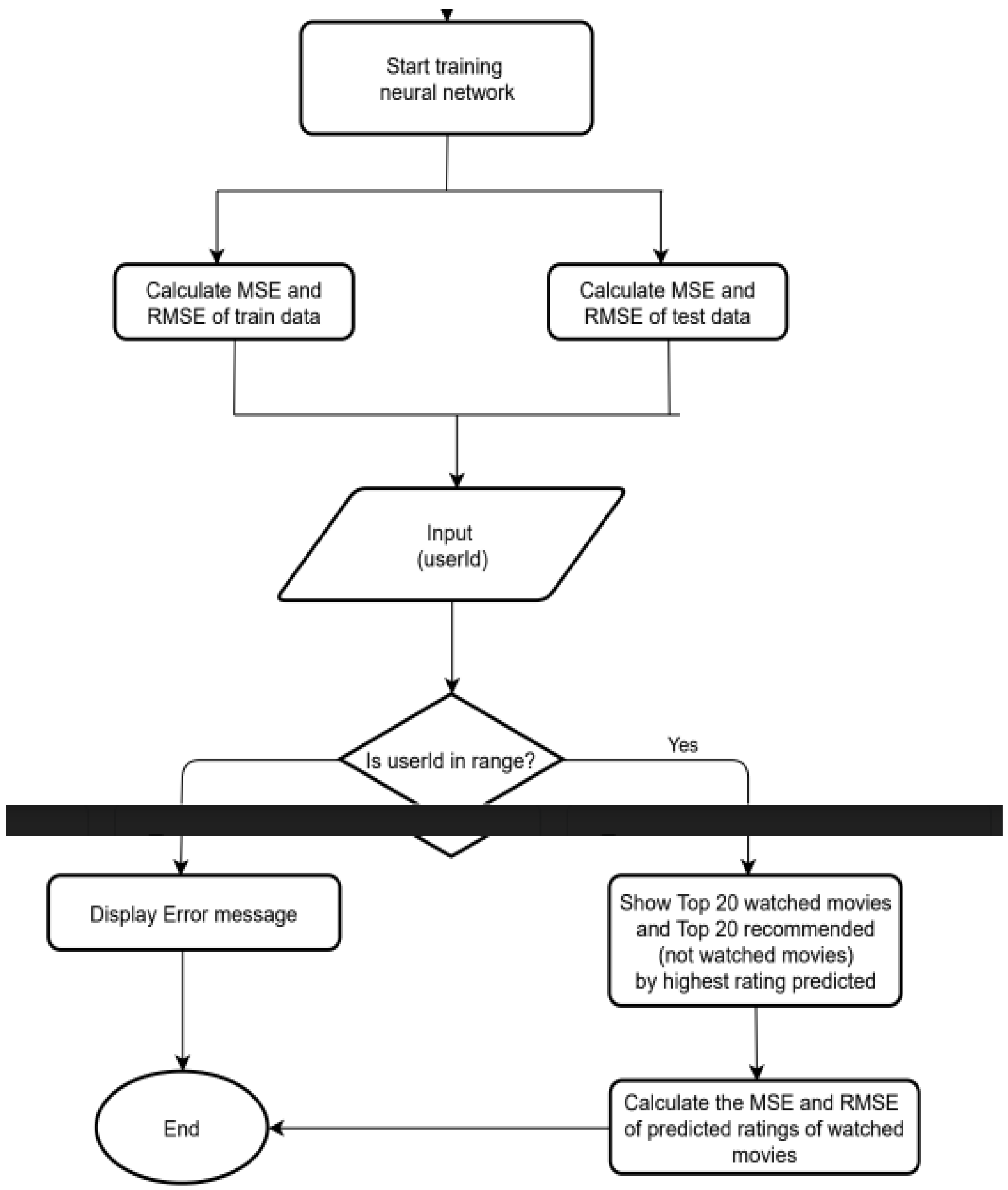


```
MSE of training data : 0.17673979881320465
RMSE of training data : 0.4204043277764926

MSE of testing data : 0.7779281195644344
RMSE of testing data : 0.882002335350896
```

Overall, I am quite satisfied with the algorithms I have chosen as it produced satisfactory results.

