Restaurant Recommendation System

Aditi Singhal • Kavya Lilhare • Rahul Madan • Sonia Khetarpaul

Abstract. Recommender Systems or Recommendation Systems are straightforward algorithms that aim to give the most relevant and precise items (movies, products, occasions, articles) to the user (clients, readers, guests, application users) by separating valuable stuff from a gigantic pool of database. Recommendation engines discover data patterns in the information dataset by learning customers' decisions and producing results that co-connect with their necessities and interests.

This recommendation system serves as a decision-making indicator that helps users by suggesting restaurants in the future by exploring the symmetry between multiple user activity characteristics.

The dataset "RestaurantRatersComplete.csv" consists of 4,090 restaurant reviews by 139 people for 130 different restaurants located in Mexico. The following analysis will look at food rating, restaurant rating, and service rating and compare them to different personal preferences (marital status, interests, profession/student), personalities, and budget.

Keywords Recommendation Systems • Content Based System • K-nearest neighbors • XGBoost

1. Introduction.

1.1 Background.

At the point when we need to know new spots to eat, it is normal to ask companions for ideas. Touristic and gastronomic guides are other choices to track down great restaurants. But, companions are probably going to know our taste, current area, and most loved ambiance. Thus, their ideas would be more exact than those given by the guides. In reality, recommender systems are normal web-based services that assist clients to adapt to data overload by recovering valuable things as indicated by their preferences. Collaborative Filtering (CF) is an effective method, which automatizes the social recommender plot. CF predicts users' inclination to think about opinions of clients with similar interests, though content-based recommendation systems construct a model only from the user's favorite items.

Common recommendation approaches just consider user-item-rating data, ignoring contextual information. The disadvantage of this plan is the absence of personalization; along these lines, a vacationer visiting Africa could get a recommendation of a café situated in Brazil. Data such as time, area, or weather conditions can generate tailored recommendations as indicated by the current situation of the user.

1.2 Motivation.

Recommendation engines are nothing but just an automated form of a person who is sitting at shop counter and predict about the customer on the basis of their previous experience or customer choice. If a customer asks him for a product then he not only to show that product, but also the related ones which you could buy. They are well trained in cross-selling and up-selling. So, does the recommendation engines. The ability of these engines to recommend personalized content, based on past behavior is incredible. It brings customer delight and gives them a reason to keep returning to the website.

2. Related Work

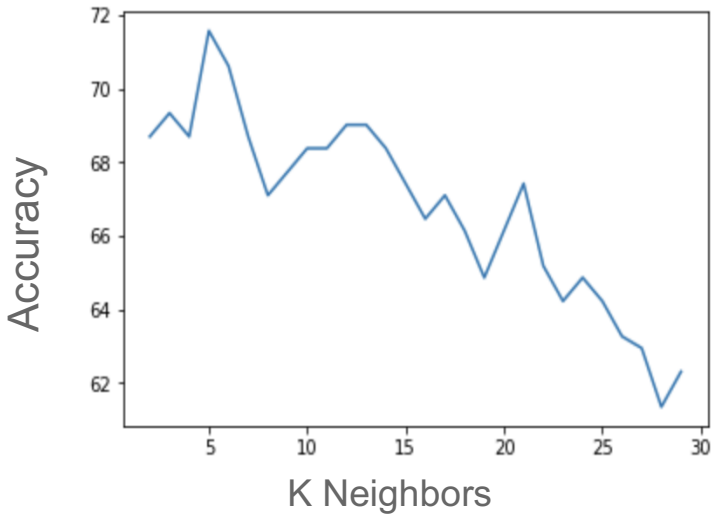
Sonia Khetarpaul
Shiv Nadar University
Email: sonia.khetarpaul@snu.edu.in

As an emerging and under-explored research area, context-aware recommender systems are receiving increasing attention. Although the impact of setting has not been widely considered, related work uncovers that contextual information is significant. For instance, in [8] the impact of context variables in a content-based system was examined. A few clusters were constructed by the measurement of statistical reliance between sets of context variables. Then, the framework was assessed over each cluster to notice changes in recommendations. Results showed that context variables improve predictive accuracy. In [13] it is shown the advantages of contextual information, both in accuracy and utilizing a philosophy to take advantage of semantic ideas. One more fascinating technique comprises parting the user profile into a few sub-profiles [2]. Each sub-profile addresses the client in a time span of the day. Utilizing sub-profiles, the framework could suggest music definitively as it considered time. It was demonstrated the way that precision could increment by making recommendations utilizing sub-profiles rather than a single profile. As a result of the combination of context-oriented data, flexibility and dimensionality reduction requirements on recommender systems have increased.

To manage these issues, machine learning and data mining strategies have proved to be effective. However, feature selection and machine learning techniques have mainly been applied to content-based systems. For instance, in [9] it is shown an intelligent recommender framework based on reinforcement learning. At the point when the framework returns a tremendous sum of results, a feature selection algorithm is applied to reduce the results list. Another approach to feature selection is presented in [3], where the similarity between users is calculated by taking into account the subset of common items that best describes the user preferences. The results show that predictive performance can be improved by a careful selection of item ratings. The recommender system is based on CF and does not include contextual information. In [4] the authors tested their methodology with a recommender system based on Semantic Web technologies and collaborative filtering. Their work focuses on the assessment of relevant model features using decision trees and feature selection techniques. However, the system was not evaluated with the new learned models.

3. Proposed Model.

KNN with K=5 was chosen as the final model based on evaluation metrics and visualizations created while tuning the model's parameters.



4. Methodology.

Different classifier models were implemented and the accuracy scores were calculated for each of them.

1. XGBoost Classifier Model

XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.

In this algorithm, decision trees are created in sequential form. Weights play an important role in XGBoost. Weights are assigned to all the independent variables which are then fed into the decision tree which predicts results. The weight of variables predicted wrong by the tree is increased and these variables are then fed to the second decision tree. These individual classifiers/predictors then ensemble to give a strong and more precise model. It can work on regression, classification, ranking, and user-defined prediction problems.

We needed to convert categorical columns to numerical columns so that the algorithm understands them. This process is called categorical encoding. Categorical encoding is a process of converting categories to numbers. One-hot encoding creates additional features based on the number of unique values in the categorical feature. Every unique value in the category will be added as a feature. One-Hot Encoding is the process of creating dummy variables. In this encoding technique, each category is represented as a one-hot vector.

