M.Tech (Geoinformatics)

Ashish Bhardwaj

2022

COMPARISON OF VARIOUS RECOMMENDATION TECHNIQUES EMPLOYING LOCATION BASED SERVICES FOR E-COMMERCE WEBSITE

A DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE

OF

MASTER OF TECHNOLOGY

IN

CIVIL ENGINEERING

(With Specialization in Geoinformatics Engineering)

By

ASHISH BHARDWAJ

(2K20/GEO/03)

Under the supervision of

DR. (COL) K.C. TIWARI, PROFESSOR

MULTIDISCIPLINARY CENTER FOR GEOINFORMATICS

DEPARTMENT OF CIVIL ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi – 110042

MAY 2022

CANDIDATE'S DECLARATION

I, Ashish Bhardwaj, Roll No. 2K20/GO/03 student of M.Tech (Geoinformatics), hereby declare that the Dissertation titled "Comparison of various recommendation techniques employing location based services for e-commerce website**"** which is submitted by me to the Multidisciplinary Center for Geoinformatics, Department of Civil Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirements for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: **ASHISH BHARDWAJ**

Date:

MULTIDISCIPLINARY CENTER OF GEOINFORMATICS

CIVIL ENGINEERING DEPARTMENT

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi – 110042

CERTIFICATE

I hereby certify that the project titled "Comparison of various recommendation techniques employing location based services for e-commerce website**"** which is submitted by Asshish Bhardwaj(2K20/GEO/03) to the Multidisciplinary Center for Geoinformatics, Department of Civil Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirements for the award of the degree of Master of Technology, is a record of the project work carried out by the student under my supervision. To the best of my knowledge, this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place:

Date:

DR. (COL) K.C. TIWARI

SUPERVISOR

Professor, Multidisciplinary center of Geoinformatics

Civil engineering

Delhi technological university

(formerly delhi college of engineering)

Bawana road, delhi – 110042

ACKNOWLEDGEMENTS

First and preeminent, I express my profound sense of appreciation to my supervisor, counselor and advisor Prof. K.C. Tiwari, (Multidisciplinary Center for Geoinformatics), Department of Civil Engineering for their steady guidance, support, motivation and encouragement throughout the period this work was carried out. His readiness for discussion at all times, educational comments, concern and help have been deeply appreciated.

I also thank Mr. Kunj Bihari (Junior Research fellow) and Mr. Shivam (Project Assistant) Multidisciplinary Center for Geoinformatics, Department of Civil Engineering, Delhi Technological University, Delhi for their unconditional technical and moral support for guiding me. I express my deep condolence for their valuable counseling that led to the completion of this project.Without the excellent direction and tips provided by Miss Kainat Khan, PHD scholar, this project would not have been started.

I would like to thank Mr. Raghuvendra Aggarwala, Ms. Smita Sahu, Dr. Kumar Gravit the founding members of TorcAI Digital Media Pvt. Ltd.for developing my persona and teaching me all the latest technologies during my internship. I will always be thankful to them.

The credit for constant push I am getting because of which I was able to reach at this level is given to my Mother Ms. Pushpa Bhardwaj and my Father Mr. Rajeev Bhardwaj and all my friends esp. Badal Mohanty.

ABSTRACT

As data has grown tremendously over the last few decades, the management of data also follows its footsteps and created the ocean of opportunities for researchers to come up with best data management techniques. One aspect of data management that is widely used now-a-days is in the E-Commerce industry to showcase the relevant items to the users and predict which items the user will be most interested in. This hurdle Race allows the data scientist/Machine learning Engineers/Data Engineers and even Geo Data Scientist to give the user the best experience once he visits the website for online shopping. In other words, every website tries to showcase the limited products that the user might be most interested in rather than displaying all its trillion items and destroying the customer experience as he will get irated in searching for his product of interest. So the website should carefully design its recommended products palette as it can destroy as well as build the customer's experience. Another aspect is that some portals display the product that are far from the customer but these match the user's profile the most. It will also be of no use as the delivery cost as well as time will also be a deciding factor whether the customer will buy that product or not. So there is a need to build location based Recommendation techniques.

In this thesis work, it has been attempted to incorporate location based services into the traditional techniques i.e. Collaborative filtering and content based filtering and name them as "LOCOL" and "LOCONT" and the displayed results are in very good terms with the customer.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

List of states in Brazil:-

AC : Acre
AL : Alagoas
AM : Amazonas
AP : Amapá
BA : Bahia
CE : Ceará
DF : Distrito Federal
ES : Espírito Santo
GO : Goiás
MA : Maranhão
MG : Minas Gerais
MS : Mato Grosso do Sul
MT : Mato Grosso
PA : Pará
PB : Paraíba
PE : Pernambuco
PI : Piauí
PR : Paraná
RJ : Rio de Janeiro
RN : Rio Grande do Norte
RO : Rondônia
RR: Roraima
RS : Rio Grande do Sul
SC : Santa Catarina
SE : Sergipe
SP : São Paulo
TO : Tocantins

Abbrevations used:-

AI : Artificial Intelligence
CF : Collaborative Filtering
CBF : Content Based Filtering
GNSS : Global Navigation Satellite System
IALBRS : Interest Aware Location Based Recommendation System
IOT : Internet of Things
KNN : K Nearest Neighbor
LBS : Location Based Services
LBRS : Location Based Recommendation System
LOCOL : Location Based collaborative filtering

LOCONT : Location Based Content Based Filtering

MAE : Mean Absolute Error

RMSE : Root Mean Square Error

CHAPTER-1

INTRODUCTION

1.1 Motivation

Now-a-days, people are opting for online shopping over traditional shopping whether it's to buy a toothbrush or a washing machine. Everything is available online for purchase, sell, and even for rent. So every shopkeeper, wholesaler, and retailer wants their products to be available for every user. But the user does not want all of them and if that customer has to spend more time on searching his products of interest, he gets irritated. Hence there is a need for a Recommendation System (RS). It presents only those items to a customer which he might be interested in. The training process for the same begins from the past searches by him on that platform or similar users' searches behavior. There are various techniques available in the market and are employed by many e-commerce websites. There is a need to find which technique is the best and matches the customer requirements the most. Let's take an example if a customer A is located in Kashmir, he has ordered a shawl from an e-commerce website and from this purchase behavior, next time the RS should recommend items related to shawl like cardigan, socks, inners etc. The second scenario is that the users (with similar interests to customer A) located in the vicinity of customer A will be shown all the shawls in the same price range. So a RS system helps the customer to get only meaningful data out of the ocean of information.

1.2 General Introduction

Data is growing exponentially due to many IOT scenarios like every device around us is connected be it a smartphone, watch, lights, TV, washing machine, fans and the list Is never ending. All these devices generate data. For a user to filter out the meaningful information

from this ocean of information, this boon of data will become bane. The traditional system of filtering information for a user according to his interests can't be applied to other users because they are specific to a particular user. Hence there is a need to develop a system which satisfies the Requirements of most users.

Recommendation system is one good example in this regard [1]. There has been a surge in the use of Global Navigation Satellite System (GNSS) in recent times and geo-located data in real time has become more accessible. All these together contribute to Location Based Services (LBS). LBS plays a significant role in RS [2][18]. The location of the user can be utilized to provide the customized information. LBS very well plays with this information like in the navigation system[19]. But LBS is not limited to this only. The smartphones are portable and this unique feature is used by retailers to advertise their products virtually irrespective of time and space[16]. Hence customer targeting is getting easier with LBS and RS. The customers are also benefiting with this amazing combination of technologies [3][17].

1.3 Recommendation system

Recommendation systems are employed by many commercial websites to showcase products of their interests to its customers. The products can be show-cased based on various parameters like:-

1) Top sellers
2) Demographics like age, gender etc.
3) Past purchase behavior.
4) Products seen/bought by similar customers.
    1) Location based similarity.
    2) Interests based similarity etc.[4]

To achieve the above said purpose, there are various algorithms/techniques already proposed which make use of different parameters to recommend products that interests users the most. This requires the recommendation system to have many requirements like accuracy, effectiveness, temporal awareness etc.

Digital Data has grown massively in recent times and the number of users that visit the internet is also on the same footprint thereby creating the ocean of information available. This overloaded information many times hinders the retrieval of desired information when a non-tech savvy user tries to search the information of his interest. Although several platforms like Google, DevilFinder and Altavista have somewhat solved this problem but far from idealism. Hence there is urgent demand for a recommender system that filters out the information as per user's profile and interests to save his time and cost [6]. The inventions in this ever expanding recommendation system started back in the 1990s when content based and collaborative filtering were engulfing the market. Nearly every firm/corporates were adopting them. The famous Netflix challenge gave wings to the new researchers to explore more and find new techniques that were more trustworthy and imitates any anonymous user's interest areas, thus Matrix Factorization (MF) came into picture and was leading since then. But thus got halted by itself only. The main drawback of MF was that it was linear. It was going smoothly till the more complex and huge data came into picture. So, in the 2010s, deep learning became famous suddenly and has revolutionized everything. So the researchers were also trying their luck with the deep neural network to work in a recommendation system and it gave them very good results thus referred to as "neural recommender system". This has laid the stone for the next generation recommendation system. Even today the research in this area is still going on with an aim to proceed towards an ideal recommendation system [7]. Even though the current research is going on, the present recommendation system is not so accurate, effective and reliable. There have always been some security, piracy or other issues. It can't be applied to a broader range of items. This has to be intelligent enough to showcase only those products to users that have higher probability to be liked by users. It should display different results to users each time they come to that same site so as to increase their interests. The recommendation tray highly affects the business of any e-commerce website. This can develop the trust and belongingness between customer and publisher and can destroy it also [8]. Hence many categories of recommendation systems developed after that also mimicked the user's interests. For example, some of them utilize social information. If someone has changed their status to be "in relationship", then the recommender system would recommend him some chocolates, flowers, new dresses, teddies, some holiday packages, etc. this is termed as "social recommendation system". Some of them use side information which includes

relations, content on their profile, ratings over other products, association behavior, interaction with that product etc. [7]. Only accuracy is not the desired characteristics of an ideal Recommendation system algorithm. An accurate recommender will only showcase the same products to a user over time like a movie recommender system will only recommend a particular movie to a user each time that user visits that portal/web-site/app etc. In some cases, an accurate recommender system may not be an ideal one and vice-versa. Following are some significant parameters for choosing an ideal recommender system:-

1) The recommender system should display the results irrespective of topic. For example if someone searches for reliable, inexpensive, new mobiles, it should display diversity in phones of all brands. But if topic diversity is not there, it could have filtered the results first by reliability, then out of filtered results; it would have searched for low cost mobiles, and then applied the search criteria for new.

2) On applying the parameter- past choices of a user, a recommender system would have given the results which the user has already seen. Hence the results will not be vicissitude.

3) Sometimes the results of the recommendation system should have a desired property of happenstance for example- if a user has some favorite items stored on any ecommerce website, a user similar to him can be shown those items. So it will be an extra free and surprise topping on your pizza.

4) There should be diversity in results of any recommendation system each time the same user visits the same website. This kind of heterogeneity is a must have property.

5) The recommender system should also track the change in user's search behavior over time and this system should be learning on its own. Presently, various sophisticated AI systems have been developed for the same those are employed by some of the websites.

6) It should be intelligent enough to keep an account for seasonal fluctuations in preferences of any user. If the Diwali season is about to commence, the website should recommend items on that website related to that season in addition to other important parameters.

7) Recommendation system should keep the current location of the user to showcase the products to its users. For example if a user of Kashmir is going for a trip to goa. In goa, it should not recommend the same winter wear clothes, boots and other stuff.

8) It should be intelligent enough to recommend items based on age, gender, sexual orientation, ethnicity, social status, and other preferences of user. Recommending a couple holiday plans to a young single will be of no use to him.

9) There should be an option for the user to rate the recommendation results from time to time and to modify his preferences if the performance is below average. For example if a user earlier was searching for a job of data analyst and now wants to explore data science. He should be given an option for relaxing his preferences. Reinforcement learning will be best in this scenario.

10) The recommender system should display items based on knowledge which guesses the goal of a user from its past searches rather than past search products, or similar user's search criteria, or items discovered by user in its area. For example if a user is trying to search "how to write on screen during an online meeting" or similar things, then the knowledge based recommender system should suggest to him products like pen-tab etc.

In general, a Recommender system recommends to a user those products that a customer might be interested in. previous works have classified the old/traditional techniques into following categories:-

1) Content Based Filtering

2) Collaborative Filtering

1.3.1 Collaborative Filtering: Ratings given to any product by a user plays a significant role in this filtering technique. The similarity between users is determined by the ratings they give on the same product. The better rated products are showcased to similar users and have high chances of being liked by that user [8]. Collaborative recommender frameworks attempt to anticipate the utility of things for a specific client based on the items already evaluated by other clients[21]. In this, we tend to discover comparative clients and suggest what comparative clients like. In this sort of proposal framework, we don't utilize the highlights of the thing to prescribe it, or maybe we classify the clients into the clusters of comparable sorts, and suggest each client agreeing to the inclination of its cluster.[9]

Collaborative filtering channels data by utilizing the intuitive and information collected by the framework from other clients. It's based on the thought that individuals who concurred in their assessment of certain things are likely to concur once more within the future[20]. The concept is straightforward: when we need to discover an unused motion picture to observe we'll frequently inquire our companions for proposals. Normally, we have more prominent beliefs within the proposals from companions who share tastes comparable to our own[23]. Most collaborative filtering systems apply the so-called likeness index-based strategy. Within the neighborhood-based approach, a number of clients are chosen based on their similarity to the dynamic client. Deduction for the dynamic client is made by calculating a weighted normal of the appraisals of the chosen users[23]. Collaborative-filtering frameworks center on the relationship between clients and things. The similitude of things is decided by the similitude of the appraisals of those things by the clients who have appraised both things[8].

CF procedures utilize a database of inclinations for items by clients to anticipate extra points or items a unused user might like. In a normal CF situation, there's a list of m users} and a list of n things and each user has a list of items which he/she has rated in the past. The appraisals can either be unequivocal indications, and so forward, on a 1–5 scale, or understood signs, such as buys or click-throughs. For CF frameworks, creating high quality expectations or proposals depends on how well they address the challenges, which are characteristics of CF assignments as well. Let’s discuss some challenges in this regard:-

1. **Data Sparsity:-** In practice, numerous commercial recommender frameworks are utilized to assess exceptionally huge product sets. The user-item network utilized for collaborative filtering will in this way be greatly meager and the exhibitions of the predictions or suggestions of the CF frameworks are challenged. The information sparsity challenge shows up in a few situations, specifically, the cold begin issue happens when a unused user or thing has fair entered the framework, it is troublesome to find similar ones since there's not sufficient information (in some writing, the cold begin issue is additionally called the new client issue or modern thing issue. Modern items cannot be prescribed until a few clients rate it.
2. **Scalability:-** When numbers of existing clients and items grow massively, conventional CF calculations will suffer serious adaptability issues, with

computational resources going past down to earth or satisfactory levels. For example, with tens of millions of clients (M) and millions of distinct catalog things (N), a CF calculation with the complexity of O(n) is as of now as well expansive. As well, numerous systems need to respond promptly to online prerequisites and make recommendations for all clients in any case of their purchases and appraisals history, which requests a high scalability of a CF system

3. **Synonymy:-** Synonymy alludes to the propensity of a number of the same or exceptionally comparable things to have different names or sections. Most recommender frameworks are unable to find this idle affiliation and in this way treat these products in an unexpected way. For case, the apparently different items "children movie" and "children film" are genuine the same thing, but memory-based CF frameworks would discover no match between them to compute likeness. Without a doubt, the degree of changeability in graphic term utilization is more noteworthy than commonly suspected. The predominance of equivalent words decreases the suggestion execution of CF frameworks.
4. **Gray sheep:-** It alludes to the clients whose opinions don't reliably concur or oppose this idea with any group of people and hence don't benefit from collaborative filtering. Dark sheep are the inverse gather whose idiosyncratic tastes make recommendations incomprehensible. Although this could be a disappointment of the recommender framework, non-electronic recommenders moreover have awesome issues in these cases, so black sheep is a worthy failure.
5. **Shilling Attacks:-** In cases where anybody can provide recommendations, individuals may deliver tons of positive proposals for their possessed materials and negative proposals for their competitors. It is alluring for CF systems to present safeguards that debilitate this kind of phenomenon[11]

There are various techniques in this regard like memory based CF, model based CF etc.