3.3. System flowchart/activity diagram


Start
Read Datasets:
ratings.csv
movies.csv
Merge the two datasets on movieId
Drop any missing ratings
Plot user to movie matrix
Convert movieId and userId into index and split into train and test set
Specify model's architecture:
Input layer
Embedding layer
Flatten layer
Dropout layers
Dot layer
Dense layers


Start training neural network
Calculate MSE and RMSE of train data
Calculate MSE and RMSE of test data
Input (userId)
Is userId in range?
Yes
[redacted]
Display Error message
Show Top 20 watched movies and Top 20 recommended (not watched movies) by highest rating predicted
End
Calculate the MSE and RMSE of predicted ratings of watched movies

3.4. Proposed test plan/hypothesis

H1 : A user that rated drama(genre) movies with high ratings in the past will get drama movies as recommendations.
H1a : A user that rated drama(genre) movies with high ratings in the past will not get drama movies as recommendations.

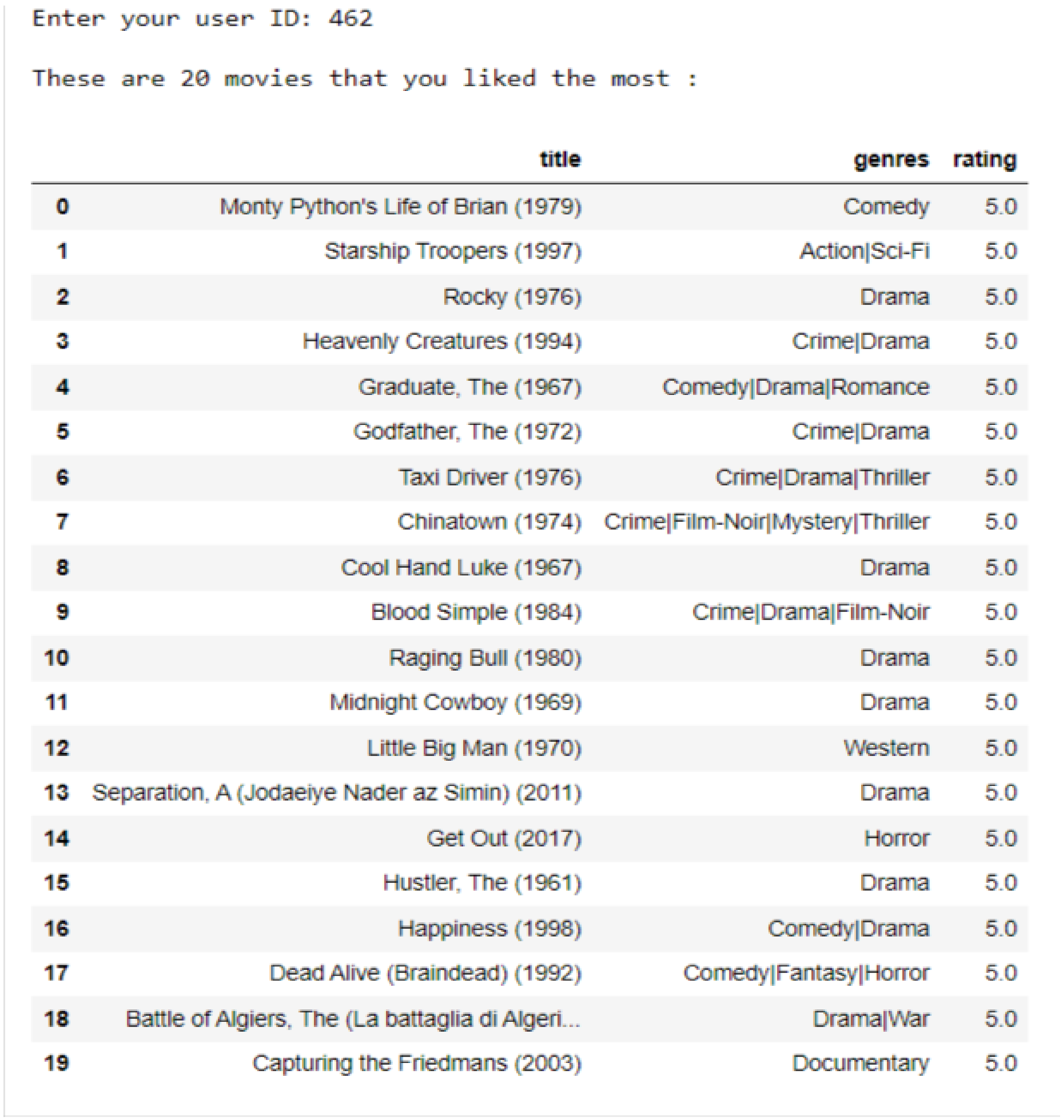
H2 : A user that rated action(genre) movies with high ratings in the past will get action movies as recommendations.
H2a : A user that rated action(genre) movies with high ratings in the past will not get action movies as recommendations.

