Title of the Project

Hotel Recommendation using Past Entries

**Project Report**

**Submitted to:**

**Training and Placement Department**

**LNCT, Bhopal**

**Submitted by:**

**Sakshi Mahashabde**[0157CS181134]

****

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LAKSHMI NARAIN COLLEGE OF TECHNOLOGY AND SCIENCE, BHOPAL SESSION 2020-21

**LAKSHMI NARAIN COLLEGE OF TECHNOLOGY** **AND SCIENCE, BHOPAL**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



**DECLARATION**

I , Sakshi Mahashabde[Enrollment no: 0157CS181134] student of “Bachelor of Engineering in Computer Science & Engineering “ **session 2020-21, Lakshmi Narain College of Technology and Science , Bhopal [M.P]** hereby declare that the work presented in this project entitled **“Hotel Recommendation Using Past Entries”** is the outcome of my work , is bonafide and correct to the best of my knowledge and this work has been carried out taking care of Engineering Ethics. The work presented does not infringe any patented work and has not been submitted to any other University or anywhere else for the award of any degree or any professional diploma.

Sakshi Mahashabde[0157Cs181134]

Place: Bhopal Date: 12.12.2020

**LAKSHMI NARAIN COLLEGE OF TECHNOLOGY**

**AND SCIENCE, BHOPAL [M.P]**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



**ACKNOWLEDGEMENT**

I express my deep sense of gratitude to , department of Computer Science and Engineering, Lakshmi Narain College of Technology and Science Bhopal, whose valuable guidance and timely help encouraged me to complete this project.

A special thanks goes to Dr. Ashish Khare Sir , Prof and Head of Department, who helped us by providing timely suggestions in completing this project work.

I would also thank our institution and all the faculty members without whom this project work would have been a distinct reality.

**Sakshi Mahashabde** (0157CS181134)

**ABSTRACT**

One of the first things to do while planning a trip is to book a good place to stay. Booking a hotel online can be an overwhelming task with thousands of hotels to choose from, for every destination. Motivated by the importance of these situations, we decided to work on the task of recommending hotels to users. dataset has a variety of features that helped us achieve a deep understanding of the process that makes a user choose certain hotels over others. The aim of this hotel recommendation task is to predict and recommend five hotel clusters to a user that he/she is more likely to book given hundred distinct clusters.

**INDEX**

|  |  |
| --- | --- |
| **Title** | **Page no** |
| 1. **Introduction…………………………………………………** |  |
| * 1. Introduction   2. Dataset   3. Features |  |
| 1. **Literature Survey…………………………………………** |  |
| 1.1 Literature Survey |  |
| 1. **Problem Statement……………………………………..** |  |
| 3.1 Problem statement |  |
| 1. **Technologies Used……………………………………………** |  |
| 1. **Learning Models and Methodology………………….** |  |
| * 1. Numpy   2. Pandas   3. CSV   4. XgBoost algorithm   5. KNN algorithm   6. Random Forest algorithm |  |
| 1. **Data Flow diagrams………………………………………..** |  |
| * 1. How recommender Systems work?   2. How Hotel Recommendation System works? |  |
| **7.Implementation…………………………………………………** |  |
| 7.1 Screenshots |  |
| **8.Limitations and Conclusion…………………………………………………..** |  |
| 8.1 Limitations  8.2 Conclusion and future work |  |
| **9. References……………………………………………………..** |  |

**1. INTRODUCTION**

* 1. **INTRODUCTION**

Everyone likes their products to be personalised and behave the way they want them to. Given a user recommender systems aim to model and predict the preference of a product. The goal is to provide not just one recommendation but to rank the predictions and return the top five most likely hotel cluster’s for each user’s particular search query in the test set. We use multiple models and techniques to arrive at our best solution. This includes: a) KNN b) Rule based approach c) Random Forest Classifier and d) XgBoost Classifier

The one Hotel Recommender System similar to this is Expedia Online Hotel Booking System. The dataset was made available by Expedia as a Kaggle Challenge. Expedia wants to take the proverbial rabbit hole out of hotel search by providing personalized hotel recommendations to their users. Currently, Expedia uses search parameters to adjust their hotel recommendations, but there aren’t enough customer specific data to personalize them for each user.

**1.2 DATASET**

The aim of this challenge is to predict the next hotel which is not present in user’s history. For the dataset we have used :

Users.txt, activity.txt, hotel.txt

**Users.txt-It has user’s features, user id, gender, continent.**

**Activity.txt- It has user id of user and hotel selected by user.**

**Hotel.txt- It has hotel number and rating.**

**1.2 FEATURES**

The location about users and temporal information about users and hotel clusters has been provided in terms of continent. Hotel has some rating based upon which we will predict the next hotel for every user which will be based on continent to which user belongs to.

**2.** **LITERATURE SURVEY**

The authors work on a similar problem for personalized hotel recommendations for Expedia. Their dataset contained features like hotel price, location, ratings, user purchase history, OTA information for searches. Among these they found price and rating to be useful along with the location feature, which is in direct agreement with our own findings. For the ranking they study various approaches including Boosting Trees, Logistic Regression, SVM Rank, Random Forests along with Ensembling over these methods. They achieved their highest score, for an overall ranking of 5, using ensembling with z-score normalization. The authors approach the problem of hotel recommendation using text- based features extracted from reviews. This allows them to tackle the cold-start problem. For this they extract hotel reviews and user information from TripAdvisor and Venere. They extract various kinds of tags from each review based on service, room, food in order to conduct community detection. Apart from this they monitor user nationalities and intent to model preferences, opinion detection using wordnet. They then combine these to get a final review score according to which they rank the hotels.

The authors propose a combination of cluster-based model and collaborative filtering to provide a good recommendation model while also giving a way to tackle cold start problem. For collaborative filtering they compute the user-hotel rating matrix and then use pearson correlation to compute similarities between users. They then cluster similar users together. Whenever a query is made, ratings given by similar users are computed to generate recommendations.

