PROJECT: MOVIE RECOMMENDER SYSTEM

BY: TAI NGO

DATE: 11/04/2020

DSC-680 – BELLEVUE UNIVERSITY

Introduction

With the rise of data science and its applications in the technology field, many data scientists have developed and successfully integrated a plethora of impactful projects and products into various computing machines. Over time, the machines continue to be developed and integrated with algorithms to become exponentially smarter when it comes to serving the customer needs with basic and complicated tasks. Many technology companies, whether they are big or small, have already built, implemented and introduced to their customers about their machine learning algorithms. The moment that the customer enters a website, it is very likely that the customer already engages in interacting with the web via their data collection system, recommender system and search system. A simple example would be a customer who goes to an online store and look for new things to buy. The customer is probably sending their current zip code, and proceeds with a search bar and will be greeted with products displayed by the recommender system. It is not a secret that the applications, especially the machine learning algorithms, of data science have brought billions worth of profits for many businesses since it provides serious competitive edges, and an exciting path to interact with customers at the highest efficiency. It is also not a coincident that the rise of many big technology companies appears to be parallel with the rise of data science applications. After all, data science plays a big part in the exponential expansion of the technology industry. This edge does not seem to fall off as companies, who may already own multiple data science technologies, continue to develop and improve their own systems while older companies have to struggle with the switch and try to catch up. A prominent example is Apple whom has recently announced that they will develop their own search engine to complete their own ecosystem after many years of neglecting their own search engine. The search engine is nothing new in the data science world. A big tech company such as Google has seen its meteoric growth since they showed the world

their search engine and algorithms. However, aside from Google the competitive edge is always there for any businesses, who wish to expand and serve their customers better. Nowadays, even a small, local market has its own website with the search engine to allow the customer to figure out if the market is selling a certain item, or whether the item is available at their nearby store. Many data science projects will continue to be developed further to be more useful and serve the customers better.

Although the data science application in this project will not be as fancy and complicated as the search engines that most businesses are working on, the application from this project should be useful for those who want to learn and understand about the basics of a popular system, the Movie Recommender System. There are many movie recommender systems available. They are built differently to capture things and produce results that are desired by the developers. However, the movie recommender system is a data science project, many parts of the process of building the movie recommender system will be similar to building other types of data science applications. The pipeline of the process should be similar to other typical data science projects. The builders will work on collecting, preprocessing and analyzing the data before building and training a machine learning model. When referring to the movie recommender system, people may think about the Netflix, Amazon or some other entertainment websites. In reality, the movie recommender systems look very different for different businesses because the expected inputs, which are the movies from the customers, and outputs, which are the recommended movies, are probably the same. However, they are different because different companies may use different datasets, or their own datasets to build their system, as well as recommending different movies due to various factors such as the rights to certain movies.

Summary of Business Problem/ Hypothesis

Many big and small businesses rely on new technology to compete and improve their customer service. While there are many applications and technology available, data science can offer the exact solution for this specific task: enhancing the customer engagement with their products by showcasing recommended items. In this project, a simple movie recommender system is built in order to recommend ten movies that a user may like based on his/her preferences. A machine learning algorithm will be built to help perform the prediction/recommendation. The result from this project can be applied for similar projects where the data includes the users and their ratings on specific items to suggest recommendation. The business may have the data of the customer when they shop online at their website, but turning the data into something useful will be a challenge. This project offers insight into how to work with very simple data and make the most out of it.

Questions

1. What are the shapes of the dataset?
2. What is the type of the downloaded file?
3. What are the features and relevant features?
4. What are the inputs and expected outputs?
5. How should the data be cleaned?
6. Do all movies have the same number of ratings?
7. How should the movies be ranked based on the rating?
8. What are the functions necessary to perform the tasks?
9. How are the ratings of the movies going to be used for prediction?
10. How are businesses going to benefit from this project?
11. What benefits can this project bring to the users?
12. How many outputs should be shown to the users?
13. How do we evaluate whether the project succeeds?
14. How do we build a model that is scalable to bigger datasets?
15. How do we deploy the model into production?

Methods

Before going into the details of the project, it will be helpful to discuss what a movie recommender system does. It is interesting to describe about this system beyond its functionality and how it is useful for the people. Basically, the movie recommender system is a system that can predict and filter the movies based on the user's preferences. The movie recommender system works similarly to other recommender systems such as music, news, book and research article recommender system. There are two ways, in which a recommender system can be used to produce a list of recommended items: collaborative filtering and content-based filtering.

