CIND 820: Literature Review, Data Description, and Approach

By: Daria Yip

Table of Contents

[Abstract 3](#_Toc127916287)

[Dataset Link 4](#_Toc127916288)

[GitHub Link 4](#_Toc127916289)

[References 4](#_Toc127916290)

[Literature Review 5](#_Toc127916291)

[Data Description 7](#_Toc127916292)

[Data Approach 18](#_Toc127916293)

[Data Approach Process 18](#_Toc127916294)

[Procedure 18](#_Toc127916295)

[Citations 20](#_Toc127916296)

# Abstract

In the past few years, with the advent of the Internet and the rise of instant gratification, companies who have been able to provide services on-demand have been the greatest winners. One of the companies that have led the transition to digital media consumption is Netflix. During the pandemic, they achieved even greater success through a significant increase in subscriptions. One of the technologies that streaming platforms like Netflix and its peers (i.e. Disney, Crave, Hulu) share is recommendation systems.

This paper looks at the logic behind recommendation systems and applying it to the Yelp dataset to provide personalized recommendations to the user based on data such as their reviews, ratings, location, and more. The dataset provides the information in .json form, and provides five files with information on users, businesses, reviews, check-ins, and tips. The sixth part of the dataset is in .jpg format, and will be disregarded for the purpose of this paper. Two common candidate generation approaches include content-based filtering, which is item-based filtering, and collaborative filtering, which uses user-based filtering (Google, 2022). The research questions that I will attempt to answer are: “For restaurant recommendations, do content-based filtering models or collaborative filtering models provide better suggestions?” and “What level of accuracy/performance can I achieve given these data points?”

To evaluate my models, I propose using the metric recall. As we are looking to create a Top-N recommendation list, recall is known to be effective in evaluating the ability of the model in finding relevant results (Cremonesi, P., & Turrin, R. (n.d.)). Currently, I am contemplating using the leave-one-out cross validation method (LOOCV), where it involves leaving a known rating to the side and later trying to predict it. Alternatives would be to use the k-fold cross-validation or the train-test split. As the Yelp dataset is very large, it may be computationally expensive to use k-fold cross-validation or the LOOCV methods. However, I will attempt to use the k-fold cross validation first, and then use the LOOCV, then the train-test split respectively.

I will be using Python to code this project. Tools that I am proposing to use include Python’s Pandas, Scikit-Learn, Numpy, Weka, and MatPlotLib libraries. With these tools, I will perform dimensionality reduction, data cleaning, exploratory data analysis, create training and test sets, and develop the content-based and collaborative. Then, I will evaluate this model with the aforementioned metric of recall.

# Dataset Link

Dataset Link: https://www.yelp.com/dataset

# GitHub Link

<https://github.com/dariayip/Final-Capstone-Project> (currently private)

# References

Google. (2022, July 18). *Candidate Generation Overview  |  machine learning  |  google developers*. Google. Retrieved January 23, 2023, from https://developers.google.com/machine-learning/recommendation/overview/candidate-generation

Cremonesi, P., & Turrin, R. (n.d.). *An evaluation methodology for Collaborative Recommender Systems - polimi.it*. Retrieved January 24, 2023, from https://chrome.deib.polimi.it/images/8/8e/Cremonesi\_2008\_CROSSMEDIA.pdf

# Literature Review

In a world where the burden of too many choices has become a widespread issue for consumers, recommender Systems (also known as RSs) have become a prevalent technology as a solution to this problem. RSs collect information on its users using explicit information such as ratings and reviews as well as implicit data such as monitoring users’ behaviour and sites visited [1]. RSs have become applicable to many diverse industries, finding a home in areas such as music (Spotify), streaming entertainment (Netflix), and e-commerce (shopping sites), among others [1]. Not only do RSs help facilitate the decision-making process for users, they are also advantageous for businesses as they boost sales, particularly sales of niche items [3].