XGBoost classifier was used and One hot encoding was applied on the following feature attributes and the outcome was 39.83% accuracy.

['Rating_sum', 'smoker', 'drink_level', 'dress_preference', 'ambience', 'transport', 'marital_status', 'hijos', 'age', 'interest', 'personality', 'religion', 'activity', 'budget', 'alcohol', 'smoking_area', 'dress_code', 'accessibility', 'price', 'Rambience', 'area', 'other_services'].

Following this, when the age, area, and dress code features were removed, 37.82% accuracy was achieved.

A final of 56.65% accuracy was obtained using XGBoost when rating_sum and user rating was removed.

2. Random Forest Classifier

It is a meta-estimator that fits a number of decision trees on various sub-samples of datasets and uses an average to improve the predictive accuracy of the model and controls over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement.

3. AdaBoost Classifier

It is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

4. Decision Tree

Given a data of attributes together with its classes, a decision tree produces a sequence of rules that can be used to classify the data.

5. SVM

Support vector machine is a representation of the training data as points in space separated into categories by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

6. Categorical Naive Bayes

Naive Bayes algorithm is based on Bayes' theorem with the assumption of independence between every

pair of features. Naive Bayes classifiers work well in many real-world situations such as document classification and spam filtering.

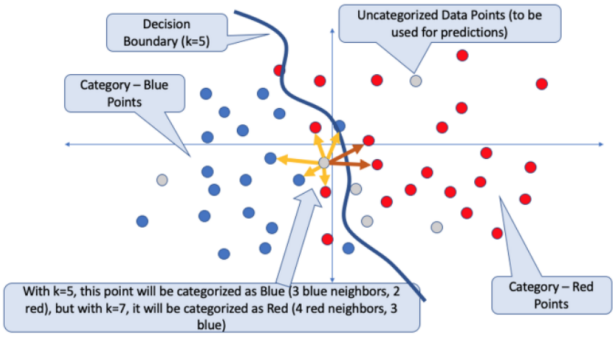
7. SGDClassifer

Stochastic gradient descent is a simple and very efficient approach to fit linear models. It is particularly useful when the number of samples is very large. It supports different loss functions and penalties for classification.

8. KNN model

The KNN classification algorithm is often used for classifying the data, as well as for regression on continuous numeric data. It is used for not just binary classification but for multilevel classifications too. It is a non-parametric machine learning method implying that it doesn't make any assumptions about the data. It also doesn't make any generalizations, and simply checks the neighboring data points to determine the classification of unknown or uncategorized data points. The KNN machine learning method works on the distance between the data points and is based on the basic premise that similar data points are close to each other. It is widely used in customer recommendation systems, image recognition, and decision-making models. If neighbors are categorized in a certain way, you will likely fall under the same category too based on the virtue of you being near to them. Conversely, those who are not near but farther away are likely to fall under a different classification than yours. There can be exceptions (points belonging to a different category
being near to points in a different category) and consequently, there can be errors in the predictions based on distance, especially around the decision

boundaries. We can implement the KNN model and check how accurately it can predict the classification of points. Think of these data points as points in an n-dimensional space and the KNN algorithm predicts their categories based on the distance between the uncategorized data points and their k nearest neighbors, where k is a parameter.



KNN Algorithm in 2D Space

As shown in the image above, gray color points are predicted as red and blue. A predictive algorithm creates a decision boundary or a contour (high-dimension space) and depending on which side of the boundary or contour a point fall, a prediction is made. Decision boundary shows which points will be predicted as red or blue. And this decision boundary changes as variable k changes.

By varying the value of k, we can get a different boundary and that boundary may be better at predicting the outcomes. Choosing a value of k which results in higher prediction accuracy, is a part of parameter tuning. There are many different parameters to the KNN model, just like k, and will use different values of these parameters to determine a set of values best suited for the data set.

Algorithm -

(i) Determine the value of K.

The first step is to determine the value of K. The determination of the K value varies greatly depending on the case. If using the Scikit-Learn Library the default value of K is 5.

(ii) Calculate the distance of new data with training data.

To calculate distances, 3 distance metrics that are often used are Euclidean Distance, Manhattan Distance, and Minkowski distance.
The Euclidean Distance metric gave the best accuracy in our dataset.

$$d_{euclidean} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

$$d_{manhattan} = \sum_{i=1}^{n} |x_i - y_i|$$

$$d_{minkowski} = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$$

In Scikit-Learn, the default distance used is Euclidean. It can be seen in the Minkowski distance formula that there is a Hyperparameter p, if set p = 1 then it will use the Manhattan distance and p = 2 to be Euclidean.

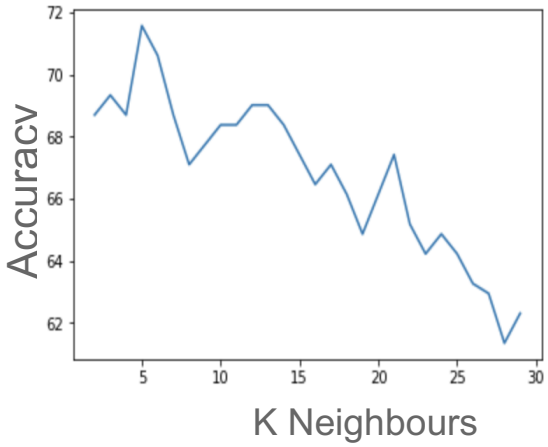
(iii) Find the closest K-neighbors from the new data. After calculating the distance, then look for

K-Neighbors that are closest to the new data. If using K = 3, look for 3 training data that is closest to the new data.

(iv) New Data Class Prediction.
To determine the class of new data, select the class of training data that closest to the new data and have the highest quantity.

