Recommending Food Items Based on History of Users

Jaskaran Singh Gujral
Department of Computer Science and Engineering
Shiv Nadar University
Greater Noida, Uttar Pradesh
jg504@snu.edu.in

May 25, 2021

Contents

1 Introduction

In the fast growing era of Internet, we have seen a huge shift towards E-Commerce. The ease of purchasing a product or an item online with just a few clicks on your smartphone or personal computer has made the lives of people easier. Furthermore, the recent onset of the COVID-19 Pandemic has forced even more people to shift towards E-Commerce.

The limitless number of items present online presents a challenge that is how to make sure the customers or users see a product that they would like to buy. The attention span of humans is on an average very low, and if the right product is not showcased at the most opportune time then the user will not buy anything and the sales of the E-Commerce platform will drop.

Firms like Swiggy and Zomato face a similar problem but in terms of Food Items. This report details the work done by the author to explore the approaches that such firms can or have implemented to recommend food items based on the previous history of the customers.

2 Requirements

2.1 Python 3.6

https://www.python.org/downloads/release/python-368/

2.2 Jupyter Notebook

- After installing Python 3, run the following commands from the command line
- pip 3 install –upgrade pip
- pip3 install jupyter

2.3 Pandas

pip3 install pandas

2.4 Running the Code

- Double click on the Jupyter Notebook or navigate to the folder where you have unpacked the zip file in the command line and run the following command
- jupyter notebook

HTML and PDF versions of the Jupyter Notebook as well as the actual notebook are present.

3 Related Works

Recommendation Systems are used by almost everything present online from content streaming platforms like Netflix and YouTube to E-Commerce Platforms like Amazon and Flipkart to even search engines like Google. Some reasons why recommendation systems are so popular are:

- They help users find the right service or product. For example, 35% products on Amazon get sold due to recommendation systems [1].
- They makes services more personalized. For example, the most watched content on Netflix is due to the Content Streaming Giant's recommendation systems.
- They increase user engagement. For example, there is a 40% increase in the clicks of Google News due to personalized recommendation systems [1].

Three main types of recommendation systems are Content Based, Collaborative Filtering, and Hybrid of the two [5].

3.1 Content Based Recommendation Systems

In this, users and products are given profiles based on their characteristics, and if the characteristics of a user matches the characteristics of any product then those products are recommended to the user. A very basic example could be that if Jaskaran listens to pop songs then any song falling under the category of pop songs could be recommended to him. Of course, the characteristics will be much more than a simple category in an actual recommendation system but the essence is the same.

3.2 Collaborative Filtering

In this, the recommendations are based on the history of users and items taken together. There are mainly two types of Collaborative Filtering methods [2].

3.2.1 User-User

In this, items are recommended to a user based on the items preferred by users whose preferences are similar to the concerned user. For example, if Jaskaran and his friend have similar tastes then if his friend orders a Chocolate Brownie then the same will be recommended to Jaskaran as well. Again, this is a crude example and the actual implementation will have more items to contemplate, and the similarity between users has to be calculated as well.

3.2.2 Item-Item

In this, items which are similar to the items ordered previously by a user shall be recommended to the user. For example, if Jaskaran likes Chocolate Smoothie and orders it a lot then the recommendation system may recommend a Butterscotch Smoothie to him.

