SHIV NADAR
INSTITUTION OF EMINENCE DEEMED TO BE
UNIVERSITY
DELHI NCR

NH-91, Tehsil Dadri
Gautam Buddha Nagar
Uttar Pradesh - 201314, India

Tel: +91-120-7170100
+91-120-2662002

Certificate

This is to certify that the project titled 'Location Recommendation System' is submitted by Kavya Lilhare, Mrityunjay Singh and Muskan Mehta, CSE, SNIOE in the partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Technology**. This work was completed under the supervision of Dr. Sonia Khetarpaul. No part of this dissertation/report has been submitted elsewhere for award of any other degree.

Signature with date: 7/12/22

Name of Supervisor: SONIA KHETARPAUL

Designation: ASSISTANT PROF.

Affiliation: SHIV NADAR IOE

Restaurant Recommendation System

Project Report Submitted in Partial Fulfillment of the Requirements for the Degree of

Bachelor of Technology

in

Computer Science and Engineering

Submitted by

Kavya Lilhare: 1910110196
Mrityunjay Singh: 1910110242
Muskan Mehta: 1910110244

Under the Supervision of

Dr. Sonia Khetarpaul

SHIV NADAR
INSTITUTION OF EMINENCE DEEMED TO BE
UNIVERSITY
DELHI NCR

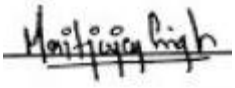
Department of Computer Science and Engineering

December, 2022

Declaration

We declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Student: Kavya Lilhare
Roll Number: 1910110196
Signature: 

Student: Mrityunjay Singh
Roll Number: 1910110242
Signature: 

Student: Muskan Mehta
Roll Number: 1910110244
Signature: 

Acknowledgement

First and foremost, we would like to express our sincere gratitude to our mentor, **Dr. Sonia Khetarpaul**, for her continuous support, patience, motivation, enthusiasm, and immense knowledge. Without her valuable guidance and dedicated involvement in every step, this project would have never been accomplished. She taught us the methodology to carry out research and to present the research works as clearly as possible. From her, we learned the importance and value of studying topics in-depth without being satisfied with surface-level knowledge. Understanding the sophisticated field of Deep Learning from scratch seemed very daunting to us initially, but her vision and support throughout made the task easy. We could not have asked for a better advisor and mentor. Completing the project required more than academic support, and we have our family, friends, and professors to thank for listening to and, at times, tolerating us. We cannot begin to express our gratitude and appreciation for their support.

Content

List of Figures

Abstract

When going out to a new restaurant or cafe people typically use websites to explore neighboring places and then select one based on an average rating. When reviewing a restaurant, different people have different interests and perspectives. The purpose of this project is to create a system that recommends places to eat according to user history, preferences and the history of other users as well. The unique selling point of our project is the liberty of being able to follow knowns through a food and restaurant app and getting the option to explore the choices of your connections.

Simple algorithms that seek to provide the most pertinent and useful recommendations are called recommender systems or recommendation systems.
By selecting relevant information from a vast database, precise objects (movies, products, events, and articles) are provided to the user (clients, readers, guests, and application users). By studying client decisions and providing outcomes that are in line with their needs and interests, recommendation engines are able to identify data patterns in the information dataset.
By examining the symmetry between various user activity variables, our recommendation system helps consumers make decisions by proposing eateries in the future.

The analysis that follows will examine the ratings for food, restaurants, and services and compare them to various personal preferences (marriage status, interests, profession/student), personalities, and financial constraints.

1. Introduction and Literature Review

The following section will give the reader the perspective of the writers of this paper and also a brief of recommendation systems in general. The main idea would further be explained in detail in the sections afterwards.

1.1 Introduction

We live in a country with the world's largest youth population. And what do we know about this youth? One thing that we see very often is the will to grow and get successful, especially in urban places. And with this, the dependance of the population on fast-food vendors and restaurants is also rapidly increasing.

Be it picking up a quick meal before office, or a casual dine-out or even a business lunch meet, everyone wants to get the best service available. And these services should also comply with the financial and time constraints of the customers.

The public's perceptions and feelings regarding a restaurant's food and cleanliness have a significant impact on the customer's perception of the restaurant. And for the same reason, every restaurant and fast-food vendor is trying their best to provide the best services, be it the aesthetics of the dining area, the aesthetics and taste of the food, the quality of service, etc.

With an abundance of dining options and home delivery options, people are heavily relying on recommendation systems. This is where big organizations like Swiggy and Zomato are working and trying to deliver to customers.

There are various algorithms that these tech giants use. Some of the most popular ones being machine learning based algorithms like Matrix Factorisation and XGBoost (Extreme Gradient Boosting). Then there are other ways to figure out predictions such as KNN (K-Nearest Neighbors), K-Means Clustering etc.

A recommendation system provides suggestions to the users through a filtering process that is based on user preferences and browsing history. Playlist creators for video and music services like Netflix, YouTube, and Spotify frequently use recommendation systems. Product recommenders are used by online retailers like Amazon and social networking sites like Facebook and Twitter. Their use has phenomenally increased over

the last decade. The objective of this project is to build a restaurant recommendation system/model according to the geographic location of the user based on their preferences and likings.

Recommendation engines are simply an automated version of people sitting at shop counters who give suggestions to customers based on their previous experiences and preferences. If a customer asks for a product, they are presented with the demanded product and related ones that the customer may be interested in. These people sitting at shop counters have received extensive training in cross-selling and up-selling. Recommendation engines work in a very similar fashion. An engine's ability to recommend personalized content based on past behavior is incredible. It delights customers and gives them a reason to return to the website.

The system would be capable of recommending places to eat or order from according to the user's history, restaurant data available and also through a social network of friend-users (similar to the system of *Suggestions* on instagram and facebook). We will be using exploratory data analysis tools and algorithms like KNN and also machine learning based algorithms like **XGBoost** and **Matrix factorisation** to build the recommendation system. These would work together to bring out the best results according to the user preferences and history.

The results will be presented through a neat-looking frontend which would also present the locations of these restaurants on a map, making it easy for the user to find out the location of the restaurant.

1.2 Motivation

In today's rapidly evolving and fast paced world, people have hectic schedules and time is of utter importance. Diverse and huge amounts of data can be found on the internet in recent times. According to a report, about 2.5 quintillion bytes of data is generated everyday. Effective recommendations can save a lot of time in browsing places by narrowing it according to the user preferences on just a click, that too at the comfort of your own hands. We aim to build a restaurant recommendation system so that customers don’t have to struggle to find new places to dine out or to order from.

Recommendation engines are nothing more than an automated version of a person who sits at a shop counter and makes predictions about the customer based on past interactions or customer preferences. When a customer requests a product, he not only displays that item but also any connected items that are available for purchase. They have received thorough cross-selling and up-selling training. The recommendation engines also agree. It's amazing how these systems can suggest customized material based on past behavior. It makes customers happy and encourages them to visit the website frequently.

1.3 Literature Review

The presently available recommendation system makes use of methods from numerous disciplines, including machine learning, data mining, databases, statistics, similarity testing, etc. **Authors in [8]** have explained three conceptual techniques that are typically used to develop the recommendation system:

- **Collaborative Filtering -**

By analyzing the preferences and data from numerous users, collaborative filtering predicts a user's interests. This is accomplished by applying approaches involving cooperation among numerous agents, data sources, etc. to filter data for information or patterns.