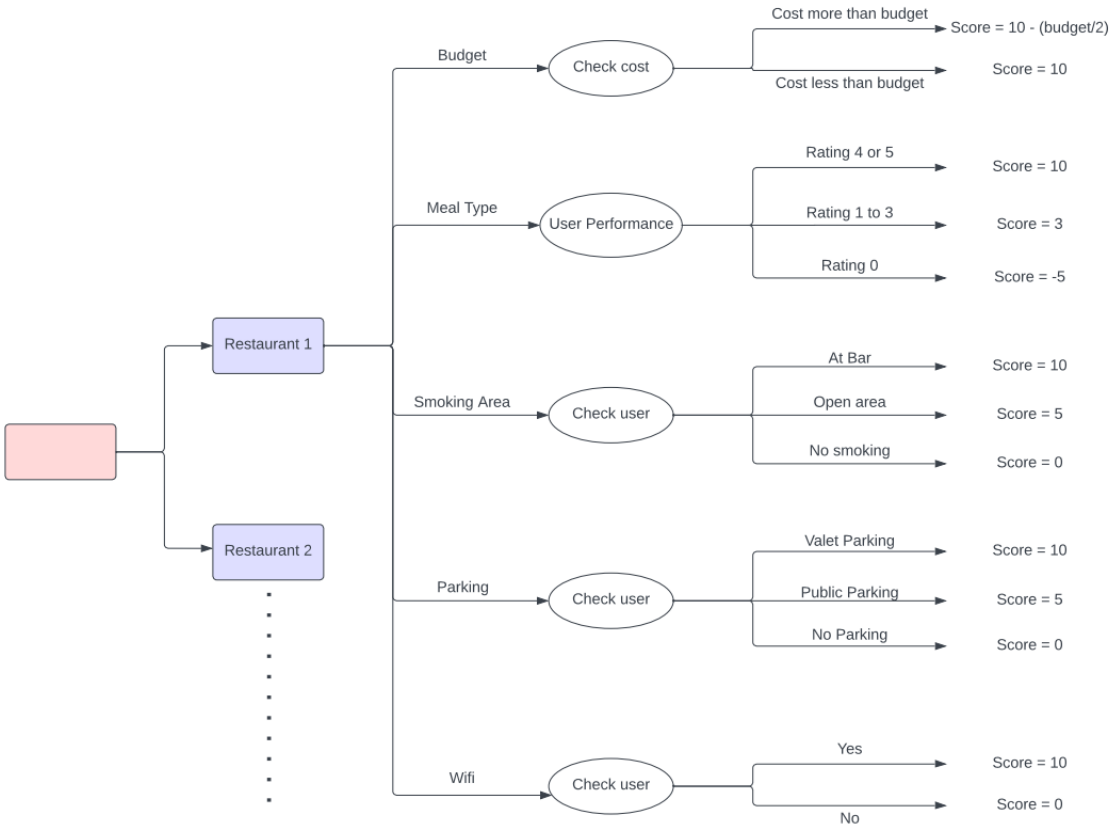
(v) Evaluation.

Calculate the accuracy of the model, if the accuracy is still low, then this process can be repeated from step 1.



Accuracy vs K line chart to select the most optimal value of K.

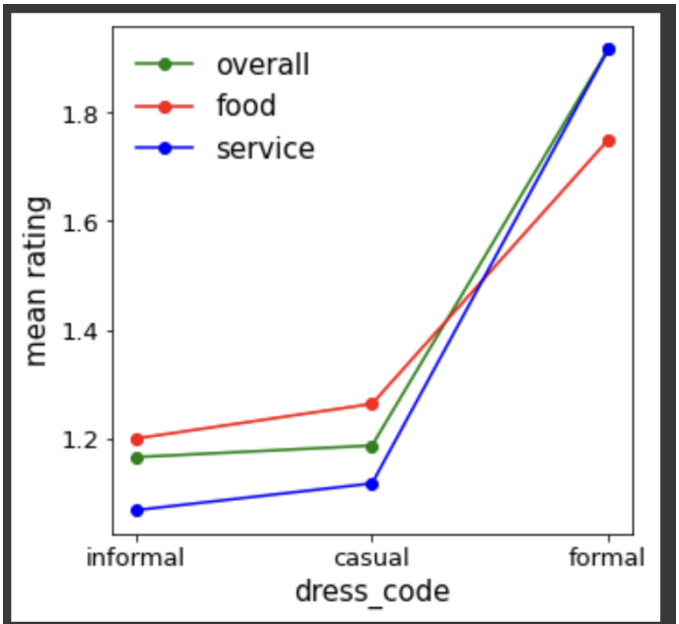
We obtained an accuracy of 71.5% at K=5, which was the maximum.



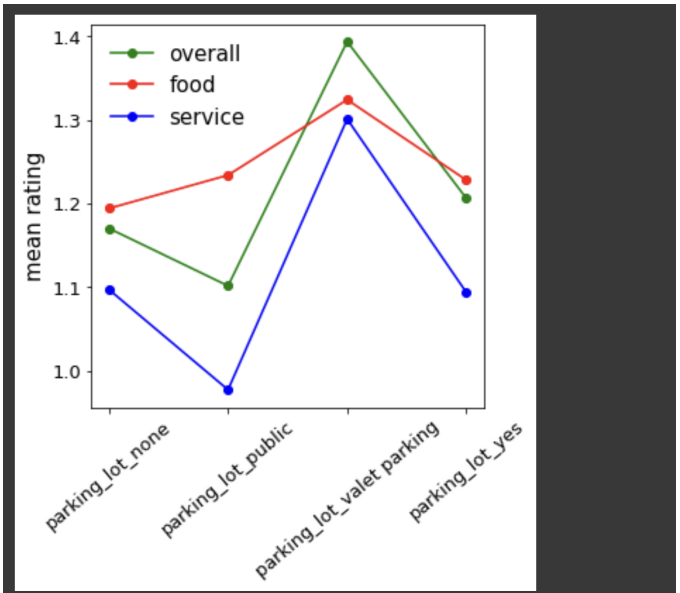
5.