4. Result

4.1. Results

H1 : A user that rated drama(genre) movies with high ratings in the past will get drama movies as recommendations.

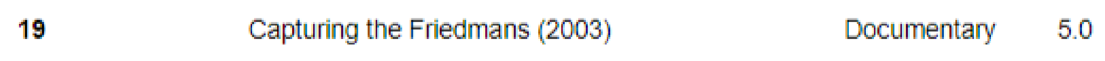
To test our hypothesis H1, I will enter a userId who has rated drama movies with high ratings before.



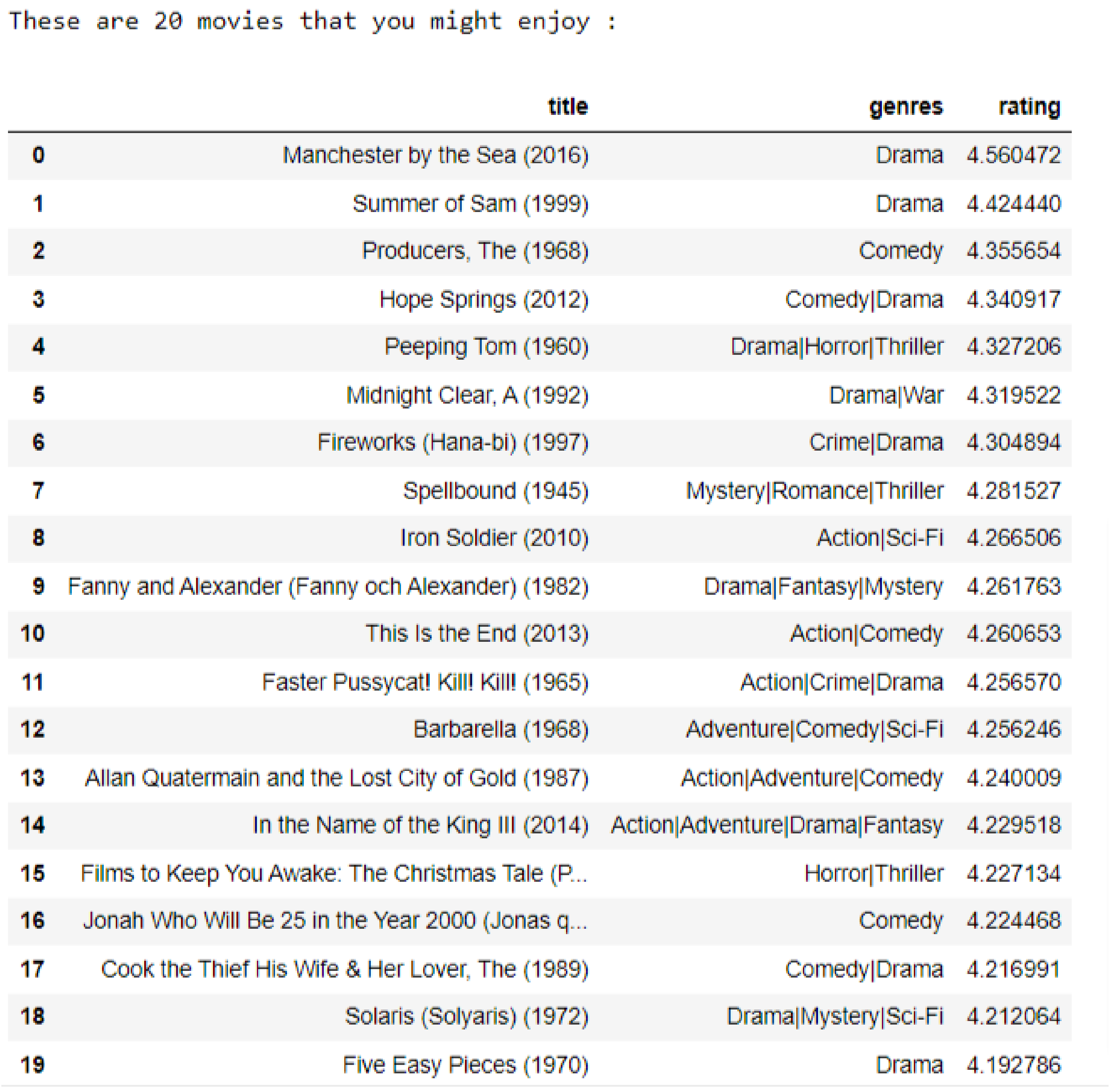
```
Enter your user ID: 462

These are 20 movies that you liked the most :
```