**3.** **PROBLEM STATEMENT**

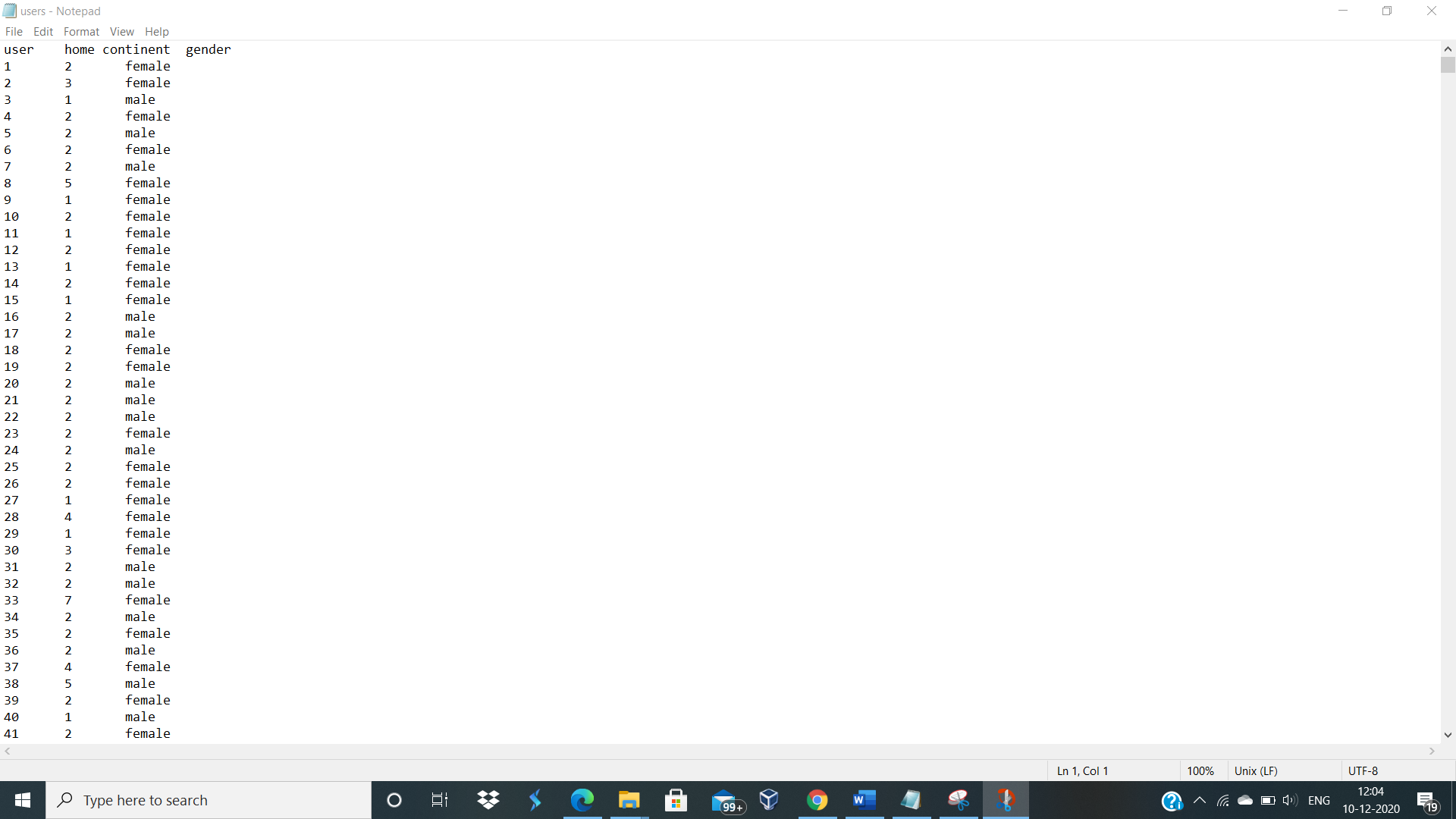
**Challenge: Given Input consists of dataset which has User’s features , User’s Activity . We have to predict User’s future hotel choices which are not present in the past choices. For this we have used the following dataset:**

**Users.txt which includes users features, user id, gender and continent.**

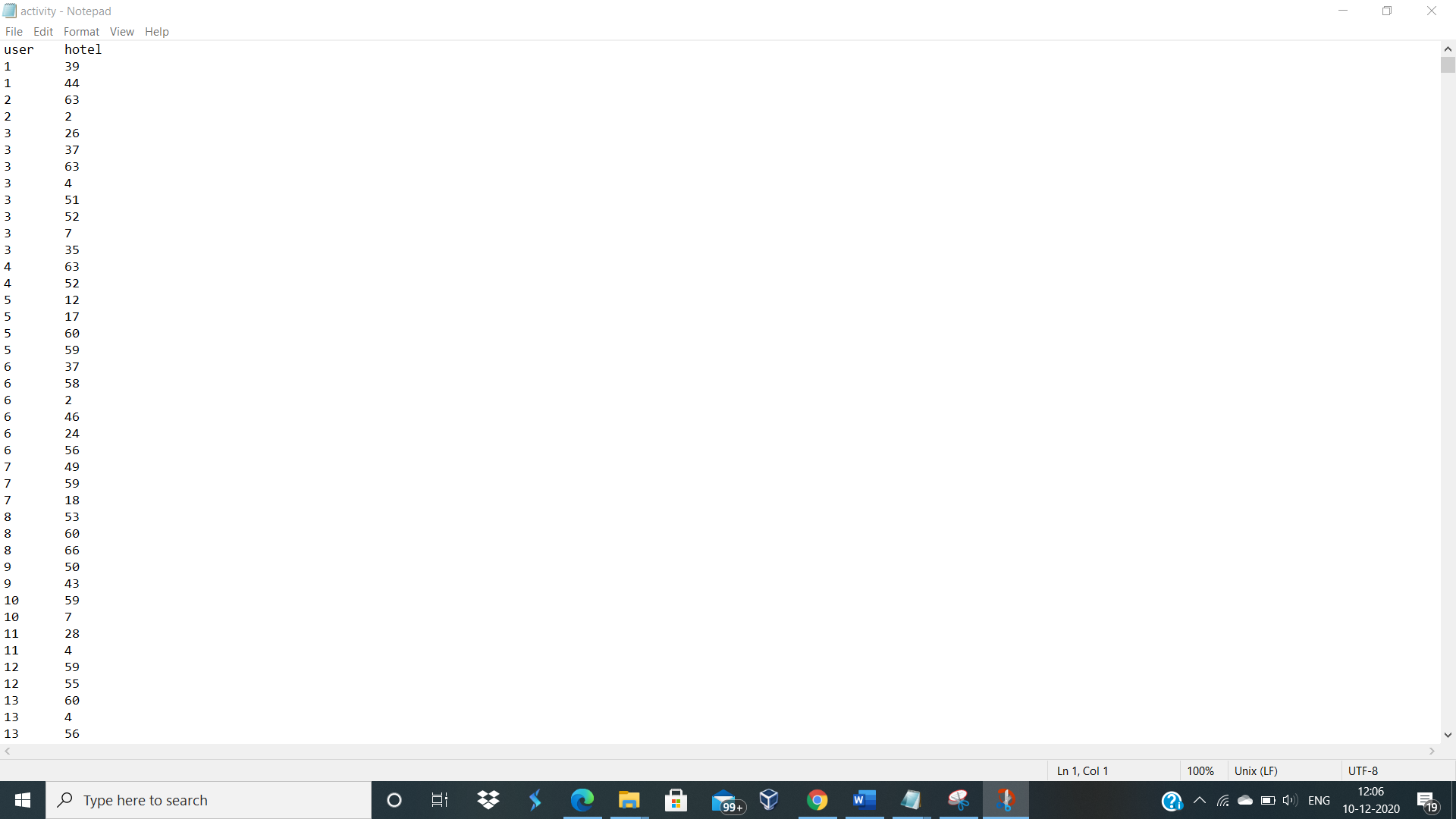
**Activity.txt which includes user id and hotel selected.**