This project will be using the collaborative filtering to build the movie recommender system. Collaborative filtering is a unique technique of the recommender systems. If people want to build a recommender system, chances are that they will very likely use this technique because of the lack of large, detailed data. Collaborative filtering has two senses: a narrow sense and a general sense. For the narrow sense, the collaborative filtering will automatically filter the predictions based on the interests of the user. This happens because it collects the preferences or information from other users, hence the term collaborating appears in the name. The researchers use this technique with the assumption that if user A and user B have the same opinion on a

certain topic, then user A may also have the same opinion as user B in some other topics. On the other hand, the general sense allows the collaborative filtering to use different data sources, viewpoints in collaboration to obtain crucial information or patterns. The collaborative filtering is a good choice for movie recommender system because it does not rely on the machine-analyzable contents, which means it can accurately recommend complex items such as movies while it does not understand the items. Some other machine learning approaches such as k-nearest neighbors and Pearson Correlation may also work for this task because they focus on similarity among the items.

While the second technique is not used in this project, it should be interesting to find out what it can be used for in other scenarios. Content-based filtering takes a completely different approach than the collaborative filtering. It relies on the descriptions of specific items and the user's specific preferences rather than the similarity among the items for the same group of users. This raises a good case to use the content-based filtering. When the data about the item, not the user, is completely known such as the name, location, and features, the content-based filtering will be a good choice. It treats this problem as a classification for specific users. It can learn a classifier for what the user likes or dislikes based on the features of a specific item. This approach is perhaps more accurate and useful when more data is given. However, for the sake of building a basic movie recommender system, the collaborative filtering is a clear superior choice.

There are many programming languages that can be used to build a movie recommender system, one of the easiest ways is to use Python. Python provides the most user-friendly environment to work with the data as well as build machine learning algorithms due to its rich module libraries such as TensorFlow, Keras and PyTorch; the simplicity and ease of use of Pandas is another positive aspect of Python in handling and preparing the data for machine learning algorithms compared to other programming languages. In addition, Python has a very rich and diverse data visualization libraries such as MatplotLib, Plotly and Seaborn. These data visualization libraries are very convenient to help us learn about the data. This project will use MatplotLib for simplicity because the data in this project does not require extensive exploratory data analysis for its depth. The dataset can be loaded into data frame by using in a few minutes before they can be cleaned up. The cleaning up of the data is one of the most time-consuming processes. Most data in real world needs to undergo preprocessing before they can be explored, analyzed and fed into the machine learning algorithms. This step allows the

researchers to design and determine what the relevant features are for the best predictions. In addition, researchers can tweak the models with different parameters to achieve the highest possible accuracy score.

The data for this project was collected and organized by GroupLens Research, which was a computer-science research lab in 1997. The file of the dataset can be downloaded directly from the GroupLens website. In this project, the Keras module will be used to obtain the zip file. The zip file will be extracted and sent to a designated folder. The ratings csv, which is one of the extracted files, will be loaded into a data frame by using Pandas. The data will be explored and examined to determine the relevant information. The data preprocessing step involves checking the number of users and how many ratings each user has made. It is important to have all users with the same number of ratings. From the dataset, the minimum rating is 0.5, and the maximum rating is 5.