	title	genres	rating
0	Monty Python's Life of Brian (1979)	Comedy	5.0
1	Starship Troopers (1997)	Action|Sci-Fi	5.0
2	Rocky (1976)	Drama	5.0
3	Heavenly Creatures (1994)	Crime|Drama	5.0
4	Graduate, The (1967)	Comedy|Drama|Romance	5.0
5	Godfather, The (1972)	Crime|Drama	5.0
6	Taxi Driver (1976)	Crime|Drama|Thriller	5.0
7	Chinatown (1974)	Crime|Film-Noir|Mystery|Thriller	5.0
8	Cool Hand Luke (1967)	Drama	5.0
9	Blood Simple (1984)	Crime|Drama|Film-Noir	5.0
10	Raging Bull (1980)	Drama	5.0
11	Midnight Cowboy (1969)	Drama	5.0
12	Little Big Man (1970)	Western	5.0
13	Separation, A (Jodaeiye Nader az Simin) (2011)	Drama	5.0
14	Get Out (2017)	Horror	5.0
15	Hustler, The (1961)	Drama	5.0
16	Happiness (1998)	Comedy|Drama	5.0
17	Dead Alive (Braindead) (1992)	Comedy|Fantasy|Horror	5.0
18	Battle of Algiers, The (La battaglia di Algeri...	Drama|War	5.0
19	Capturing the Friedmans (2003)	Documentary	5.0

After entering the userId : 462, a table will be created showing the top 20 movies that user 462 has rated before, based on highest ratings. From the table, we can clearly see that user 462 is highly interested in the drama genre, as he or she has rated a lot of drama movies with 5.0 rating. Now we're interested to see which movies will be recommended to this user.