- **Content-Based Filtering -**

Recommendations are generated by content-based systems based on the preferences and profile of the user. They strive to match customers with products that they have previously loved. The characteristics of products that the user likes serve as a general basis for determining the degree of similarity between items.

- **Hybrid Approach -**

Each type of recommendation system has advantages and disadvantages. When applied alone, several of these techniques can appear to be limiting, especially when there are numerous data sources available for the problem. Hybrid recommender systems are those created to produce reliable inferences from a variety of available data sources.

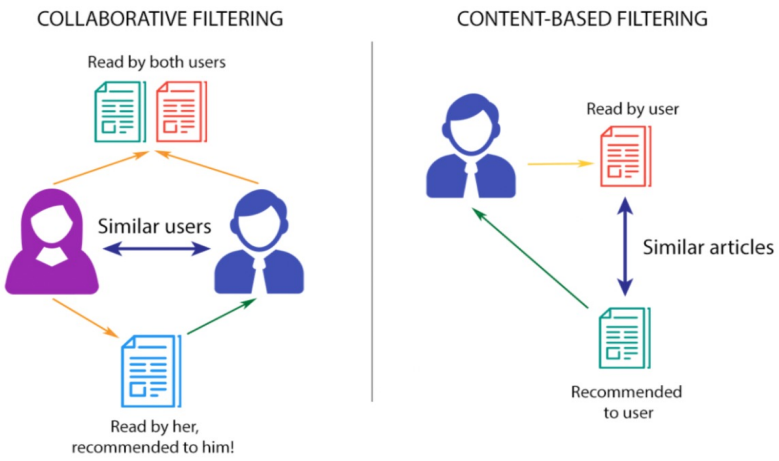


Figure 1 Content based and Collaborative filtering (Src: Ref[3])

Few authors have proposed the use of user's reviews (in the form of text and rating data) in order to examine the behavior of a user. **Authors in [6]** have proposed using NLP machine learning algorithm and analyzing the reviews further helping in recognising the likings of the user.

Many researchers have used methods which have implemented Slope One, k-Nearest Neighbors Algorithm and multiclass SVM classification.

Authors in [5] have shown that collaborative filtering is the major technique but has a major drawback to being unable to solve the cold start problem. Hybrid filtering gives the best results.Many projects use hybrid recommendation systems which make use of a number of strategies, including integrating several recommenders, such as content-based filtering using several criterias, collaborative models' properties etc.

Authors in [9] suggest a content-filtering recommender system that ranks individual internet reviews for each of the five customer segments and provides each review a numerical score. Later, web reviews for specific consumers can be sorted according to their preferences for restaurants using the numerical scores.

Authors in [10] use weight based score calculation and cosine similarity matrix to build the model for Dhaka city restaurants.The user will provide preferences such as location, price range, food type, and ratings, after which the model will filter the dataset based on score and give the user the best recommendation.When the user selects one restaurant from the list, the model will run once more and use content-based filtering to identify similar restaurants and recommend them to the user

Authors in [4] have used hybrid matrix factorisation where user features and item features were fitted to the interaction matrix. AUC score was used as a metric to find out the accuracy of the model.

We aim to build a recommendation system which uses different algorithms like **Matrix factorisation** and **XGBoost** which takes in the geographical location of the user and its preferences as an input.

2. Dataset and Preprocessing

The following section explains how the dataset has been obtained and gives its description. It also explains the steps used for preprocessing the data.

2.1 DataSet

The dataset was obtained from a recommender system prototype. The task was to generate a top-n list of restaurants according to the consumer preferences.

Table 1: Context attributes

Service model (23 attributes)
latitude,longitude,address,city,state,country,fax,ZIP, alcohol,smoking,dress,accessibility,price,franchise, ambiance,space,services,parking,cuisine,phone,accepts, days,hours
User model (21 attributes)
latitude,longitude,smoking,alcohol,dress,ambiance,age, transportation,marital-status,children,interests, personality,religion,occupation,favorite-color,weight, height,budget,accepts,accessibility,cuisine
Environment model (2 attributes)
time,weather

Figure 2 Dataset (Src: Ref[16])

Service model. It describes the restaurant's characteristics. The model has 23 attributes; 6 of them: cuisine, alcohol, smoking, dress, acceptance (type of payment) and parking were defined according to http://chefmoz.org, an online dining guide. Values are selected by the user from several possible options shown by a GUI when he/she rates a new restaurant.

User model. It describes the user profile. The model has 21 attributes; 19 of them are provided by the user when he/she signs into the system the first time or modifies his/her personal information.

Environment model. It refers to the time and weather of the user's location; their values are acquired from Web services. This information restricts the search to available restaurants that have appropriate installations.

The accuracy scores obtained from various Classifier models were compared and analyzed and are shown in the chart below.

Through data collection, preprocessing, and data cleaning, we combined these datasets in accordance with our preferences. The attributes of the dataset we used are listed below:

Dataset 1 (chefmozaccepts.csv)

- **Instances: 1314**
- Attributes: 2
- placeID: Nominal
- Rpayment: Nominal, 12
  [cash,VISA,MasterCard-Eurocard,American_Express,bank_debit_cards,checks,
  Discover,Carte_Blanche,Diners_Club,Visa,Japan_Credit_Bureau,gift_certificate
  s]

Dataset 2 (chefmozcuisine.csv)

- **Instances: 916**
- Attributes: 2
- placeID: Nominal
- Rcuisine: Nominal, 59
  [Afghan,African,American,Armenian,Asian,Bagels,Bakery,Bar,Bar_Pub_Brewe
  ry,Barbecue,Brazilian,Breakfast-Brunch,Burgers,Cafe-Coffee_Shop,
  Cafeteria,California,Caribbean,Chinese,Contemporary,Continental-European,De
  li-Sandwiches,Dessert-Ice_Cream,Diner,Dutch-Belgian,Eastern_European,Ethio
  pian,Family,Fast_Food,Fine_Dining,French,,Game,German,Greek,Hot_Dogs,
  International,Italian,Japanese,Juice,Korean,Latin_American,Mediterranean,Mex
  ican,Mongolian,Organic-Healthy,Persian,
  Pizzeria,Polish,Regional,Seafood,Soup,Southern,Southwestern,Spanish,Steaks,S
  ushi,Thai,Turkish,Vegetarian,Vietnamese]
- Dataset 3 (chefmozhours4.csv)

- Instances: 2339
- Attributes: 3
- placeID: Nominal
- hours: Nominal, Range:00:00-23:30
- days:Nominal, 7 [Mon;Tue;Wed;Thu;Fri;Sat;Sun]

Dataset 4 (chefmozparking.csv)

- **Instances: 702**
- Attributes: 2
- placeID: Nominal
- parking_lot:Nominal, 7[public,none,yes,valet_parking,free,street,validated_parking]

Dataset 5 (geoplaces2.csv)

- **Instances: 130**
- Attributes: 21
- placeID: Nominal
- latitude: Numeric
- longitude: Numeric
- the_geom_meter: Nominal (Geospatial)
- name: Nominal
- address: Nominal,Missing: 27
- city: Nominal, Missing: 18
- state: Nominal, Missing: 18
- country: Nominal, Missing: 28
- fax: Numeric, Missing: 130
- zip: Nominal,Missing: 74
- alcohol: Nominal, Values: 3 [No_Alcohol_Served,Wine_Beer,Full_Bar]
- smoking_area: Nominal, 5 [none,only_at_bar,permitted,section,not_permitted]
- dress_code: Nominal, 3 [informal,casual,formal]
- accessibility: Nominal, 3 [no_accessibility,completely,partially]
- price: Nominal, 3 [medium,low,high]