Exploratory data analysis

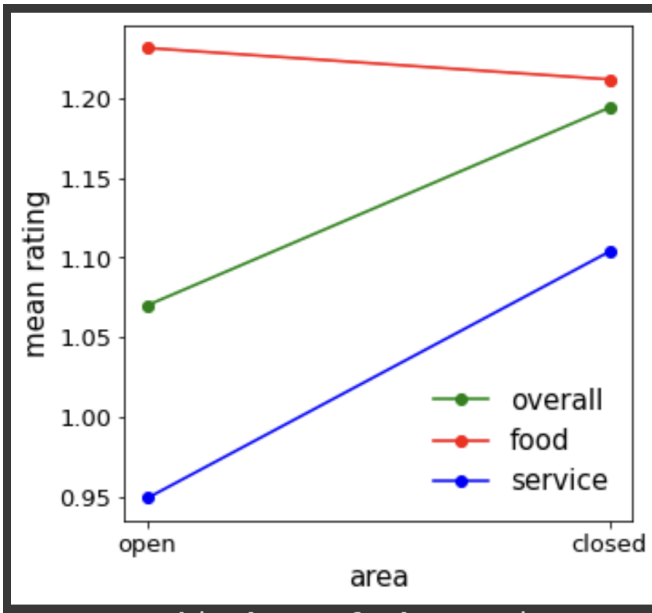
Initial analysis of the data led to multiple outcomes about the preferences of the customers - their likings and dislikings. These have been elaborated in the form of various graphs-



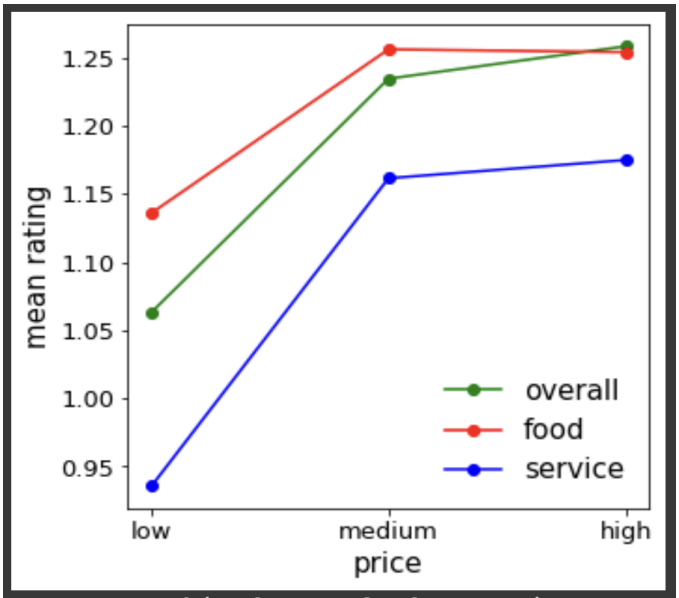
	combined	food	service
informal	1.166441	1.200562	1.068978
casual	1.187739	1.264256	1.118116
formal	1.916667	1.750000	1.916667



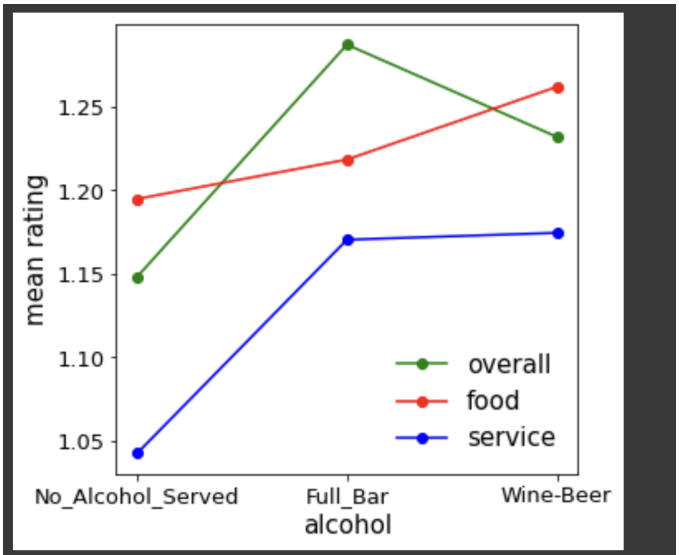
	combined	food	service
parking_lot_none	1.170202	1.194014	1.097035
parking_lot_public	1.101438	1.233839	0.977173
parking_lot_valet parking	1.393519	1.324074	1.300926
parking_lot_yes	1.206176	1.227920	1.093676



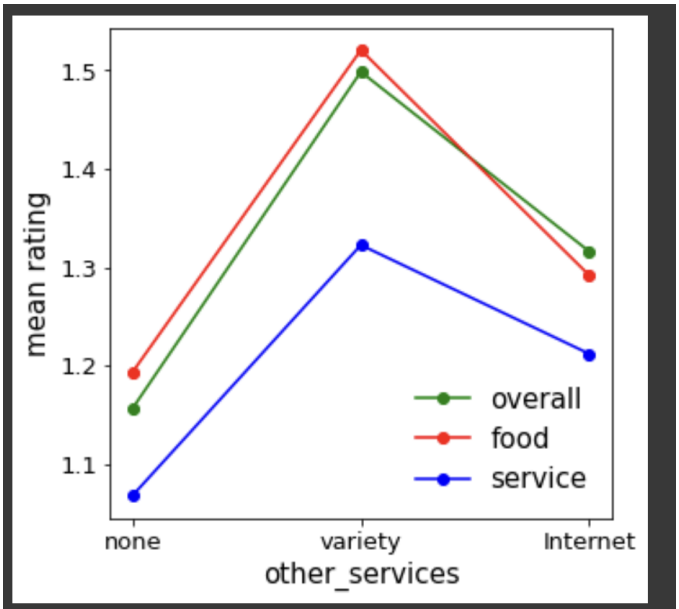
	combined	food	service
open	1.069992	1.231315	0.949101
closed	1.193921	1.211645	1.103630



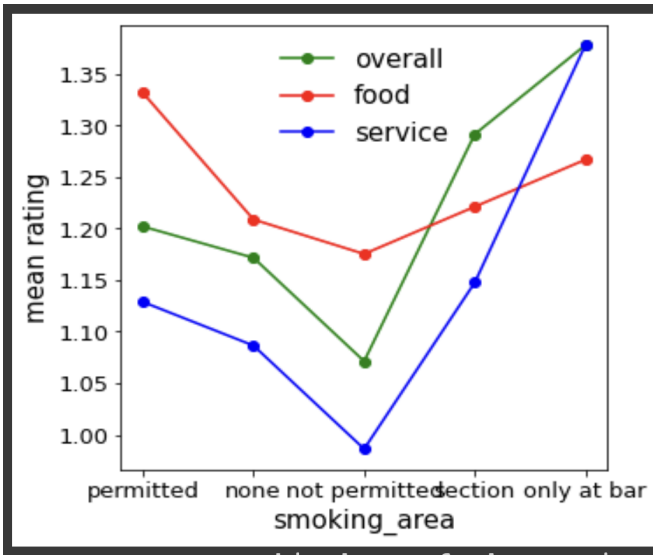
	combined	food	service
low	1.063059	1.135805	0.935632
medium	1.234342	1.255871	1.161361
high	1.258106	1.253816	1.174754