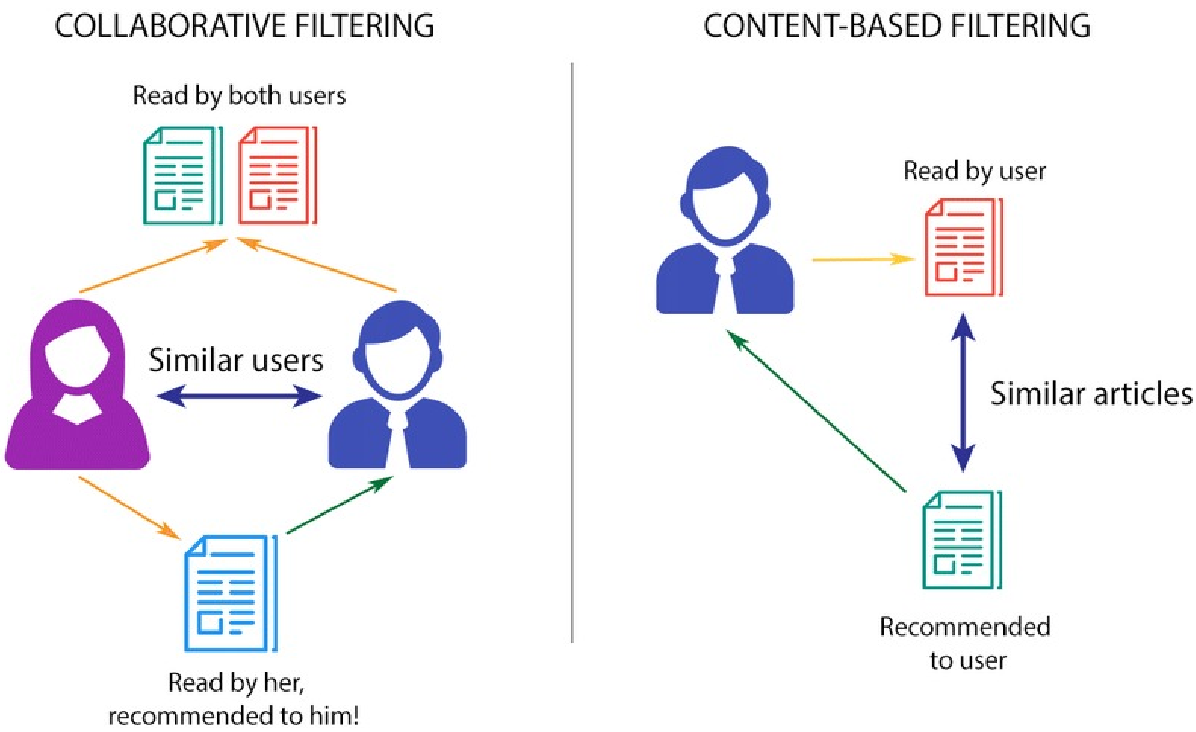

Figure 1: Difference Between Collaborative Filtering and Content-Based Filtering [3]

3.3 Hybrid Recommendation Systems

In this, both the history of users and items, as well as the characteristics of the users and items are taken into account before recommending anything. This type of model is usually developed using Deep Learning. The most common example of such a model would be YouTube's recommendation model [5]. Such methods may depend on content based recommendations if less data is available for a given user, and then move towards the hybrid approach as more data starts being available, and hence, increasing accuracy.

4 Methodology

Recommendation systems are mainly of three types as mentioned in the previous section. Based on that the author has created multiple approaches to the problem.

4.1 Item Ordered the Maximum Number of Times

This is perhaps the simplest algorithm that can exist for any recommendation system. In this approach, the author segregated the dishes ordered previously by the user that the system needs to recommend a dish to. This is done using the User ID of the user. From the segregated data, the frequency of each dish is calculated, and then the dish that is ordered the maximum number of times by the user is recommended.

- The first step is to segregate the data based on the user ID and store it in a variable termed 'data'.
- The second and final step is to find out the dish which has been ordered most frequently. This dish is the final recommendation by the algorithm.

4.2 Collaborative Filtering

Collaborative Filtering is one of the most popular approaches for any recommendation system, and the author has used this approach as well. The author has utilized User-User Collaborative Filtering to recommend a food item to the concerned user.

- The first step is to store unique dishes previously ordered by the concerned user and store it in a variable termed 'dishes'.
- The second step is to find the User IDs of all the users whose history (list of unique dishes ordered) is a superset of 'dishes' and store these IDs in a variable termed 'IDs'.
- The third step is to take out the data which corresponds to the 'IDs' that we have taken out in the previous step and store the data in a variable termed 'newData'.
- The fourth step is to store unique dishes in 'newData' that are not present in 'dishes' in a variable termed 'newDishes'.
- The fifth and last step is to use 'newDishes' and 'IDs' to find out the dish that is ordered the maximum number of times by users whose ordering habits are similar to the concerned user. This dish is the final recommendation by the algorithm.

4.3 Modified KNN

This approach is not a classical Machine Learning approach that is clearly defined. The author believes that it could be called a modified version of the K-Nearest Neighbours Algorithm.

4.3.1 K-Nearest Neighbours Algorithm

KNN Algorithm is a supervised Machine Learning algorithm which is used to solve both classification and regression problems [4]. It assumes that similar items exist in close proximity. It works by calculating the distance between a query and all the examples in the data, and then selects 'k' closest values and classifies based on the most frequent label or averages the labels in case of regression.

4.3.2 The Variation Used

- The first step is assigning numerical values to each dish based on the similarity of each dish. This step was done during the creation of the dataset itself and the specifics are mentioned later.

- The second step is averaging the value of each dish ordered by the concerned user. In this, each occurrence of a dish is taken into account and not just once.
- The third and final step is finding out the nearest value to the obtained value and finding out which dish it corresponds to. This dish is the final recommendation by the algorithm.

4.4 Popularity

This approach is yet another easy approach but one which is highly popular among recommendation system enthusiasts. It could be argued that this is not a Machine Learning approach per se because it is not personalized, that is, it does not change it's recommendation based on the user it is recommending the item to. To make it slightly personalized, we have only considered the cuisine which the user has ordered most frequently. The approach simply takes the average ratings of each dish of the relevant cuisine into account and then recommends the dish which has the highest average rating.