- url: Nominal, Missing: 116
- Rambience: Nominal, 2 [familiar,quiet]
- franchise: Nominal, 2 [t,f]
- area: Nominal, 2 [open,closed]
- other_services: Nominal, 3 [none,internet,variety]

Dataset 6 (rating_final.csv)

- **Instances: 1161**
- Attributes: 5
- userID: Nominal
- placeID: Nominal
- rating: Numeric, 3 [0,1,2]
- food_rating: Numeric, 3 [0,1,2]
- service_rating: Numeric, 3 [0,1,2]

Dataset 7 (usercuisine.csv)

- **Instances: 330**
- Attributes: 2
- userID: Nominal
- Rcuisine: Nominal, 103
  [Afghan,African,American,Armenian,Asian,Australian,Austrian,Bagels,Bakery,Bar,Bar_Pub_Brewery,Barbecue,Basque,Brazilian,Breakfast-Brunch,British,Burgers,Burmese,Cafe-Coffee_Shop,Cafeteria,Cajun-Creole,California,Cambodian,Canadian,Caribbean,Chilean,Chinese,Contemporary,Continental-European,Cuban,Deli-Sandwiches,Dessert-Ice_Cream,Dim_Sum,Diner,Doughnuts,Dutch-Belgian,Eastern_European,Eclectic,Ethiopian,Family,Fast_Food,Filipino,Fine_Dining,French,Fusion,Game,German,Greek,Hawaiian,Hot_Dogs,Hungarian,Indian-Pakistani,Indigenous,Indonesian,International,Irish,Israeli,Italian,Jamaican,Japanese,Juice,Korean,Kosher,Latin_American,Lebanese,Malaysian,Mediterranean,Mexican,Middle_Eastern,Mongolian,Moroccan,North_African,Organic-Healthy,Pacific_Northwest,Pacific_Rim,Persian,Peruvian,Pizzeria,Polish,Polynesian,Portuguese,Regional,Romanian,Russian-Ukrainian,Scandinavian,Seafood,Soup,Southeast

_Asian,Southern,Southwestern,Spanish,Steaks,Sushi,Swiss,Tapas,Tea_House,Tex-Mex,Thai,Tibetan,Tunisian,Turkish,Vegetarian,Vietnamese]

Dataset 8 (userpayment.csv)

- **Instances: 177**
- Attributes: 2
- userID: Nominal
- Upayment: Nominal, 5 [cash,bank_debit_cards,MasterCard-Eurocard,VISA,American_Express]

Dataset 9 (userprofile)

- **Instances: 138**
- Attributes: 19
- userID: Nominal
- latitude: Numeric
- longitude: Numeric
- the_geom_meter: Nominal (Geospatial)
- smoker: Nominal, Missing: 3, 2 [false,true]
- drink_level: Nominal, 3 [abstemious,social drinker,casual drinker]
- dress_preference:Nominal, Missing: 5, 4 [informal,formal,no preference,elegant]
- ambience: Nominal, Missing: 6, 3 [family,friends,solitary]
- transport: Nominal, Missing: 7, 3 [on foot,public,car owner]
- marital_status: Nominal, Missing: 4, 3 [single,married,widow]
- hijos: Nominal, Missing: 11, 3 [independent,kids,dependent]
- birth_year: Nominal
- interest: Nominal, 5 [variety,technology,none,retro,eco-friendly]
- personality: Nominal, 4 [thrifty-protector,hunter-ostentatious,hard-worker,conformist]
- religion: Nominal, 5 [none,Catholic,Christian,Mormon,Jewish]
- activity: Nominal, Missing: 7, 4 [student,professional,unemployed,working-class]

- color: Nominal, 8 [black,red,blue,green,purple,orange,yellow,white]
- weight: Numeric
- budget: Nominal, Missing: 7, 3 [medium,low,high]
- height: Numeric

2.2 Combined DataSet

We combined the datasets after analysis in accordance with our preferences.

Dataset final (dataset.csv)

- **Instances : 1161**
- userID
- placeID
- rating
- food_rating
- Service_rating
- Rating_sum
- Latitude
- Longitude
- Smoker
- Drinker_level
- Dress_preferences
- Ambience
- Transport
- Marital_status
- Hijos
- Birth_year
- Age
- Interest
- Personality
- Religion
- Activity
- Color

- Weight
- Budget
- Height
- Res_latitude
- Res_longitude
- The_geo_meter
- Name
- Address
- City
- State
- Country
- Fax
- Zip
- Alcohol
- Smoking_area
- Dress_code
- Accessibility
- Price
- Url
- Rambience
- Franchise
- Area
- Other_services

2.3 Data Pre-processing

A copy of the final dataset ‘df’ was made, named ‘df2’. With the original dataset intact, processing was performed on the new copy. Unnecessary columns that are not required for recommending restaurants to the users were removed.

The removed columns are - *food_rating, service_rating, latitude, longitude, birth_year, color, weight, height, res_latitude, res_longitude, the_geom_meter, name, address, city, state, country, fax, zip, url* and *franchise.*

Further, the columns *dress_preference, ambience, transport, marital_status, hijos, interest, personality, religion, activity, budget, alcohol, smoking_area, dress_code, accessibility, price, Rambience, area, other_services* were checked for null values. Depending on the column and how the values affect the result, the null values were replaced by either the mean, median or the row with the missing value was removed.

The final dataset ‘df2’ with the unnecessary columns and rows removed and with no null values has 1042 rows and 26 columns.

3. Approach

In the section ahead, we will explain our idea of how recommendations will be made and by using which algorithms.

3.1 Model

The model consists of four major algorithms to go study the data and predict restaurants based on what it learns. The algorithms are : Random Forest, XGBoost, K-Nearest Neighbors and Matrix Factorisation.
Matrix Factorisation is the most accurate algorithm out of the four. The final prediction to the user would be made based on only Matrix Factorisation and KNN because of the higher accuracies.

3.1.1 Algorithm 1: Random Forest

In order to understand Random Forest Classifier properly, we need to start by understanding what decision trees are.

Decision Trees : Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation.

Some advantages of decision trees are:

- Simple to understand and to interpret. Trees can be visualized.
- Requires little data preparation. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Note however that this module does not support missing values.
- The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.
- Able to handle both numerical and categorical data. However, the scikit-learn implementation does not support categorical variables for now. Other techniques are

usually specialized in analyzing datasets that have only one type of variable. See algorithms for more information.

- Able to handle multi-output problems.
- Uses a white box model. If a given situation is observable in a model, the explanation for the condition is easily explained by boolean logic. By contrast, in a black box model (e.g., in an artificial neural network), results may be more difficult to interpret.
- Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model.
- Performs well even if its assumptions are somewhat violated by the true model from which the data were generated.

The disadvantages of decision trees include:

- Decision-tree learners can create over-complex trees that do not generalize the data well. This is called overfitting. Mechanisms such as pruning, setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem.
- Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble.
- Predictions of decision trees are neither smooth nor continuous, but piecewise constant approximations as seen in the above figure. Therefore, they are not good at extrapolation.
- The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts. Consequently, practical decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement.
- There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.

■ Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

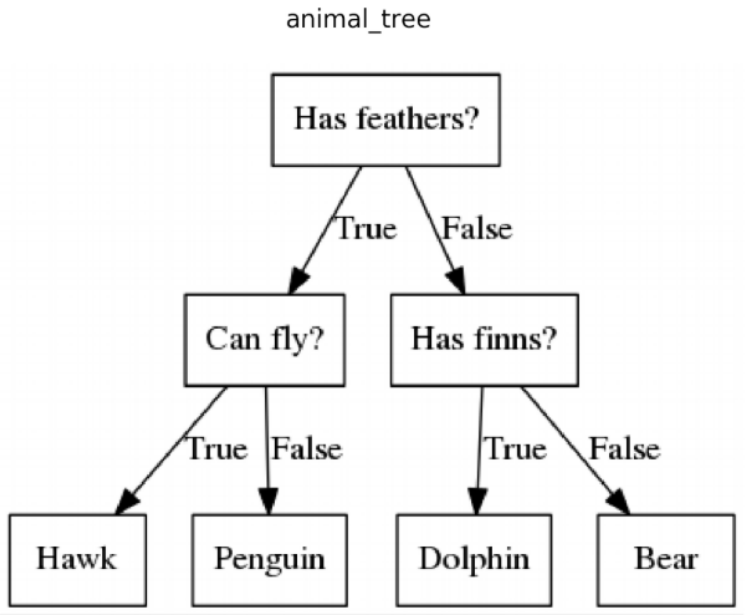


Figure 3 Animal Decision Tree

Now coming to Random Forest Classifier, it is a commonly used machine learning algorithm which combines the output of multiple decision trees to reach a single result. Random forest algorithms have three main hyperparameters which need to be set before training- node size, the number of trees, and the number of features sampled. The random forest algorithm is made up of a collection of decision trees, and each tree in the ensemble consists of a data sample drawn from a training set with replacement, called the bootstrap sample. Of that training sample, one-third of it is set aside as test data, known as the out-of-bag (OOB) sample.

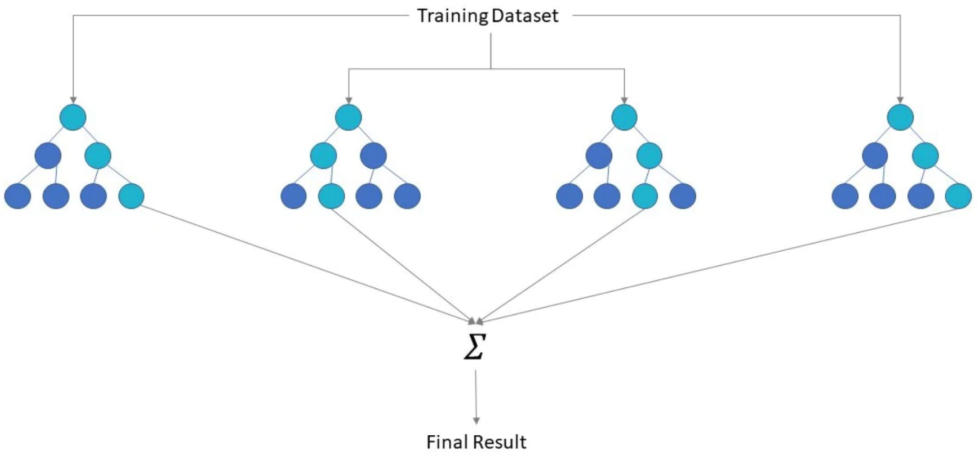


Figure 4 Random Forest classifier (Src:Ref[13])

3.1.2 Algorithm 2: XGBoost

In order to understand the concept of XGBoost, we need to start by understanding the concept of **Gradient Boosting**.

Gradient Boosting : Gradient boosting is a machine learning technique used in regression and classification tasks, among others. It gives a prediction model in the form of an ensemble of weak prediction models, which are typically decision trees. When a decision tree is the weak learner, the resulting algorithm is called gradient-boosted trees; it usually outperforms **random forest**. A gradient-boosted tree model is built in a stage-wise fashion as in other boosting methods, but it generalizes the other methods by allowing optimization of an arbitrary differentiable loss function.

Regression Analysis - *is a set of statistical processes for estimating the relationships between a dependent variable and one or more independent variables. The most common form of regression analysis is linear regression, in which one finds the line that most closely fits the data according to a specific mathematical criterion*

XGBoost, which stands for **Extreme Gradient Boosting**, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library. It provides parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems.

At this point, it's a little crucial to also understand "supervised machine learning". Supervised machine learning uses algorithms to train a model to find patterns in a dataset with labels and features and then uses the trained model to predict the labels on a new dataset's features.

A **Gradient Boosting Decision Trees** (GBDT) is a decision tree ensemble learning algorithm similar to random forest, for classification and regression. Ensemble learning algorithms combine multiple machine learning algorithms to obtain a better model.

Both random forest and GBDT build a model consisting of multiple decision trees. The difference is in how the trees are built and combined.

Random forest uses a technique called bagging to build full decision trees in parallel from random bootstrap samples of the data set. The final prediction is an average of all of the decision tree predictions.

The term "gradient boosting" comes from the idea of "boosting" or improving a single weak model by combining it with a number of other weak models in order to generate a collectively strong model. Gradient boosting is an extension of boosting where the process of additively generating weak models is formalized as a gradient descent algorithm over an objective function. Gradient boosting sets targeted outcomes for the next model in an effort to minimize errors. Targeted outcomes for each case are based on the gradient of the error (hence the name gradient boosting) with respect to the prediction.

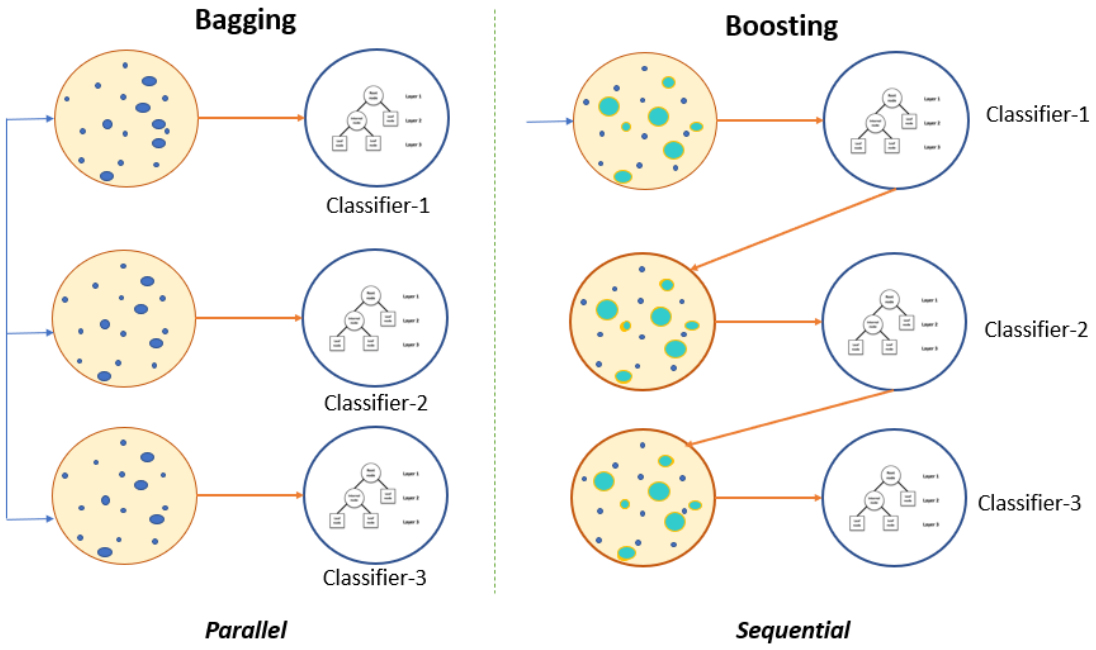


Figure 5 Bagging vs Boosting (Src: Ref[14])

GBDTs iteratively train an ensemble of shallow decision trees, with each iteration using the error residuals of the previous model to fit the next model. The final prediction is a weighted sum of all of the tree predictions. Random forest “bagging” minimizes the variance and overfitting, while GBDT “boosting” minimizes the bias and underfitting.

