Amazon Fine Foods Recommendation System

Sasidev Mahendran, Harshavardhan Devaraj, Harish Raju Ganapathi Senthilkumar

ABSTRACT

Recommendation Systems are algorithms that aid users in product/service selection in the domain in which they are deployed. Recommendation systems are built on a bed of large quantities of data which serve as backbone in making decisions and recommendations to users in the particular domains. Recommendation systems incorporate a wide variety of Artificial Intelligence, Machine Learning and Data Mining techniques. Recommendation systems do not just make blind assumptions and make the recommendations, they take in personalized preferences and make user specific recommendations. In this project, we will try to build a recommendation system for a Fine Foods industry and discuss the techniques used and critical challenges faced.

INTRODUCTION

The project titled Amazon Fine Foods Recommendation System is a recommendation system built by us using the Amazon Fine Food Reviews dataset from Kaggle (Amazon Fine Food Reviews | Kaggle). This dataset is originally from Stanford University and contains reviews of a variety of fine food on Amazon. The data logged in this dataset span over 10 years and have around 500000 reviews up to 2012. The dataset has mainly information of the users, products, ratings, and a standard text review.

The main objective of the project here is to use various data mining techniques to clean, process and visualize the data to get a good understanding of how the data in the dataset looks like. Further we use various techniques to build a robust recommendation system which groups similar users together based on their reviews and recommends products which might relate to their liking based on these reviews. Various machine learning techniques have been applied to obtain these results.

DATA SOURCE

The dataset is obtained from a Kaggle competition and is present in a CSV format. The dataset primarily consists of reviews over a time period from Oct 1999 – Oct 2012. Around 548,454 reviews are present in the dataset which have been submitted by 256,059 users for 74,258 products. The initial dataset has around 10 columns which are ID, Product ID (Identifier for product), User ID (Identifier for user), profile name, Helpfulness Numerator (Number of users who found the review helpful), Helpfulness Denominator (Number of users who indicated if the review was helpful to them), Score, Time, Summary and Review. Most of these columns have a data type of int64 or object.

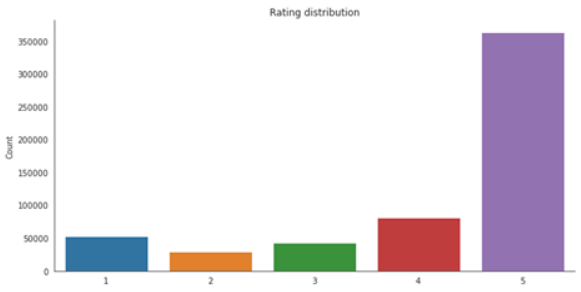
METHODS

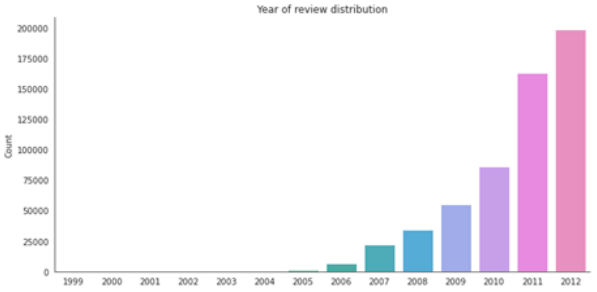
Data Pre-processing

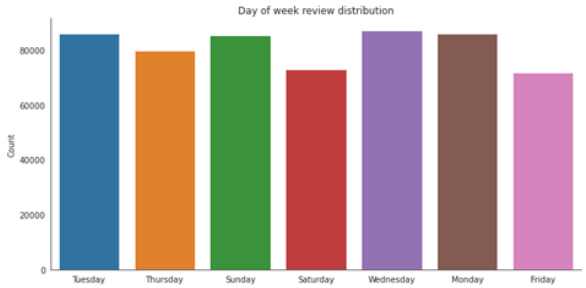
Since the time data in our dataset is in a scalar time series data format, the inbuilt pandas function (pd.to_datetime) is used to extract features into a usable date format. Further extraction from the date is carried out using (pd.DatetimeIndex) to obtain the year, day of week and month data to get a better understanding of the data present in the dataset.

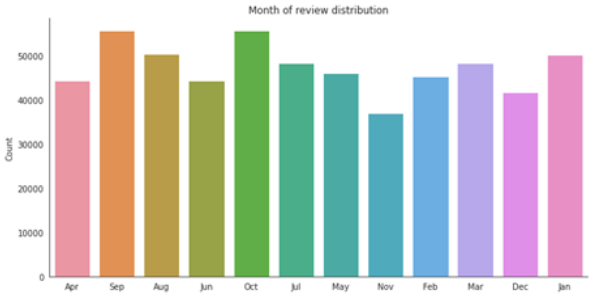
Data Visualization

To get a clear and better understanding of the data, basic data visualization was carried out.