1.3.1.1 Model based CF:-

Model based CF strategies expect the cloud ratings following learning to demonstrate from the basic information using machine learning or measurable strategies. Model-based CF strategies, for example, dependence systems, Bayesian and clustering models have to be

recognized to handle those shortcomings around memory-based CF approaches. Model-based techniques comprises Bayesian classifiers, relapse based strategies and cluster-based CF. The concept of the demonstrate based CF adaptations is to construct a show on the premise of the assessment information, or is to construct a few of the extraction of information from the information collection and dataset that is utilized and to create suggestions without having to use the dataset at a time.

1.3.1.2 Memory based CF:-

Memory based CF strategies point to utilize past information to predict the obscure rating depending on a few heuristics. They are working over the whole client database to foresee the results. We can discover that the commonly memory based strategies are based on the concept of closest neighbors, employing a differing quality of distance measures . The method of the foreseeing appraisals via referring to clients whose evaluations are comparable to the queried user based on neighborhood-based strategies, which are common memory-based CF approaches. With respect to the generation of memory-based CF strategies forecast, a test of the user item database is utilized. Each client has a place for a bunch of people that have comparable interfaces. The memory based CF has some features such as clarity. In other words, memory based CF is respected as a critical perspective of recommendation systems, ease of utilization, encouraging unused information effortlessly, and independent content of components that are suggested to extend the scope of a great rating with the common components. In expansion, the disadvantages of memory based CF are the issues on points clear and moderate. Too, the trouble of improvement Particularly in the setting of the genuine frameworks that generates recommendations in genuine time on the premise of huge datasets.

Memory based CF can also be of 2 types:-

1. Item based
2. User based

1.3.1.2.1 Item based:- There are two stages in item based collaborative filtering. Firstly, discover the similarities between things that are computed by utilizing one of the number similarity measures. Secondly, evaluate similarity values by using anticipated ratings for unknown things. Item-based nearest neighbor strategies are transpose of the user-based

nearest neighbor strategies. In any case, it can make expectations depend on likenesses among things.

1.3.1.2.1 User-based:- this collaborative filtering method for determining the similarity of the user by analyzing the elements participating in the vote between the user and that of the user uses the similarities and predicted weight in order to assess the important rating of the user on the efficiency of the element. [11]

1.3.2 Content Based Filtering: In this, only the customer's surfing behavior/past purchases/profile is used. The similarity here is product-user. Depending on product description and customer's profile, items are recommended. If a user has shown interest in getting admission to an online course, it can recommend the books, gadgets etc. related to that course [8].

Suppose a user U1 has watched a movie A of genre G1 that has the lead actor as A1 and the lead actress as A2 and has given rating 5 out of 5 to it. The next time he come back to the same platform he must be getting following recommendations of movies:-

1) Another movie B that has genre G1, actor A1 and actress A2.
2) Another movie C that has genre G2, actor A1 and actress A2.
3) Movie of genre different from G1 but has lead actor A1.
4) Movie of genre different from G1 but has the lead actress A2.
5) Movie of genre G1 but with different actors and actresses.

Likewise there can be possibly other combinations also. They are picking up the data from the user's past behavior. This is called "Content Based Filtering". What is happening in the above scenario, let's discuss that in detail [24].

A content-based filtering framework chooses things based on the relationship between the content of the things and the user's inclinations as contradicted to a collaborative filtering system that chooses things based on the relationship between individuals with similar preferences. PRES may be a content-based sifting framework. It makes proposals by comparing a client profile with the substance of each report within the collection [25]. The content of a report can be spoken to with a set of terms. Terms are extracted from archives by running through a number of parsing steps. To begin with, all HTML tags and halt words (words that happen exceptionally regularly and cannot be utilized as discriminators) are

removed. The remaining words are decreased to their stem by evacuating prefixes and suffixes. For instance the words “computer”, “computers” and “computing” may all be decreased to “comput”. The client profile is spoken to with the same terms and built up by analyzing the substance of archives that the client found interesting. Which archives the client found curiously can be decided by using either express or verifiable input. Unequivocal criticism requires the client to evaluate examined archives on a scale. In understood criticism the user’s interface is inferred by watching the user’s activities, which is more helpful for the client but more difficult to execute.[13].

CB methods make recommendations by analyzing the description of the things that have been appraised by the user and the depiction of things to be suggested. An assortment of calculations have been proposed for analyzing the substance of content archives and finding regularities in this substance that can serve as the premise for making suggestions. Numerous approaches are specialized adaptations of classification learners, in which the objective is to learn a work that predicts which course a report has a place to (i.e., either liked or not-liked). Other calculations would treat this as a relapse issue in which the objective is to memorize a work that predicts a numeric esteem (i.e., the rating of the report). There are two critical subproblems in designing a content-based sifting framework. The primary is finding a representation of reports. The moment is to form a profile that permits for inconspicuous reports to be suggested [14]

1.4 LOCATION BASED SERVICES

The availability of GIS technology and the Location-Based Social Networks (LBSNs) adds a new dimension of recommendations. They work on the user’s score to map the interests and to reduce the cold start problem they considered the user which is new in the system. Based on the location of users, LBS provides information which is very helpful to analyze the behavior of users. Decision-making ability, the interaction between users, and privacy on social media are some challenging features which affect the services. The location of the user can be used to list only items/stores etc that are popular within the vicinity of a user to better showcase recommended products. [7][18]

As each customer has his own states of mind and discernments toward his or her buying behavior, it is important to get it that each customer has his or her own point of view towards LBM [17]. In any case, there are a few common variables which impact the consumer acceptance of such administrations. More than half of the buy choices are made inside the store. Therefore, marketers are attempting to inform the buyer around an item as they pass by which clearly boosts the number of purchases.[15]

1.5 RESEARCH GAPS

There exist many wonderful models in the market, some are available from open source and others are privately owned by certain companies. The open source one are very few in number. With the introduction of Hybrid techniques, some algorithms fit on certain features of the ideal recommendation system. This creates a demand for building a still better recommendation system. Some MNCs have built robust systems and we see certain sections on their website that recommend items based on several criteria like past purchases, items popular in their area, related items, searches over their social media pages, etc. this itself covers so much area on webpage which may irritate the customers. Ideally, Recommended products should come under 1 palette and contain limited products which have all the features of an ideal recommendation system. This is still under research. One of the popular features of it is location based services. The recommender system should recommend the products in the vicinity of the customer so as to build customer's experience.

One another important feature is protection from malicious voting, the redundant reviews, fake comment/voting etc which can fail any robust model. As most of the model detects good products based on these criteria like voting, review score, reviews etc.

1.6 OBJECTIVES

Under this thesis, we will:-

1) Develop location based collaborative filtering (LOCOL)
2) Develop location based content based filtering (LOCONT)
3) Compare results of LOCOL and LOCONT

1.7 ORGANIZATION OF THESIS

The thesis are organized into following chapters :-

1) Chapter 1 contains all the theoretical background of Recommendation System and location based services, research gaps and objectives of study are covered.
2) Chapter 2 contains review of the various research papers studied for each topic that is being used in this thesis.
3) Chapter-3 contains information about the Data, Methodologies, Tools, Software and Libraries to be followed, It further contains the implementation, results and discussion of results of LOCOL. The data analysis is also discussed here.
4) Chapter-4 contains the implementation, results and discussion of results of LOCONT.
5) Chapter-5 contains the Geographic and Analytical comparison of results of LOCOL and LOCONT
6) Chapter-6 contains the conclusion and future prospects
7) After that the source code is attached in Annexure-A , Brazil state map is depicted in Annexure-B and data description in Annexure-C

CHAPTER-2

LITERATURE REVIEW

2.1 RECOMMENDATION ENGINE

Isinkaye, et al. (2020) have quoted that the recommendation system is a very helpful tool in the decision making process in complicated environments. It is characterized from the point of view of E-commerce as an instrument that makes a difference when clients look through records of information which is related to users' interest and inclination. It was characterized as a means of assisting and increasing the social handle of utilizing recommendations of others to form choices when there's no adequate personal knowledge or encounter of the options . Recommender systems handle the issue of data over-burden that users normally experience by giving them personalized, exclusive substance and benefit proposals.

Throat, et al. (2015) have quoted that the Recommender systems have ended up exceptionally prevalent in later years and are utilized in different web applications. Recommender Systems (RSs) are program instruments that are utilized to supply suggestions to clients agreeing to their prerequisite. The suggestions relate with different decision-making forms, such as which things to purchase, what music to tune in to. "Item" is the general term utilized to represent what the framework prescribes to clients. A RS regularly centers on a particular sort of thing, its design, its graphical client interface and the center recommendation technique used to produce the recommendations are all customized to supply valuable and effective proposals for that particular sort of thing. Due to the increasing significance of the proposal, it has ended up an autonomous inquiry about the field since the mid 1990s. Broadly speaking, a RS recommends to a client those things that may well be of interest to users intrigued.

2.2 COLLABORATIVE FILTERING

Mustafa, et al, (2017) demonstrated the broad classification of collaborative filtering and its sub types. Ratings given to any product by a user plays a significant role in this filtering technique. The similarity between users is determined by the ratings they give on the same product. The better rated products are showcased to similar users and have high chances of being liked by that user. It also depicted the basic mathematics behind all these subtypes.

Tan, et al. (2008) demonstrated how to deliver the precise and successful recommendations and guarantee the real-time necessity of the framework, analysts proposed a few distinctive algorithms, a few of which determine from the achievements of information mining. The e-commerce recommending calculations contains user-based collaborative filtering , Item-based collaborative filtering , Cluster-based collaborative sifting , Dimension decrease based collaborative filtering , Horting Graph-theoretic collaborative sifting , Bayesian arrange based suggestion , association rules based suggestion .

Roy, et al. (2020) depicted that Collaborative Filtering (CF) may be a well-known strategy utilized within the most RS. CF is the information filtering handle to discover the imperative designs among the clients which leads to future exercises . The main motivation behind the CF is that somebody can get the most excellent proposals from individuals with similar exercises. It works with the past user-item collaborations to foresee the conceivable outcomes. CF is of two sorts: user-based CF and item-based CF. The user-based CF strategy calculates the similarity between clients to analyze the forecast. It more often than not distinguishes clients who share comparative designs and generates a neighborhood of clients.it further depicted the wonderful measures for assessment of this algorithm.

2.3 CONTENT BASED FILTERING

Mustafa, et al, (2017) CBF approaches recommend items that are similar in content to the items that the user liked in the past or match to the attributes of the user [5]. In a content based filtering method, each item is represented by a feature vector or an attribute profile. The feature holds numeric or nominal values representing certain aspects of the item such as color,

price, etc. A variety of (dis) similarity measures between the feature vectors may be used to calculate the similarity between two items.

Gaudar, et al. (2015)In content-Based filtering proposals depend on clients previous choices. Thing depiction and a profile of the user‟s introduction play a critical role in Content-based sifting. Content-based sifting calculations attempt to suggest things based on similarity checks.

2.4 HYBRID FILTERING

Barve, et al. (2015)The hybrid filtering may be a mix and match of more than one filtering approach. It is presented to overcome a few common problems that are related with over sifting approaches such as cold begin issue, overspecialization issue and sparsity issue. Another thought process behind the execution of half breed filtering is to progress the precision and productivity of recommendation handling.

Adomavicius, et al. (2005) have depicted that a few recommendation systems utilize a hybrid approach by combining collaborative and content-based techniques, which helps to dodge certain limitations of content-based and collaborative systems. Different ways to combine collaborative and content-based methods into a hybrid recommender system can be classified as follows:

1. Executing collaborative and content-based methods independently and combining their predictions,
2. Consolidating a few content-based characteristics into a collaborative approach,
3. Joining a few collaborative characteristics into a content-based approach
4. Building a common binding together show that joins both content-based and collaborative characteristics

2.5 LOCATION BASED SERVICES

Albanna, et al. (2002) depicted that utilizing geotagged places, it is given a RS system to investigate open intrigued. They work on the user's score to outline the interface and to decrease the cold begin issue they considered the client which is unused within the system. Advances in area securing and versatile innovations driven to the expansion of the location

dimension to Social Systems (SNs) and to the development of a more current course called Location-Based Social Systems (LBSNs). Whereas LBSNs are wealthier in their demonstration and capacities than SNs, they fall flat so far to draw in as numerous clients as SNs. On the other hand, SNs have expansive sums of geo-tagged media that are under-utilized. In this paper, we propose an Interest-Aware Location-Based Recommender system (IALBR), which combines the preferences of both LBSNs and SNs, in order to provide interest-aware location-based suggestions.

Jaradat, et al. (2015) depicted that As each customer has his own states of mind and discernments toward his or her buying behavior, it is important to get it that each customer has his or her own point of view towards LBM. In any case, there are a few common variables which impact the consumer acceptance of such administrations. More than half of the buy choices are made inside the store. Therefore, marketers are attempting to inform the buyer around an item as they pass by which clearly boosts the number of purchases.

CHAPTER-3

LOCOL

3.1 OVERVIEW

Location based collaborative filtering (LOCOL) is developed by incorporating location based services into traditional collaborative filtering technique. The advantage it will offer over traditional technique is that the customer will be provided the results in the vicinity of him. The similar users calculation will be modified here. This will certainly add advantage to the recommendation system. Moreover the malicious reviews are also removed and filtered results are fed to the algorithm. To detect those malicious reviews, duplicity is detected by taking combination of unique customer id, product id and seller id and hence forth removed.

3.2 DATA USED

The data used here is open source data available on kaggle with the following link:-
https://www.kaggle.com/olistbr/brazilian-ecommerce

It was uploaded by Olist stores in 2018. Olist is a Brazilian startup that works within the e-commerce section, primarily through the commercial center.It is well spread inside the nation. The first Olist dataset has data of 100k orders from 2016 to 2018 made at numerous marketplaces in Brazil.

Its highlights permits seeing an arrange from different measurements from arrange status, cost, installment and cargo execution to client area, item traits and at last audits composed by clients. The whole data is available in 9 different csv files as follows:-

- **sellers**=olist_sellers_dataset.csv
- **customers**=olist_customers_dataset.csv
- **products**=olist_products_dataset.csv
- **payments**=olist_order_payments_dataset.csv

- **orders**=olist_orders_dataset.csv
- **order_items**=olist_order_items_dataset.csv
- **reviews**=olist_order_reviews_dataset.csv
- **english_names**=product_category_name_translation.csv
- **coordinates**=olist_geolocation_dataset.csv

The columns/Attributes in above files are as under:-

Seller:-['seller_id', 'seller_zip_code_prefix', 'seller_city', 'seller_state']

Customer:-['customer_id', 'customer_unique_id', 'customer_zip_code_prefix', 'customer_city', ‘customer_state']

Products:-['product_id', 'product_category_name', 'product_name_lenght', 'product_description_lenght', 'product_photos_qty', 'product_weight_g', 'product_length_cm', 'product_height_cm', 'product_width_cm']

Payments:- ['order_id', 'payment_sequential', 'payment_type', 'payment_installments', 'payment_value']

Orders:- ['order_id', 'customer_id', 'order_status', 'order_purchase_timestamp', 'order_approved_at', 'order_delivered_carrier_date', 'order_delivered_customer_date', 'order_estimated_delivery_date']

Order_items:- ['order_id', 'order_item_id', 'product_id', 'seller_id', 'shipping_limit_date', 'price', 'freight_value']

Reviews:- ['review_id', 'order_id', 'review_score', 'review_comment_title', 'review_comment_message', 'review_creation_date', 'Review_answer_timestamp']

English_names:- ['product_category_name', 'product_category_name_english'], dtype='object'),

Coordinates:- ['geolocation_zip_code_prefix', 'geolocation_lat', 'geolocation_lng', 'geolocation_city', 'geolocation_state']

The above datasets are interlinked by foreign keys as depicted in figure 1. Although some more foreign keys exist in this database which can be inferred from above attribute wise description of tables in this database which are termed as hidden keys.