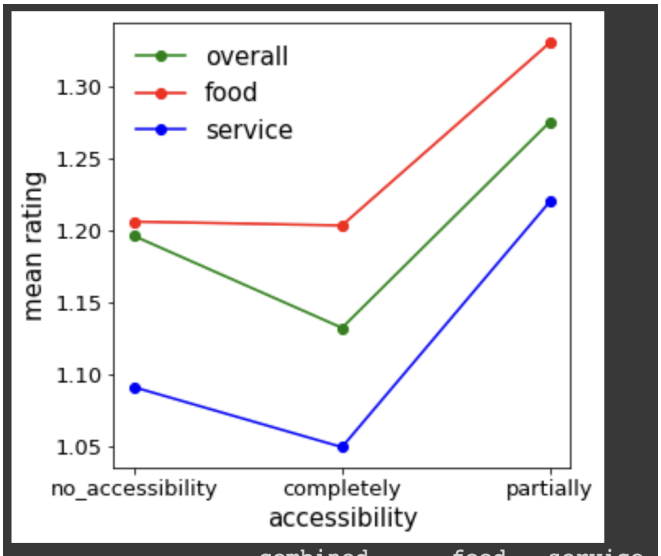
	combined	food	service
No_Alcohol_Served	1.148075	1.194730	1.042417
Full_Bar	1.287124	1.218315	1.170311
Wine-Beer	1.231887	1.261840	1.174437



	combined	food	service
none	1.156273	1.193252	1.067643
variety	1.498639	1.520748	1.322449
Internet	1.315972	1.291667	1.211806



	combined	food	service
permitted	1.201852	1.331481	1.128704
none	1.171497	1.208648	1.086371
not permitted	1.071238	1.175244	0.986184
section	1.291368	1.221073	1.147479
only at bar	1.377778	1.266667	1.377778



	combined	food	service
no_accessibility	1.196189	1.206242	1.091278
completely	1.132494	1.203597	1.049709
partially	1.275356	1.330294	1.219991

Link to the dataset:- https://archive.ics.uci.edu/ml/datasets/Restaurant+%26+consumer+data

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

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 showed 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.

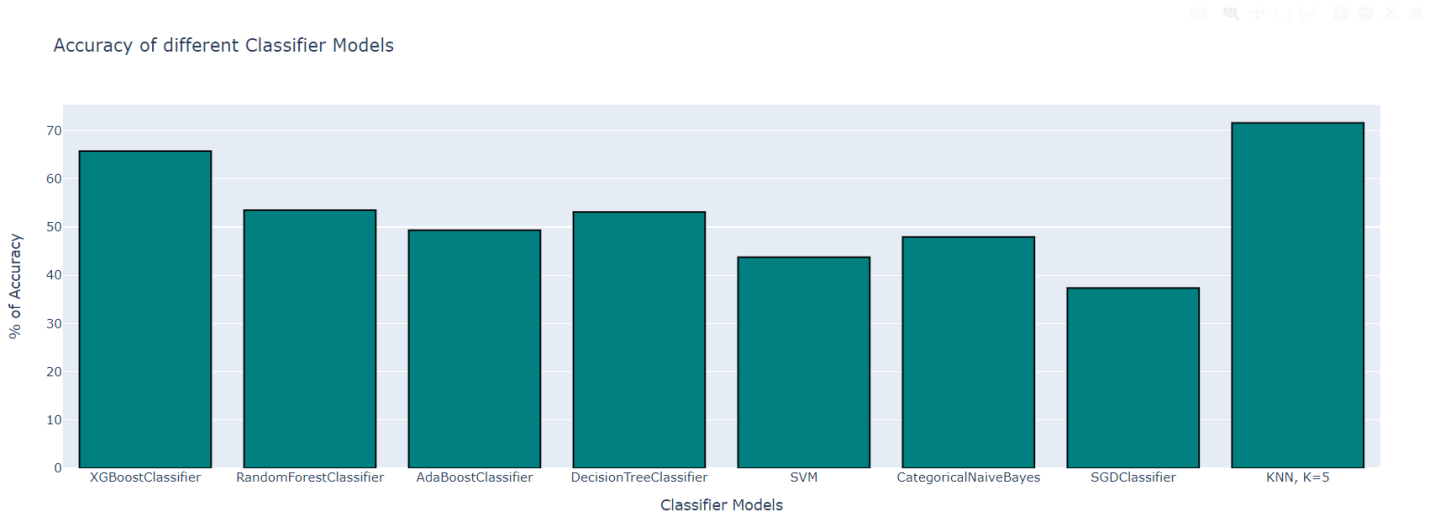
The accuracy scores obtained from various Classifier models were compared and analyzed and are shown in the chart below.

EXPERIMENTATION AND RESULTS:

We tried multiple models on our data. We picked all the models which can be used as a classifier on categorical data.

