Zahid Amaar

University of Sunderland

Computer Science

Student ID: BH19GO

Student Number: 1349 107 102

Contents

1. Development

1.1. Introduction

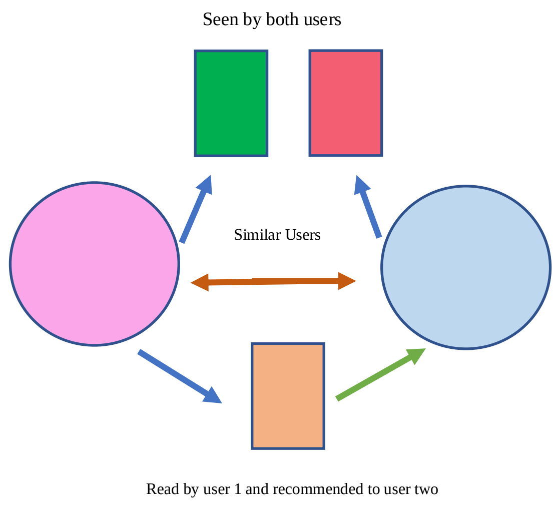
During the course of the previous professional practice week it was established, that due to previous work undertaken during the course of the module, and current comfort levels and understanding that the project which would be developed for this professional practice week would be an application which uses machine learning as apposed to a pathfinding project.

During the course of the last professional practice week it was decided that the prototype built in this assessment would be a movie recommendation system. There were two main approaches to the system, one of which is known as content based, and the other is user-based systems.

These two systems essentially, do as they are named, a content-based system is able to recommend movies, or items, depending on the service, to users based on their specific previously viewed movies or items. This can be thought of as a simplified algorithm which used to be adopted by Netflix.

A collaborative or also commonly referred to as a user-based system essentially recommends items to users based on high ratings, for example if one movie completely unrelated to another has high rating from a large number of users, that item will be recommended regardless of context. This was and is still adopted by some online retailers.

Fig 1. Showing the methodology and frame work for this prototype system.



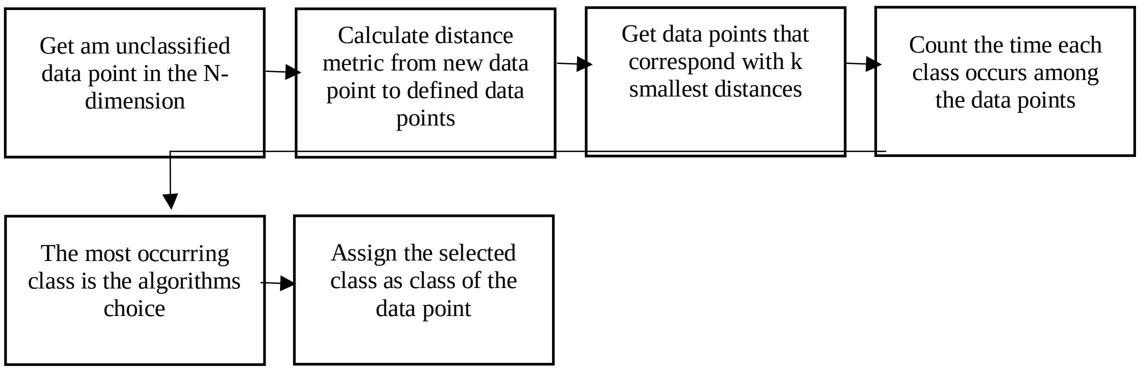
1.2. Planning

During early development it was decided that the system developed would use both content-based recommendation and collaborative, as this is proven to be most effective as Basilico and Hofmann (2004) and Marovic et al. (2011) state, using a unified system which uses both content-based filtering and collaborative-based filtering offers a substantial improvement in results. This method is also proven to be highly successful as Netflix currently uses a recommendation system based on a unified filtering system which incorporates both content-based, and collaborative-based filtering as stated by Help Center (2020).

The way in which this is being approached in the prototype system is by essentially using two datasets, one with Movies, and one with user reviews, both of these together allow for a correlation to be built. Essentially, the user will be able to search for a movie in the system, and based on the ratings from all users, and the similarity result of the searched movie a correlation score will be calculated which will then offer a list of highest to lowest recommendations to the end user.