XGBoost is a scalable and highly accurate implementation of gradient boosting that pushes the limits of computing power for boosted tree algorithms, being built largely for energizing machine learning model performance and computational speed. With XGBoost, trees are built in parallel, instead of sequentially like GBDT. It follows a level-wise strategy, scanning across gradient values and using these partial sums to evaluate the quality of splits at every possible split in the training set.

3.1.3 Algorithm 3: KNN

The k-nearest neighbors algorithm(KNN) is a **non-parametric**, supervised learning classifier which uses proximity (based on the idea that the observations closest to a given data point are the most "similar" observations in a data set, and we can therefore classify unforeseen points based on the values of the closest existing points.) to make

predictions about the grouping of an individual data point. While it can be used for either regression or classification problems, it is typically used as a classification algorithm, working off the assumption that similar points can be found near one another.

A learning model that summarizes data with a set of parameters of fixed size (independent of the number of training examples) is called a parametric model. No matter how much data you throw at a parametric model, it won't change its mind about how many parameters it needs. On the other hand, algorithms that do not make strong assumptions about the form of the mapping function are called nonparametric machine learning algorithms. By not making assumptions, they are free to learn any functional form from the training data.

K is the number of nearest neighbors to use. For classification, a majority vote is used to determine which class a new observation should fall into. Larger values of K are often more robust to outliers and produce more stable decision boundaries than very small values (K=3 would be better than K=1, which might produce undesirable results).

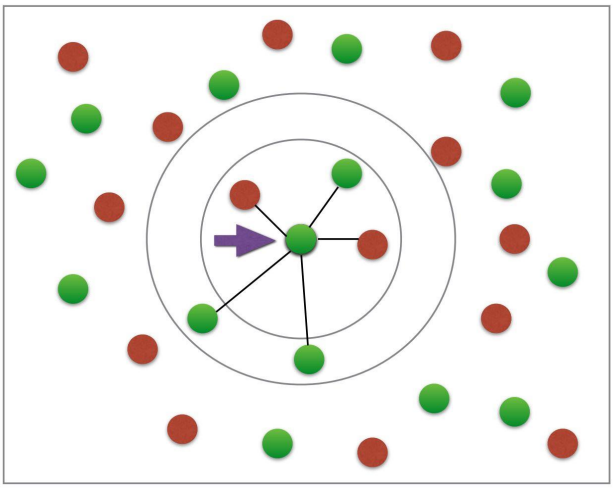


Figure 6 KNN

Figure 6 is an example of the K-Nearest Neighbors algorithm. The inner circle shows the closest 3 nodes to the green node to which the arrow is pointing.

It's also worth noting that the KNN algorithm is also part of a family of "lazy learning" models, meaning that it only stores a training dataset versus undergoing a training stage. This also means that all the computation occurs when a classification or prediction is being made. Since it heavily relies on memory to store all its training data, it is also referred to as an instance-based or memory-based learning method.

3.1.4 Algorithm 4: Matrix Factorisation

Suppose we have a table(depicted in Figure 7) containing the data ranking of 5 restaurants by the 4 customers for the places that they have visited. As each consumer has not visited all the places, the matrix contains null values. Matrix factorisation is used to predict these missing values and hence recommend places based on the preferences of the user.

	Res1	Res2	Res3	Res4	Res5
User1		5	4	2	1
User2	1			5	3
User3	1	4	4	1	
User4		2			2

Figure 7 Restaurant-User Rating Matrix

Matrix factorisation is an embedding model which decomposes the original matrix into two matrices.It has K latent features. The relationship between a user and the features is represented by matrix P, whereas the relationship between an object and the features is represented by matrix Q. Given with the input of two matrices P (|U|*k) and Q (|D|*k), it would generate the product result R.

$$\mathbf{R} \approx \mathbf{P} \times \mathbf{Q}^T = \hat{\mathbf{R}}$$

It uses collaborative filtering to ascertain the relationship between the entities of places and users. Latent features are used to compare and predict based on both item and user entities by determining the link between users and restaurant matrices.

Matrix decomposition can be classified into three type-

1. **LU decomposition** - Decomposition of matrix into L and U matrix where L is lower triangular matrix and U is upper triangular matrix, generally used for finding the coefficient of linear regression. This decomposition failed if matrix can't have decomposed easily
2. **QR matrix decomposition** - Decomposition of matrix into Q and R where Q is square matrix and R is upper triangular matrix (not necessary square). Used for eigen system analysis
3. **Cholesky Decomposition** - This is the mostly used decomposition in machine learning. Used for calculating linear least square for linear regression.

Steps-

Single Value Decomposition(SVD) is used to find the values of the matrices.
It works in the following ways:

1. The matrix M is transformed into a square matrix by multiplying it by it's transpose i.e $M*M^T$
2. The eigenvalues and eigenvectors of Matrix $M*M^T$ are calculated. The results of this will correspond to the Σ and U matrices
3. V is solved using $V = 1/\Sigma * M^T * U$

It uses Gradient Descent Algorithm for optimization i.e. to minimize the mean square error function.Let r is rating defined for user u and item i, p is row of M for a user and q is the column of transpose(U) for a specific item i.

$$MSE = (Y - \widetilde{Y})^2 = (r_{ui} - \sum_{a}^{b} p_u q_i)^2$$

Regularization is done to prevent data from overfitting.

4. Results and Accuracies

The model is successful in making recommendations based on user profile and the accuracy of the model is analyzed using the following metrics-

MAE (Mean Absolute Error) : It measures the average of the residuals in the dataset.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|$$

MSE (Mean Squared Error) : It measures the variance of the residuals.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$$

RMSE (Root Mean Squared Error) : It measures the standard deviation of residuals.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2}$$

The various results and accuracies obtained by the different algorithms are shown below-

4.1 Random Forest:

The final accuracy for Random Forest Classifier is **53.68%**. The reason for the relatively low accuracy of Random Forest is the limited data available for its training.

```
Accuracy: 61.24%
Accuracy: 48.80%
Accuracy: 50.72%
Accuracy: 51.20%
Accuracy: 51.20%
Accuracy: 53.59%
Accuracy: 53.59%
Accuracy: 54.55%
Accuracy: 56.46%
Accuracy: 55.50%
Final Accuracy: 53.68%
```

Figure 8 Ransom Forest Result

4.2 XGBoost: (across different values of n_estimator parameter)

"n_estimators" refers to the number of turns or iterations in which the algorithm will try to learn from the dataset.