**Hotel.txt which includes hotel number and rating.**

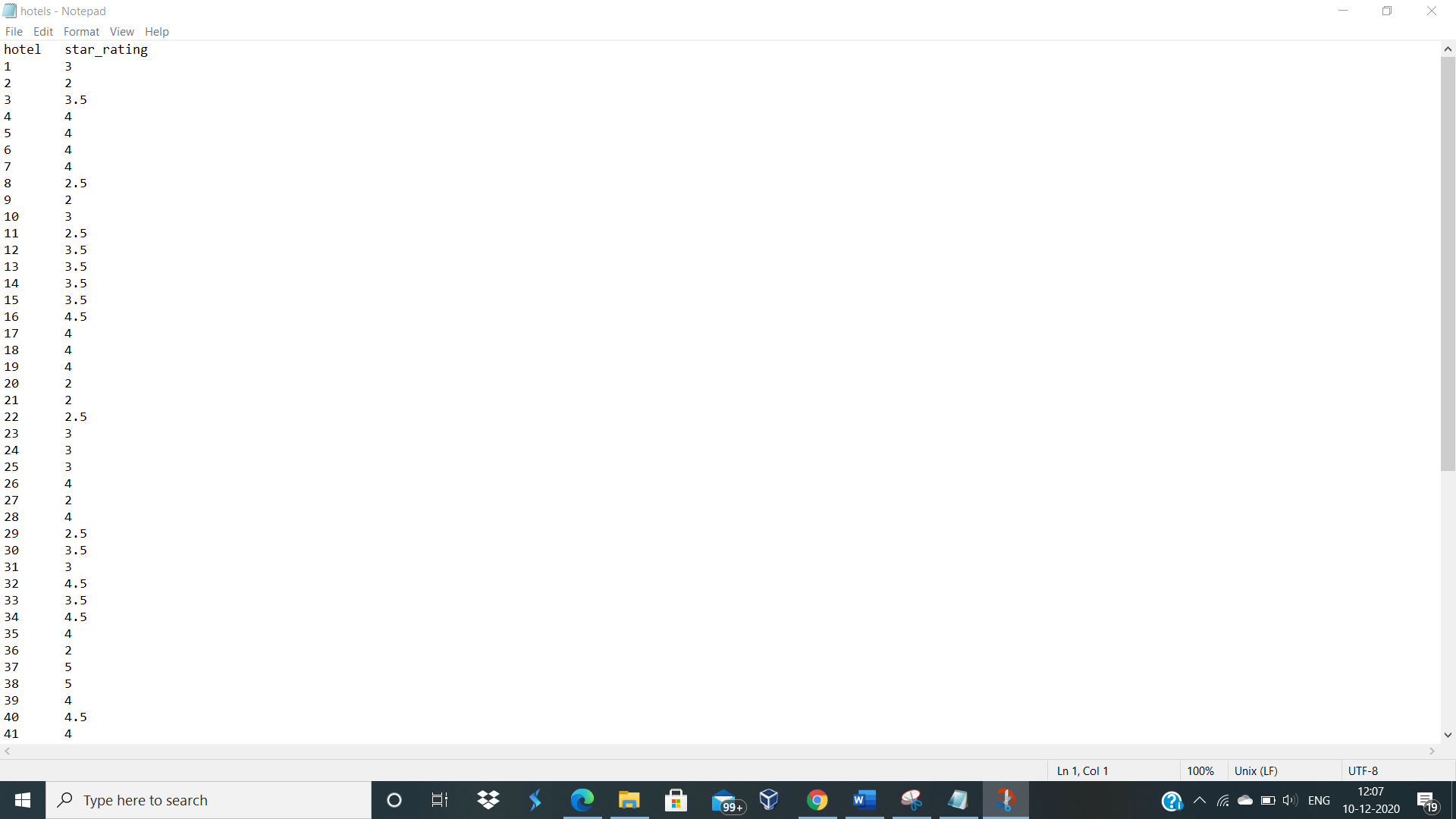
**Users.txt**

****

**Activity.txt**

****

**Hotel.txt**

****

**4. TECHNOLOGIES USED**

1. Python ( 3.9.1) – Python is an interpreted and general purpose programming language.
2. Numpy – NumPy is a library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform and matrices.
3. Pandas- Pandas is fast , powerful , flexible and easy to use open source data analysis and manipulation tool.
4. CSV- With this we can read and write csv files in python.
5. Machine Learning Algorithms – KNN, Rule based approach, Random Forest Classifier and XGBoost Algorithm. We have experimented with all algorithms and the algorithm which gave us the best results was selected to implementthe testing and training of the dataset.

**5.** **LEARNING MODELS AND METHODOLOGIES**

**5.1 NUMPY -**  NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open- source project and you can use it freely. NumPy stands for Numerical Python. In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important. NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently. This behaviour is called locality of reference in computer science. This is the main reason why NumPy is faster than lists. Also it is optimized to work with latest CPU architectures.