19	Capturing the Friedmans (2003)	Documentary	5.0



```
These are 20 movies that you might enjoy :
```

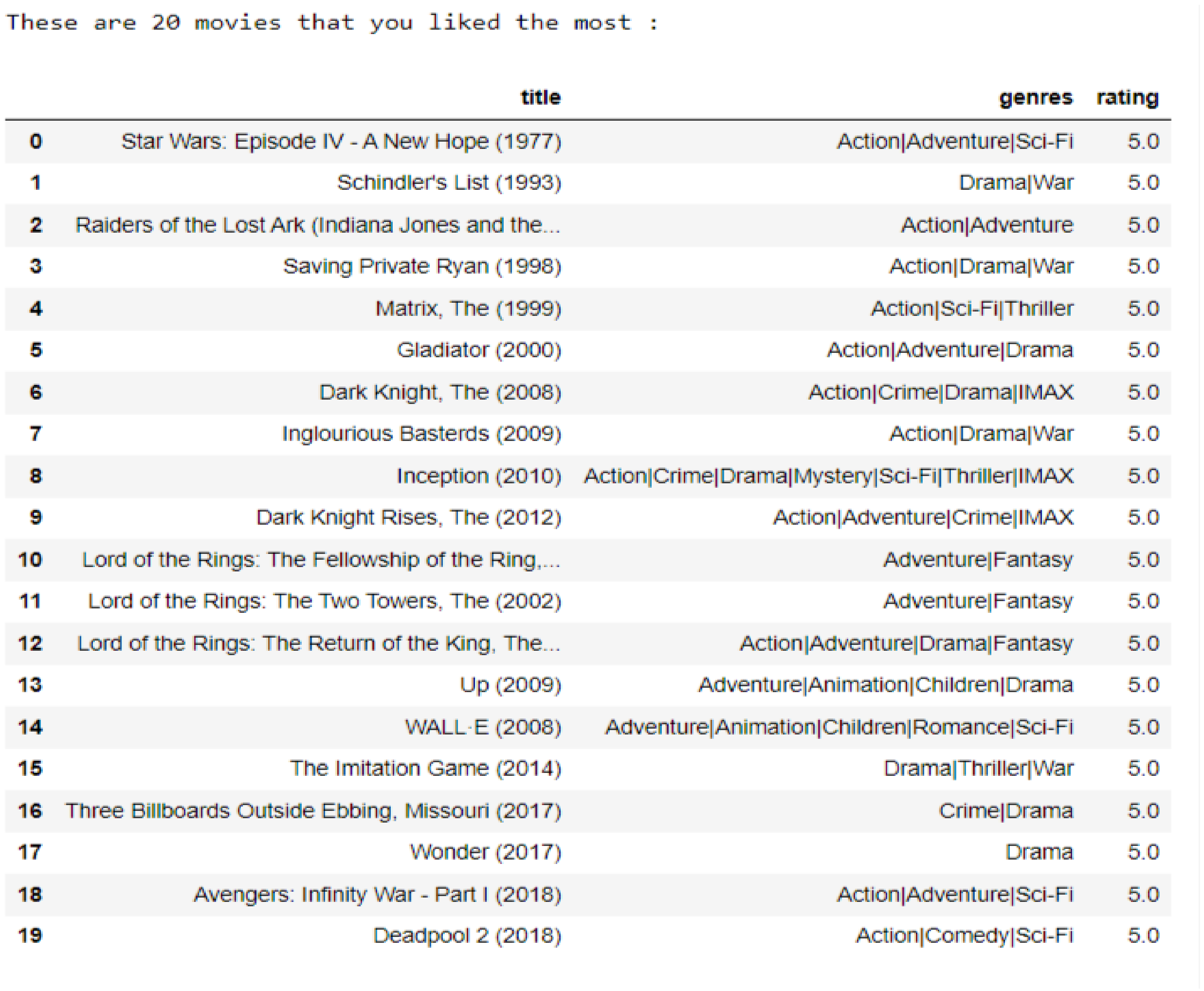
	title	genres	rating
0	Manchester by the Sea (2016)	Drama	4.560472
1	Summer of Sam (1999)	Drama	4.424440
2	Producers, The (1968)	Comedy	4.355654
3	Hope Springs (2012)	Comedy|Drama	4.340917
4	Peeping Tom (1960)	Drama|Horror|Thriller	4.327206
5	Midnight Clear, A (1992)	Drama|War	4.319522
6	Fireworks (Hana-bi) (1997)	Crime|Drama	4.304894
7	Spellbound (1945)	Mystery|Romance|Thriller	4.281527
8	Iron Soldier (2010)	Action|Sci-Fi	4.266506
9	Fanny and Alexander (Fanny och Alexander) (1982)	Drama|Fantasy|Mystery	4.261763
10	This Is the End (2013)	Action|Comedy	4.260653
11	Faster Pussycat! Kill! Kill! (1965)	Action|Crime|Drama	4.256570
12	Barbarella (1968)	Adventure|Comedy|Sci-Fi	4.256246
13	Allan Quatermain and the Lost City of Gold (1987)	Action|Adventure|Comedy	4.240009
14	In the Name of the King III (2014)	Action|Adventure|Drama|Fantasy	4.229518
15	Films to Keep You Awake: The Christmas Tale (P...	Horror|Thriller	4.227134
16	Jonah Who Will Be 25 in the Year 2000 (Jonas q...	Comedy	4.224468
17	Cook the Thief His Wife & Her Lover, The (1989)	Comedy|Drama	4.216991
18	Solaris (Solyaris) (1972)	Drama|Mystery|Sci-Fi	4.212064
19	Five Easy Pieces (1970)	Drama	4.192786

The model will predict all the ratings that user 462 will give to movies that he or she has not watched and the table above will output the top 20 predicted ratings for user 462. From the table above, we can see a lot of drama movies that will be recommended to this user.