Accuracy for XGBoost maxed out at around **68.2**% with the n_estimator value being a little above 420

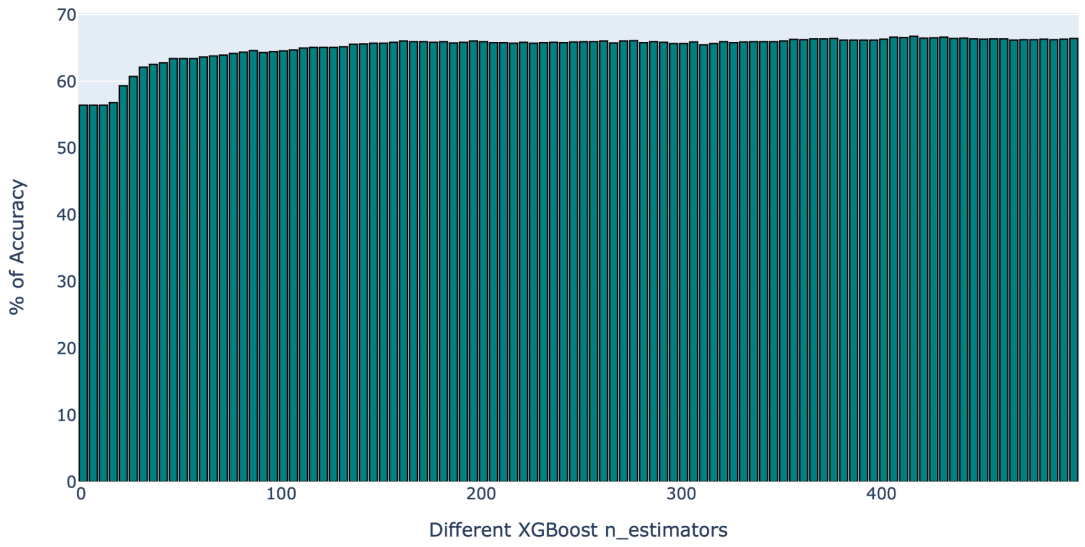


Figure 9 XGBoost Result

4.3 K-Nearest Neighbors:

K-Nearest Neighbors can work for different values of K (or different number of neighbors). After checking accuracy for different values of K, the result is presented in the above graph. It can be seen that the highest accuracy was shown with a K value of 5. Accuracy maxed out at **71.2%** with **K = 5**.

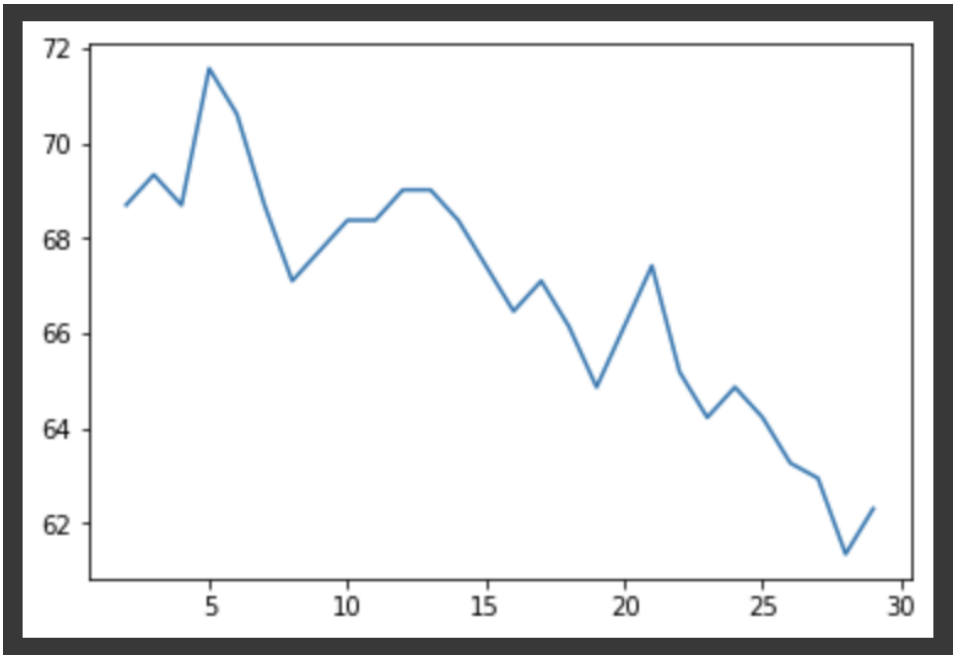


Figure 10 KNN Result

4.4 Matrix Factorisation:

After tuning the parameters a little and splitting the data in an 80:20 ratio for training and testing, we finally got an accuracy of **87.34%** for Matrix Factorisation

```
Final Accuracy: 87.34%
```

Figure 11 Matrix Factorisation Result

4.5 Final comparison of all accuracies:

As can be seen from Figure 12, the most accurate model amongst the four algorithms is Matrix Factorisation followed by KNN with K = 5.

None of the algorithms have crossed 90% accuracy. The reason behind the low accuracy is the limited data available for us to process and work on. A larger dataset will lead to more data available for training and hence better learning by the algorithms and also better comparisons made.

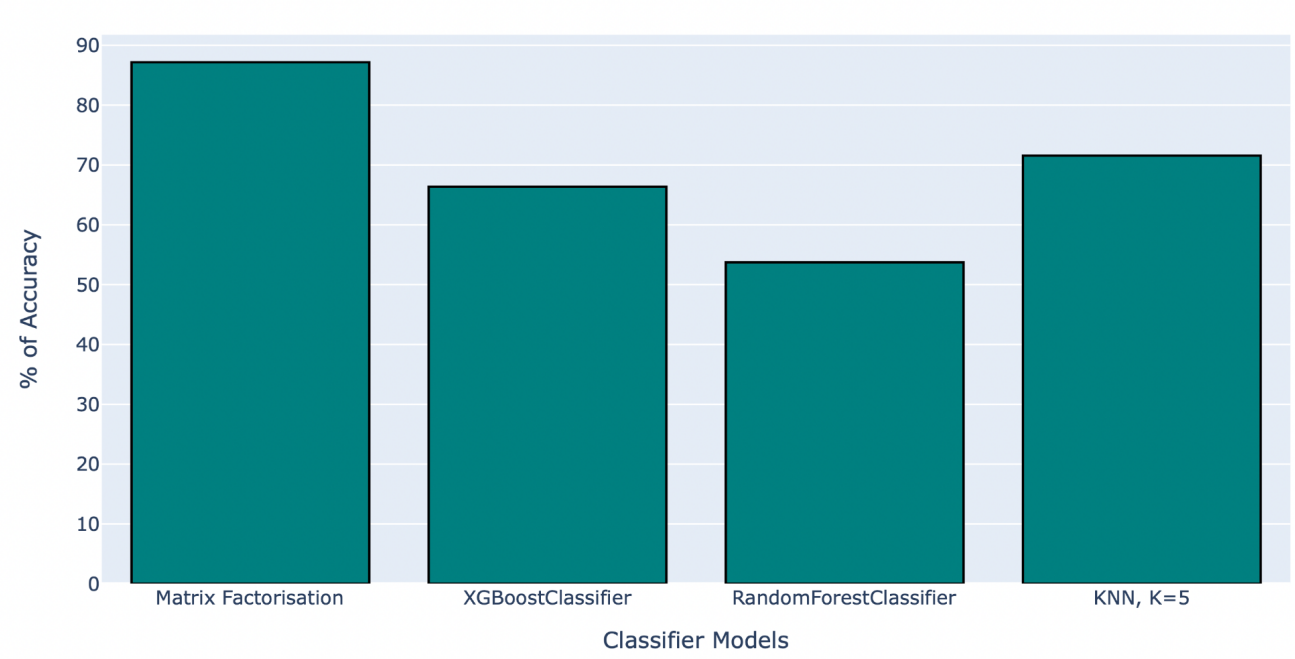


Figure 12 Bar graph comparing the models

5. Web Application

The web application is user friendly and helps user to navigate through it easily and interactive as well.