- The first step is to store all the cuisines ordered before by the concerned user in a variable termed 'cuisine'.
- The second step is to find the cuisine which has been ordered most frequently. This cuisine is stored in 'cuisine'.
- The third step is to store the data of all the rows having the cuisine as stored in 'cuisine' in a variable termed 'dish'.
- The fourth and final step is to find out the dish that has the highest average rating in 'dish'. This dish is the final recommendation by the algorithm.

4.5 Rating

This approach takes into account rating yet again but it gives personalized recommendations.

- The first step is to segregate the data based on the given user's ID and store the data into a variable termed 'dish'.
- The second step is to sort the data based on the average rating of each dish and store it in 'dish' again.
- The third step is to store the highest rating in a variable termed 'rating' and replace the data in 'dish' with the name of the dish having the highest rating.
- The fourth step is to store the user IDs of each user who has given a similar rating to the dish (between rating - 1 and rating + 1) in a variable termed 'IDs'.
- The fifth step is to store the data of all the users whose IDs were stored in 'IDs' in a variable termed 'newData'.
- The sixth step and final step is to find the dish which has the highest average rating in 'newData'. This dish is the final recommendation by the algorithm.

5 Experimentation and Results

5.1 Challenges Faced

- The dataset on which the author's method could be applied is not available on the internet due to privacy issues. Hence, the author had to create a new dataset on which to apply the methods.
- The lack of a pre-existing dataset and the fact that a recommendation system in itself is subjective, cross checking the validity of the algorithms has proven to be extremely difficult.

5.2 Dataset Creation

To create the dataset, the author initially took 12 Cuisines that are listed below.

- American
- Beverage
- Chaat
- Chinese
- Dessert
- Fast Food
- Indian
- Italian
- Mexican
- Mughlai
- South Indian
- Thai

In these cuisines, the author added 36 arbitrary dishes that are listed below.

- Aloo Tikki (2)
- Burrito (32)
- Butter Naan (3)
- Cake (24)
- Challupa (33)
- Chilli Paneer (17)
- Chilli Potato (18)

- Chocolate Ice-Cream (23)
- Chole Bhature (5)
- Chowmein (19)
- Dal Makhani (4)
- GolGappe (1)
- Gulab Jamun (25)
- Idli Sambhar (9)
- Kebab (13)
- Masala Dosa (11)
- Mutton Biryani (14)
- Noodles (20)
- Onion Uttappam (12)
- Palak Paneer (7)
- Pasta (30)
- Pav Bhaji (6)
- Pizza (31)
- Quesadilla (34)
- Raj Kachori (0)
- Rasgulla (26)
- Rice Bowl (21)
- Sambhar Vada (10)
- Shahi Paneer (8)
- Strawberry Smoothie (22)
- Subway (28)
- Tacos (35)
- Thai Curry (15)
- Tom Yam (16)
- Vada Pav (27)
- Veg Burger (29)

Now, from the list of these dishes, the author created a dataset of 100 users containing 10,000 entries. Each row has an ID, Cuisine, Dish, Rating, Value. The Ratings were generated randomly between 0 and 5 both included. The Values were generated based on the similarity of dishes and this was done manually due to the lack of a pre-existing dataset. For example, Shahi Paneer and Palak Paneer are similar items and hence their values were taken as 8 and 7 respectively. These values are written next to each dish above. The dataset generated using the described method was then saved in a CSV file named 'Data.csv'.

6 Conclusion and Future Scope

Earlier when eating out was still the norm, regular customers of a restaurant would sometimes not even have to order their food since the waiters would already know what they'd order. If not, the waiters would know what to offer. Similarly, platforms like Swiggy and Zomato work like online waiters by recommending food items that the users may like.

With the advent of the internet, E-Commerce platforms are here to stay. Recommendation Systems are the lifelines of these platforms and it would be hard to imagine a future where we were scrolling through Amazon and not see something that we like.

In the future, not far from now, Recommendation Systems will evolve even further, and will become the backbone of most of the online platforms, be it content delivering websites like Netflix, or food delivery apps like Swiggy and Zomato. In the future, with more data available to build upon, we can develop better and more accurate Recommendation Algorithms. We can connect this to social media platforms like Snapchat or Instagram where food bloggers or users who like to share their meals with the world can be taken into account, and those items could also be considered. Some highly known food bloggers can join and rate the food, and based on their rating, recommendations can be made.

The advances could be enormous, and the only thing restricting us is our imagination, and the data available to work on. Hopefully, in the near future we would not have to choose what we want to order at all because the recommendations would be perfect.

References

[1] Kaggle, last accessed on: 23^{rd} May, 2021

[2] Medium, last accessed on: 23^{rd} May, 2021

[3] Ngoupeyou Tondji, Lionel. (2018). Web Recommender System for Job Seeking and Recruiting. 10.13140/RG.2.2.26177.61286.

[4] towardsdatascience KNN, last accessed on: 23^{rd} May, 2021

[5] towardsdatascience Recommender Systems, last accessed on: 23^{rd} May, 2021