From the outputs, we accept H1 and conclude that a user that rated drama movies with high ratings in the past will get drama movies as recommendations.

H2 : A user that rated action(genre) movies with high ratings in the past will get action movies as recommendations.

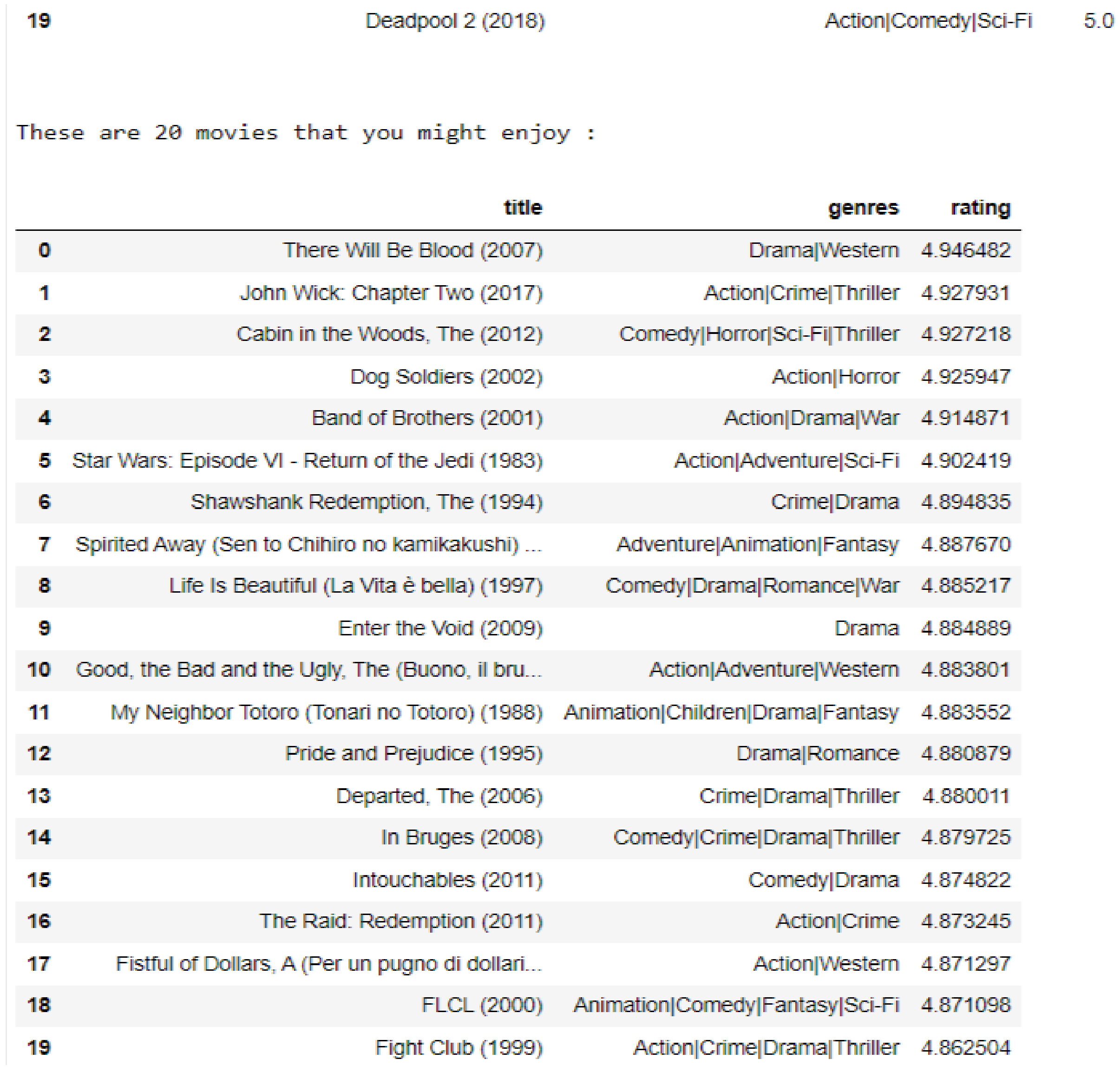
To test our hypothesis H2, I will enter a userId who has rated action movies with high ratings before.



```
These are 20 movies that you liked the most :
```