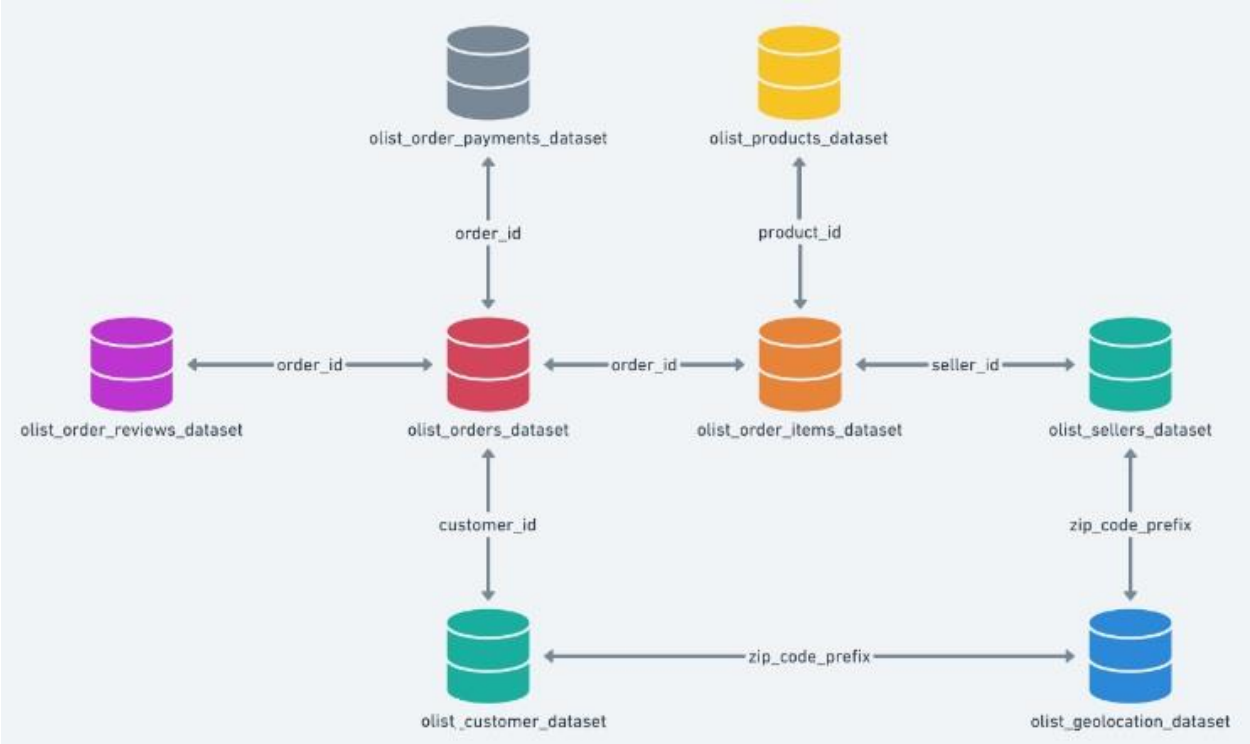

Figure 1:- Olist dataset interlinking foreign keys

(source: https://www.kaggle.com/olistbr/brazilian-ecommerce)

3.3 SOFTWARES, TOOLS and LIBRARIES USED

Table 1: Tools/softwares/libraries used

Tools/softwares/libraries	**versions**
Python	3.9
Microsoft visual studio	1.67.2
Jupyter	IPython : 8.2.0 ipykernel : 6.9.1 ipywidgets : 7.6.5 jupyter_client : 6.1.12 jupyter_core : 4.9.2 jupyter_server : 1.13.5 jupyterlab : 3.3.2 nbclient : 0.5.13 nbconvert : 6.4.4 nbformat : 5.3.0 notebook : 6.4.8 qtconsole : 5.3.0

	traitlets : 5.1.1
Pandas	1.4.2
Numpy	1.21.5
Surprise	0.1
Matplotlib	3.5.1
Seaborn	0.11.2
haversine	2.5.1

3.4 DATA ANALYSIS

The orders and customers dataset are merged to get "**number of orders per state**" and the results are displayed in following table 1:-

Table 2:- Number of orders per state

State code	number of orders	State code	number of orders
SP	41746	PA	975
RJ	12852	MT	907
MG	11635	MA	747
RS	5466	MS	715
PR	5045	PB	536
SC	3637	PI	495
BA	3380	RN	485
DF	2140	AL	413
ES	2033	SE	350
GO	2020	TO	280
PE	1652	RO	253
CE	1336	AM	148

This is graphically presented as a percentage of total orders through python in jupyter notebook. All those states whose total orders are less than 100, are grouped into the "others" category to simplify evaluations and shown in figure 2.

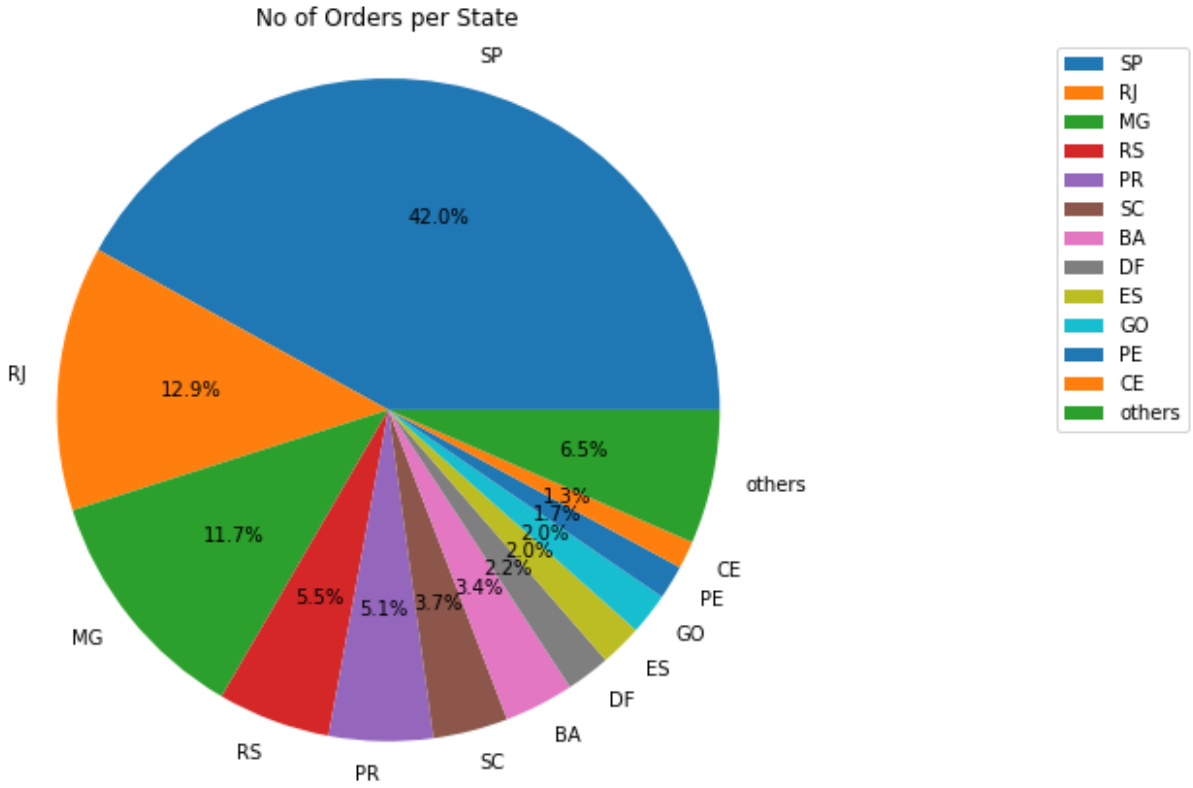

Figure 2:- Numbers of orders per state as a percentage of total number of orders

Now, for depicting the total number of sales per month, order and order_items dataset are merged to get the results and are plotted on a line graph in jupyter notebook through python and shown in figure 3 :- the y-axis denotes the sales/10000. The peak around 2017-10 to 2018-04 and 2018-05 to 2018-09 etc can be observed.

The initial effect or break even analysis can be seen here also, as in initial phases very less sales data is seen.

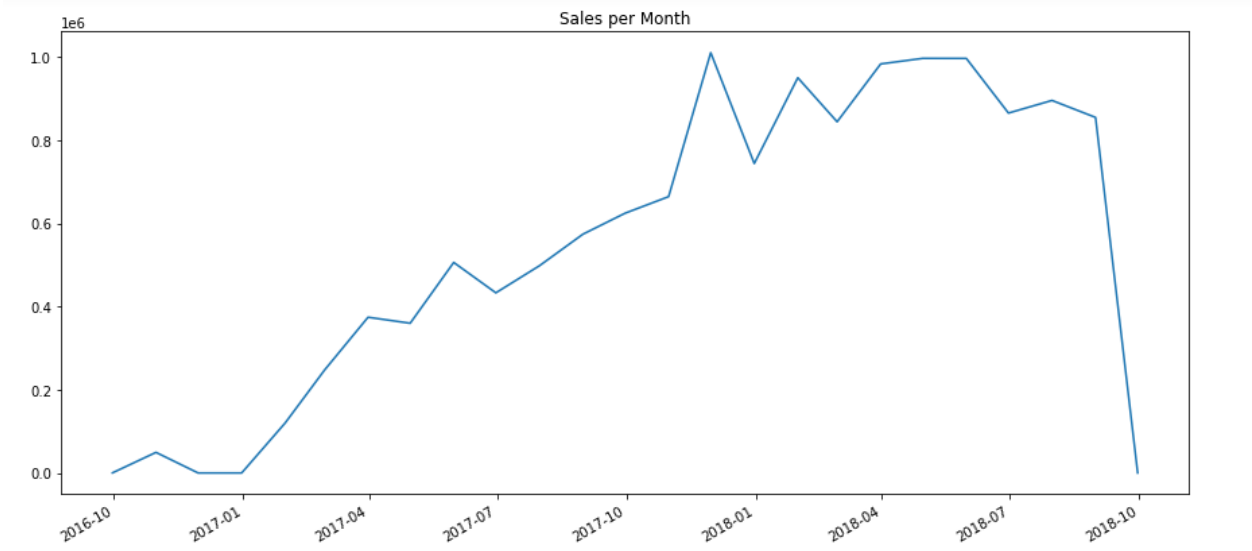

Figure 3:- Sales per Month Analysis

The payment modes are also analyzed and shown in a pie chart in figure 4. The values denote the number of transactions done by that payment mode as a percentage of total number of payments.

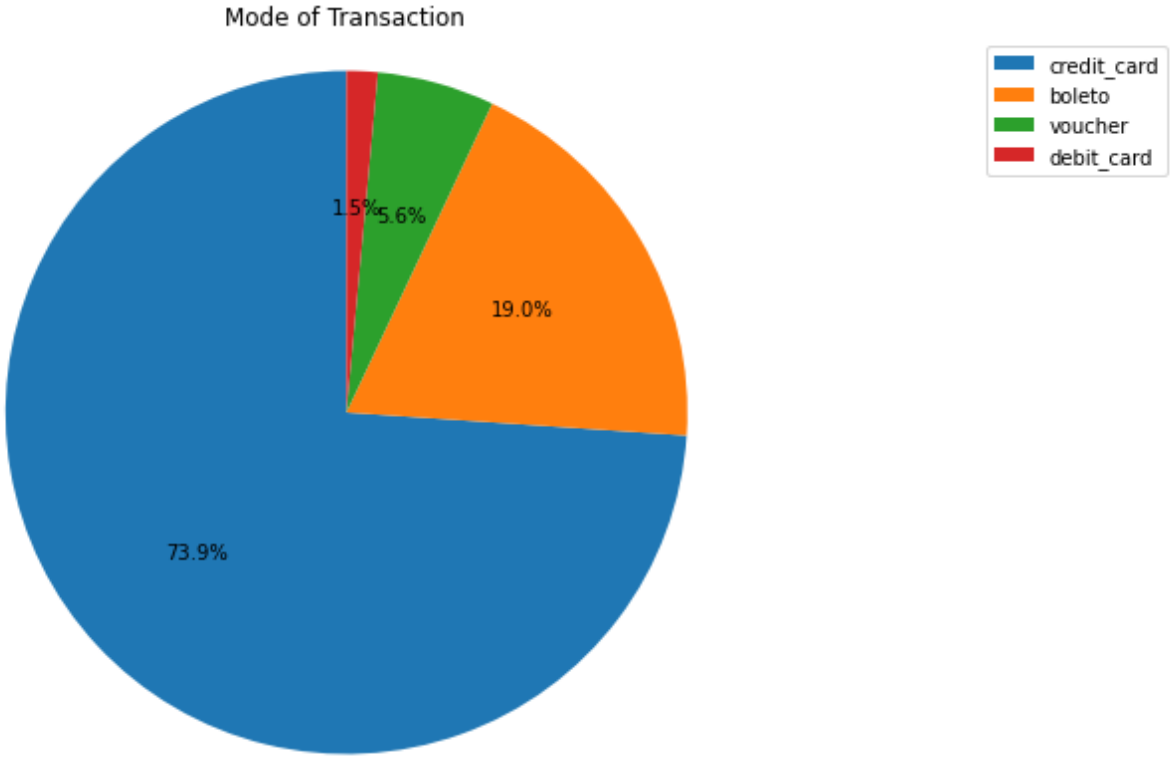

Figure 4:-Mode of transaction analysis

Next, this payment mode analysis is extended state wise. Here the y-axis represents the count of transactions done. It is depicted in fig 5.

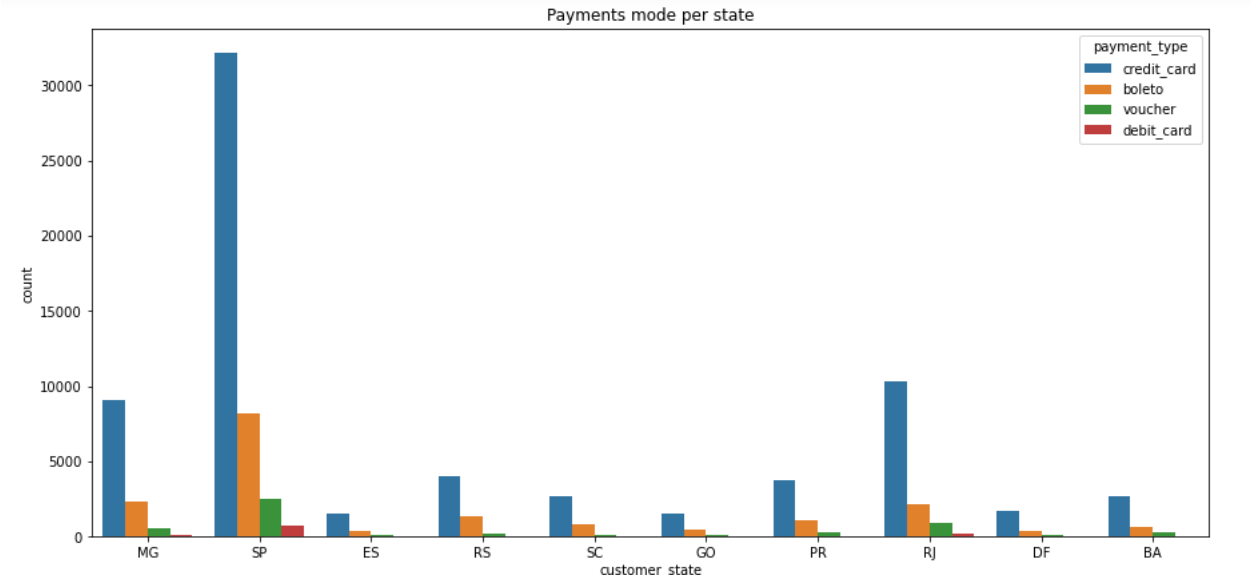

Figure 5:- Mode of transaction analysis state wise

Here top 3 performing states can be identified as “SP”, “RJ”, “MG” and hence are taken further in analysis part. The mode of transaction plays a vital role in any business and hence any company wants to know which mod of payment best suits the customer. In other words, which mode of payment is more convenient to users so as to stress more on that mode. From business point of view also, it is in close agreement. As the credit card here wins this analysis part, we can say that lots of people are still more comfortable in traditional mode of payment rather than the newer ones. To promote the newer methods, this company needs to provide attractive offers.