Before this can be done however, it will be important to clean the data set as there will be dozens of movies with no ratings as people may not have viewed or rated them, once the processing is done, a data frame can be built, once the movies have been sorted and a correlation has been built using pandas series, it will be possible to send the movies to the data frame, when visualising this the movies displayed will have the highest similarity result. The next few steps require for sorting, and removing movies which have under a certain amount of reviews to give an accurate and reliable result. This prototype system uses a KNN (K-Nearest Neighbour) algorithm to classify the ratings of each movie and user rating to allow for accurate results when compared to the data, in this approach, the model will be using machine learning algorithms in this case, KNN to predict a user's rating of unrated items, some examples of this are KNN, clustering, matrix, factorization and deep learning models. The model used in this system is inspired by the proposed solution mentioned by Werneck et al. (2018), this framework is applied to movie recommendation systems which incorporate ai and machine learning, in this proposed framework, the raw data is input into the training sets and data sets which are normalised and run through feature extraction, which is then sent to build a classifier. Because of this framework, it was possible to arrive at the conclusion that KNN is the most appropriate model for this system, the algorithm is able to predict using the input data sets as explained in the following framework diagram in Fig 2.

Fig 2. Showing the KNN model


Get am unclassified data point in the N-dimension
Calculate distance metric from new data point to defined data points
Get data points that correspond with k smallest distances
Count the time each class occurs among the data points
The most occurring class is the algorithms choice
Assign the selected class as class of the data point

One of the most important decisions made when developing this system was the format, being able to develop the prototype system as a Jupyter notebook file, allows for the use of both Python, but also selective code running. This is much more effective as it allows for code to be ran on its own without needing to re run the whole script. Using what was learned within the tutorials of this module it was possible to develop a system, which uses machine learning to filter results and recommend movies as a system based on a dataset, this could be a historical dataset, or an ever evolving dataset when pushed to a live service, however the use of pandas in the way in which it was used in this system, is a subject that was not so familiar, but with a little research and reading documentation it was easily digestible. Matplotlib was also used which allowed for data plotting which was helpful for visualising the data before processing to allow for an easier understanding of the data, as it shows in plain and simple graphs and histograms. The work conducted during this module, and the work uploaded to the Eportfolio have helped a lot to understand the fundamentals of machine learning and AI in general, with the help of reading datasets, and building data frames and correlations it was much simpler implanting the hybrid flirting system, which gave results which were both accurate and easy to see the correlation between. Using libraries such as sklearn and matplotlib were made a lot easier due to the work conducted during lectures and tutorials, as it was heavily relevant to this subject and an essential part of the system. One final Library to make note of during the development section of this report and the planning documentation is the fuzzywuzzy library, this essentially allows for the use of searching the dataset through an input which does not match exactly to the name within the database making the system much easier to test but also more user friendly.

2. Evaluation and Testing

During the course of this professional practice week the task was to develop a system of choice, which was selected in the previous professional practice week. The system chosen for this assessment was a Movie recommendation software, this software would be akin to those used within Netflix, Amazon, YouTube, and many other online media services and Shopping services.

The ultimate goal of this assessment was to put together the knowledge acquired throughout this module, to create robust prototype that would fulfil the needs of the system if developed further. As time was limited to only a week the system had constraints which meant liabilities and priorities had to be taken into account.

Because of this creating a full website or live service to implement the system into has been made a task which cannot be completed, because of this the testing and evaluation of this system will examine the final Ipynb script.

2.1. Evaluation

Before mentioning the testing and the errors present which needed to be addressed and or proactively stopped, it is important to remember that KNN does not make any assumptions on the underlying context of data, however it does use feature similarity when it is to make a prediction based on a movie within the dataset as Subramaniyaswamy and Logesh (2017) states. It is also stated that the KNN model uses distance to calculate predictions about each movie within the database.

Because of this it is likely that a bias within movies which have a much larger quantity of reviews may be present within the system, this will be further addressed in the evaluation of the testing of this system.

When developing and testing this application it was apparent that data processing was important and essential as the data sets needed to be read correctly within the system, because of this Pandas was used, however there were persistent issues with the data as it was not being interpreted correctly and was unreliable, because of this there were modifications made to the data frame which moved the user ID into rows and Movie IDs into columns, the next step in fixing this problem was to implement a threshold value which would essentially tell the system if a user and a movie is worth adding to a list based on if they have rated a certain amount of movies or if the movie has a certain amount of ratings, this is to combat low number of ratings and data sparsity. in the prototype system this is handled by ensuring that the threshold is at 50 essentially meaning that the user must have rated at least 50 movies out of the whole dataset of movies. This also applies to the movies within the system, if they have not been rated at least 50 times they will not be included in the data frame, this is best practice as stated by Samin and Azim (2019) who proposed the method of removing ratings which return NaN or 0 and assign none negative numbers to each movie to achieve a more accurate result.