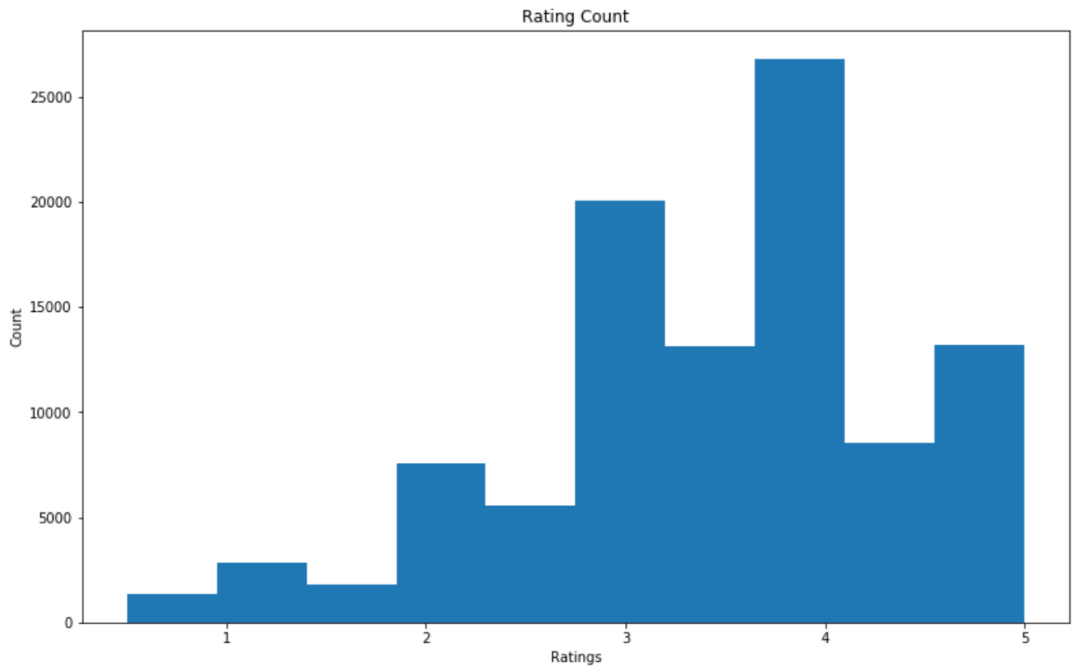

Figure 1. The rating counts of movie reviews.

Figure 1 shows that the graph is left-skewed, and there are more high rating counts than low rating counts. This suggests that most people either feel neutral about the movie or feel good about the movie. This is a positive side for those who want to build a movie recommender systems because chances are people will enjoy the recommended movies. Although this step is not implemented in this project, it is worth thinking about. Movie recommender system can filter out the movies with lower rating and only recommend movies with higher ratings. This will improve the user experience with the system. However, the downside is it can be very

biased because there are movies with low review rating, and yet some people would love to see them.

Rows with missing data will be removed. After examining the data, the userID and movieID columns are converted into two additional columns, which are user and movie. The reason for the conversion into new columns is because we need to encode the users and the movies as indices. Later on, the user and movie columns will be fed into the machine learning algorithms. Below are the first ten rows of the data frame after preprocessing.

	userId	movieId	rating	timestamp	user	movie
67037	432	77866	4.5	1335139641	431	4730
42175	288	474	3.0	978465565	287	474
93850	599	4351	3.0	1498524542	598	2631
6187	42	2987	4.0	996262677	41	194
12229	75	1610	4.0	1158989841	74	727
7433	51	177	4.0	1230930634	50	535
53802	354	51662	3.5	1200869930	353	899
65098	416	750	4.5	1187495659	415	722
68041	438	6503	0.5	1105654319	437	1882
11854	73	8641	3.5	1464197978	72	1226

Figure 2. The first ten rows of the data after being randomized.

As a side note, the users do not appear in order because the entire data frame has been shuffled to ensure that the data is as randomized as possible before going into the algorithm.

70% of the data is used for training while the remaining 30% is used for validation. The features for the model are the values of user and movies. In order to build the model, the model function of the Keras library is used. The MovieRecommender function is created, which takes three inputs: the number of users, the number of rated movies and the number of vector dimensions. To compile the model, the BinaryCrossentropy is chosen for the loss function, Adam is chosen for the optimizer. The regularization for the optimizer is 0.001. The role of the regularization in the optimizer is to modify the model performance so that it can perform well with unknown data outside of the training data. Once every parameter is set in place, the training process can take place. Below is the image of the training process.

```
Epoch 1/10
1103/1103 [==============================] - 4s 4ms/step - loss: 0.6409 - val_loss: 0.6212
Epoch 2/10
1103/1103 [==============================] - 4s 4ms/step - loss: 0.6164 - val_loss: 0.6232
Epoch 3/10
1103/1103 [==============================] - 4s 4ms/step - loss: 0.6106 - val_loss: 0.6141
Epoch 4/10
1103/1103 [==============================] - 4s 4ms/step - loss: 0.6082 - val_loss: 0.6141
Epoch 5/10
1103/1103 [==============================] - 4s 4ms/step - loss: 0.6071 - val_loss: 0.6112
Epoch 6/10
1103/1103 [==============================] - 4s 3ms/step - loss: 0.6064 - val_loss: 0.6102
Epoch 7/10
1103/1103 [==============================] - 4s 3ms/step - loss: 0.6052 - val_loss: 0.6116
Epoch 8/10
1103/1103 [==============================] - 4s 3ms/step - loss: 0.6062 - val_loss: 0.6100
Epoch 9/10
1103/1103 [==============================] - 4s 4ms/step - loss: 0.6050 - val_loss: 0.6122
Epoch 10/10
1103/1103 [==============================] - 5s 4ms/step - loss: 0.6041 - val_loss: 0.6104
```

Figure 3. The training process of building a machine learning model.

Figure 3 shows that there are 10 epochs for the training. This means that there are 10 passes of the entire dataset that the machine learning model has performed. After each pass, the accuracy of the model improves as shown by the lower loss in both training and validation.