The product sales per state are analyzed by the categories of product in below multiple bar charts and result is as shown figure 6. The top 3 states sales wise are only taken into account.

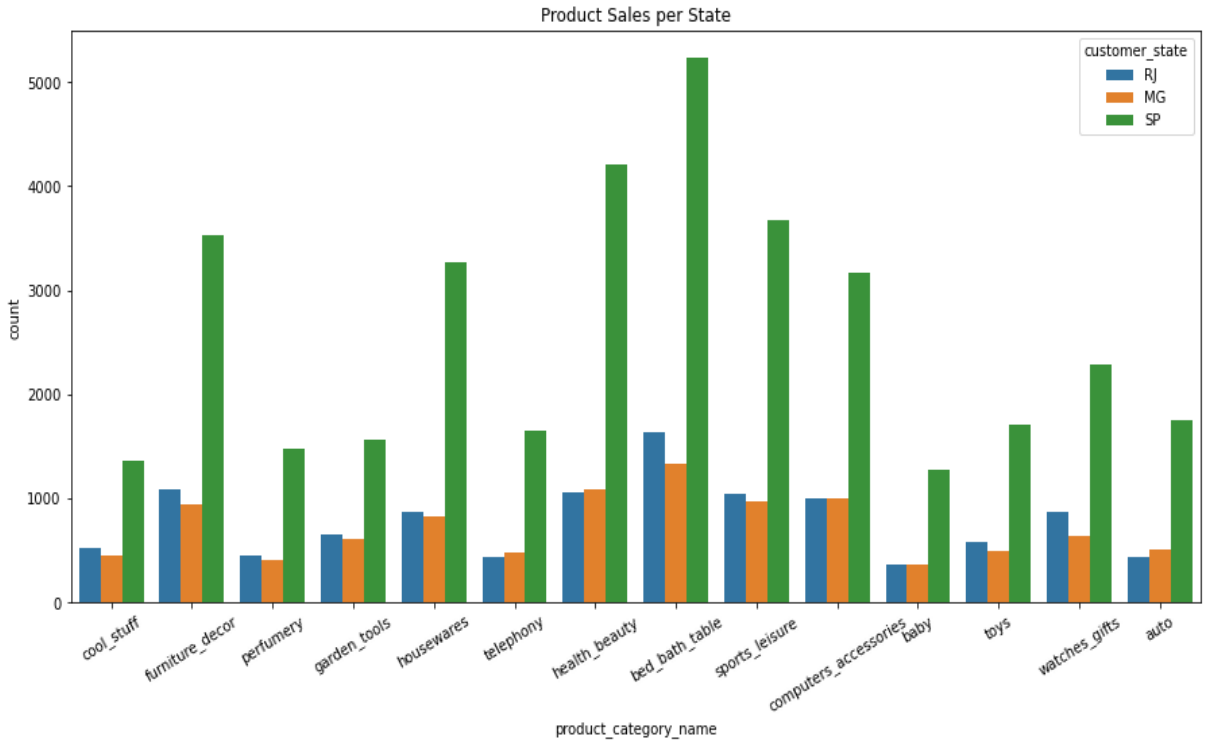

Figure 6:- Product sales per state w.r.t. Category of products.

3.5 METHODOLOGY OF LOCOL

The methodology followed here is quite simple we need filter the input of collaborative filtering so that limited data goes to algorithm. The complete methodology is divided into 3 parts :-

Part-1:- Location based filtering

Part-2:- Collaborative filtering

Part-3:- Final recommendation

From the user point of view, the user's location is used as input and recommendation results are displayed based on the similar user criteria. (here similar user means those users who have rated the products previously in similar fashion as done by him). The products purchased by similar users will be recommended to that user as they have higher chances of being liked by him.

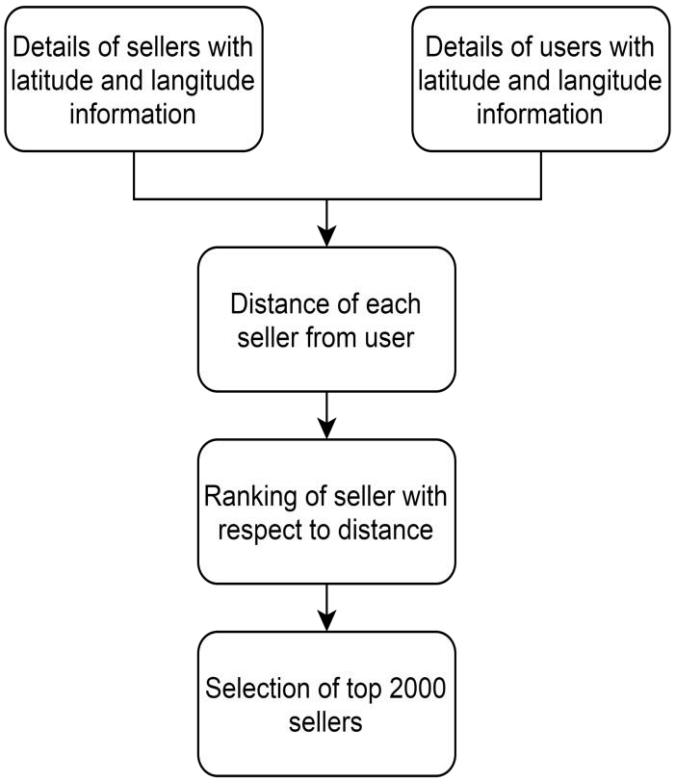

Figure 7:- LOCOL Part-1(Location based filtering)

Haversine distance is used for calculation of distance between user and seller as the haversine equation calculates the shortest distance between two points on circle utilizing their latitudes and longitudes measured along the surface. It is imperative for utilize in route. However it assumes that earth is a perfect sphere but is not hence some accuracy concerns are always there [28].

Singular value decomposition (SVD) is a matrix factorization method that generalizes the eigen decomposition of a square lattice (n x n) to any matrix (n x m). It is similar to Principal Component Analysis (PCA), but more common. PCA accept that input square matrix. PCA accept that input square matrix, SVD doesn't have this presumption. To calculate the dominant singular value and singular vector we may begin from power iteration strategy. This strategy may be balanced for calculating n-dominant particular values and vectors. For synchronous singular value decomposition we could utilize block version of Power Iteration[29].

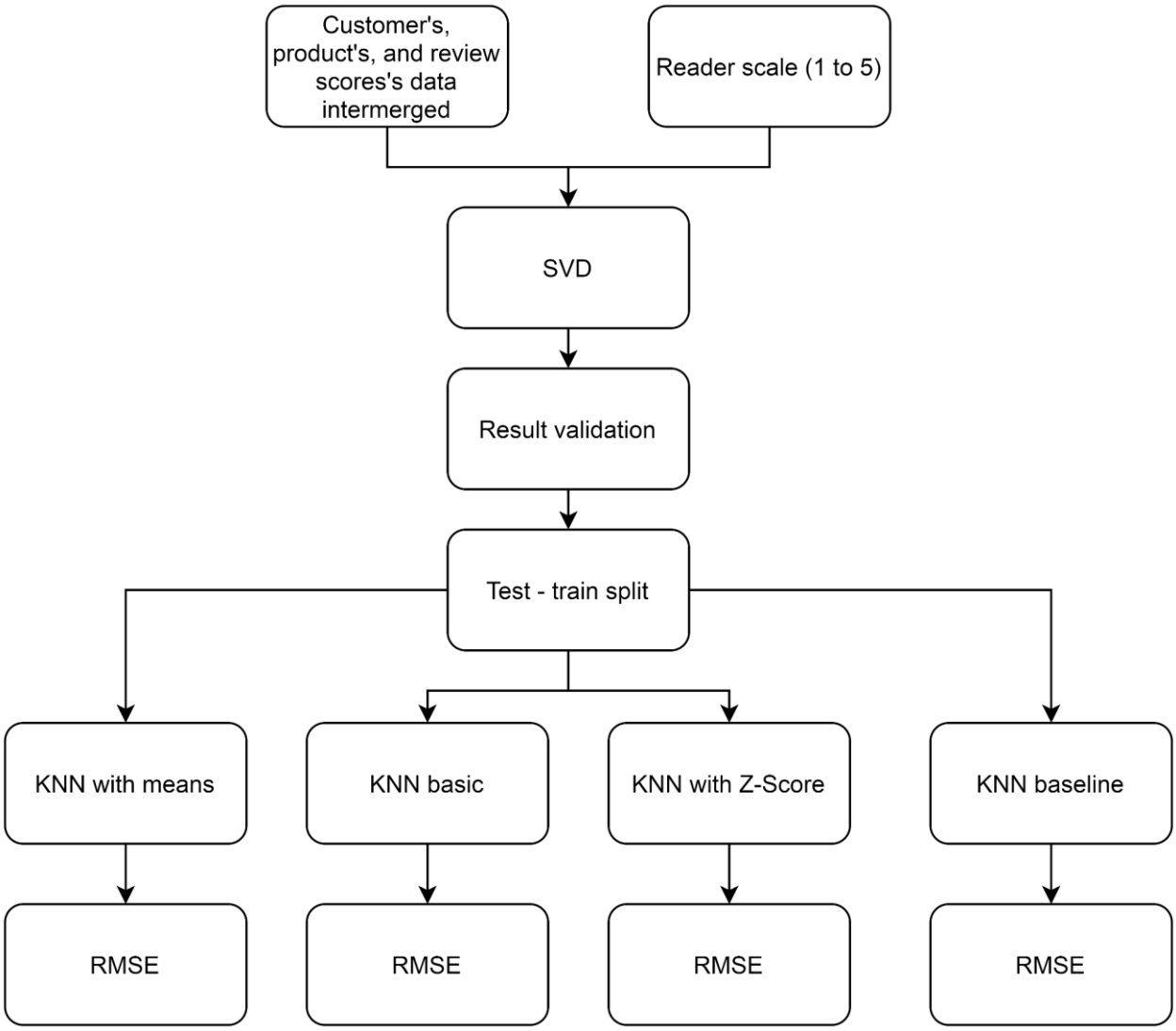

Figure 8:- LOCOL Part-2(Collaborative filtering)

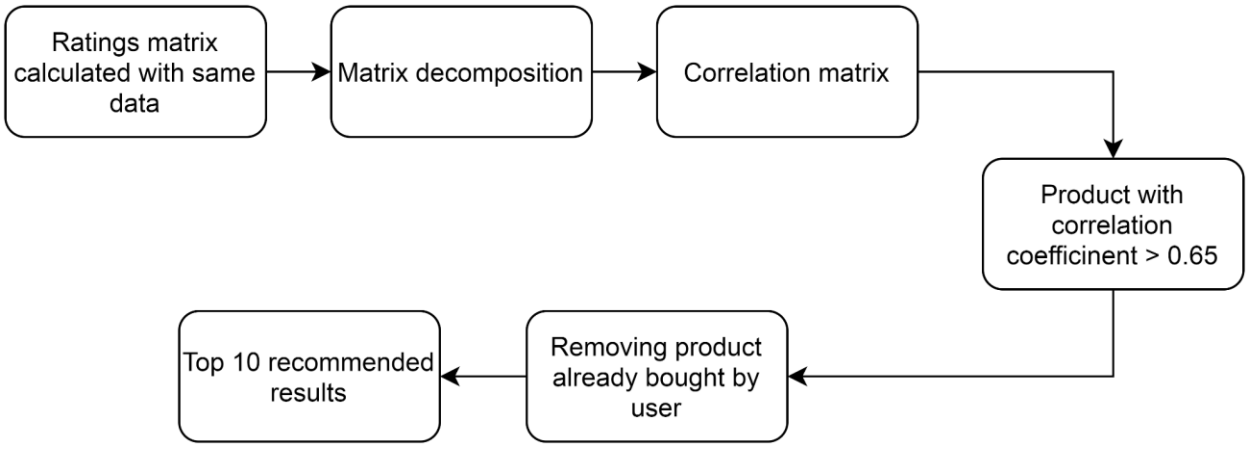

Figure 9:- LOCOL Part-3(Final recommendation)

3.6 IMPLEMENTATION AND RESULTS OF LOCOL

Locol has been run and following results are collected:-

The data is interconnected and following brief is calculated and displayed in fig 10:-

```
Total data
-----------------------------------------------------

Total no of ratings : 85441
Total No of Users   : 1356
Total No of products  : 617
```

Figure 10:- Data Brief

The whole data is arranged according to the number of ratings given by the user to find the most active users so as to ensure that its reference can be taken further. This is as shown in Fig 11.

```
customer_id
61a81040aee00781a11ef32ebf346f1a    325
9c31813dad0c6a85c3fc5ae81aadc0fe    256
848a418025af5ffbbadfdedd85081da7    220
9d58f19e8d5d306f2d15eb225c3adeb5    195
09d9f8180c32474c99e6d50b0400687b    195
                                   ...
9d640876024b3161ed854d9ac3d4c1f8     35
35de295c9053e824f7df2631a2b60b00     35
02be3c1e80d42583e71d6c2eb9cade45     35
3591f887df153da2719e91fbfb37817c     35
079da973ecd0fb654da4ffdb946c9ea1     35
Name: review_score, Length: 1356, dtype: int64
```

Figure 11:- Arrangement of customers based on count of ratings given by them

Let's dig into the product side also and see which products are highly rated. The results are shown in figure 12:-

```
product_id
d285360f29ac7fd97640bf0baef03de0    1782
574597aaf385996112490308e37399ce     840
4df83a41105e00e0845b9d12b5fe601c     810
2ff17002562478fb03cd44f09e7ca51a     810
103cc8c441a717d9a54fca312ae8ec3c     810
                                    ...
4b70e5e2b965baa6e9bae5b0d8a54ef7      35
299a30066c04cbcaa68a1c4271c87aaf      35
d66d9dba1b0d3aab8f3b6b02e4cd3f8e      35
7237404d77298a92c6f0af9184bdbafd      35
d23f2f3a0d52d3cb9effc0e11bae4bff      35
Name: review_score, Length: 617, dtype: int64
```

Figure 12:- Arrangement of products based on count of ratings given by them

After applying SCD in 5 folds, following RMSE and MAE are calculated and shown in fig 13:-

```
Evaluating RMSE, MAE of algorithm SVD on 5 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    0.0207  0.0200  0.0199  0.0201  0.0201  0.0202  0.0003
MAE (testset)     0.0136  0.0138  0.0137  0.0138  0.0139  0.0137  0.0001
Fit time          7.75    7.61    8.38    8.36    8.11    8.04    0.31
Test time         0.26    2.42    0.36    0.34    0.36    0.75    0.84

{'test_rmse': array([0.02068954, 0.02001849, 0.01992534, 0.02011383, 0.02006237]),
 'test_mae': array([0.0135742 , 0.01379597, 0.01370843, 0.0137967 , 0.01385962]),
 'fit_time': (7.750737190246582,
  7.6053078174591064,
  8.375831365585327,
  8.355842590332031,
  8.110999584197998),
 'test_time': (0.26284170150756836,
  2.4196157455444336,
  0.36177515983581543,
  0.33779215812683105,
  0.3557746410369873)}
```

Figure 13:- RMSE and MAE after applying SVD

After computing the cosine matrix similarity, RMSE is calculated w.r.t. to

1)KNN with means

2)KNN basic

3)KNN with baseline

4)KNN with z-score

This is shown in fig -14.

```
Computing the cosine similarity matrix...
Done computing similarity matrix.
RMSE of  KNNWithMeans
RMSE: 0.0426
Computing the cosine similarity matrix...
Done computing similarity matrix.
RMSE of  KNNBasic
RMSE: 0.0000
Computing the cosine similarity matrix...
Done computing similarity matrix.
RMSE of  KNNWithZScore
RMSE: 0.0444
Estimating biases using als...
Computing the cosine similarity matrix...
Done computing similarity matrix.
RMSE of  KNNBaseline
RMSE: 0.0182
```

Figure 14:- After computing cosine similarity matrix, RMSE with KNN with means, KNN basic, KNN with baseline and KNN with z-score

Finally, after removing the products already purchased by user, following items are recommended as shown in figure 15 as follows:-

	product_id
0	00ffe57f0110d73fd84d162252b2c784
1	014d94f219fbff1166b9cf700eee36b2
2	03894c9a40bf32b84a349ef0ddfccc1d
3	04d5a0ec1f6db00db01676199151df4c
4	0527af46c46b7a345d2086ea304ae8d7
5	05c24fd82c39e05337fcc418f0a61e54
6	0ac092fcd35bffaeb624017d498172d1
7	0b28e43610654365d0266f77070a71d9
8	0b6a32b32a7dc68b12e8a0e0805c44d5
9	0c52fe573de020553e4f524ac4267c92

Figure 15:- LOCOL Recommended products.