The final major issue which was ran into in the development of this project and which needed thorough testing was the value of the k within the KNN algorithm in this system. As stated by Subramaniyaswamy and Logesh (2017)When choosing the value for k it is important to pick a value which works well with the specific system and in this case data set, Essentially the system before hand had a much lower value of k which lead to inaccurate results for both the comparison of data and the actual final list output, however this was fixed by increasing the value, as having a larger value for k increases the accuracy of the results and reduces overall noise. The Historical optimal value for k has been known to be at 3-10.

Overall these were the major testing hurdles which had to be overcome during the development of this system, below is a testing table conducted to show results of the system as it stands currently.

2.2. Testing Table

Test	Predicted Result	Actual Result	Passed or Failed
Defining data path and the data files as variables	The code runs successfully without issue	Data gets loaded into the data frame with the defined columns and separators	p
Reading the files defined above as data frames using pandas	Data gets loaded into the data frame with the defined columns and separators	Data gets loaded into the data frame with the defined columns and separators	P
Using df.Shape to return a tuple representing the dimensionality of the Movie data frame	The shape of the Data frame is returned	The shape of the Data frame is returned	P
Using df.Shape to return a tuple representing the dimensionality of the Movie data frame	The shape of the Data frame is returned	The shape of the Data frame is returned	P
Pivot the rating information from ratings.csv to the movie features data frame to allow for parsing into a crs matrix	Using the head feature of Pandas, the matrix shows up with values	Using the head feature of Pandas, the matrix shows up with values	P
Import and assign the KNN model	Knn model is set and does not throw an error	Knn model is set and does not throw an error	P
Get the count of each rating score	Shows count in rating count data frame	Shows count in rating count data frame	P
Use numpy to normalise the data	Shows normalised count next to count in the data frame rating count	Shows normalised count next to count in the data frame rating count	P
Use matplotlib to show the output	Visualises data in a graph	Visualises data in a graph	P
Get movie rating count and review count of each user	Out puts tables and saves into data frame the count for both	Out puts tables and saves into data frame the count for both	P

	user review count and movie review count	user review count and movie review count	
Filter Data	Threshold of 50 is added to user review and movie reviews, the original data frames for the data are successfully shaped and dropped`	Threshold of 50 is added to user review and movie reviews, the original data frames for the data are successfully shaped and dropped	P
Running Method for Fuzzy Wuzzy which allows for string match making when in terms of searching for movies	The search input from the user should search the data frame for matching movies	The search input from the user should search the data frame for matching movies	P
Running method which uses the KNN methodology to search for similar movies	The search result returns movies with high ratings in the same genre	If Spiderman is input the result shows Spiderman 2 and the Dark Knight Rises as recommendations	P

3. Conclusion

During the course of this module and this professional practice week there were many challenges to overcome, however with a strong idea of what the final system should look like and behave it was possible to create an accurate system, this system met the initial goals and is able to predict and recommend movies to users based on a title input through the system. With a live data models attached to a live service it would be possible to take this system further and test the lengths and success of this application thoroughly, however due to time frames the system was evaluated on its own, with a two fixed data sets, one for reviews and one for movie ratings. These were merged using Pandas which allowed for accurate data analysis.

Initially the system was using a simple correlation metric along with user reviews and movie reviews to arrive at a recommendation list, however, through research and looking through past work, it was evident that using a system which incorporated sklearn was essential to have the system evolve and scale in an intelligent manner, because of this the final system was implemented using the KNN algorithm and methodology. Which allows for the system to make predictions based off of user reviews using the plurality vote of the nearest neighbour of the class.

Apart from this for the most part the system developed ran into minor issues which were expected beforehand due to research done on the previous professional practice week and the module at hand, for example, some work had been done on using the sklearn library previously, and Pandas work had been accomplished beforehand due to work needed for the CET 300 project.