	title	genres	rating
0	Star Wars: Episode IV - A New Hope (1977)	Action|Adventure|Sci-Fi	5.0
1	Schindler's List (1993)	Drama|War	5.0
2	Raiders of the Lost Ark (Indiana Jones and the...	Action|Adventure	5.0
3	Saving Private Ryan (1998)	Action|Drama|War	5.0
4	Matrix, The (1999)	Action|Sci-Fi|Thriller	5.0
5	Gladiator (2000)	Action|Adventure|Drama	5.0
6	Dark Knight, The (2008)	Action|Crime|Drama|IMAX	5.0
7	Inglourious Basterds (2009)	Action|Drama|War	5.0
8	Inception (2010)	Action|Crime|Drama|Mystery|Sci-Fi|Thriller|IMAX	5.0
9	Dark Knight Rises, The (2012)	Action|Adventure|Crime|IMAX	5.0
10	Lord of the Rings: The Fellowship of the Ring,...	Adventure|Fantasy	5.0
11	Lord of the Rings: The Two Towers, The (2002)	Adventure|Fantasy	5.0
12	Lord of the Rings: The Return of the King, The...	Action|Adventure|Drama|Fantasy	5.0
13	Up (2009)	Adventure|Animation|Children|Drama	5.0
14	WALL·E (2008)	Adventure|Animation|Children|Romance|Sci-Fi	5.0
15	The Imitation Game (2014)	Drama|Thriller|War	5.0
16	Three Billboards Outside Ebbing, Missouri (2017)	Crime|Drama	5.0
17	Wonder (2017)	Drama	5.0
18	Avengers: Infinity War - Part I (2018)	Action|Adventure|Sci-Fi	5.0
19	Deadpool 2 (2018)	Action|Comedy|Sci-Fi	5.0

After entering the userId : 25, a table will be created showing the top 20 movies that user 25 has rated before, based on highest ratings. From the table, we can clearly see that user 25 is highly interested in the action genre, as he or she has rated a lot of action movies with 5.0 rating. Now we're interested to see which movies will be recommended to this user.



19	Deadpool 2 (2018)	Action|Comedy|Sci-Fi	5.0

```
These are 20 movies that you might enjoy :
```

	title	genres	rating
0	There Will Be Blood (2007)	Drama|Western	4.946482
1	John Wick: Chapter Two (2017)	Action|Crime|Thriller	4.927931
2	Cabin in the Woods, The (2012)	Comedy|Horror|Sci-Fi|Thriller	4.927218
3	Dog Soldiers (2002)	Action|Horror	4.925947
4	Band of Brothers (2001)	Action|Drama|War	4.914871
5	Star Wars: Episode VI - Return of the Jedi (1983)	Action|Adventure|Sci-Fi	4.902419
6	Shawshank Redemption, The (1994)	Crime|Drama	4.894835
7	Spirited Away (Sen to Chihiro no kamikakushi) ...	Adventure|Animation|Fantasy	4.887670
8	Life Is Beautiful (La Vita è bella) (1997)	Comedy|Drama|Romance|War	4.885217
9	Enter the Void (2009)	Drama	4.884889
10	Good, the Bad and the Ugly, The (Buono, il bru...	Action|Adventure|Western	4.883801
11	My Neighbor Totoro (Tonari no Totoro) (1988)	Animation|Children|Drama|Fantasy	4.883552
12	Pride and Prejudice (1995)	Drama|Romance	4.880879
13	Departed, The (2006)	Crime|Drama|Thriller	4.880011
14	In Bruges (2008)	Comedy|Crime|Drama|Thriller	4.879725
15	Intouchables (2011)	Comedy|Drama	4.874822
16	The Raid: Redemption (2011)	Action|Crime	4.873245
17	Fistful of Dollars, A (Per un pugno di dollari...	Action|Western	4.871297
18	FLCL (2000)	Animation|Comedy|Fantasy|Sci-Fi	4.871098
19	Fight Club (1999)	Action|Crime|Drama|Thriller	4.862504

The model will predict all the ratings that user 25 will give to movies that he or she has not watched and the table above will output the top 20 predicted ratings for user 25. From the table above, we can see a lot of action movies that will be recommended to this user. Similar genres to action like adventure will also be recommended as action movies usually contain a bit of adventure element in them.