Let's see how far these are from the user's location i.e. (-22.898536428530225,-47.063125168330544)

The distances are calculated and as shown in figure 16 as follows:-

	product_id	geolocation_lat	geolocation_lng	distance
0	00ffe57f0110d73fd84d162252b2c784	-22.738242	-46.895859	24.730522
1	014d94f219fbff1166b9cf700eee36b2	-22.828655	-47.267296	22.315674
2	03894c9a40bf32b84a349ef0ddfccc1d	-22.738242	-46.895859	24.730522
3	04d5a0ec1f6db00db01676199151df4c	-22.828655	-47.267296	22.315674
4	0527af46c46b7a345d2086ea304ae8d7	-22.689713	-46.982493	24.647486
5	05c24fd82c39e05337fcc418f0a61e54	-22.828655	-47.267296	22.315674
6	0ac092fcd35bffaeb624017d498172d1	-22.828655	-47.267296	22.315674
7	0b28e43610654365d0266f77070a71d9	-22.828655	-47.267296	22.315674
8	0b6a32b32a7dc68b12e8a0e0805c44d5	-22.828655	-47.267296	22.315674
9	0c52fe573de020553e4f524ac4267c92	-22.828655	-47.267296	22.315674

Figure 16:- Distance of recommended products from user by LOCOL

3.7 DISCUSSION

As we are able to visualize Sao Paulo has got most number of orders coming to up to 15k. We can see the list of best 15 cities having most orders, and the list is: ['sao paulo', 'rio de janeiro', 'belo horizonte', 'curitiba', 'porto alegre', 'salvador', 'guarulhos', 'brasilia', 'sao bernardo do campo', 'osasco', 'santo andre', 'niteroi', 'recife', 'goiania']

There are 5 types of strategy utilized for shopping and credit card has been utilized most sharing 73.92% among other sorts of payment. Boleto is bank wallet sort of mode and able to say that Olist gives way better voucher payments because it has got more sum of share than debit cards.

The product category "bed-bath-table" has most number of orders (around 5000) followed by "health-beauty" (around 4000). This indicates the potential of commerce in these sectors.

Further, the sales from 2017-11 to 2018-01 shows a peak which is obviously due to Christmas, New year etc. and another peak is observed around 2018-04 which can be a result of Easter, Carnival of Rio De Janerio etc.

The results are fairly good as the farthest seller in recommended products by LOCOL is 24.7 Km and the nearest one is just around 22.3 Km away. However we can get more accurate results if we take whole dataset into consideration.

CHAPTER-4

LOCONT

4.1 OVERVIEW

Location based content based filtering (LOCONT) is another technique proposed in this thesis work. Here the past purchases of customer are taken into account but only those items are recommended which are available in the vicinity of him. Like in LOCOL, here also malicious voting are taken care of and removed accordingly.

Same data, tools, softwares, etc. are considered here as done for LOCOL and can be referred from section 3.2,3.3,3.4 and 3.6.

4.2 METHODOLOGY OF LOCONT

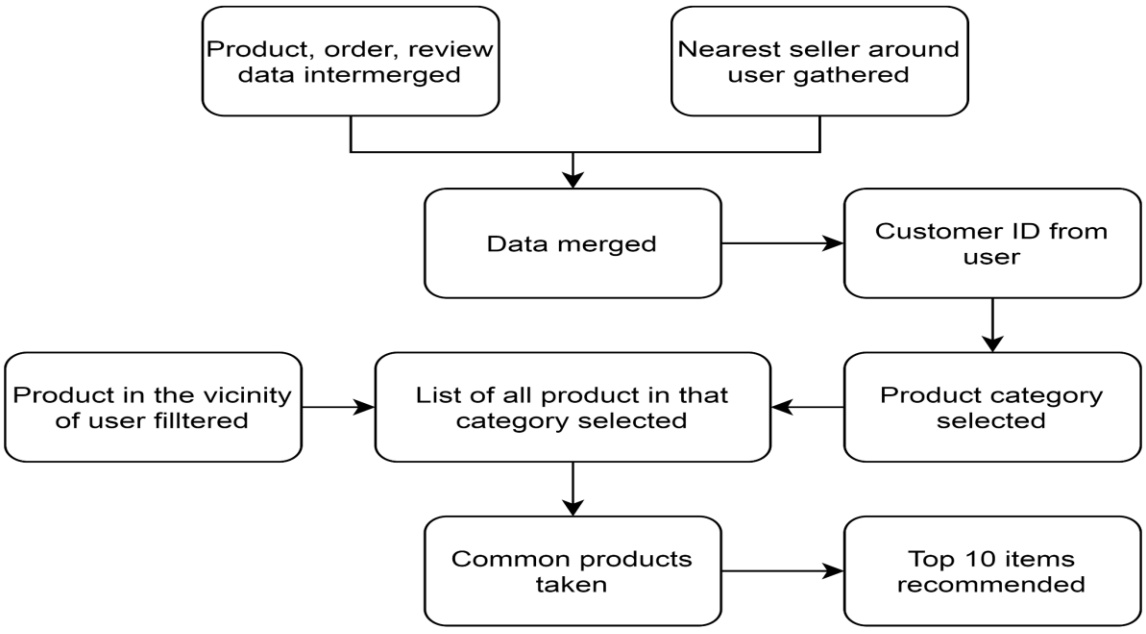

Figure 17: LOCONT Algorithm

4.3 IMPLEMENTATION AND RESULTS OF LOCONT

For this algorithm, various datasets are inter merged and the customers who have given ratings greater than 4 (i.e. 4 or 5) to products are analyzed and the following count of ratings w.r.t. The customers is given in figure 18 :-

```
customer_id
10de381f8a8d23fff822753305f71cae    15
adb32467ecc74b53576d9d13a5a55891    15
48d6a6909a9bd0b2128ea79630e52d13    12
daf15f1b940cc6a72ba558f093dc00dd    12
0d93f21f3e8543a9d0d8ece01561f5b2    12
Name: customer_id, dtype: int64
```

Figure 18:- Count of high ratings (4 or 5) given by customers

Any customer is picked at random and the product category of their orders are collected and stored in a variable, in our case, we got only 1 product category, but it can be large in number if we take the whole dataset. This is shown in fig- 19.

```
['ferramentas_jardim']
```

Figure 19:- Product category of chosen user

Now all those products in the above category (ies) are filtered and as shown in following figure 20:-

```
array(['18209df52bc87a69b84db4df602397c1',
       '1ab21e6620cb6346784529bff0a0c43b',
       'ef69da85d3bc28e624e5bc8fd34456c7',
       'f4daca35389849edde800f3532be3719',
       '10ee3576fcc55c71728a897eed8d2f95',
       '0df2a1aed6dc8c10577115fa9e4b6de0',
       'fc2d2cbc50b7d69ce961df503cb25e6b',
       '842fb98abb484ddb50b2682d0be9c3f2',
       'a037c98ed926711d7513cf779d788536',
       '463013aa75c3e698fc887b04b225f435',
       'b6046194183ce54d8fcaa7a2d28b1095',
       '9cd4e242b488f919520445a0623cc055',
       'da3d93a1cb86a9e0369c04900f413a5d',
       'ee47f3323aacfeba9e122400e5200fe5',
       '2f774e1d1ee6937ae8294771c662066d',
       'fad4e554c85c77f187ef2c6744495d31',
       '66ffe28d0fd53808d0535eee4b90a157',
       '714e305d7c565927c0ede7012c1c5c7c',
       '98ae5b3166d58f0055db5dc330ff1914',
```

Figure 20:- Products in category “ferramentas_jardim”

The list is very big, but it contains many products that are very far from the user. So, let's filter them according to user location. For comparison in later stages, let's keep the user's location same i.e. (-22.898536428530225,-47.063125168330544)

So, the finally recommended products are as shown in following figure 21:-

	product_id	geolocation_lat	geolocation_lng
0	18209df52bc87a69b84db4df602397c1	-22.901241	-47.055128
1	1ab21e6620cb6346784529bff0a0c43b	-22.929912	-47.071896
2	ef69da85d3bc28e624e5bc8fd34456c7	-22.738242	-46.895859
3	f4daca35389849edde800f3532be3719	-22.738242	-46.895859
4	10ee3576fcc55c71728a897eed8d2f95	-22.607003	-46.914150
5	0df2a1aed6dc8c10577115fa9e4b6de0	-22.759673	-47.364737
6	fc2d2cbc50b7d69ce961df503cb25e6b	-22.559120	-47.165168
7	842fb98abb484ddb50b2682d0be9c3f2	-23.112774	-46.548885
8	a037c98ed926711d7513cf779d788536	-23.112774	-46.548885
9	463013aa75c3e698fc887b04b225f435	-23.112774	-46.548885

Figure 21:- Recommended products by LOCONT

Let's see how far these are from the user's location i.e. (-22.898536428530225,-47.063125168330544)

The distances are calculated and as shown in figure 22. We can see some products have same latitudes and longitudes. The products are recommended irrespective of the seller address. The seller could have more than 1 product of any customer interest. This will usually attracts the customer as it could save the delivery charges or freight charges and the customer will be befitted. If these products could be from different sellers and the customer in case is interested in many of the recommended items then instead of saving money, he could be wasting money in delivery of those items. In other words, he must be sorting the items according to cost of the items also from the tray of ours recommended items which contains already distance filtered products.

	product_id	geolocation_lat	geolocation_lng	distance
0	18209df52bc87a69b84db4df602397c1	-22.901241	-47.055128	0.872590
1	1ab21e6620cb6346784529bff0a0c43b	-22.929912	-47.071896	3.602621
2	ef69da85d3bc28e624e5bc8fd34456c7	-22.738242	-46.895859	24.730522
3	f4daca35389849edde800f3532be3719	-22.738242	-46.895859	24.730522
4	10ee3576fcc55c71728a897eed8d2f95	-22.607003	-46.914150	35.836160
5	0df2a1aed6dc8c10577115fa9e4b6de0	-22.759673	-47.364737	34.552651
6	fc2d2cbc50b7d69ce961df503cb25e6b	-22.559120	-47.165168	39.165540
7	842fb98abb484ddb50b2682d0be9c3f2	-23.112774	-46.548885	57.773217
8	a037c98ed926711d7513cf779d788536	-23.112774	-46.548885	57.773217
9	463013aa75c3e698fc887b04b225f435	-23.112774	-46.548885	57.773217

Figure 22:- Distance of recommended products from user by LOCONT

4.4 DISCUSSION

The results are fairly good as the farthest seller in recommended products by LOCONT is 57.7 Km while the nearest one is as close as 872 m. However we can get more accurate results if we take whole dataset into consideration. If we get a large number of product categories, we could have used the NLP for computing similar products and hence recommended results would be better. This is similar to the scenario we see while buying some product like laptop online , we see similar items quoted below it like charger, laptop cover, case, bag, mouse, etc or even similar laptops in that budget range or similar featured laptops. Likewise we can improve the customer experience if we could have got diverse and scattered data.

CHAPTER-5

COMPARISON OF LOCOL AND LOCONT

5.1 GEOGRAPHIC COMPARISON

The results of LOCOL and LOCONT are plotted geographically and the results are pretty much close enough and are in close agreement of the requirements. This is shown in fig- 23 and that result is prepared in ArcMap 10.8. Some points are overlapping because of the same coordinates which can be the result of same store address for those products. The items are suggested independent of the seller address. The seller may have more than 1 item of any customer interest. This will ordinarily draws in the client because it may spare the conveyance charges or cargo charges and the client will be benefitted. If these items may well be from diverse venders and the customer in case is curious about numerous of the suggested things at that point rather than sparing cash, he might be squandering cash in conveyance of those things. In other words, he must be sorting the things concurring to fetch of the things too from the plate of our own suggested things which contains as of now separate sifted items.

5.2 ANALYTICAL COMPARISON

The results from Table 3 show more scatteredness in the results of LOCONT than LOCOL. This property sometimes greatly benefits the customer if the recommended results greatly suites the customer's taste and requirement. If the customer has to pick different products from different sellers, he has to pay separate delivery fees to all sellers. In that way, LOCOL will be preferred. But this concentrated results sometimes is not preferred from industry point of view, as it will give less Product diversity in results.`

The results can be analytically summarized into the below table 3:-

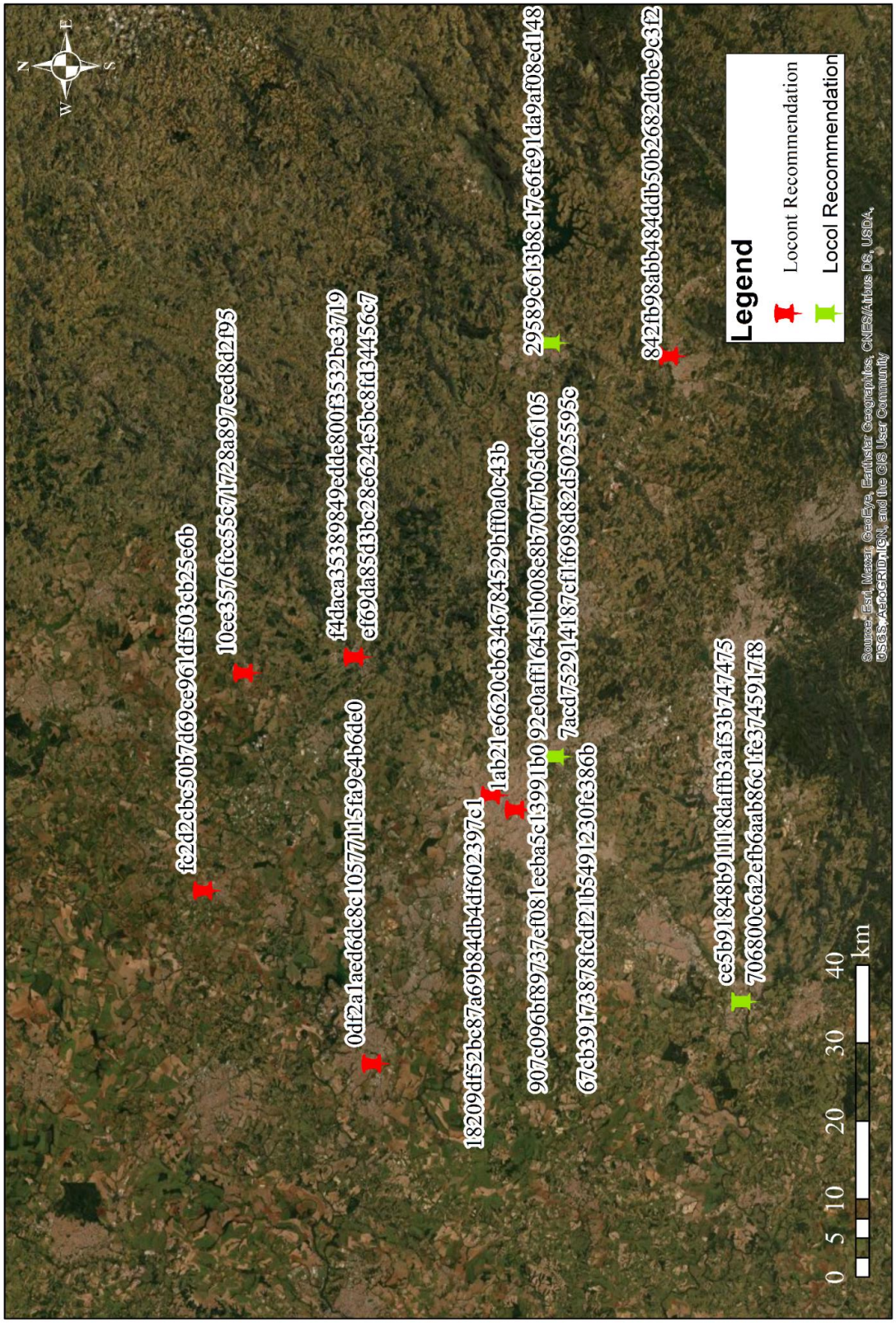

Figure 23: Geographic representation of results of LOCOL and LOCONT

Table 3:- Analytical Analysis of Results of LOCOL and LOCONT

Distance	**Number of products recommended by LOCOL**	**Number of products recommended by LOCONT**
Within 1Km	0	1
1-5 Km	0	1
5- 20 Km	0	0
20- 50 Km	10	5
More than 50 Km	0	3

5.3 CUMULATIVE ANALYSIS

We can do cumulative analysis of table 3 to see the results in more descriptive way. Cumulative studies (CS) consolidates data over a scale may be time or distance. Here cumulative over distance best describes the scenario. This will also help in decision making also as we can measure the performance of ours algorithm i.e. LOCOL and LOCONT w.r.t distance. The cumulative analysis of results of LOCOL and LOCONT are hence summarized in Table 3.