5.1 User Interface

A screenshot of our website is shown here. It was created with JavaScript, CSS, and HTML. It has a simple design and is intended to be easily understood by the user.

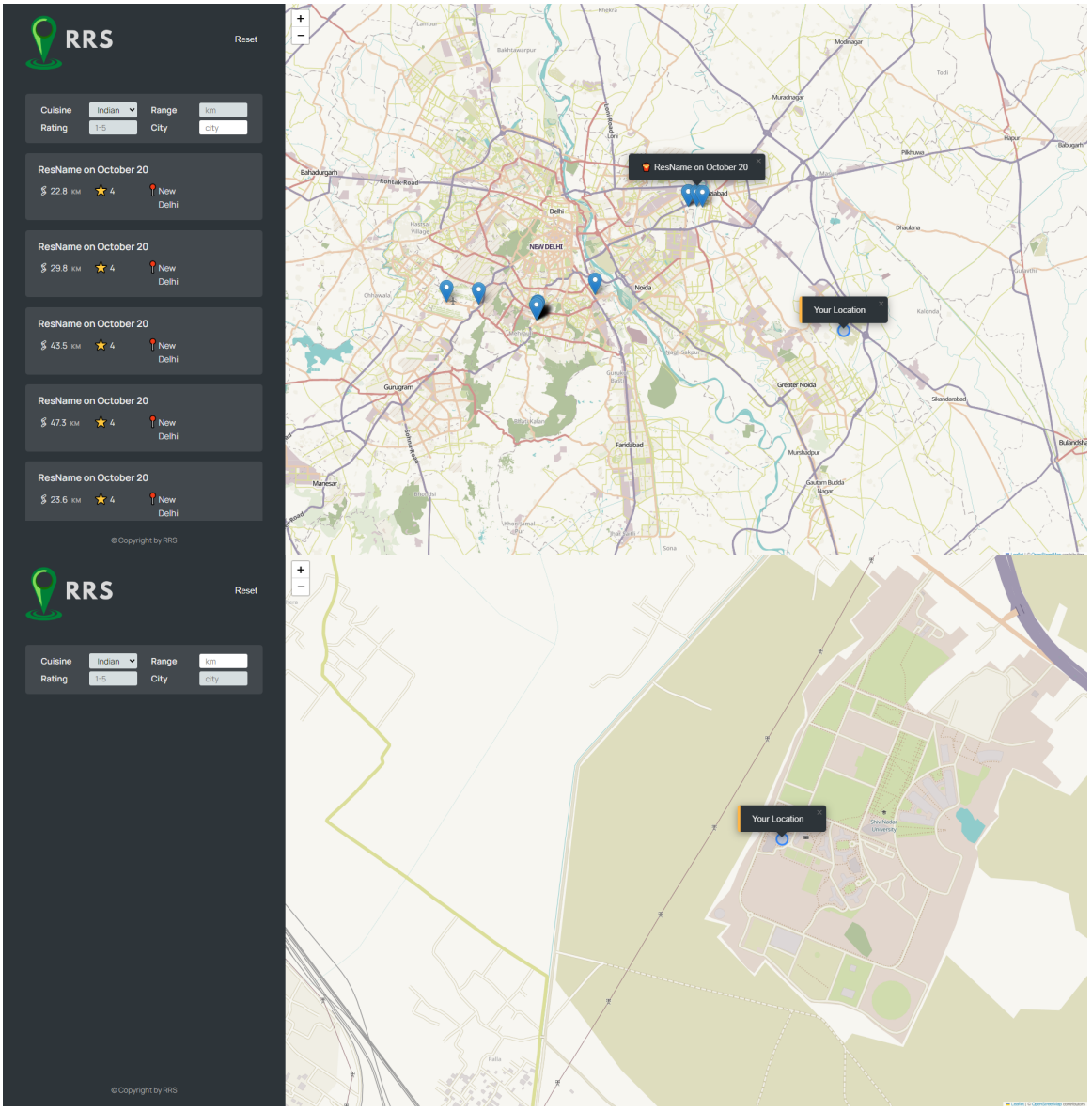


Figure 13 Website UX

5.2 FlowChart for UX

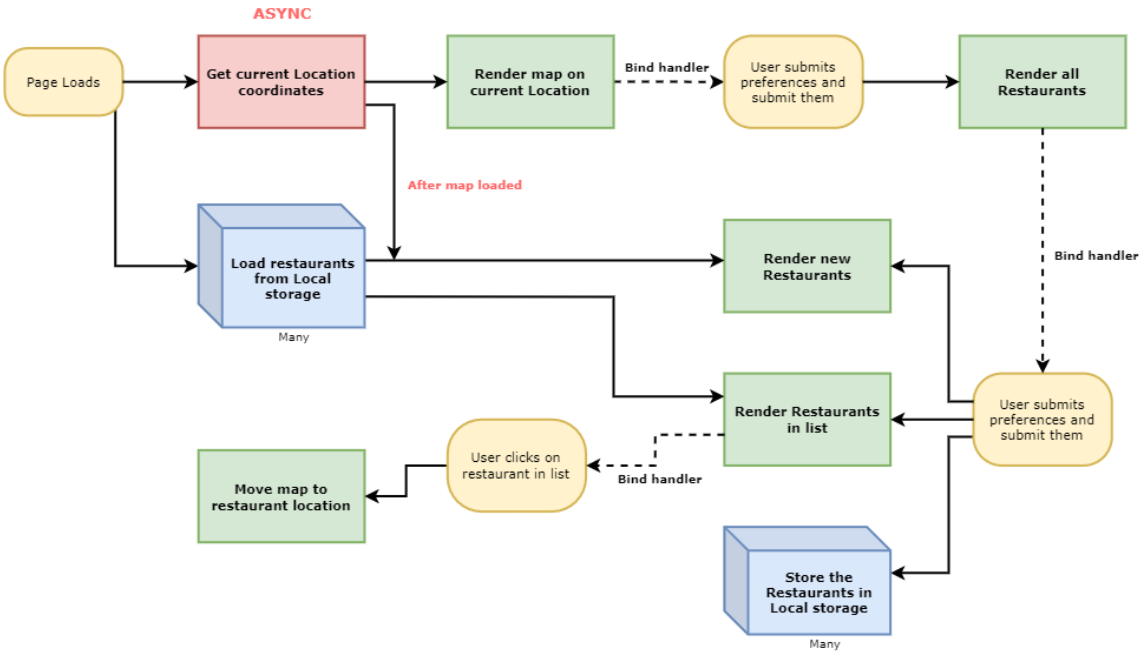


Figure 14 Flowchart for UX

5.3 Architecture

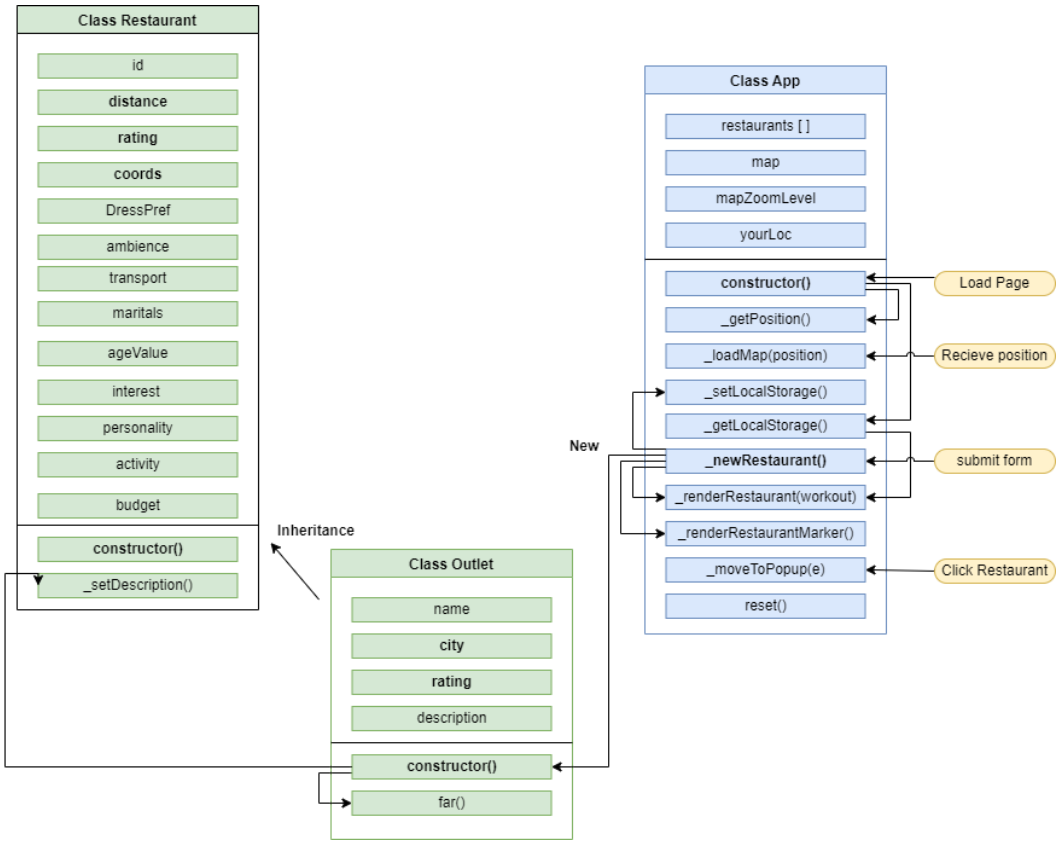


Figure 15 UML Diagram

5.3 FastAPI

FastAPI is a modern, high-performance web framework for building APIs with Python based on standard type hints. It has the following key features:

- Fast to run: It offers very high performance, on par with NodeJS and Go, thanks to Starlette and pydantic.
- Fast to code: It allows for significant increases in development speed.
- Reduced number of bugs: It reduces the possibility for human-induced errors.
- Intuitive: It offers great editor support, with completion everywhere and less time debugging.
- Straightforward: It’s designed to be uncomplicated to use and learn, so you can spend less time reading documentation.
- Short: It minimizes code duplication.
- Robust: It provides production-ready code with automatic interactive documentation.
- Standards-based: It’s based on the open standards for APIs, OpenAPI and JSON Schema.

We **constructed** our **custom-designed** FastAPI that's being hosted on PORT http://127.0.0.1:8000/, And the request we're sending to this API is from http://127.0.0.1:5500/.

6. Future Work and Conclusions

In this paper, we aimed to build a restaurant recommendation system. We experimentally found the best model to be matrix factorisation with an accuracy of 87.34%. This model can be improved and expanded on by adding additional features wherein the user can follow people in his social circle or knowns and can get recommendations based on their preferences.

If in a hypothetical situation our project turns into a product, then as our recommendation system is used by more and more users, more data will be generated for the algorithms to process and the accuracies will get trained better, giving out better results.

Further, an extension for restaurant owners can be made, enabling them to be able to study the data and see where users are inclined towards. This will help the restaurant owners to improve their services and delicacies. The geographic distribution of data would also enable potential restaurants to meet demands by studying where demands are high and where requirement of workforce is high.

To conclude, we'd like to firstly appreciate the existing recommendation systems in the market. With this project we've come to a realization of why giants like Swiggy and Zomato keep their recommendation systems private and don't make their studies public like Netflix. With the growing reliance of people on restaurants and fast food vendors, the amount of data generated by and for these recommendation systems is just gonna increase, and hence it is very crucial to do more and more research in the field to curate more efficient and fast algorithms to process the vast amount of data being generated every single day.