The image below shows the results after the model has been trained.

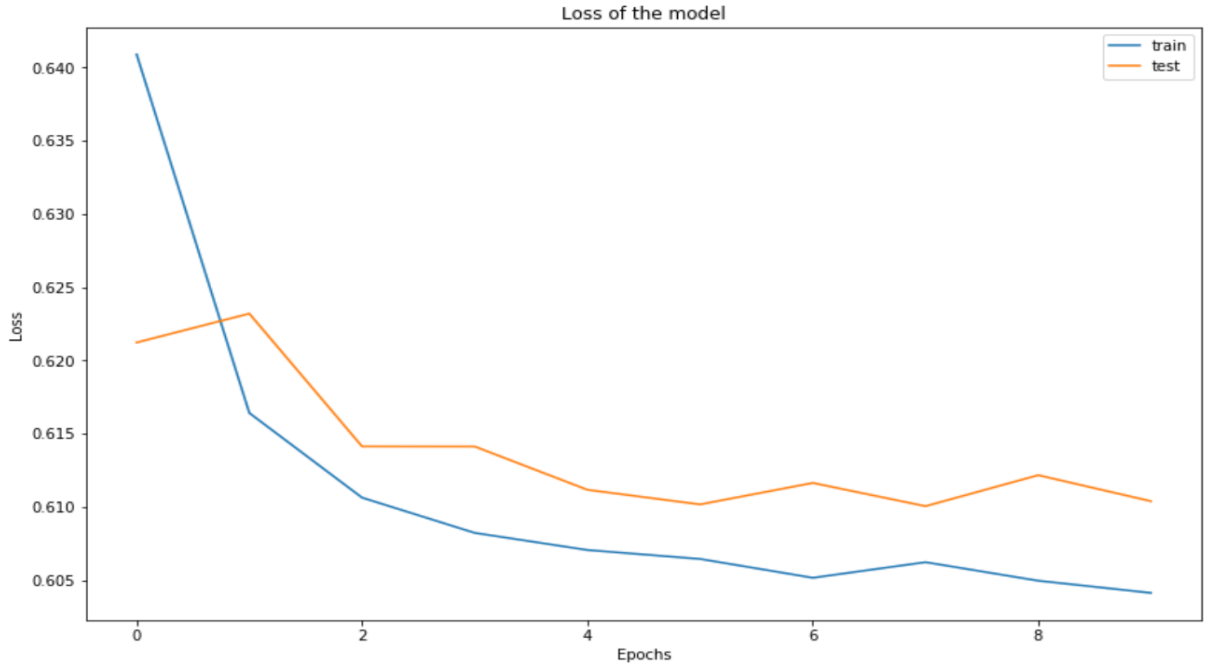

Figure 4. The graphs of the training and test data after fitting.

There are ten epochs for both training and test. The training takes 70% and the test takes 30% of the dataset. It is typical for a model to show a big drop in the loss for the training data because the model relies on the training data to learn. The loss for the test data is a bigger concern as the lower the loss the better it is for the learning process. While the training loss continues to drop after 10 epochs, the test loss starts forming a plateau. The lowest loss value for the test data occurs at 8th epoch with the value of 0.6100.

Once the model has been put, the predict function can be used to perform recommendations for the movies. Twenty movies will be recommended for any random user in the database. Below is an example of a random user receiving their recommended movies based on their preferences.

```
Recommended Movies for user: 522


Highly Rated Movies by user


Scott Pilgrim vs. the World (2010) - Action|Comedy|Fantasy|Musical|Romance
Tron: Legacy (2010) - Action|Adventure|Sci-Fi|IMAX
Louis C.K.: Chewed Up (2008) - Comedy
Avengers, The (2012) - Action|Adventure|Sci-Fi|IMAX
Louis C.K.: Live at the Beacon Theater (2011) - Comedy


Top 20 Recommended Movies


Toy Story (1995) - Adventure|Animation|Children|Comedy|Fantasy
Usual Suspects, The (1995) - Crime|Mystery|Thriller
Braveheart (1995) - Action|Drama|War
Forrest Gump (1994) - Comedy|Drama|Romance|War
Searching for Bobby Fischer (1993) - Drama
Blade Runner (1982) - Action|Sci-Fi|Thriller
Fargo (1996) - Comedy|Crime|Drama|Thriller
Godfather, The (1972) - Crime|Drama
Die Hard (1988) - Action|Crime|Thriller
Reservoir Dogs (1992) - Crime|Mystery|Thriller
Monty Python and the Holy Grail (1975) - Adventure|Comedy|Fantasy
L.A. Confidential (1997) - Crime|Film-Noir|Mystery|Thriller
Matrix, The (1999) - Action|Sci-Fi|Thriller
Memento (2000) - Mystery|Thriller
Cowboy Bebop: The Movie (Cowboy Bebop: Tengoku no Tobira) (2001) - Action|Animation|Sci-Fi|Thriller
Batman Begins (2005) - Action|Crime|IMAX
Last King of Scotland, The (2006) - Drama|Thriller
Hot Fuzz (2007) - Action|Comedy|Crime|Mystery
Inglourious Basterds (2009) - Action|Drama|War
Harry Potter and the Deathly Hallows: Part 2 (2011) - Action|Adventure|Drama|Fantasy|Mystery|IMAX
```