Table 4: Cumulative Analysis of Results of LOCOL and LOCONT

Distance	**Cumulative Number of products recommended by LOCOL**	**Cumulative Number of products recommended by LOCONT**
Within 1Km	0	1
Within 5 Km	0	2
Within 20 Km	0	2
Within 50 Km	10	7
More than 50 Km	10	10

The results of this cumulative study are also plotted in Figure -24. We see that both the algorithms give most of the recommended products within 20-50 Km range. If we make a threshold for 50 km, then LOCOL have performed better as most of the recommendations are within 50 km.

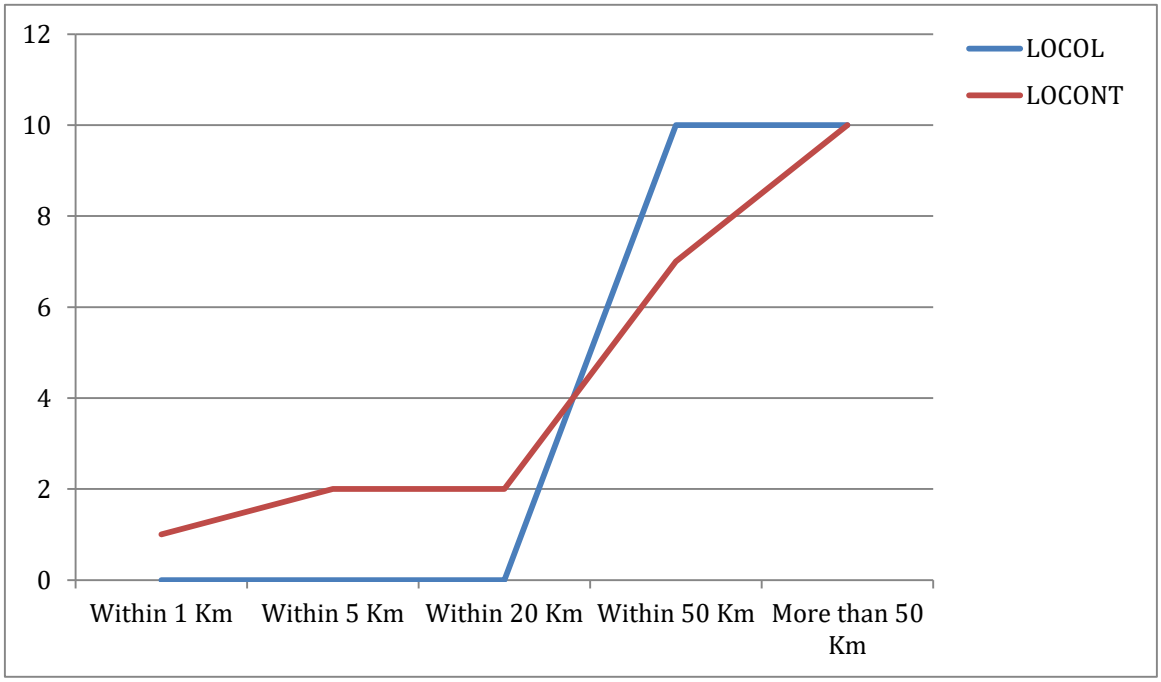

Figure 24: Cumulative results of LOCOL and LOCONT

CHAPTER-6

CONCLUSION AND FUTURE PROSPECTS

6.1 CONCLUSION

LOCOL and LOCONT are algorithms designed and implemented on OLIST dataset. The main spirit of these algorithms is to recommend products to users that best matches its interest but these should be near to him. The recommended products' sellers distance are computed using haversine distance (as it takes care of earth's curved surface).

We can draw following conclusion from them:-

1) LOCOL can recommend multiple products in the same store.
2) LOCONT recommend products in more vicinity but the results are scattered as we have sellers ranging from 0.3 Km to 57.7 Km
3) LOCOL takes about 1.5 hrs to run for a dataset of 120 MB in the system of RAM 4 GB and LOCONT takes about 2 hrs in a similar scenario.

6.2 FUTURE PROSPECTS

The run time of these proposed algorithms can be shortened by adopting more advanced merging techniques in the dataset. We can use Relational DBMS for this also. Code segment for the same has been enclosed in Annexure A. for LOCONT, we get limited data after applying filters, if the data is huge as we get in commercial scenarios then we can apply NLP for checking the similar products to user's interest. Moreover the recommended results can be stored in the same database and can be linked to UI and displayed to the user.

We can also go for hybrid filtering, deep neural networks, Matrix Factorization by introducing a layer of Distance in them and the results can be compared.

ANNEXURE-A

SOURCE CODE

A.1 DATA ANALYSIS

```
#!/usr/bin/env python
# coding: utf-8
# # Importing Libraries and reading CSV Files
# In[1]:
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
get_ipython().run_line_magic('matplotlib', 'inline')
import seaborn as sns
# In[2]:
sellers=pd.read_csv('D:/major_project/data/brazillion/olist_sellers_dataset.csv')
customers=pd.read_csv('D:/major_project/data/brazillion/olist_customers_dataset.csv')
products=pd.read_csv('D:/major_project/data/brazillion/olist_products_dataset.csv')
payments=pd.read_csv('D:/major_project/data/brazillion/olist_order_payments_dataset.csv')
orders=pd.read_csv('D:/major_project/data/brazillion/olist_orders_dataset.csv')
order_items=pd.read_csv('D:/major_project/data/brazillion/olist_order_items_dataset.csv')
reviews=pd.read_csv('D:/major_project/data/brazillion/olist_order_reviews_dataset.csv')
```

```
english_names=pd.read_csv('D:/major_project/data/brazillion/product_category_name_translation.csv')
coordinates=pd.read_csv('D:/major_project/data/brazillion/olist_geolocation_dataset.csv')
# In[3]:
sellers.head()
# In[4]:
customers.head()
# In[5]:
products.head()
# In[6]:
payments.head()
# In[7]:
orders.head()
# In[8]:
order_items.head()
# In[9]:
reviews.head()
# In[10]:
english_names.head()
# In[11]:
coordinates.head()
# # Counting number of orders per state
# In[12]:
df1=pd.merge(orders, customers, on="customer_id", how="left")
df1=df1[["order_id", "customer_id", "customer_state"]]
count_states=df1['customer_state'].value_counts()
# In[13]:
```

```
count_states
# In[14]:
count_states['others']=count_states[count_states<1000].sum()
x=np.array(count_states[count_states>1000].index)
y=np.array(count_states[count_states>1000].values)
# In[15]:
fig1, ax1 = plt.subplots(figsize=(15,7))
ax1.pie(y, labels=x, autopct='%1.1f%%', radius=2000, startangle=0)
ax1.axis('equal')
plt.title("No of Orders per State")
plt.legend(x)
plt.show()
# In[16]:
#count_states=pd.DataFrame(count_states,columns=["state","No. of orders"])
# In[17]:
count_states
# # Number of Sales per month
# In[19]:
df2=pd.merge(orders, order_items, on="order_id", how="right")
df2=df2[["order_id", "order_purchase_timestamp", "price"]]
df2['datetime'] =  pd.to_datetime(df2['order_purchase_timestamp'])
value_month = df2[['datetime', 'price']].copy()
value_month.set_index('datetime', inplace=True)
value_month = value_month.groupby(pd.Grouper(freq="M"))['price'].sum()
fig, ax = plt.subplots(figsize=(15,7))
ax.plot(value_month.index, value_month.values)
fig.autofmt_xdate()
```

```
ax.fmt_xdata = mdates.DateFormatter('%Y-%m')
ax.set_title('Sales per Month')
plt.show()
# # Payments Modes
#
# In[20]:
count_paymentMode=payments["payment_type"].value_counts()
count_paymentMode=count_paymentMode[count_paymentMode>100]
fig1, ax1 = plt.subplots(figsize=(15,7))
ax1.pie(count_paymentMode.values, autopct='%1.1f%%', radius=2000, startangle=90)
ax1.axis('equal')
plt.title("Mode of Transaction")
plt.legend(count_paymentMode.index)
plt.show()
# In[21]:
df3=pd.merge(payments, df1, on="order_id", how="left")
a=df3["customer_state"].value_counts()
a["others"]=a[a<2000].sum()
a=a[a>2000]
df4=df3[df3["customer_state"].isin(a.index) & df3["payment_type"].isin(["credit_card",
"boleto", "voucher", "debit_card"])]
plt.figure(figsize=(15,7))
sns.countplot(x="customer_state", hue="payment_type", data=df4 )
plt.title("Payments mode per state")
# # Product Sales per State
# In[22]:
btoe=pd.Series(english_names.product_category_name_english.values,
index=english_names.product_category_name).to_dict()
```

```
products["product_category_name"]=products["product_category_name"].map(btoe)
df5=pd.merge(order_items, products, on="product_id", how="left")
df5=df5[["order_id", "product_id", "product_category_name"]]
df6=pd.merge(orders, customers, on="customer_id", how="left")
df6=df6[["order_id", "customer_id", "customer_city", "customer_state" ]]
df7=pd.merge(df5, df6, on="order_id", how="left")
df7=df7[df7["customer_state"].isin(["SP", "RJ", "MG"])]
b=df7["product_category_name"].value_counts()
b["others"]=b[b<2000].sum()
b=b[b>2000]
df7=df7[df7["product_category_name"].isin(b.index)]
plt.figure(figsize=(15,7))
chart=sns.countplot(x="product_category_name", hue="customer_state", data=df7 )
chart.set_xticklabels( chart.get_xticklabels(), rotation=30 )
plt.title("Product Sales per State")
plt.show()
# In[9]:
df_customer_order_coordinates.to_csv("customer's lat long.csv")
# In[10]:
df_customer_order_coordinates.head()
# In[5]:
df5=pd.merge(sellers,coordinates,
right_on="geolocation_zip_code_prefix",left_on="seller_zip_code_prefix", how="inner")
df6=df5[["seller_id", "geolocation_lat", "geolocation_lng"]]
# In[6]:
df6.to_csv("seller's lat long.csv")
# In[11]:
```

```
df6.head()
# In[15]:
print(df6.shape)
len(df6["seller_id"].unique())
# In[6]:
get_ipython().system('pip install folium')
# In[7]:
get_ipython().system('pip install geopy')
# In[8]:
get_ipython().system('pip install minisom')
# In[9]:
## for geospatial
import folium
import geopy
## for machine learning
from sklearn import preprocessing, cluster
import scipy
## for deep learning
import minisom
# In[13]:
df4.shape
# In[14]:
df4.head()
# In[17]:
len(df4["customer_id"].unique())
# In[18]:
df5=df4.drop_duplicates()
```

```
df5.shape
customers.columns()
```

A.2 LOCOL

```
#!/usr/bin/env python
# coding: utf-8
# In[40]:
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
get_ipython().run_line_magic('matplotlib', 'inline')
import seaborn as sns
import haversine as hs
# In[74]:
sellers=pd.read_csv('D:/major_project/data/brazillion/olist_sellers_dataset.csv')
customers=pd.read_csv('D:/major_project/data/brazillion/olist_customers_dataset.csv')
products=pd.read_csv('D:/major_project/data/brazillion/olist_products_dataset.csv')
payments=pd.read_csv('D:/major_project/data/brazillion/olist_order_payments_dataset.csv')
orders=pd.read_csv('D:/major_project/data/brazillion/olist_orders_dataset.csv')
order_items=pd.read_csv('D:/major_project/data/brazillion/olist_order_items_dataset.csv')
reviews=pd.read_csv('D:/major_project/data/brazillion/olist_order_reviews_dataset.csv')
english_names=pd.read_csv('D:/major_project/data/brazillion/product_category_name_trans
lation.csv')
coordinates=pd.read_csv('D:/major_project/data/brazillion/olist_geolocation_dataset.csv')
# In[42]:
```

```
sellers.columns,customers.columns,products.columns,payments.columns,orders.columns,ord
er_items.columns,reviews.columns,english_names.columns,coordinates.columns
# In[43]:


df=pd.merge(sellers,coordinates,
right_on="geolocation_zip_code_prefix",left_on="seller_zip_code_prefix", how="inner")
df['coordinates']=df['geolocation_lat'].astype(str) + ',' + df['geolocation_lng'].astype(str)
df1=df[["seller_id","geolocation_lat","geolocation_lng","coordinates"]].iloc[0:200000]
df1.head()
# In[44]:
df1["distance"]=0
user=(-22.898536428530225,-47.063125168330544)
for i in df1.index:

df1["distance"][i]=hs.haversine(user,(df1["geolocation_lat"][i],df1["geolocation_lng"][i]))
# In[45]:
df_sorted=df1.sort_values(by="distance", ascending=True)
df_sliced_sellers=df_sorted.iloc[0:2000]
# In[46]:
df2 = reviews.merge(orders, on= 'order_id')
del reviews
del orders
df3 = df2.merge(order_items,on = 'order_id')
del order_items
data_coll = df3[['customer_id','seller_id','product_id','review_score']]
new_data = pd.merge(df_sliced_sellers, data_coll, how='inner',on='seller_id')
new_data=new_data[['customer_id','product_id','review_score']]
del data_coll
```

```
# In[47]:
new_data.describe()['review_score'].T
# In[48]:
# Check the distribution of the rating
with sns.axes_style('white'):
    g = sns.factorplot("review_score", data=new_data, aspect=2.0,kind='count')
    g.set_ylabels("Total number of ratings")
# In[49]:
print("Total data ")
print("-"*50)
print("\nTotal no of ratings :",new_data.shape[0])
print("Total No of Users   :", len(np.unique(new_data.customer_id)))
print("Total No of products  :", len(np.unique(new_data.product_id)))
# In[50]:
#Analysis of rating given by the user
no_of_ratings_given_by_users =
new_data.groupby(by='customer_id')['review_score'].count().sort_values(ascending=False)
no_of_ratings_given_by_users
# In[51]:
no_of_ratings_given_by_users.describe()
# In[52]:
#Analysis of ratings for every product
no_of_rated_products_per_user =
new_data.groupby(by='product_id')['review_score'].count().sort_values(ascending=False)
no_of_rated_products_per_user
# In[53]:
product_avg_rate =
new_data.groupby(by='product_id')['review_score'].mean().sort_values(ascending=False)
```

```
# In[54]:
product_avg_rate.describe()
# # item-item collaborative- location based
#
# In[55]:
from surprise import KNNWithMeans, KNNBasic, KNNWithZScore, KNNBaseline
from surprise import Dataset
from surprise import accuracy
from surprise import Reader
from surprise.model_selection import GridSearchCV
from surprise import SVD
import os
from surprise.model_selection import train_test_split
from surprise.model_selection import cross_validate
# In[56]:
reader = Reader(rating_scale=(1, 5))
data = Dataset.load_from_df(new_data,reader)
# single value decomposition
algo = SVD()
# Run 5-fold cross-validation and then print results
cross_validate(algo,data, measures=['RMSE', 'MAE'], cv=5, verbose=True)
# In[57]:
# #Splitting the dataset
trainset, testset = train_test_split(data, test_size=0.3,random_state=10)
# In[58]:
# # Use user_based true/false to switch between user-based or item-based collaborative
filtering
```

```
models = []
models.append(('KNNWithMeans',KNNWithMeans(k=5, sim_options={'name': 'cosine',
'user_based': False})))
models.append(('KNNBasic', KNNBasic(k=5, sim_options={'name': 'cosine', 'user_based':
False})))
models.append(('KNNWithZScore', KNNWithZScore(k=5, sim_options={'name': 'cosine',
'user_based': False})))
models.append(('KNNBaseline', KNNBaseline(k=5, sim_options={'name': 'cosine',
'user_based': False})))
# In[59]:
for name, model in models:
    model.fit(trainset)
    test_pred = model.test(testset)
    print("RMSE of ",name)
    accuracy.rmse(test_pred, verbose=True)
# # model based location  based collaborative filtering
# In[60]:
#new_df1=new_df.sample(10000)
ratings_matrix = pd.pivot_table(new_data,values='review_score', index='customer_id',
columns='product_id', fill_value=0)
ratings_matrix.head()
# In[61]:
ratings_matrix.shape
# In[62]:
X = ratings_matrix.T
X.head()
# In[63]:
X.shape
```

```
# In[64]:
#Decomposing the Matrix
from sklearn.decomposition import TruncatedSVD
SVD = TruncatedSVD(n_components=10)
decomposed_matrix = SVD.fit_transform(X)
decomposed_matrix.shape
# In[65]:
#Correlation Matrix
correlation_matrix = np.corrcoef(decomposed_matrix)
correlation_matrix.shape
# In[66]:
X.index[20]
# In[68]:
i = "09738baf9be60e4c294fa5d0bfcc690d"
product_names = list(X.index)
product_ID = product_names.index(i)
product_ID
# In[69]:
correlation_product_ID = correlation_matrix[product_ID]
correlation_product_ID.shape
# In[70]:
Recommend = list(X.index[correlation_product_ID > 0.65])
# Removes the item already bought by the customer
Recommend.remove(i)
Recommend[0:5]
# In[71]:
len(Recommend)
```

```
# In[72]:
reco=pd.DataFrame(Recommend[0:10],columns=["product_id"])
reco
# In[75]:
comb_product = pd.merge(products, order_items, on ='product_id')
comb_product_seller = pd.merge(comb_product,sellers, on = 'seller_id')
df=pd.merge(comb_product_seller,coordinates,
right_on="geolocation_zip_code_prefix",left_on="seller_zip_code_prefix", how="inner")
#comb_products_orders=pd.merge(comb_product_review,orders, on = 'order_id')
df=df[['product_id','geolocation_lat','geolocation_lng']]
df.head()
# In[76]:
df=df.drop_duplicates(subset=['product_id'])
# In[77]:
df_recom=pd.merge(df,reco,on='product_id',how='right')
# In[78]:
df_recom["distance"]=0
user=(-22.898536428530225,-47.063125168330544)
for i in df_recom.index:
df_recom["distance"][i]=hs.haversine(user,(df_recom["geolocation_lat"][i],df_recom["geolo
cation_lng"][i]))
# In[79]:
df_recom
# In[80]:
df_recom.to_csv("locol_recommendation_new.csv")
```