**5.2 PANDAS- Pandas**is a Python package that provides fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series data both easy and intuitive. It aims to be the fundamental high-level building block for doingpractical**, real world**data analysis in Python.Additionally, it has the broader goal of becoming**the most powerful and flexible open source data analysis / manipulation tool available in any language. It is already well on its way toward this goal.** Pandas is well suited for many different kinds of data:

* Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
* Ordered and unordered (not necessarily fixed-frequency) time series data.
* Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels
* Any other form of observational / statistical data sets. The data actually need not be labelled at all to be placed into a pandas data structure.

The two primary data structures of pandas, Series (1-dimensional) and Data Frame (2-dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. For R users, Data Frame provides everything that R’s data frame provides and much more. pandas is built on top of [NumPy](https://www.numpy.org/) and is intended to integrate well within a scientific computing environment with many other 3rd party libraries.

* 1. **CSV -** Python has a vast library of modules that are included with its distribution. The csv module gives the Python programmer the ability to parse CSV (Comma Separated Values) files. A CSV file is a human readable text file where each line has a number of fields, separated by commas or some other delimiter. You can think of each line as a row and each field as a column. The CSV format has no standard, but they are similar enough that the csv module will be able to read the vast majority of CSV files. You can also write CSV files using the csv module.
  2. **XG Boost –** XG Boost is short for Extreme Gradient Boosting, which refers to using ensembles of weak learners to make a prediction. XG Boost is based on this model. It is used for supervised learning problems, where we use the training data (with multiple features) xi to predict a target variable y. XG Boost and Random forests are similar in that they both use tree ensembles as the model. The primary difference between them is the way in which they are trained. The random forest treats all its trees uniformly, whereas the boosted trees use different levels of learners to classify progressively difficult samples. We try to optimize one level of the tree at a time, since picking the best out of all trees would be impractical. We try to split a leaf into two leaves, and d the score it gains is given by the following formula: Gain = 1 2 G2 L HL +λ + G2 R HR +λ − (GL+GR) 2 HL +HR +λ − γ where the first term refers to the score of the first leaf, the second term refers to the score of the second leaf, and the third term refers to the score before the split into two leaves, i.e. the score at the previous level. The fourth term refers to the regularization on the additional leaf.

**5.5 KNN Algorithm-**  This algorithm is used to solve the classification model problems. K-nearest neighbour or K-NN algorithm basically creates an imaginary boundary to classify the data. When new data points come in, the algorithm will try to predict that to the nearest of the boundary line.

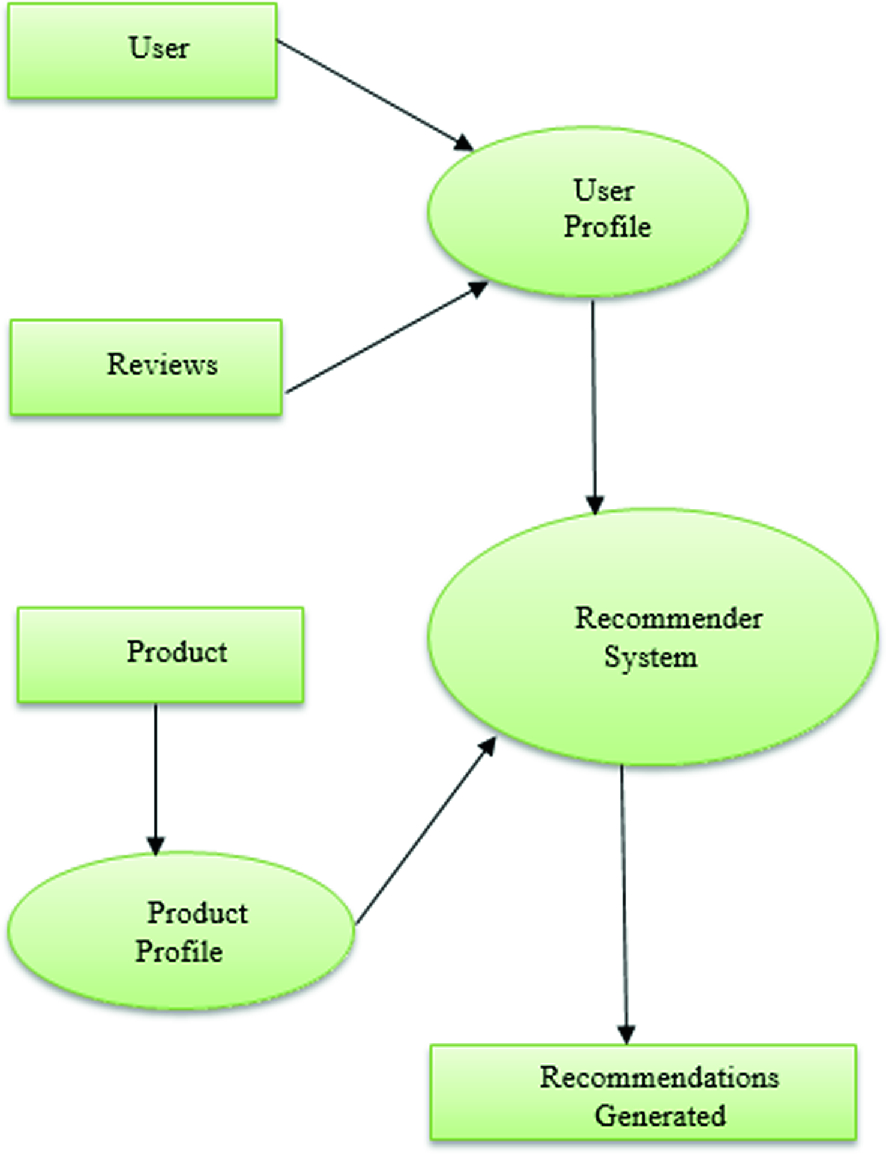
Therefore, larger k value means smother curves of separation resulting in less complex models. Whereas, smaller k value tends to overfit the data and resulting in complex models.

**Note:**It’s very important to have the right k-value when analysing the dataset to avoid overfitting and underfitting of the dataset.