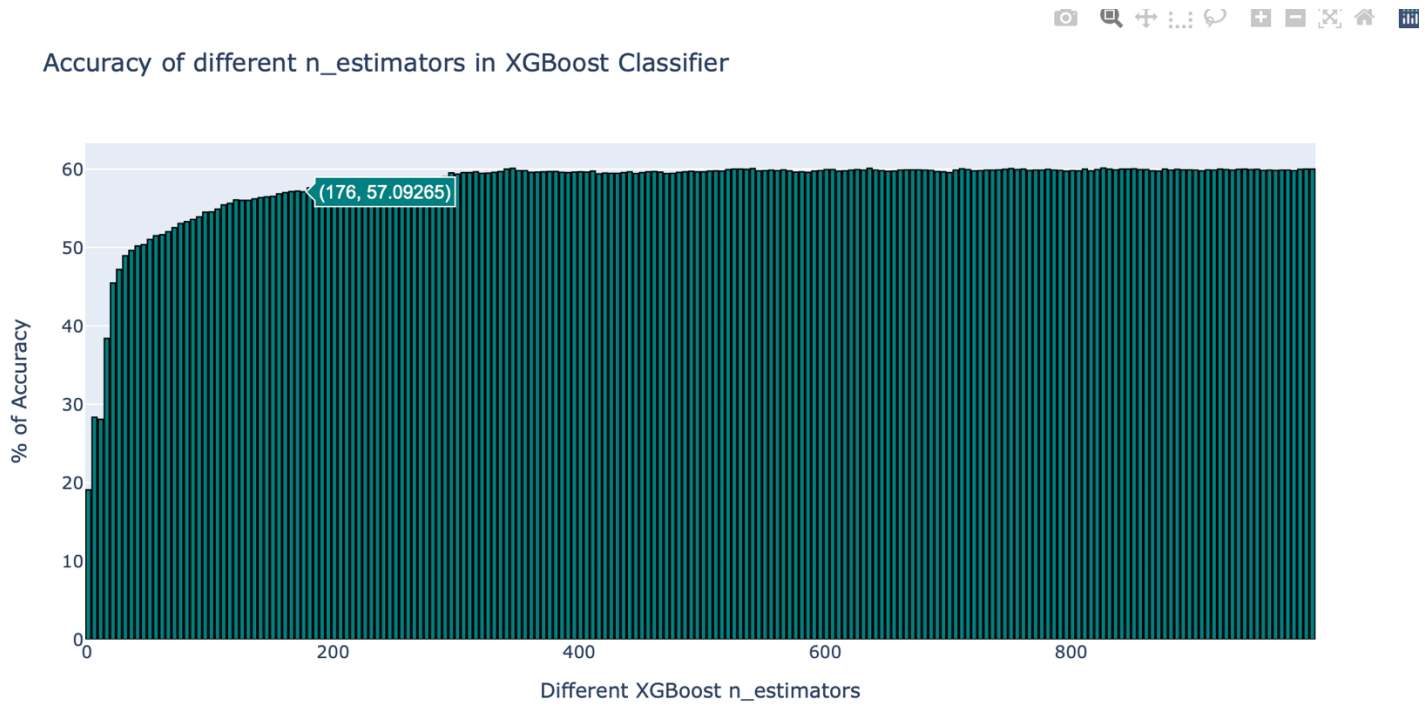
1. XGBoost Classifier
2. Random Forest Classifier
3. AdaBoost Classifier
4. Decision Tree Classifier
5. SVM Classifier
6. Categorical Naïve Bayes
7. SGD Classifier
8. K Nearest Neighbors

Accuracy Results:

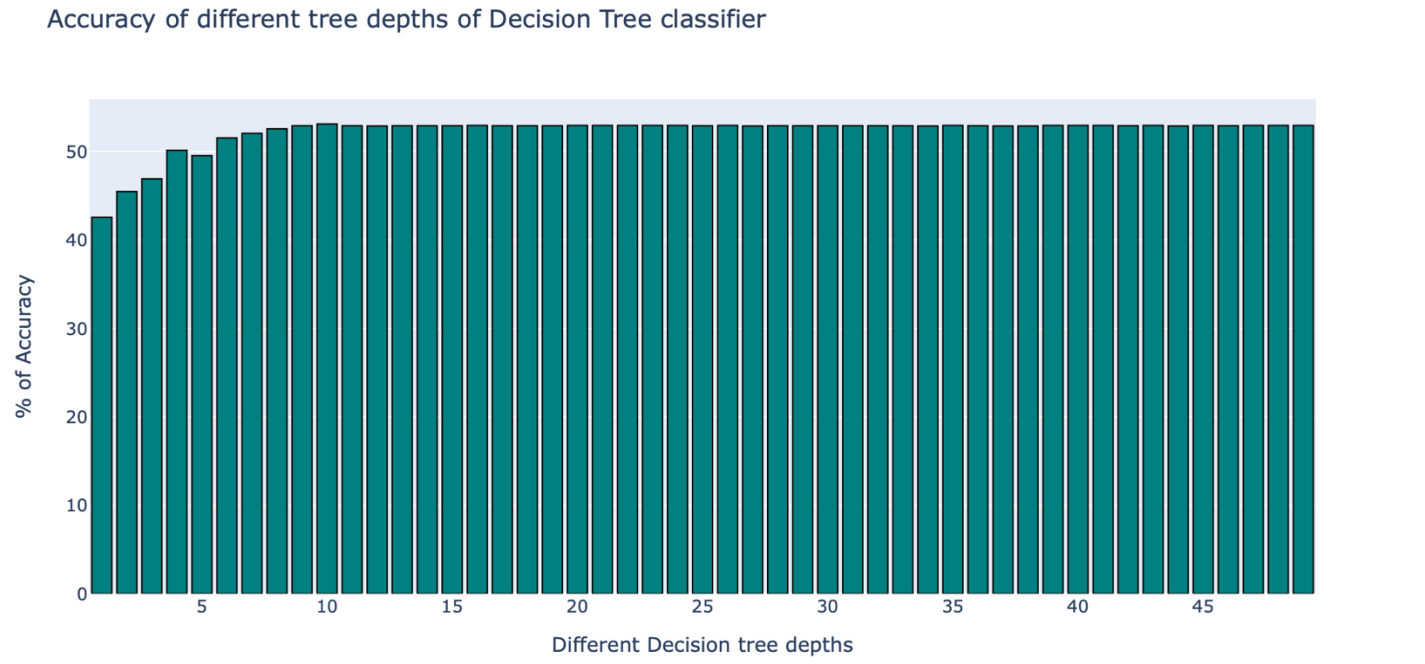


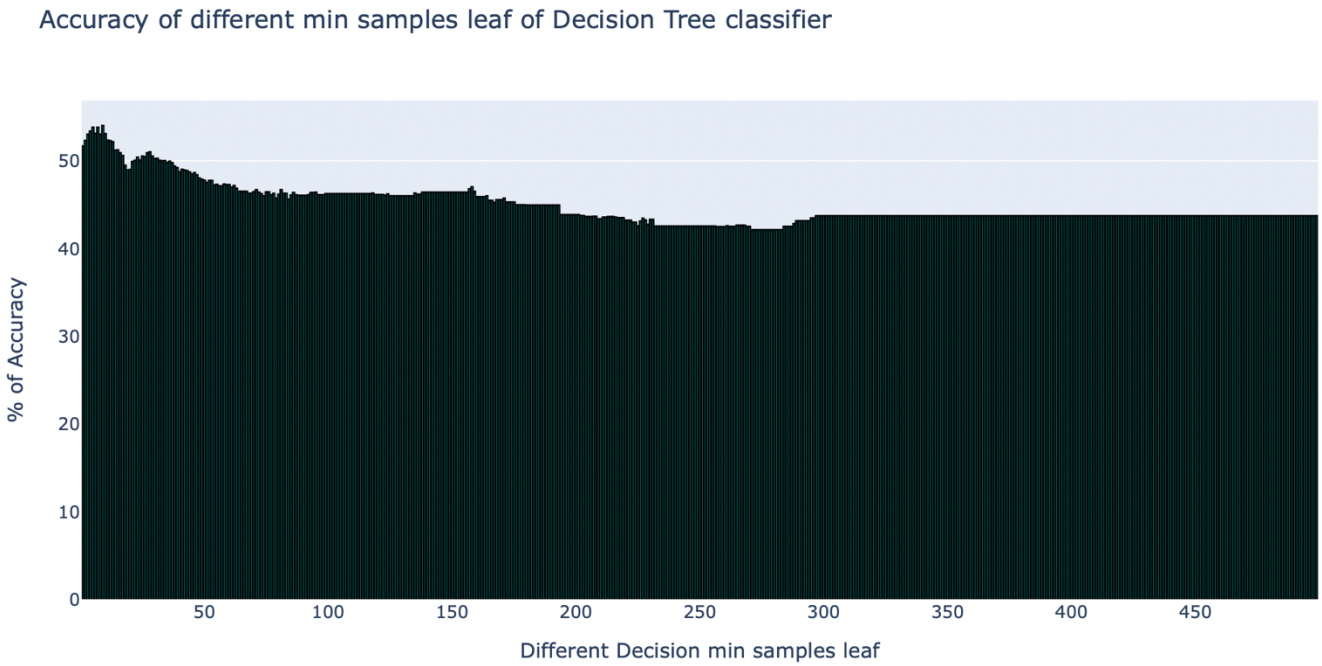
Model wise analysis:

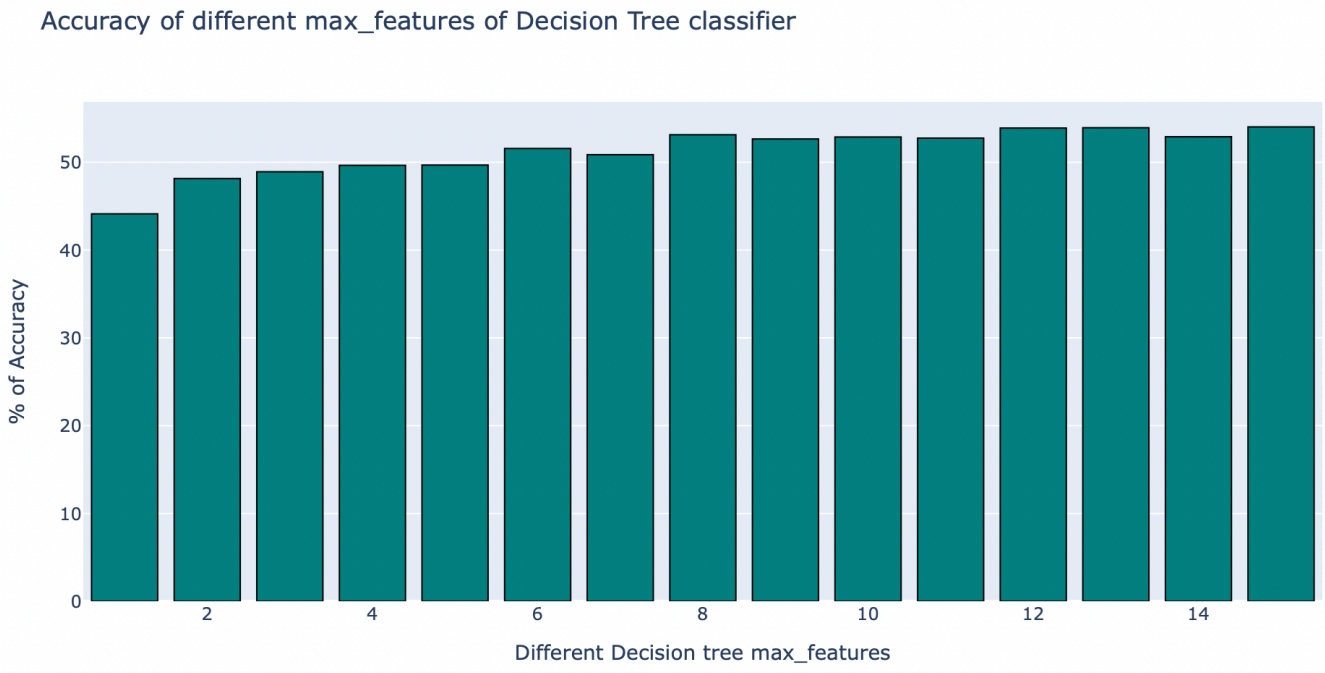
<u>XGBoost Classifier:</u>



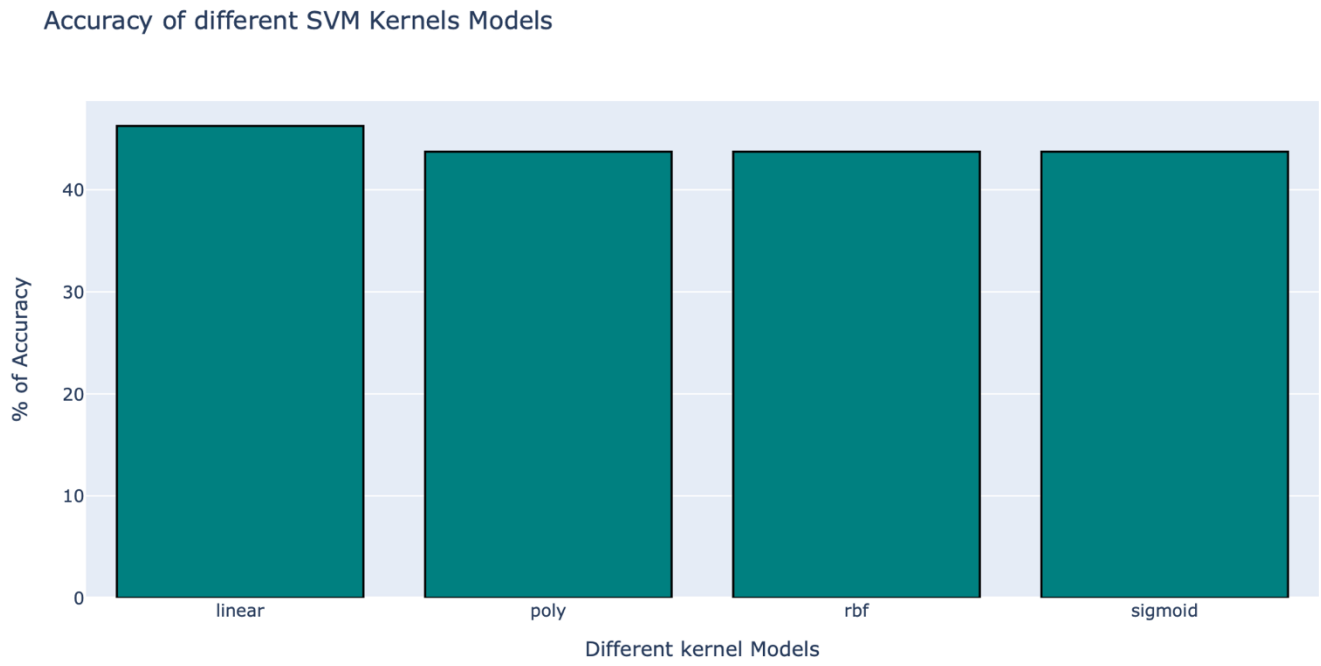
Decision Tree Classifier:







Support Vector Machine:



Conclusion:

To conclude, various measures have been used to accurately recommend a suitable restaurant to any user - XGBoost gave us an accuracy of 56.6% in predicting the correct restaurants, KNN which gave an accuracy of 71.5%. Aside these, Random Forest Classifier, AdaBoost Classifier, Decision Tree, SVM, Categorical Naive Bayes and SGDClassifier have also been put to use. The Latitude and Longitude functions have been put to use by finally sorting the recommended restaurants according to the distance from the user's location. The dataset has been put to precise use by keeping in view all the effective parameters that can increase the rating of a restaurant.

Top restaurants obtained using User Profile:-

Rank	Name	City
1	Tortas Locas Hipocampo	San Luis Potosi
2	Restaurant la Chalita	San Luis Potosi
3	puesto de tacos	s.l.p.
4	Restaurante Marisco Sam	San Luis Potosi
5	vips	?
6	Carreton de Flautas y Migadas	Ciudad Victoria
7	tacos abi	victoria
8	Taqueria EL amigo	Cd Victoria
9	palomo tec	victoria
10	Gorditas Dona Tota	?
11	Little Cesarz	Ciudad Victoria
12	puesto de gorditas	victoria
13	carnitas_mata	victoria
14	little pizza Emilio Portes Gil	victoria
15	carnitas mata calle Emilio Portes Gil	victoria
16	Cafeteria cenidet	Cuernavaca
17	McDonalds Centro	Cuernavaca
18	Hamburguesas La perica	victoria
19	Pollo_Frito_Buenos_Aires	victoria
20	TACOS CORRECAMINOS	victoria

Restaurants obtained by distance sorting from User -

Rank	Name	City	Distance
1	Restaurante Marisco Sam	San Luis Potosi	137.32 KM
2	Restaurant la Chalita	San Luis Potosi	137.45 KM
3	Tortas Locas Hipocampo	San Luis Potosi	138.00 KM
4	puesto de tacos	s.l.p.	138.38 KM
5	Taqueria EL amigo	Cd Victoria	188.11 KM
6	vips	?	188.36 KM
7	puesto de gorditas	victoria	188.49 KM
8	tacos abi	victoria	188.49 KM
9	Carreton de Flautas y Migadas	Ciudad Victoria	188.52 KM
10	Gorditas Dona Tota	?	188.66 KM
11	carnitas_mata	victoria	188.69 KM
12	carnitas mata calle Emilio Portes Gil	victoria	188.73 KM
13	palomo tec	victoria	188.75 KM
14	Pollo_Frito_Buenos_Aires	victoria	188.78 KM
15	Hamburguesas La perica	victoria	188.82 KM
16	little pizza Emilio Portes Gil	victoria	188.86 KM
17	Little Cesarz	Ciudad Victoria	188.94 KM
18	TACOS CORRECAMINOS	victoria	191.41 KM
19	McDonalds Centro	Cuernavaca	528.70 KM
20	Cafeteria cenidet	Cuernavaca	534.06 KM

References:

[1] G. Adomavicius, R. Sankaranarayanan, S. Sen, and
A. Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. ACM Transactions on Information Systems,
23:103–145, January 2005.

[2] L. Baltrunas and X. Amatriain. Towards time-dependent recommendation based on implicit feedback. In RecSys'09: Workshop on context-aware
systems (CARS-2009), 2009.

[3]https://stackoverflow.com/questions/38694457/how-can-i-tweak-xgboost-to-assign-more-weight-to-a-variable

[4] A. Bellog´ın, I. Cantador, P. Castells, and A. Ortigosa. Discovering relevant preferences in a personalised recommender system using machine learning techniques. In Proceedings of the 8th European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, Preference Learning Workshop, pages 82–96, 2008.

[5]https://machinelearningmastery.com/develop-first-xgboost-model-python-scikit-learn/

[6]https://www.analyticsvidhya.com/blog/2020/03/one-hot-encoding-vs-label-encoding-using-scikit-learn/

[7]https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/

8] S. Lombardi, S. Anand, and M. Gorgoglione. Context and customer behavior in the recommendation. In RecSys'09: Workshop on context-aware systems
(CARS-2009), 2009.
[9] T. Mahmood and F. Ricci. Learning and adaptivity in interactive recommender systems. In Proceedings of the ninth international conference on Electronic commerce, pages 75–84, New York, USA, 2007. ACM.

[10] B. Mobasher. Contextual user modeling for a recommendation. In RecSys'10: Workshop on Context Aware Recommender Systems (CARS-2010),
Barcelona, Spain, 2010.

[11]https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions/

[12]https://analyticsindiamag.com/7-types-classification-algorithms/

[13] D. Vallet, M. Fern´andez, P. Castells, P. Mylonas, and Y. Avrithis. A contextual personalization approach based on ontological knowledge. 17th European Conference on Artificial Intelligence - Contexts and Ontologies: Theory, Practice and Applications Workshop, Riva del Garda, Italy, 28 August, 2006.