A.3 LOCONT

```
#!/usr/bin/env python
# coding: utf-8
# In[1]:
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
get_ipython().run_line_magic('matplotlib', 'inline')
import seaborn as sns
import haversine as hs
# In[2]:
sellers=(pd.read_csv('D:/major_project/data/brazillion/olist_sellers_dataset.csv')).drop_duplicates(keep='first')
customers=(pd.read_csv('D:/major_project/data/brazillion/olist_customers_dataset.csv')).drop_duplicates(keep='first')
products=(pd.read_csv('D:/major_project/data/brazillion/olist_products_dataset.csv')).drop_duplicates(keep='first')
payments=(pd.read_csv('D:/major_project/data/brazillion/olist_order_payments_dataset.csv')).drop_duplicates(keep='first')
orders=(pd.read_csv('D:/major_project/data/brazillion/olist_orders_dataset.csv')).drop_duplicates(keep='first')
order_items=(pd.read_csv('D:/major_project/data/brazillion/olist_order_items_dataset.csv')).drop_duplicates(keep='first')
reviews=(pd.read_csv('D:/major_project/data/brazillion/olist_order_reviews_dataset.csv')).drop_duplicates(keep='first')
```

```
english_names=(pd.read_csv('D:/major_project/data/brazillion/product_category_name_translation.csv')).drop_duplicates(keep='first')
coordinates=(pd.read_csv('D:/major_project/data/brazillion/olist_geolocation_dataset.csv')).drop_duplicates(keep='first')
# In[3]:
comb_product = pd.merge(products, order_items, on ='product_id')
comb_product_review = pd.merge(comb_product,reviews, on = 'order_id')
comb_products_orders=pd.merge(comb_product_review,orders, on = 'order_id')
high_review_score = comb_products_orders[comb_products_orders.review_score >= 4]
# In[4]:
high_review_score.shape
# In[5]:
type(high_review_score)
# In[6]:
high_review_score.columns
# In[7]:
high_review_score_1=high_review_score[['product_id',
'product_category_name','seller_id','customer_id']]
# In[8]:
high_review_score_1 =
high_review_score_1.groupby(by='customer_id')['customer_id'].count().sort_values(ascending=False)
high_review_score_1.head()
# In[9]:
high_review_score_2 =
high_review_score.groupby(by='product_id')['product_id'].count().sort_values(ascending=False)
high_review_score_2.head()
```

```
high_review_score=high_review_score.drop_duplicates()
# In[10]:
data=high_review_score[high_review_score['customer_id'] ==
"adb32467ecc74b53576d9d13a5a55891"]
unique_product_category=data['product_category_name'].unique()
# In[11]:
print(unique_product_category)
# In[12]:
df=pd.merge(sellers,coordinates,
right_on="geolocation_zip_code_prefix",left_on="seller_zip_code_prefix", how="inner")
df['coordinates']=df['geolocation_lat'].astype(str) + ',' + df['geolocation_lng'].astype(str)
df1=df[["seller_id","geolocation_lat","geolocation_lng","coordinates"]].iloc[0:200000]
df1.head()
# In[13]:
df1["distance"]=0
user=(-22.898536428530225,-47.063125168330544)
for i in df1.index:

df1["distance"][i]=hs.haversine(user,(df1["geolocation_lat"][i],df1["geolocation_lng"][i]))
# In[14]:
df_sorted=df1.sort_values(by="distance", ascending=True)
df_sliced_sellers=pd.merge(df_sorted,order_items,on="seller_id")
df_sliced_products=pd.merge(df_sliced_sellers,products,on="product_id")
# In[15]:
df_sliced_products.columns
# In[16]:
df_required=df_sliced_products[df_sliced_products["product_category_name"]==unique_pr
oduct_category[0]]
```

```
# In[17]:
recom=df_required["product_id"].unique()
recom[:5]
# In[18]:
recom=df_required["product_id"].unique()
recom
# In[20]:
comb_product = pd.merge(products, order_items, on ='product_id')
comb_product_seller = pd.merge(comb_product,sellers, on = 'seller_id')
df=pd.merge(comb_product_seller,coordinates,
right_on="geolocation_zip_code_prefix",left_on="seller_zip_code_prefix", how="inner")
#comb_products_orders=pd.merge(comb_product_review,orders, on = 'order_id')
df=df[['product_id','geolocation_lat','geolocation_lng']]
df.head()
# In[21]:
df=df.drop_duplicates(subset=['product_id'])
df
# In[31]:
recommend=pd.DataFrame(recom,columns=["product_id"])
df_reco=pd.merge(df,recommend,on ='product_id', how="right")
df_reco=df_reco.iloc[:10]
# In[32]:
df_reco
# In[33]:
df_reco.to_csv("locont_recommendation_new.csv")
df_reco["distance"]=0
user=(-22.898536428530225,-47.063125168330544)
```

```
for i in df_reco.index:
        df_reco["distance"][i]=hs.haversine(user,(df_reco["geolocation_lat"][i],df_reco["geo
location_lng"][i]))
# In[35]:
df_reco
```

A.4 FRAMEWORK FOR DATA MANAGEMENT

A.4.1 main.py

```
import os
from turtle import clear
from configDecoder import *
from dataDumper import *
conf = configP()
dataCleaning()
```

A.4.2 configDecoder

```
def configP():
   import configparser
   class Conf:
      def __init__(self, config):
         if isinstance(config, configparser.ConfigParser):
            self.host = config['DatabaseSectionLocal']['DatabaseSectionLocal.host']
            self.dbname = config['DatabaseSectionLocal']['DatabaseSectionLocal.dbname']
            self.user = config['DatabaseSectionLocal']['DatabaseSectionLocal.user']
            self.password = config['DatabaseSectionLocal']['DatabaseSectionLocal.password']
```

```
        self.input_csv = config['pathsLocal']['pathsLocal.input_csv']
        self.resources = config['pathsLocal']['pathsLocal.resources']
        self.processed = config['pathsLocal']['pathsLocal.processed']
        self.customers=config['files']['files.customers']
        self.geolocation=config['files']['files.geolocation']
        self.orderItems=config['files']['files.orderItems']
        self.orderPayments=config['files']['files.orderPayments']
        self.orderReviews=config['files']['files.orderReviews']
        self.orders=config['files']['files.orders']
        self.products=config['files']['files.products']
        self.sellers=config['files']['files.sellers']
        self.trancription=config['files']['files.trancription']


def configExtract(confPath):
    config = configparser.ConfigParser()
    config.read(confPath)
    return config
def initConfig(confPath: str):
    """
    Initializer Function
    :param confPath: configuration file path
    :return: GenericConf Object, HiveConf Object, MysqlConf Object
    """
    config = configExtract(confPath)
    return Conf(config)


confPath='D:/major_project/algo-collaborative/resources/ConfigFile.properties'
```

```
    conf = initConfig(confPath)
    return(conf)
```

A.4.3 config.properties

```
[DatabaseSectionLocal]
DatabaseSectionLocal.host=127.0.0.1
DatabaseSectionLocal.dbname=ab_ginf_olist
DatabaseSectionLocal.user=root
DatabaseSectionLocal.password=root


[pathsLocal]
pathsLocal.input_csv=D:/major_project/algo-collaborative/input/
pathsLocal.resources=D:/major_project/algo-collaborative/resources/
pathsLocal.processed=D:/major_project/algo-collaborative/processed/

[files]
files.customers=olist_customers_dataset
files.geolocation=olist_geolocation_dataset
files.orderItems=olist_order_items_dataset
files.orderPayments=olist_order_payments_dataset
files.orderReviews=olist_order_reviews_dataset
files.orders=olist_orders_dataset
files.products=olist_products_dataset
```

files.sellers=olist_sellers_dataset

files.trancription=product_category_name_translation

A.4.4 dataDumper

```
from configDecoder import *
from MysqlUtility import *
def dataCleaning():
    import logging
    import os
    import shutil
    import pymysql as sql
    import sys
    import pandas as pd
    conf=configP()
    host=conf.host
    dbname=conf.dbname
    user=conf.user
    password=conf.password

    input=conf.input_csv
    resources=conf.resources
    processed=conf.processed
    #input file
    list = os.listdir(input)
    for i in list:
        if conf.customers in i:
```

```
        data_file = os.path.basename(input + i)
        df = pd.DataFrame(pd.read_csv(input+i))
        os.chdir(resources)
        print(df.columns)
        print(df.isnull().sum())
        df_final_customers = df[['customer_id', 'customer_unique_id',
'customer_zip_code_prefix','customer_city', 'customer_state']]
        df_final_customers.columns = [['ac_customer_id', 'ac_customer_unique_id',
'ac_customer_zip_code_prefix','ac_customer_city', 'ac_customer_state']]
        mysqlUtilTB_insert = MysqlUtil(host=host, db=dbname, user=user,
password=password)

mysqlUtilTB_insert.insert(df_final_customers.where(pd.notnull(df_final_customers),
None),'ago_customers')
    elif conf.geolocation in i:
        data_file = os.path.basename(input + i)
        df = pd.DataFrame(pd.read_csv(input+i))
        os.chdir(resources)
        print(df.columns)
        df_final_geolocation = df[['geolocation_zip_code_prefix', 'geolocation_lat',
'geolocation_lng','geolocation_city', 'geolocation_state']]
        df_final_geolocation.columns = [['ag_geolocation_zip_code_prefix',
'ag_geolocation_lat', 'ag_geolocation_lng','ag_geolocation_city', 'ag_geolocation_state']]
        mysqlUtilTB_insert = MysqlUtil(host=host, db=dbname, user=user,
password=password)

mysqlUtilTB_insert.insert(df_final_geolocation.where(pd.notnull(df_final_geolocation),
None),'ago_geolocation')
```

```
    elif conf.orderItems in i:
        data_file = os.path.basename(input + i)
        df = pd.DataFrame(pd.read_csv(input+i))
        os.chdir(resources)
        print(df.columns)
        df_final_orderItems = df[['order_id', 'order_item_id', 'product_id',
'seller_id','shipping_limit_date', 'price', 'freight_value']]
        df_final_orderItems.columns = [['aoi_order_id', 'aoi_order_item_id',
'aoi_product_id', 'aoi_seller_id','aoi_shipping_limit_date', 'aoi_price', 'aoi_freight_value']]
        mysqlUtilTB_insert = MysqlUtil(host=host, db=dbname, user=user,
password=password)

mysqlUtilTB_insert.insert(df_final_orderItems.where(pd.notnull(df_final_orderItems),
None),'ago_orders_items')

    elif conf.orderPayments in i:
        data_file = os.path.basename(input + i)
        df = pd.DataFrame(pd.read_csv(input+i))
        os.chdir(resources)
        #print(df.columns)
        df_final_orderPayments = df[['order_id', 'payment_sequential',
'payment_type','payment_installments', 'payment_value']]
        df_final_orderPayments.columns = [['aop_order_id', 'aop_payment_sequential',
'aop_payment_type','aop_payment_installments', 'aop_payment_value']]
        mysqlUtilTB_insert = MysqlUtil(host=host, db=dbname, user=user,
password=password)

mysqlUtilTB_insert.insert(df_final_orderPayments.where(pd.notnull(df_final_orderPaymen
ts), None),'ago_orders_payments')
```

```
    elif conf.orderReviews in i:
        data_file = os.path.basename(input + i)
        df = pd.DataFrame(pd.read_csv(input+i))
        os.chdir(resources)
        #print(df.columns)
        df_final_orderReviews = df[['review_id', 'order_id', 'review_score',
'review_comment_title','review_comment_message',
'review_creation_date','review_answer_timestamp']]
        df_final_orderReviews.columns = [['aor_review_id', 'aor_order_id',
'aor_review_score', 'aor_review_comment_title','aor_review_comment_message',
'aor_review_creation_date','aor_review_answer_timestamp']]
        mysqlUtilTB_insert = MysqlUtil(host=host, db=dbname, user=user,
password=password)

mysqlUtilTB_insert.insert(df_final_orderReviews.where(pd.notnull(df_final_orderReviews)
, None),'ago_orders_reviews')


    elif conf.orders in i:
        data_file = os.path.basename(input + i)
        df = pd.DataFrame(pd.read_csv(input+i))
        os.chdir(resources)
        #print(df.columns)
        df_final_orders = df[['order_id', 'customer_id', 'order_status',
'order_purchase_timestamp','order_approved_at',
'order_delivered_carrier_date','order_delivered_customer_date',
'order_estimated_delivery_date']]
        df_final_orders.columns = [['ao_order_id', 'ao_customer_id', 'ao_order_status',
'ao_order_purchase_timestamp','ao_order_approved_at',
```

```
'ao_order_delivered_carrier_date','ao_order_delivered_customer_date',
'ao_order_estimated_delivery_date']]
        mysqlUtilTB_insert = MysqlUtil(host=host, db=dbname, user=user,
password=password)
        mysqlUtilTB_insert.insert(df_final_orders.where(pd.notnull(df_final_orders),
None),'ago_orders')

    elif conf.products in i:
        data_file = os.path.basename(input + i)
        df = pd.DataFrame(pd.read_csv(input+i))
        os.chdir(resources)
        #print(df.columns)
        df_final_products = df[['product_id', 'product_category_name',
'product_name_lenght','product_description_lenght', 'product_photos_qty',
'product_weight_g','product_length_cm', 'product_height_cm', 'product_width_cm']]
        df_final_products.columns = [['ap_product_id', 'ap_product_category_name',
'ap_product_name_lenght','ap_product_description_lenght', 'ap_product_photos_qty',
'ap_product_weight_g','ap_product_length_cm', 'ap_product_height_cm',
'ap_product_width_cm']]
        mysqlUtilTB_insert = MysqlUtil(host=host, db=dbname, user=user,
password=password)
        mysqlUtilTB_insert.insert(df_final_products.where(pd.notnull(df_final_products),
None),'ago_products')

    elif conf.sellers in i:
        data_file = os.path.basename(input + i)
        df = pd.DataFrame(pd.read_csv(input+i))
        os.chdir(resources)
        #print(df.columns)
```

```
        df_final_sellers = df[['seller_id', 'seller_zip_code_prefix', 'seller_city', 'seller_state']]
        df_final_sellers.columns = [['as_seller_id', 'as_seller_zip_code_prefix',
'as_seller_city', 'as_seller_state']]
        mysqlUtilTB_insert = MysqlUtil(host=host, db=dbname, user=user,
password=password)
        mysqlUtilTB_insert.insert(df_final_sellers.where(pd.notnull(df_final_sellers),
None),'ago_sellers')

    elif conf.trancription in i:
        data_file = os.path.basename(input + i)
        df = pd.DataFrame(pd.read_csv(input+i))
        os.chdir(resources)
        #print(df.columns)
        df_final_trancription = df[['product_category_name',
'product_category_name_english']]
        df_final_trancription.columns = [['at_product_category_name',
'at_product_category_name_english']]
        mysqlUtilTB_insert = MysqlUtil(host=host, db=dbname, user=user,
password=password)

mysqlUtilTB_insert.insert(df_final_trancription.where(pd.notnull(df_final_trancription),
None),'ago_customers')
```