While recommendation systems have become more common as of recent years, RSs already have a history of more than 29 years [4]. The first collaborative filtering model was proposed in the 1990s [4]. The technology of recommendation systems can be divided into a data mining part that collects data about items and users, and a recommendation filtering model part [4]. The best type of filtering to use, collaborative or content-based, depends on the field [4]. Content-based filtering is where recommendations are generated based on similar items, whereas collaborative based filtering is where recommendations are generated based on similar users [4]. For instance, for online social network services (SNS) such as Yelp, Facebook, and Instagram, collaborative filtering models and hybrid recommendation models are common due to the availability of user data [4]. The vulnerabilities of collaborative filtering models include cold start (for new users), sparsity, and gray sheep (users with unique and difficult to predict tastes) [4]. In order to address these vulnerabilities, many studies have been conducted, such as Yang et al. who studied the use of a collaborative filtering model and the Matrix Factorization Technique for SNS recommendation systems [4]. Matrix factorization is where a user-item matrix will be split into two lower-ranked matrices, one for items and one for users to assist with prediction accuracy when there are issues with sparsity [4].

Content-based techniques develop their recommendations by analyzing an item’s attributes. This technique is best used when suggesting content such as web pages, publications, and news [5]. In order to find similarity between documents, it can use models such as the Naïve Bayes Classifier, Decision Trees, or Neural Networks [5]. One advantage of this technique is that it does not require the profile of other users as it does not generate predictions based on the profiles of similar users like collaborative filtering does [5]. One disadvantage is that it requires in-depth knowledge and description of the features of the items [5]. Within the recommendation system technology, recommendation techniques such as text mining, KNN (k-Nearest Neighbours), Clustering, Matrix Factorization, and Neural Network have been mainly utilized [4].

On the other hand, for collaborative filtering techniques, they have the ability to generate recommendations based on users with similar profiles enjoy, thus creating a “neighbourhood” [5]. The technique of collaborative filtering can be further broken down into memory-based and model-based categories [5]. Within memory-based techniques, one can further break down the categories into user-based techniques and item-based techniques [5]. User based collaborative filtering (UBCF) calculates the similarity between users based on their ratings on the same item, whereas item based collaborative filtering (IBCF) computes similarity between items and not users [5]. For model-based techniques, they utilize historic ratings for data mining or machine learning [5]. Examples include the Singular Value Decomposition (SVD) technique, and regression and clustering [5].

When comparing the performance of SVD, SVD++, and NMF (Non-negative matrix factorization) in a study, the RMSE of the SVD method demonstrated the lowest average error rate, while NMF had the largest value [2]. Using MAE as the performance metric, NMF maintained the highest value, but SVD++ showed the smallest value [2]. The results showed that using SVD for collaborative filtering in order to predict restaurant ratings was appropriate and efficient as it had the lowest RMSE and the second-lowest MAE [2]. The author continues to state that the combination of reduced data storage requirements and improved accuracy is a great justification for the use of SVD on the subject dataset [2].

The use of RMSE and MAE are popular as they are easy to compute and simple to understand [6]. However, the author states that the use of MAE for classification tasks is not meaningful [6]. The paper also talks about precision and recall in terms of classification metrics that are popular and appropriate for Top-N models [6]. The author emphasizes that they should not be viewed on an absolute basis, but only on a comparative basis to compare different algorithms’ results on the same dataset [6].

# Data Description

The dataset is from the Yelp website, available for public download at <https://www.yelp.com/dataset>. At the time of download, the dataset contained 6,990,280 reviews; 150,346 businesses; 200,100 pictures; and 11 metropolitan areas. There were also 908,915 tips made by 1,987,897 users. Finally, there were also check-ins for each of the 131,930 businesses. The dataset is comprised of five .JSON files, named business.json, review.json, user.json, tip.json, and checkin.json.

**Business.json**

Within this .json file, there are 14 variables (Figure 2). Note, the variable is\_open is a numeric variable displayed in Figure 1 as having a count and mean, amongst other variables, but it comprises of “0” and “1” in order to signify if the business is open (“1”) or closed (“0”). Thus, the interquartile ranges as described below are not meaningful. There are five numeric variables, including latitude, longitude, stars, review\_count, and is\_open. Is\_open may be better classified as binary. Contains location data, attributes, and categories for the business. According to the df.agg function, there are 150,346 records, and 14 attributes/columns.