Overall the system works in a proficient and expected manner, however there are some aspects which could be improved further or delved into with more experience, for example, due to the application using a KNN model, it has a natural bias to recommending Movies which have a higher quantity of reviews when compared to those which do not, along with this there arises an issue when a new movie is added to the system, if the system is fairly new and on a small scale website, then it will only recommend older Movies as they have higher interactions, this would cause a cycle of increasing the already popular and decreasing the new Movies added, this issue can be fixed by enabling a default value for each movie added to the database, and allowing for it to be recommended upon reviews solely until it reaches it’s first 100 or 1000 reviews. As the KNN method is a lot more intelligent than the previously incorporated method it is important to understand the scalability in a sense of larger live datasets can be an issue, when taking into account that the system stores every user and movie within the dataset matrix, whenever users join or new movies are added to the system, many scores and rating at 0 will be stored which essentially wastes storage space. However as this prototype system meets the requirements and these issues can be resolved with more experience and further development on a live service. The aforementioned issues can be resolved if one were to develop the system further using Spark’s ALS, if the system was pushed live, then as Spark.apache.org, (2020) states the use of the Spark ALS library would allow for the use of a coldStartStrategy parameter which would drop any rows in the data frame with NaN values, the evaluation metric will then computed over the none NaN values, this can be used to beat the popularity bias issue and the cold start of the recommendation system for new movies with low interactions. This however would only be viable through a live dataset as the current dataset is fixed and has static values.

4. References

Basilico, J. and Hofmann, T. (2004). Unifying collaborative and content-based filtering. *Twenty-first international conference on Machine learning - ICML '04*.

Help Center. (2020). *How Netflix's Recommendations System Works*. [online] Available at: https://help.netflix.com/en/node/100639 [Accessed 13 Jan. 2020].

Marovic, M., Mihokovic, M., Miksa, M., Pribil, S. and Tus, A. (2011). Automatic movie ratings prediction using machine learning. In: 2011 Proceedings of the 34th International Convention MIPRO MIPRO, 2011 Proceedings of the 34th International Convention. :1640-1645 May, 2011. [online] Opatija: University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, Zagreb, Croatia, pp.1640 -1645. Available at: http://eds.b.ebscohost.com/eds/detail/detail?vid=0&sid=d8c55b9e-de9d-4584-9ad3-85014ae18409%40pdc-v-sessmgr03&bdata=JnNpdGU9ZWRzLWxpdmUmc2NvcGU9c2l0ZQ%3d%3d#AN=edseee.5967324&db=edseee [Accessed13 Jan. 2020].

Samin, H. and Azim, T. (2019). Knowledge Based Recommender System for Academia Using Machine Learning: A Case Study on Higher Education Landscape of Pakistan. *IEEE Access*, 7, pp.67081-67093.

Spark.apache.org. (2020). *Collaborative Filtering - Spark 2.2.0 Documentation*. [online] Available at: https://spark.apache.org/docs/2.2.0/ml-collaborative-filtering.html [Accessed 16 Jan. 2020].
Subramaniyaswamy, V. and Logesh, R. (2017). Adaptive KNN based Recommender System through Mining of User Preferences. *Wireless Personal Communications*, 97(2), pp.2229-2247.

Werneck, R., de Almeida, W., Stein, B., Pazinato, D., Júnior, P., Penatti, O., Rocha, A. and Torres, R. (2018). Kuaa: A unified framework for design, deployment, execution, and recommendation of machine learning experiments. *Future Generation Computer Systems*, 78, pp.59-76.

4. GitHub- User: amkurian (2018) – Available at: https://github.com/amkurian/movie-recommendation-system/blob/master/recommender.py (Accessed on: 13/01/2020)

5. GitHub- User: jalajthanaki (2018) – Available at: https://github.com/jalajthanaki/Movie_recommendation_engine/blob/master/Lightfm_movie_recommendation.ipynb(Accessed on: 13/01/2020)

6. GitHub- User: PierreGe (2016) – Available at: https://github.com/PierreGe/RL-movie-recommender/blob/master/src/Data%20Filtering.ipynb (Accessed on: 13/01/2020)

7. GitHub- User: gauravtheP (2018) – Available at: https://github.com/gauravtheP/Netflix-Movie-Recommendation-System/blob/master/NetflixMoviesRecommendation.ipynb (Accessed on: 13/01/2020)

8. GitHub- User: KevinLiao159 (2018) – Available at: https://github.com/KevinLiao159/MyDataSciencePortfolio/blob/master/movie_recommender/movie_recommendation_using_KNN.ipynb (Accessed on: 13/01/2020)

9. Github User: Goktug (2017) – Available at: https://github.com/Goktug/knn-movie-recommender-system/blob/master/knn.py (Accessed on: 13/01/2020)