7. Limitations

1. In order to get higher accuracy, a deep study about the different hyperparameters and how each field finally affects the result needs to be done. The way every field/parameter contributes to the result is useful to calculate what all parameters are actually significantly important and which ones can be avoided.

2. After processing the data and removing unwanted information, we were left with around 1070 rows of data. This is a very small number compared to the vast data available with big recommendation systems such as that of Swiggy, Zomato or Netflix. More data results in more training data as well as more testing data and hence more accuracy.

As an example, Swiggy marks over a million orders every single day, and a single order might actually include meals from more than one outlet. So that means, Swiggy is generating over a million rows of data every single day.

8. References

1. https://medium.com/
2. https://towardsdatascience.com/
3. https://www.kaggle.com
4. Restaurant Recommendation System using Machine Learning by Ketan Mahajan, Varsha Joshi, Mohini Khedkar, Jacky Galani, Mayuri Kulkarni. In the International Journal of Advanced Trends in Computer Science and Engineering, 2021.
5. Machine Learning Techniques for Recommender Systems – A Comparative Case Analysis by Binu Thomas , Amruth K John. IOP Conference Series: Materials Science and Engineering, 2020.
6. R. M. Gomathi, P. Ajitha, G. H. S. Krishna and I. H. Pranay, "Restaurant Recommendation System for User Preference and Services Based on Rating and Amenities," *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*, 2019, pp. 1-6, doi: 10.1109/ICCIDS.2019.8862048.
7. Bushra Ramzan, Imran Sarwar Bajwa, Noreen Jamil, Riaz Ul Amin, Shabana Ramzan, Farhan Mirza, Nadeem Sarwar, "An Intelligent Data Analysis for Recommendation Systems Using Machine Learning", *Scientific Programming*, vol. 2019, Article ID 5941096, 20 pages, 2019.
8. Recommender systems: An overview of different approaches to recommendations by Kunal Shah, Akshaykumar Salunke, Saurabh Dongare, Kisandas Antala.Conference: 2017 4th International Conference on Innovations in Information, Embedded and Communication Systems
9. Restaurant Recommendation System Using Machine Learning Algorithms by Sathya Seelan Krishnaraj. May 2021, RIET-IJSET International Journal of Science Engineering and Technology
10. N. Vaidya and A. R. Khachane, "Recommender systems-the need of the ecommerce ERA," 2017 International Conference on Computing Methodologies and Communication (ICCMC), 2017, pp. 100-104, doi: 10.1109/ICCMC.2017.8282616.
11. https://leafletjs.com/reference.html
12. https://scikit-learn.org/
13. https://towardsai.net/

14.https://www.ibm.com/
15. *https://www.pluralsight.com/*
16. https://archive.ics.uci.edu/
17. https://fastapi.tiangolo.com/