*Latitude: Latitude of the business’s location*

*Longitude: Longitude of the business’s location*

*Stars: Out of 5 stars, with 5 being the highest rating and 1 being the lowest, the rating of the business*

*Review\_Count: The amount of reviews the business has*

*Is\_Open: “0” stands for closed, and “1” stands for open*

**Table

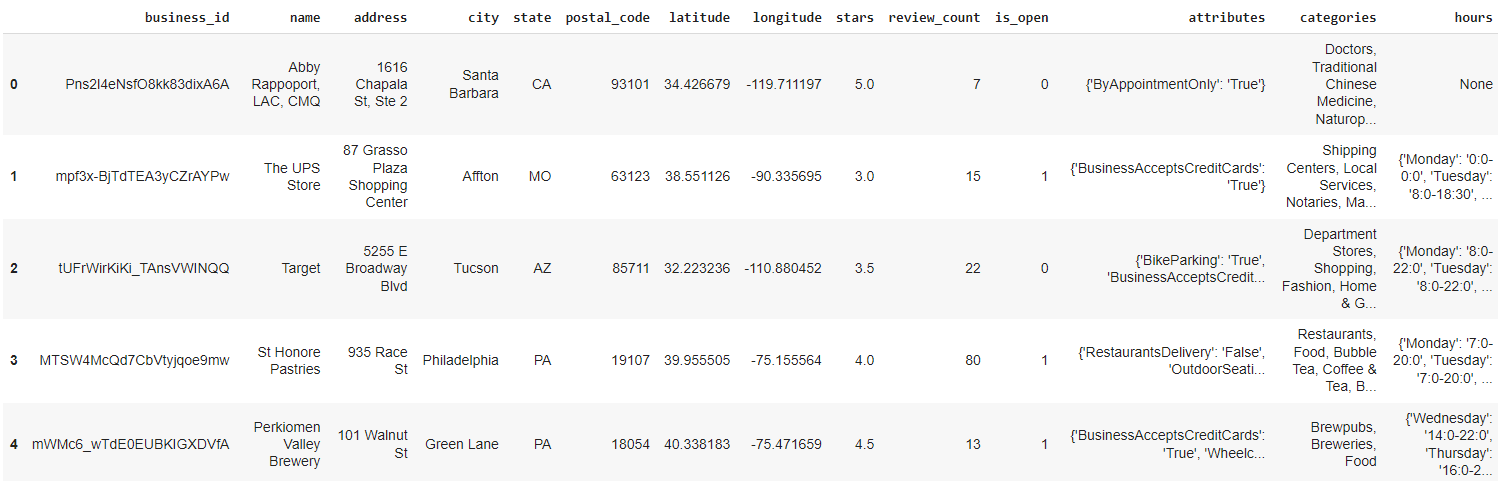
Description automatically generated**

*Figure 1: Summary statistics for the numeric attributes in business.json*

*A picture containing text, receipt

Description automatically generated*

*Figure 2: Variables for business.json*



*Figure 3: First 5 entries within the business.json dataset*

*Text, letter

Description automatically generated*

*Figure 4: df.agg*

**Tip.json**

Within this .json file, there are 5 variables (Figure 6). There is one numeric variable, compliment\_count. According to the df.agg function, there are 908,915 records, and 5 attributes/columns. Contains tips written by a user for a business – differ from reviews in that they are shorter and are there to convey fast suggestions.

*User\_id: Refers to the user\_id of the user who wrote the review*

*Business\_id: Refers to the business\_id of the respective business*

*Text: The full review text as written by the user*

*Date: Date Time format for the date and time the review was written, formatted as YYYY-MM-DD*

*Compliment\_count: How many compliments it has*

Table

Description automatically generated

*Figure 5: Summary statistics for the numeric attributes in tip.json*

*Text

Description automatically generated with medium confidence*

*Figure 6: Variables for tip.json*

*Graphical user interface, text, application

Description automatically generated*

*Figure 7: First 5 entries within the tip.json dataset*

A picture containing text

Description automatically generated*Figure 8: df.agg*

**Checkin.json**

*Within this .json file, there are 2 variables (Figure 10). There are no numeric variables, and only two string variables, business\_id and date. According to the df.agg function, there are 131,930 records, and 2 attributes/columns. Contains the date and the business\_id of the checkins.*