The visualizations carried out show us that the majority of the reviews were 5-star reviews. A huge chunk of reviews came from 2012. The day of week distribution shows us that there is around a ±1000 review count difference between the various days of the week. There is a similar ± 1000 review count difference in the month distribution of the reviews as well. The Word Cloud model generated shows us the most frequent words occurring in the review's column. The larger the text size, the larger the frequency of the word occurring in the review column.

Modeling

To build our recommendation system, we need to build a robust model which will take in input data and apply various data mining and machine learning techniques to produce results in the form of recommendations. Recommendation systems are generally based on two types: Content based and Collaborative based. In this project we have implemented a version similar to the content based recommendation system.

A separate data frame holding ProductId, UserId and score is taken. This dataframe contains more than 500,000 rows with 3 columns. From our initial data, we can see that multiple users have submitted reviews for some products. Since the number of users is very high, it is very difficult to take into account all the data and build our model using this data due to constraints caused by memory issues and training time. Thus we randomly sample around 20000 users from the total user list. We then get the matching reviews made by those 20000 users and load it into a dataframe which results in ~43000 reviews. This data frame is used to create a matrix with UserId as row and ProductId as columns and score (rating) as the value. This matrix is used to compute the cosine similarities to identify similar users based on the ratings they have given to individual products.

To further try various models, a dataframe containing itemID, userID and rating were passed to a gridsearchCV model with SVD as the base estimator. This returned {'n_factors': 20, 'reg_all': 0.02} as the best parameters.

	test_rmse	fit_time	test_time
Algorithm			
KNNBaseline	1.072512	2.793663	0.130227
SVDpp	1.130172	2.942381	0.125000
SVD	1.188844	1.126602	0.052070
BaselineOnly	1.198309	0.062506	0.041659
NMF	1.231064	2.442430	0.088525

SVD, SVDpp, NMF, KNNBaseline, BaselineOnly models were further implemented to find the best models. These models are popularly used in recommendation systems and are part of python's surprise library.

From the results we see that KNNBaseline is found to be the best model with the lowest test rmse of 1.072.

Further gridsearch using KNN baseline provided us with the best parameter combination of {'k': 10, 'min_k': 1}. When these parameters were implemented individually in a KNNBaseline model, the following accuracy metrics were obtained. RMSE: 1.0646. After gridsearch, the rmse has dropped from 1.072 to 1.064 for the KNNBaseline model. This KNNBaseline model which is a collaborative filtering baseline model will now be used to perform recommendations for users. This algorithm works by clustering similar itemIDs together for the particular userID based on how ratings were given by other users for similar products.

RESULTS

```
The given UserId is : A10123JRH5MVEA
The similar users for the above UserId, based on their ratings is ['A3F20TLZIS4VRY', 'A151AUGCSXA18H', 'A1P0S2FVBUXTF6', 'ATREKVFZ4P3
N', 'AO5WLEFYKL5T2', 'A1JFNZ4UAMEZN3', 'A3IADNY5QO1LZX', 'A3IAP9SMS6DPFZ', 'A3IEDM84ASW3QW']
```

When a particular userID is given, the cosine similarity is used to find similar users with similar ratings for related products.

Based on similar users and their ratings, using the product ID we can extract the reviews made by those similar users.

From these reviews, we can see that Gerber Fruit Strips and Salt are the products being recommended to the user the most.