From the outputs, we accept H2 and conclude that a user that rated action movies with high ratings in the past will get action movies as recommendations.

4.2. Discussion/Interpretation

In user 462’s case, we imply that the user likes drama movies and not a lot of action movies as he or she has rated drama movies with 5.0 ratings and not so much for action movies. This means that the feature of this user is that he or she likes movies with high weightage of drama and low weightage of action in it. So as expected, we recommended the movies that had a lot of drama elements in it like “Manchester by the sea” and “Summer of Sam” and these movies have little to no action element in them. Even though some action movies were recommended to this user, those movies still had a bit of a drama element in them, like the movies “Faster Pussycat! Kill! Kill! “ *and* “In the Name of the King III “.

From the second user 25, we imply that he or she likes a lot of action movies and quite enjoys adventure movies as well as he or she has rated these types of movies with 5.0 ratings. This means that the feature of this user is that he or she likes movies with high weightage of action and decent weightage of adventure in it. . As expected we recommended movies with a lot of action and a decent amount of adventure in it like “Spirited Away (Sen to Chihiro no kamikakushi) “ to the user.

5. Discussion and Conclusion

5.1. Achievements

After completing this project, I have successfully achieved my initial objective, which is to improve my understanding and knowledge about the recommender system which is a core part of Artificial Intelligence. Other than that, I gained knowledge about various different algorithms to solve the sparsity problem in recommender systems, which are Singular Value Decomposition, Alternating Least Squares, and Stochastic Gradient Descent. I also learned how to use deep learning to solve regression problems. Overall, I am grateful to my tutor, Miss Ashikin and lecturer Dr Goh Chin Pang for allowing me to participate in such a meaningful project.

5.2. Limitations and Future Works

One limitation of this project is the dataset I have chosen. The dataset consists of 100837 ratings. The number of ratings is not a lot and is very sparse as there are 610 unique users and 9724 unique movies, which should give us 5931640 total ratings. Due to this lack of data, the prediction of ratings could be inaccurate. The other limitation is the lack of prior research on recommender systems. I was not able to do enough research on my topic of study due to busy schedules. As a result, I could not implement an alternative algorithm such as the Alternating Least Squares algorithm to compare with my model and choose the better performing one.

There are a few improvements to be made in the future. The first one being, I can find a larger dataset with more number of ratings to increase the accuracy of my model. The second one is I should manage my time properly and find sources online such as kaggle, youtube, github, and such to help understand the topic better. This will lead to better results and predictions of ratings. It will make me more interested and knowledgeable in the field of study as well.

Reference & Source

Provide the sources of the dataset and tool(s) used for development

List the articles or other references you have cited in the text using the Harvard Referencing system.

[1] GroupLens. (2019). *MovieLens*. [online] Available at: https://grouplens.org/datasets/movielens/.

[2] Kohersen, W. (2018). *Neural Network Embeddings Explained*. [online] Medium. Available at: https://towardsdatascience.com/neural-network-embeddings-explained-4d028e6f0526.

[3] Google Developers. (2019). *Embeddings | Machine Learning Crash Course | Google Developers*. [online] Available at: https://developers.google.com/machine-learning/crash-course/embeddings/video-lecture.

[4] Databricks (2018). *Deep Learning for Recommender Systems with Nick pentreath*. [online] Available at: https://www.slideshare.net/databricks/deep-learning-for-recommender-systems-with-nick-pentreath [Accessed 12 Sep. 2021].

[5] www.youtube.com. (n.d.). *Tutorial 9- Drop Out Layers in Multi Neural Network*. [online] Available at: https://www.youtube.com/watch?v=XmLYl17DbbA [Accessed 12 Sep. 2021].

[6] kaggle.com. (n.d.). *Matrix Factorization*. [online] Available at: https://www.kaggle.com/colinmorris/matrix-factorization?select=movie.csv [Accessed 12 Sep. 2021].

[7] Koren, Y., Bell, R. and Volinsky, C. (2009). Matrix Factorization Techniques for Recommender Systems. *Computer*, 42(8), pp.30–37.