*Business\_id: Connected to business\_id in business.json. A string variable representing the ID of the checked-in business*

*Date: Date of the check-in, a string variable, a comma-separated list of timestamps for each check-in in YYYY-MM-DD HH:MM:SS format*

**Graphical user interface, text, application

Description automatically generated**

*Figure 9: Summary statistics for the attributes in checkin.json*

*Text

Description automatically generated with low confidence*

*Figure 10: Variables for checkin.json*

*Graphical user interface, text, chat or text message

Description automatically generated*

*Figure 11: First 5 entries within the checkin.json dataset*

*Text, letter

Description automatically generated*

*Figure 12: df.agg*

**User.json**

Within this .json file, there are 22 variables (Figure 14). There are many numeric variables, such as the ones shown in Figure 13 (ie. Review\_count). According to the df.agg function, there are 1,987,897 records, and 22 attributes/columns. User.json contains information and metadata about the user.

*user\_id: A string variable representing the ID of the user*

*name: A string variable representing the first name of the user*

*review\_count: An integer that stands for the amount of reviews the user has written*

*Yelping\_since: A string that holds a “YYYY-MM-DD” format date, representing when the user joined Yelp*

*Friends: An array of strings, representing the user’s friends written as user\_ids*

*Useful: An integer, the number of votes the user has sent that are “useful”*

*Funny: An integer, the number of votes the user has sent that are “funny”*

*Cool: An integer, the number of votes the user has sent that are “cool”*

*Fans: An integer, the amount of fans that the user has*

*Elite: An array of integers, contains the years that the user was “elite”*

*Average Stars: A float figure, representing the average rating of all reviews*

*Compliment\_Hot: An integer representing the number of times the user received the compliment “hot”*

*Compliment\_More: An integer representing the number of times the user has received the compliment “more”*

*Compliment\_Profile: An integer representing the number of times the user has received a compliment on their profile*

*Compliment\_Cute: An integer representing the number of times the user has received the compliment “cute”*

*Compliment\_List: An integer representing the number of times the user has received a list compliment*

*Compliment\_Note: An integer representing the number of note compliment received by the user*

*Compliment\_Plain: An integer representing the number of times the user received “plain” compliments*

*Compliment\_Cool: An integer representing the number of times the user received “cool” compliments*

*Compliment\_Funny: An integer representing the number of times the user received “funny” compliments*

*Compliment\_Writer: An integer representing the number of times the user received “writer” compliments*

*Compliment\_Photos: An integer representing the number of times the user received photo compliments*

Text

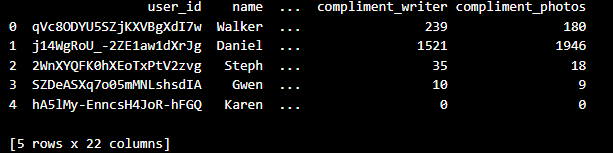
Description automatically generated

*Figure 13: Summary statistics for the attributes in user.json*

*Text

Description automatically generated with medium confidence*

*Figure 14: Variables for User.json*

**

*Figure 15: First 5 entries within the user.json dataset*

*A computer screen capture

Description automatically generated with low confidenceFigure 16: df.agg*

**Review.json**

Within this .json file, there are 9 variables (Figure 18). There are four numeric variables, such as the ones shown in Figure 17 (stars, useful, funny, and cool). According to the df.agg function, there are 6,990,280 records, and 9 attributes/columns. Review.json contains metadata about the review including the full text of the review, and contains information such as relevant user\_id and business\_id the review is written for.