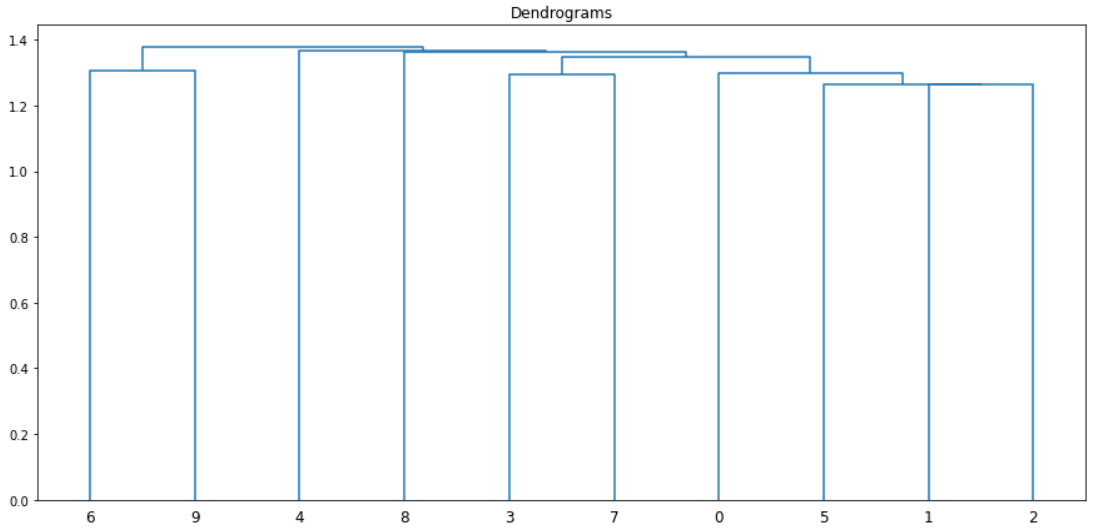
This method is similar to collaborative filtering and would allow us to recommend products to an user based on other similar users.

	index	0
0	0	I purchased alternative Gerber fruit strips to...
1	1	My kids picky fruit strips favorite treat Now ...
2	2	If looking yummy chewy treat I would suggest s...
3	3	Great deal great taste So much cheaper buying ...
4	4	My son loves snacks lunch car rides I used buy...
5	5	I used get Gerber fruit strips kids expensive ...
6	6	best tasting salt I ever tried The mineral con...
7	7	I friend turn coconut water running high amoun...
8	8	Be VERY careful eating cereal There tiny dried...
9	9	I buying Johnny Seasoning Salt years perfect a...

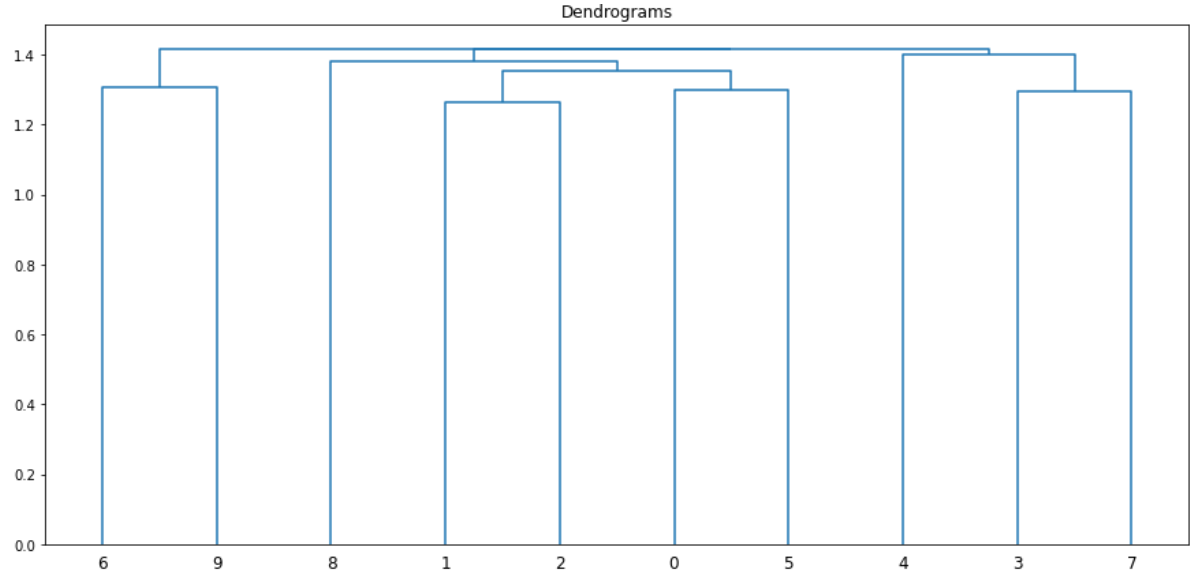
The products that could be recommended to this particular user would be the ones that the other users have rated the best.

We also took these reviews and used TFIDF vectorizer to vectorize them and plotted single link and complete link hierarchical clustering dendrograms to showcase the level of similarity between the reviews.

Single Link



Complete Link



Both dendrograms show us the same output level of similarity between similar reviews which backs up our model. Fruit strips, salt and other product reviews are clustered individually and both the dendrograms show their similarity level as well.