ANNEXURE-B

BRAZIL STATE MAP

Figure 24:- political map of Brazil depicting various states

(source:- https://mapsofworld.com)

ANNEXURE-C

DATA DESCRIPTION

The various dataset are loaded in jupyter notebook and examined and the results are displayed in fig. 24 to 14 as follows:-

	seller_id	seller_zip_code_prefix	seller_city	seller_state
0	3442f8959a84dea7ee197c632cb2df15	13023	campinas	SP
1	d1b65fc7debc3361ea86b5f14c68d2e2	13844	mogi guacu	SP
2	ce3ad9de960102d0677a81f5d0bb7b2d	20031	rio de janeiro	RJ
3	c0f3eea2e14555b6faeea3dd58c1b1c3	4195	sao paulo	SP
4	51a04a8a6bdcb23deccc82b0b80742cf	12914	braganca paulista	SP

Figure 25:-Sample seller dataset

	customer_id	customer_unique_id	customer_zip_code_prefix	customer_city	customer_state
0	06b8999e2fba1a1fbc88172c00ba8bc7	861eff4711a542e4b93843c6dd7febb0	14409	franca	SP
1	18955e83d337fd6b2def6b18a428ac77	290c77bc529b7ac935b93aa66c333dc3	9790	sao bernardo do campo	SP
2	4e7b3e00288586ebd08712fdd0374a03	060e732b5b29e8181a18229c7b0b2b5e	1151	sao paulo	SP
3	b2b6027bc5c5109e529d4dc6358b12c3	259dac757896d24d7702b9acbbff3f3c	8775	mogi das cruzes	SP
4	4f2d8ab171c80ec8364f7c12e35b23ad	345ecd01c38d18a9036ed96c73b8d066	13056	campinas	SP

Figure 26:- Sample Customer dataset

	product_id	product_category_name	product_name_lenght	product_description_lenght	product_photos_qty	product_weight_g	produc
0	1e9e8ef04dbcff4541ed26657ea517e5	perfumaria	40.0	287.0	1.0	225.0	
1	3aa071139cb16b67ca9e5dea641aaa2f	artes	44.0	276.0	1.0	1000.0	
2	96bd76ec8810374ed1b65e291975717f	esporte_lazer	46.0	250.0	1.0	154.0	
3	cef67bcfe19066a932b7673e239eb23d	bebes	27.0	261.0	1.0	371.0	
4	9dc1a7de274444849c219cff195d0b71	utilidades_domesticas	37.0	402.0	4.0	625.0	

Figure 27:- Sample Product dataset

	order_id	payment_sequential	payment_type	payment_installments	payment_value
0	b81ef226f3fe1789b1e8b2acac839d17	1	credit_card	8	99.33
1	a9810da82917af2d9aefd1278f1dcfa0	1	credit_card	1	24.39
2	25e8ea4e93396b6fa0d3dd708e76c1bd	1	credit_card	1	65.71
3	ba78997921bbcdc1373bb41e913ab953	1	credit_card	8	107.78
4	42fdf880ba16b47b59251dd489d4441a	1	credit_card	2	128.45

Figure 28:- Sample Payment dataset

	order_id	customer_id	order_status	order_purchase_timestamp	order_approved_at	order_delivered_carrier_d
0	e481f51cbdc54678b7cc49136f2d6af7	9ef432eb6251297304e76186b10a928d	delivered	2017-10-02 10:56:33	2017-10-02 11:07:15	2017-10-04 19:55
1	53cdb2fc8bc7dce0b6741e2150273451	b0830fb4747a6c6d20dea0b8c802d7ef	delivered	2018-07-24 20:41:37	2018-07-26 03:24:27	2018-07-26 14:31
2	47770eb9100c2d0c44946d9cf07ec65d	41ce2a54c0b03bf3443c3d931a367089	delivered	2018-08-08 08:38:49	2018-08-08 08:55:23	2018-08-08 13:50
3	949d5b44dbf5de918fe9c16f97b45f8a	f88197465ea7920adcdbec7375364d82	delivered	2017-11-18 19:28:06	2017-11-18 19:45:59	2017-11-22 13:39
4	ad21c59c0840e6cb83a9ceb5573f8159	8ab97904e6daea8866dbdbc4fb7aad2c	delivered	2018-02-13 21:18:39	2018-02-13 22:20:29	2018-02-14 19:46

Figure 29:- Sample Orders dataset

	order_id	order_item_id	product_id	seller_id	shipping_limit_date	price	freigh
0	00010242fe8c5a6d1ba2dd792cb16214	1	4244733e06e7ecb4970a6e2683c13e61	48436dade18ac8b2bce089ec2a041202	2017-09-19 09:45:35	58.90	
1	00018f77f2f0320c557190d7a144bdd3	1	e5f2d52b802189ee658865ca93d83a8f	dd7ddc04e1b6c2c614352b383efe2d36	2017-05-03 11:05:13	239.90	
2	000229ec398224ef6ca0657da4fc703e	1	c777355d18b72b67abbeef9df44fd0fd	5b51032eddd242adc84c38acab88f23d	2018-01-18 14:48:30	199.00	
3	00024acbcdf0a6daa1e931b038114c75	1	7634da152a4610f1595efa32f14722fc	9d7a1d34a5052409006425275ba1c2b4	2018-08-15 10:10:18	12.99	
4	00042b26cf59d7ce69dfabb4e55b4fd9	1	ac6c3623068f30de03045865e4e10089	df560393f3a51e74553ab94004ba5c87	2017-02-13 13:57:51	199.90	

Figure 30:- Sample Orders items dataset

	review_id	order_id	review_score	review_comment_title	review_comment_message	review_creation_date
0	7bc2406110b926393aa56f80a40eba40	73fc7af87114b39712e6da79b0a377eb	4	NaN	NaN	2018-01-18 00:00:00
1	80e641a11e56f04c1ad469d5645fdfde	a548910a1c6147796b98fdf73dbeba33	5	NaN	NaN	2018-03-10 00:00:00
2	228ce5500dc1d8e020d8d1322874b6f0	f9e4b658b201a9f2ecdecbb34bed034b	5	NaN	NaN	2018-02-17 00:00:00
3	e64fb393e7b32834bb789ff8bb30750e	658677c97b385a9be170737859d3511b	5	NaN	Recebi bem antes do prazo estipulado.	2017-04-21 00:00:00
4	f7c4243c7fe1938f181bec41a392bdeb	8e6bfb81e283fa7e4f11123a3fb894f1	5	NaN	Parabéns lojas lannister adorei comprar pela I...	2018-03-01 00:00:00

Figure 31:- Sample Review dataset

	product_category_name	product_category_name_english
0	beleza_saude	health_beauty
1	informatica_acessorios	computers_accessories
2	automotivo	auto
3	cama_mesa_banho	bed_bath_table
4	moveis_decoracao	furniture_decor

Figure 32:- Sample Transcription dataset

	geolocation_zip_code_prefix	geolocation_lat	geolocation_lng	geolocation_city	geolocation_state
0	1037	-23.545621	-46.639292	sao paulo	SP
1	1046	-23.546081	-46.644820	sao paulo	SP
2	1046	-23.546129	-46.642951	sao paulo	SP
3	1041	-23.544392	-46.639499	sao paulo	SP
4	1035	-23.541578	-46.641607	sao paulo	SP

Figure 33:- Sample GeoCoordinates data of sellers and customers

References

[1]: Cui, Z., Xu, X., Fei, X. U. E., Cai, X., Cao, Y., Zhang, W., & Chen, J. (2020). Personalized recommendation system based on collaborative filtering for IoT scenarios. *IEEE Transactions on Services Computing*, *13*(4), 685-695.

[2]: Roy, A. C., Arefin, M. S., Kayes, A. S. M., Hammoudeh, M., & Ahmed, K. (2020). An empirical recommendation framework to support location-based services. *Future Internet*, *12*(9), 154.

[3] Uitz, I., & Koitz, R. (2013). Consumer acceptance of location based services in the retail environment. *Int. J. Adv. Comput. Sci. Appl*, *4*.

[4] Tan, H., Guo, J., & Li, Y. (2008, December). E-learning recommendation system. In *2008 International conference on computer science and software engineering* (Vol. 5, pp. 430-433). IEEE.

[5] Thorat, P. B., Goudar, R. M., & Barve, S. (2015). Survey on collaborative filtering, content-based filtering and hybrid recommendation system. *International Journal of Computer Applications*, *110*(4), 31-36.

[6] Isinkaye, F. O., Folajimi, Y. O., & Ojokoh, B. A. (2015). Recommendation systems: Principles, methods and evaluation. *Egyptian informatics journal*, *16*(3), 261-273.

[7] Wu, L., He, X., Wang, X., Zhang, K., & Wang, M. (2022). A Survey on Accuracy-oriented Neural Recommendation: From Collaborative Filtering to Information-rich Recommendation. *IEEE Transactions on Knowledge and Data Engineering*.

[8] Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, *17*(6), 734-749.

[9] Wang, F., Zhu, H., Srivastava, G., Li, S., Khosravi, M. R., & Qi, L. (2021). Robust collaborative filtering recommendation with user-item-trust records. *IEEE Transactions on Computational Social Systems*.

[10] Lops, P., Gemmis, M. D., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. *Recommender systems handbook*, 73-105.

[11] Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in artificial intelligence*, *2009*.

[12] Mustafa, N., Ibrahim, A. O., Ahmed, A., & Abdullah, A. (2017, January). Collaborative filtering: Techniques and applications. In 2017 International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE) (pp. 1-6). IEEE.

[13] Van Meteren, R., & Van Someren, M. (2000, May). Using content-based filtering for recommendation. In *Proceedings of machine learning in the new information age: MLnet/ECML2000 workshop* (Vol. 30, pp. 47-56).

[14] Salter, J., & Antonopoulos, N. (2006). CinemaScreen recommender agent: combining collaborative and content-based filtering. *IEEE Intelligent Systems*, *21*(1), 35-41.

[15] AlBanna, B., Sakr, M., Moussa, S., & Moawad, I. (2016). Interest aware location-based recommender system using geo-tagged social media. *ISPRS International Journal of Geo-Information*, *5*(12), 245.

[16] Jueajan, B.; Naleg, K.; Pipanmekaporn, L.; Kamolsantiroj, S. Development of location-aware place recommendation system on Android smart phones. In Proceedings of the 2016 Fifth ICT International Student Project Conference (ICT-ISPC), Nakhon Pathom, Thailand, 27–28 May 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 125–128.

[17] Singh, S.; Gupta, R.; Panjabi, S.; Tribhuvan, A.; Jeswani, J. Place Recommendation System. *Int. Res. J. Eng. Technol. (IRJET)* **2017**, *4*, 367–369.

[18] Lee, M.J.; Chung, C.W. A user similarity calculation based on the location for social network services. In Proceedings of the International Conference on Database Systems for Advanced Applications, Hong Kong, China, 22–25 April 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 38–52.

[19] Liao, G.; Jiang, S.; Zhou, Z.; Wan, C.; Liu, X. POI recommendation of location-based social networks using tensor factorization. In Proceedings of the 2018 19th IEEE International Conference on Mobile Data Management (MDM), Aalborg, Denmark, 26–28 June 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 116–124

[20] Mu, W.; Meng, F.; Chu, D. A collaborative filtering recommendation algorithm based on user preferences on service properties. In Proceedings of the 2014 International Conference on Service Sciences, Wuxi, China, 22–23 May 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 43–46.

[21] Sharma, R.; Gopalani, D.; Meena, Y. Collaborative filtering-based recommender system: Approaches and research challenges. In Proceedings of the 2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT), Ghaziabad, India, 9–10 February 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–6.

[22] Takács, G.; Pilászy, I.; Németh, B.; Tikk, D. Scalable collaborative filtering approaches for large recommender systems. *J. Mach. Learn. Res.* **2009**, *10*, 623–656.

[23] Zhang, R.; Liu, Q.D.; Wei, J.X. Collaborative filtering for recommender systems. In Proceedings of the 2014 Second International Conference on Advanced Cloud and Big Data, Huangshan, China, 20–22 November 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 301–308.

[24] Y. Blanco-Fernandez, Gil-Solla A. Pazos-Arias, J. J., M. Ramos-Cabrer, and M. LopezNores. Providing Entertainment by Content-based Filtering and Semantic Reasoning in Intelligent Recommender Systems. IEEE Transactions on Consumer Electronics, 54(2):727–735, 2008.

[25] Van Meteren, R., & Van Someren, M. (2000, May). Using content-based filtering for recommendation. In *Proceedings of the machine learning in the new information age: MLnet/ECML2000 workshop* (Vol. 30, pp. 47-56).

[26] Geetha, G., Safa, M., Fancy, C., & Saranya, D. (2018, April). A hybrid approach using collaborative filtering and content based filtering for recommender system. In *Journal of Physics: Conference Series* (Vol. 1000, No. 1, p. 012101). IOP Publishing.

[27] Ghazanfar, M. A., & Prugel-Bennett, A. (2010, January). A scalable, accurate hybrid recommender system. In *2010 Third International Conference on Knowledge Discovery and Data Mining* (pp. 94-98). IEEE.

[28] Alkan, H., & Celebi, H. (2019, September). The implementation of positioning system with trilateration of haversine distance. In *2019 IEEE 30th annual international symposium on personal, indoor and mobile radio communications (PIMRC)* (pp. 1-6). IEEE.

[29] Drmač, Z., & Veselić, K. (2008). New fast and accurate Jacobi SVD algorithm. II. *SIAM Journal on matrix analysis and applications*, *29*(4), 1343-1362.