*Review\_id: A string representing the unique ID of the review*

*User\_id: Maps to the user\_id in the user.json file. A string unique to the user who wrote the review*

*Business\_id: Maps to the business\_id in the business.json file. A string unique to the business who the review is written for*

*Stars: An integer, the star rating in the review*

*Date: A string, formatted as “YYYY-MM-DD” to represent the date the review was posted*

*Text: A string, the review itself*

*Useful: An integer representing the number of votes for “useful” the review received*

*Funny: An integer representing the number of votes for “funny” the review received*

*Cool: An integer representing the number of votes for “cool” the review received*

Text, chat or text message

Description automatically generated

*Figure 17: Summary Statistics for the attributes in review.json*

*A screenshot of a computer

Description automatically generated with low confidence*

*Figure 18: Variables for review.json*

*Text

Description automatically generated*

*Figure 19: The first 5 entries in review.json*

*Text

Description automatically generated*

*Figure 20: df.agg*

# Data Approach

The approach taken can be broken down into the following steps, with the smaller sub steps to the right. The graphic below describes this in detail, and the steps are explained in detail following the diagram.

## Data Approach Process

## Procedure

1. **Step 1: Data Preparation**

The process of data preparation can be split into loading the dataset and performing any necessary cleaning, transformations, and performing treatment of missing data. Other possibilities include handling special characters and creating strategies for error handling. Specific to the Yelp Dataset, it was provided in five files, with each file created as a .JSON file, one JSON object per line. As Pandas provides a way to read .JSON files through the pandas.read\_json functionality, I will keep the files in .JSON rather than converting to .CSV. Through this, I can load the dataset as a data frame in Python. Once loaded, one of the things that I am considering includes filtering the data set into restaurants only as well as focusing on reviews and users within a certain city. As there is approximately 8GB of semi-structured data, it is essential to strategize the model building in this step and keep the models and their required data in mind as I continue.

1. **Exploratory Analysis and Visualization**

After finishing the data cleaning and processing, the next step is exploratory data analysis and visualization. This step is instrumental in understanding the data better and beginning feature engineering/dimensionality reduction. Sub steps that I will be taking include generating summary statistics along with visualizations. Within this step, I can also perform correlation analysis to determine which variables/features are most important to the models in terms of predictive ability.

1. **Modeling**

Within this step, I will begin building out the models in order to answer my research questions. To answer my first question regarding whether content-based filtering models are “better” or collaborative filtering models are “better”, I will need to build both types of models and compare their metrics and recommendation lists to evaluate and answer this question. For my second research question regarding which level of accuracy/performance I can achieve given this dataset, I have decided to extend upon my original abstract and include the metrics of precision, recall, RMSE, and MAE within my analysis. According to my literature review, not only are these metrics popular and simple to understand, but I will also be able to measure the performance of my models on a deeper level and on different facets through the inclusion of these other metrics.

1. **Evaluation**

While this step may need to be performed several times along with step 3 (modeling) in order to evaluate the performance of the models accurately (or alternatively, with different variables/scenarios), the aforementioned four metrics will be integral in validating the models.

# Citations

[1] Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender Systems survey. Knowledge-Based Systems, 46, 109–132. https://doi.org/10.1016/j.knosys.2013.03.012

[2] Al-Nafjan, A., Alrashoudi, N., & Alrasheed, H. (2022). Recommendation system algorithms on location-based social networks: Comparative study. *Information*, *13*(4), 188. <https://doi.org/10.3390/info13040188>

[3] Silveira, T., Zhang, M., Lin, X., Liu, Y., & Ma, S. (2017). How good your recommender system is? A survey on evaluations in recommendation. *International Journal of Machine Learning and Cybernetics*, *10*(5), 813–831. <https://doi.org/10.1007/s13042-017-0762-9>

[4] Ko, H., Lee, S., Park, Y., & Choi, A. (2022). A survey of recommendation systems: Recommendation models, techniques, and Application Fields. *Electronics*, *11*(1), 141. <https://doi.org/10.3390/electronics11010141>

[5] Isinkaye, F. O., Folajimi, Y. O., & Ojokoh, B. A. (2015). Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, *16*(3), 261–273. https://doi.org/10.1016/j.eij.2015.06.005

[6] Cremonesi, P., & Turrin, R. (n.d.). *An evaluation methodology for Collaborative Recommender Systems - polimi.it*. Retrieved January 24, 2023, from https://chrome.deib.polimi.it/images/8/8e/Cremonesi\_2008\_CROSSMEDIA.pdf