The best parameters were then used to implement a KNNBaseline Model to perform product recommendations for the user.

	index	iid	predictions	user_id
0	0	2734888454	4.218258	A2IW4PEEKO2R0U
1	12760	B002Y2QTEI	4.218258	A2IW4PEEKO2R0U
2	12758	B002Y2QSMQ	4.218258	A2IW4PEEKO2R0U
3	12757	B002Y2QSBW	4.218258	A2IW4PEEKO2R0U
4	12756	B002Y2QS0S	4.218258	A2IW4PEEKO2R0U

KNN baseline is a collaborative filtering algorithm which takes a baseline rating into consideration. This algorithm when given a particular userID gives us a recommendation of various products.

Thus we have successfully implemented a Recommendation system in two different ways of using cosine similarity and KNNBaseline and we have achieved the desired recommendation results as well.

DISCUSSION

We first thought of implementing a model which tokenizes the reviews from customers and applying techniques such as cosine similarity to find the level of similarity between the reviews and to recommend products to users based on such review similarity. Tokenizing the words was easy but we faced multiple issues in finding the cosine similarity as we kept running into memory issues due to the large size of the unpacked sparse matrix. We tried splitting the reviews up but were still facing memory issues and thus we adapted to a method to find similarity between how users rated products and extracted reviews from those products. This was a very innovative idea carried out by us which contributed to the overall

success of the project. Due to the massive size of the dataset,it was not feasible to calculate the similarity matrix using pivot option as it has some size limitation. This was a major difficulty. We tried to manually create a matrix, but that was very time consuming.So to fix this, the only way was to drop the data. But this has to be done without inducing any bias in the system. So we decided to pick a number of users by random sampling and dropping them.

We were successful in analyzing and designing a personalized recommender system based on both content based filtering and collaborative filtering.The recommendations were based on the past behavior of the user and also based on the other similar users.The techniques learned from this project have multiple use cases and can be used on a variety of datasets to recommend meaningful products and services to the end users.

KEYWORDS

Data Mining, Machine Learning, Kaggle, pandas, amazon, fine food, reviews, python, SVD, dendrogram, stopwords, NLTK, TFIDFvectorizer, WordCloud, word_tokenize, datetime, seaborn, cosine similarity, tokenizer, hierarchical clustering, Collaborative filtering. SVDpp, KNNBaseline, BaselineOnly, NMF

REFERENCES

- Sharedalal, R. (n.d.). *AMAZON FINE FOOD REVIEWS DESIGN AND IMPLEMENTATION OF AN AUTOMATED CLASSIFICATION SYSTEM*. Retrieved April 28, 2022, from https://etda.libraries.psu.edu/files/final_submissions/18847
- Chu, Y., & Wu, Y. (2015). Predicting Rating of Amazon Fine Food from Reviews CSE 190 Assignment 2.
- Binu Thomas and Amruth K John 2021 IOP Conf. Ser.: Mater. Sci. Eng. 1085 012011
- Sahu, Satya & Nautiyal, Anand & Prasad, Mahendra. (2017). Machine Learning Algorithms for Recommender System - a comparative analysis. International Journal of Computer Applications Technology and Research. 6. 97-100. 10.7753/IJCATR0602.1005.
- A. Fanca, A. Puscasiu, D. -I. Gota and H. Valean, "Recommendation Systems with Machine Learning," 2020 21th International Carpathian Control Conference (ICCC), 2020, pp. 1-6, doi: 10.1109/ICCC49264.2020.9257290.
- Parvattikar, Suhasini and Parasar, Dr. Deepa, Recommendation System Using Machine Learning (June 26, 2020). Proceedings of the International Conference on Recent Advances in Computational Techniques (IC-RACT) 2020, Available at SSRN: https://ssrn.com/abstract=3702439 or http://dx.doi.org/10.2139/ssrn.3702439
- https://www.kaggle.com/code/sunyuanxi/surprise/notebook
- https://surprise.readthedocs.io/en/stable/knn_inspired.html#actual-k-note
- https://medium.com/analytics-vidhya/collaborative-based-recommendation-system-using-svd-9adc5b6b3b8
- https://www.kaggle.com/code/bavalpreet26/recommender-system-part2/notebook
- https://github.com/NicolasHug/Surprise/issues/20
- https://learn.co/lessons/dsc-4-39-04-singular-value-decomposition-numpy-scipy-lab
- https://medium.com/analytics-vidhya/collaborative-based-recommendation-system-using-svd-9adc5b6b3b8
- https://towardsdatascience.com/building-and-testing-recommender-systems-with-surprise-step-by-step-d4ba702ef80b