Figure 5. Twenty recommended movies for a random user.

The user's index is 522, and based on five highly rated movies of this user, the model shows twenty recommended movies. New users with new data can be given recommended movies using this model.

This project has displayed basic instructions and overall necessary knowledge on how to build a simple movie recommender system. The codes for this project will be attached in the link below for references. Recommender system has proven to be very useful for businesses to gain competitive edge and improve their customer service. Modelers have the freedom to tweak their models to maximize the user experience by showing them highly rated items. Furthermore, understanding and practicing the movie recommender system allows researchers or machine learning enthusiasts a deeper knowledge in data science applications. This is an important stepping stone to gain more knowledge in this exciting and impactful technology field.

References

1. F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. DOI=http://dx.doi.org/10.1145/2827872
2. Christakou, C., Vrettos, S., Stafylopatis, A. World Scientific. *A Hybrid Movie Recommender System Based on Neural Network.* Retrieved from https://www.worldscientific.com/doi/abs/10.1142/S0218213007003540
3. Katarya, R., Verma, O. *An Effective Collaborative Movie Recommender System with Cuckoo Research.* Retrieved from https://www.sciencedirect.com/science/article/pii/S1110866516300470
4. Odic, A., Tkalcic, M., Tasic, J. *Predicting and Detecting the Relevant Contextual Information in a Movie-Recommender System.* Retrieved from https://academic.oup.com/iwc/article-abstract/25/1/74/768060
5. Azaria, A., Hassidim, A., Kraus, S., Eshkol, A., Weintraub, O., Netanely, I. *Movie Recommender System for Profit Maximization.* Retrieved from https://dl.acm.org/doi/abs/10.1145/2507157.2507162
6. Yamada, P. IEEE. *A Movie Recommender System Based on Inductive Learning.* Retrieved from https://ieeexplore.ieee.org/abstract/document/1460433
7. Mukherjee, R., Jonsdottir, G., Sen, S., Sarathi, P. *MOVIES2GO: An Online Voting Based Movie Recommender System.* Retrieved from https://dl.acm.org/doi/abs/10.1145/375735.376018
8. Kumar, M., Yadav, D., Singh, A., Gupta, V. *A Movie Recommender System: MOVREC.* Retrieved from

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.736.6037&rep=rep1&type=pdf

9. Perny, P., Zucker, J. *Preference-based Search and Machine Learning for Collaborative Filtering: the "Film-Conseil" Movie Recommender System.* Retrieved from https://www.cin.ufpe.br/~glr/tmp/recommend/film-conseil.pdf
10. Ho, A., Menezes, I., Tagmouti, Y. *E-MRS: Emotion-based Movie Recommender System.* Retrieved from http://www.ptidej.net/courses/ift6251/fall06/article/Projet%20Ai%20-%20Ilusca%20-%20Yousra.doc.pdf
11. Wikipedia. *Recommender System.* Retrieved from https://en.wikipedia.org/wiki/Recommender_system#Content-based_filtering
12. Wikipedia. *Collaborative Filtering.* Retrieved from https://en.wikipedia.org/wiki/Collaborative_filtering
13. Liang, Y. *Deep Learning Basics.* Retrieved from https://www.cs.princeton.edu/courses/archive/spring16/cos495/slides/DL_lecture3_regularization_I.pdf
14. Wu, Shan-Hung. *Neural Networks: Optimization & Regularization.* Retrieved from https://nthu-datalab.github.io/ml/slides/11_NN_Optimization_Regularization.pdf
15. Banerjee, S. *Collaborative Filtering for Movie Recommendations.* Retrieved from https://keras.io/examples/structured_data/collaborative_filtering_movielens/

Appendix

Figure 1 – The rating counts of movie reviews

Figure 2 - The first ten rows of the data after being randomized

Figure 3. The training process of building a machine learning model

Figure 4. The graphs of the training and test data after fitting

Figure 5. Twenty recommended movies for a random user