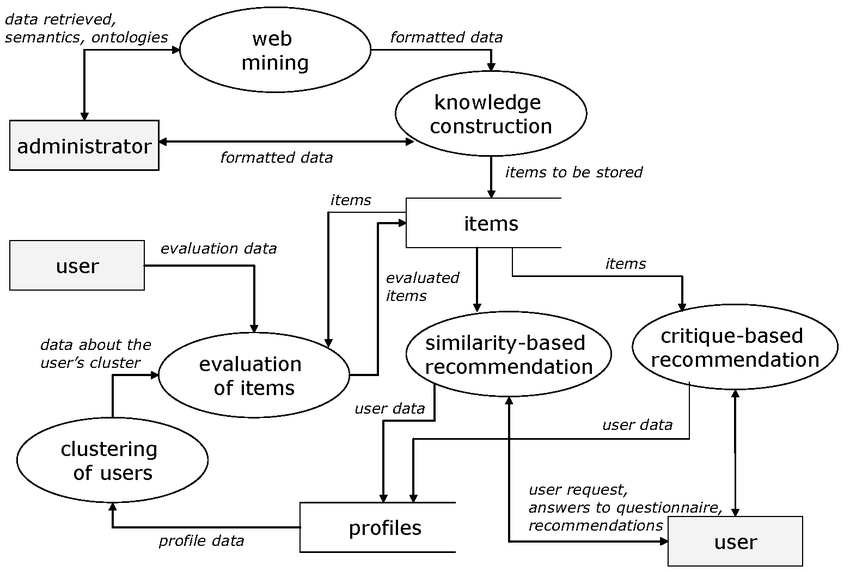
Using the k-nearest neighbour algorithm we fit the historical data (or train the model) and predict the future.

1. The k-nearest neighbor algorithm is imported from the scikit-learn package.
2. Create feature and target variables.
3. Split data into training and test data.
4. Generate a k-NN model using neighbors value.
5. Train or fit the data into the model.
6. Predict the future.
   1. **Random Forest algorithm-** A random forest classifier is a meta-estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting. In Python’s sklearn implementation3 , the sub-sample size is always the same as the original input sample size but the samples are drawn with replacement by default. As in boosting, many weak learner trees are grown and aggregated into a unified model. Random forest uses regression trees as building blocks. Using a greedy approach, the feature space is partitioned into a specified number of distinct nonoverlapping regions. The number of partitioned regions affects the bias-variance trade off. Random forests are a bagging method but in them, each grown tree only uses a random sample of p predictors out of all the n predictors. The motivation behind this is to de-correlate trees. When all predictors are used, all grown trees look like each other. Therefore, bagged trees are very correlated, which negatively affects the reduction in variance by averaging. Random forests solves this problem by considering only (np)/n portion of predictors at each tree split. For p = n, random forests turn into simple bagging. The parameters to be tuned for random forests are (1) the number of random predictors, (2) the maximum number of leaf nodes (which determines how deep a tree is grown) and (3) the number of trees to be grown and averaged. Grid Search functionality of Python’s sklearn was used for this.
7. **DATA FLOW DIAGRAMS**

Data flow diagram of recommender system works:



Data Flow diagram of how hotel recommendation system works:

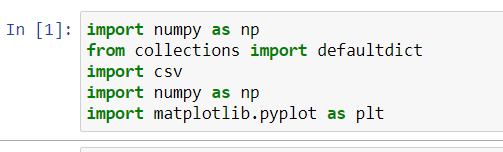


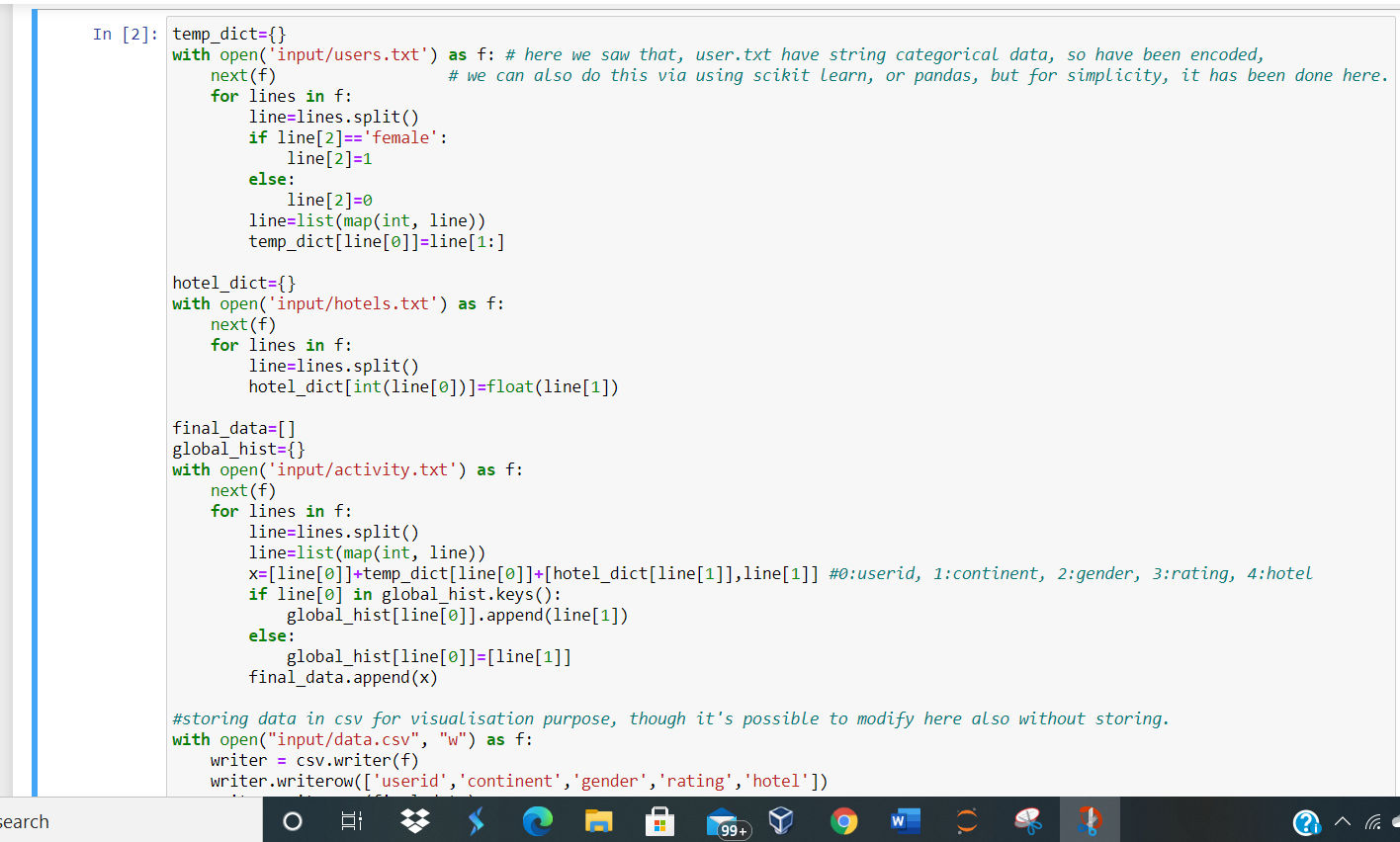
**7.IMPLEMENTATION**

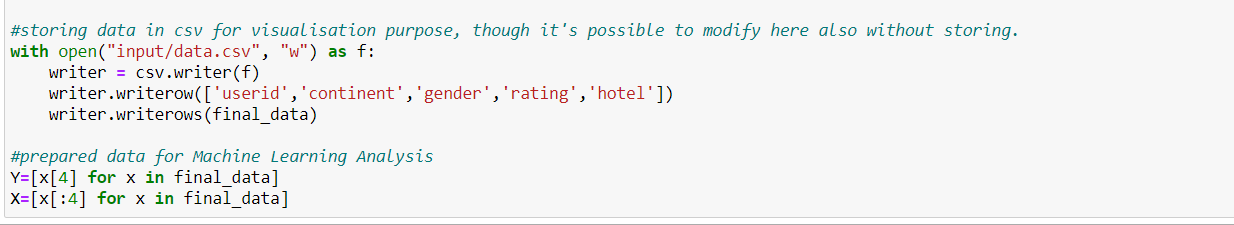
Important Points: Model finalized: XG Boost Classifier

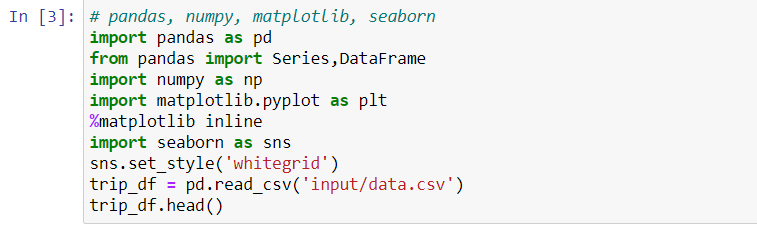
Experimented with Rule Based Approach, Random Forest, KNN (not good accuracy, optimal neighbour=1), Two Layer Neural Network (even with 100 epochs, it was giving 2% less than XG Boost, perhaps with more layers it would have worked, but then again the issue was, that we have less number of parameters, there could be a case of overfitting if we go for parametric machine learning approach).

Experimented with hybrid method which will return the hotel present in both most frequent + Machine Learning output. Didn't give good results.

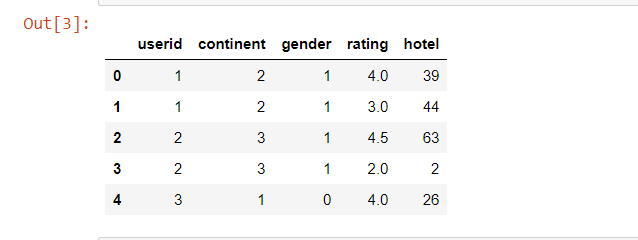




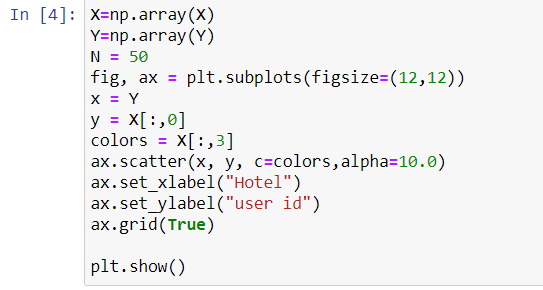




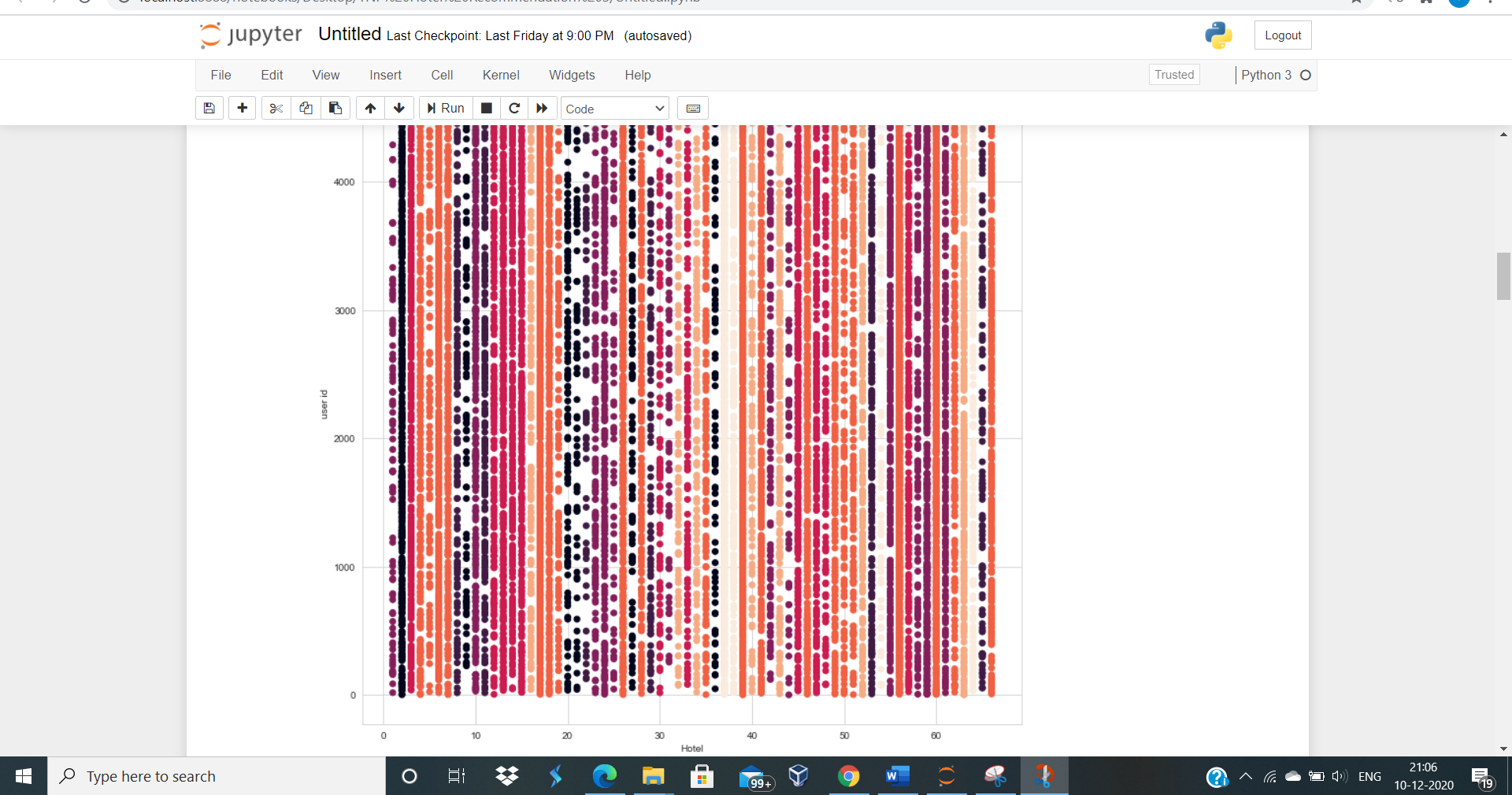
This code generates following output:

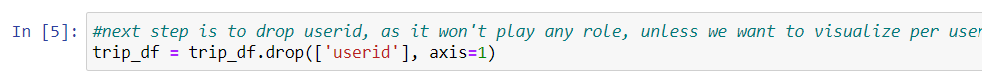


Let’s analyse user chosen hotel and their ratings. We can observe that most frequent hotels are likely to come to each user’s entry. So, popularly chosen hotel plays important role in user chosen hotel.

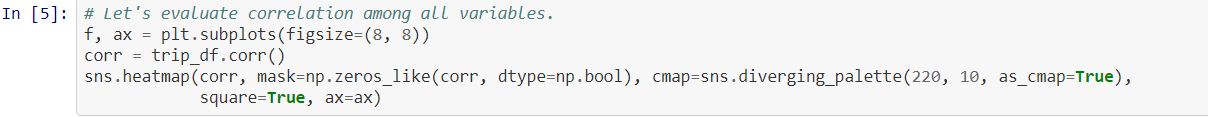


Output generated would be:

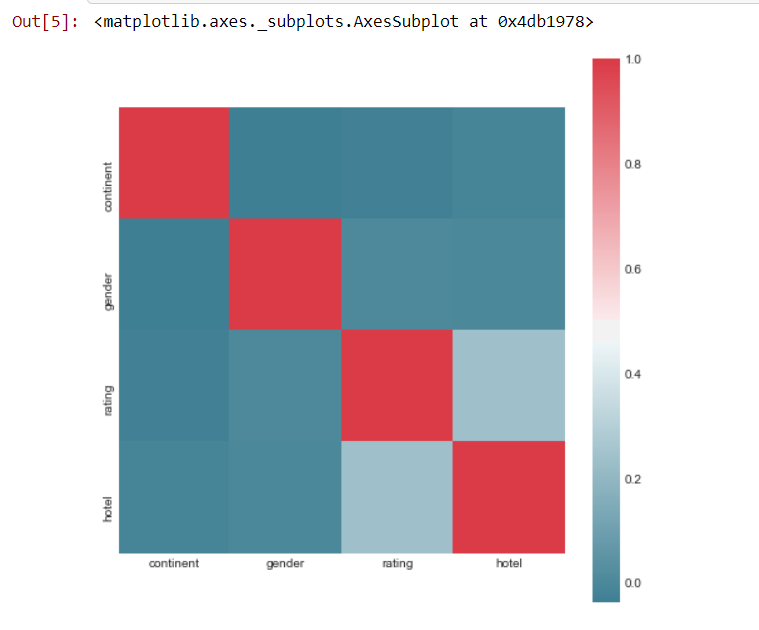




Next step is to see if there is strong correlation among variables.

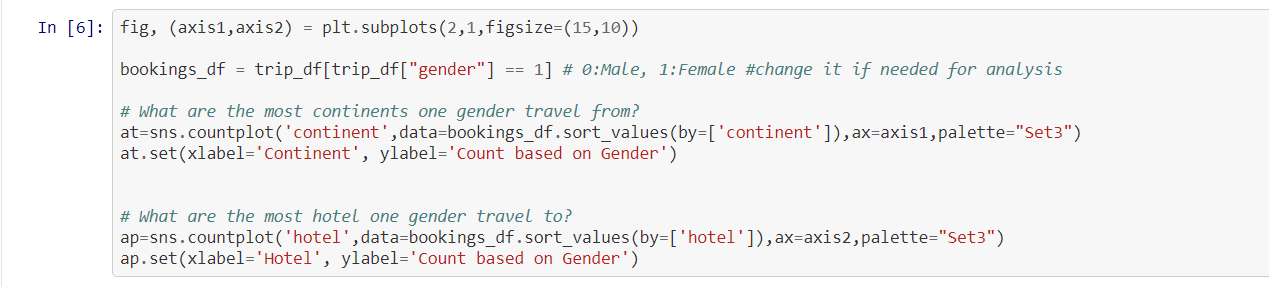


Output generated will be:

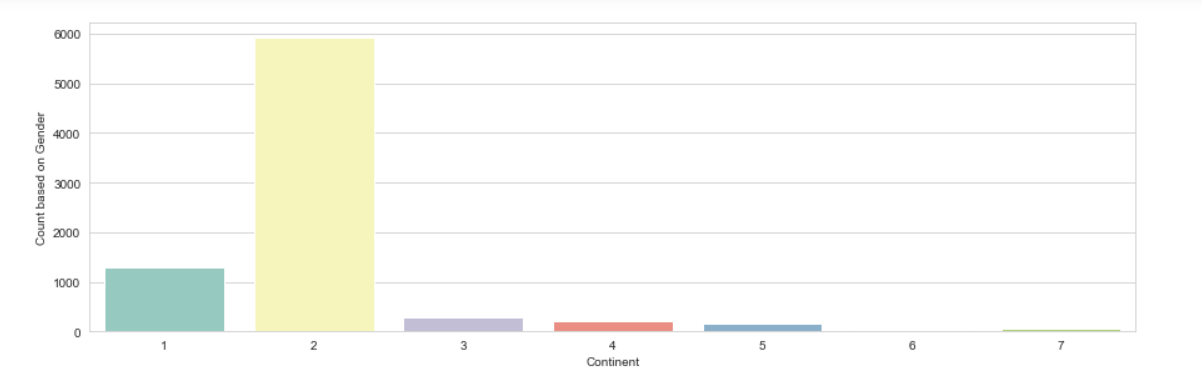


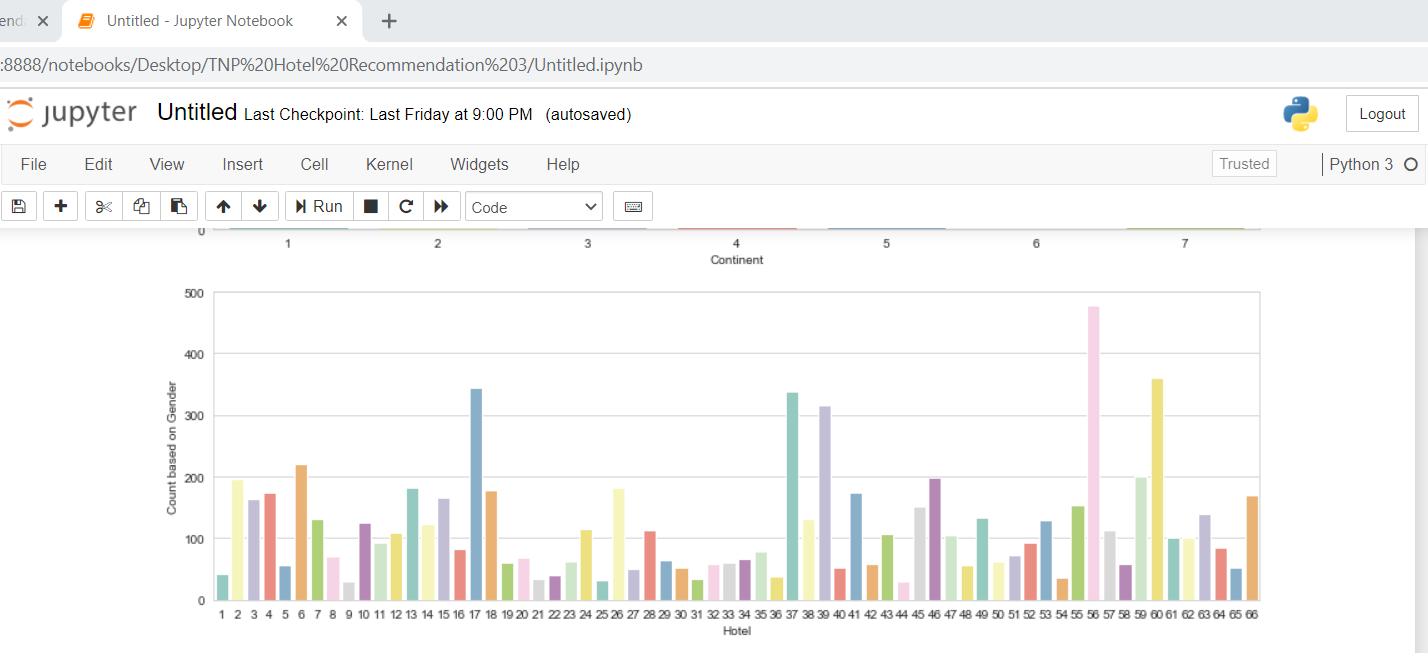
Let’s explore popular hotel and continent among genders. We got results as follows: Gender: Male, Continent: 2, Hotel :56 and Gender: Female, Continent: 2, Hotel :56.

So, we can say Gender has no role, in popular Hotel determination, but yes frequency varied.



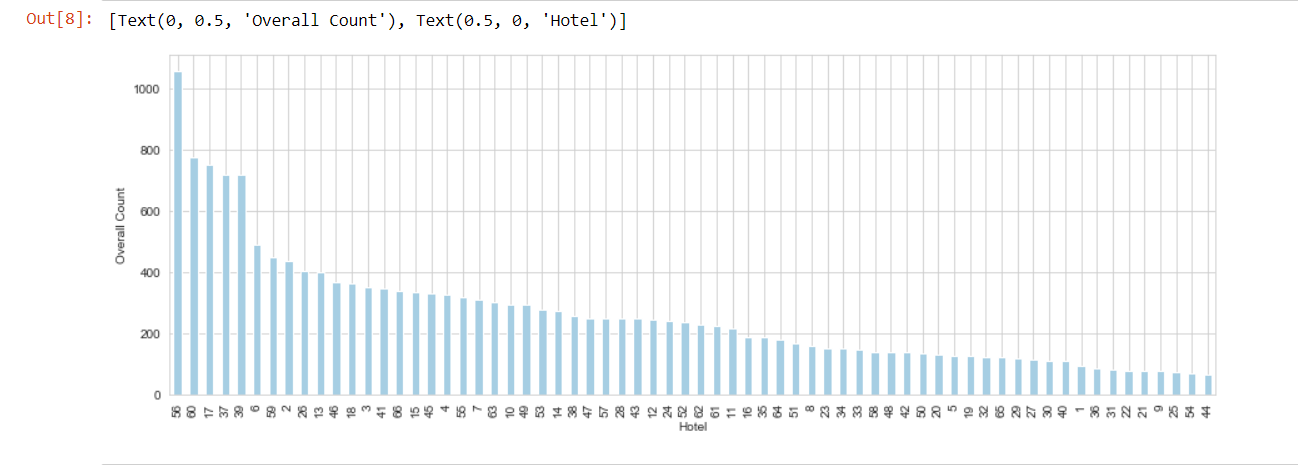
Output generated would be:

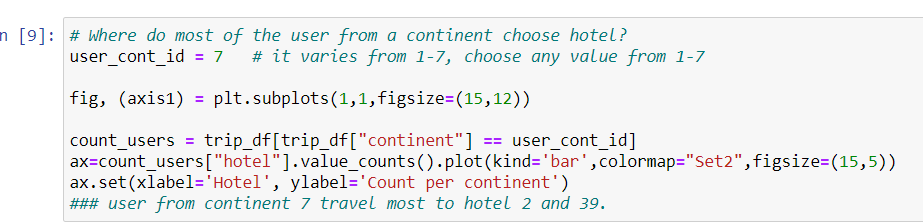


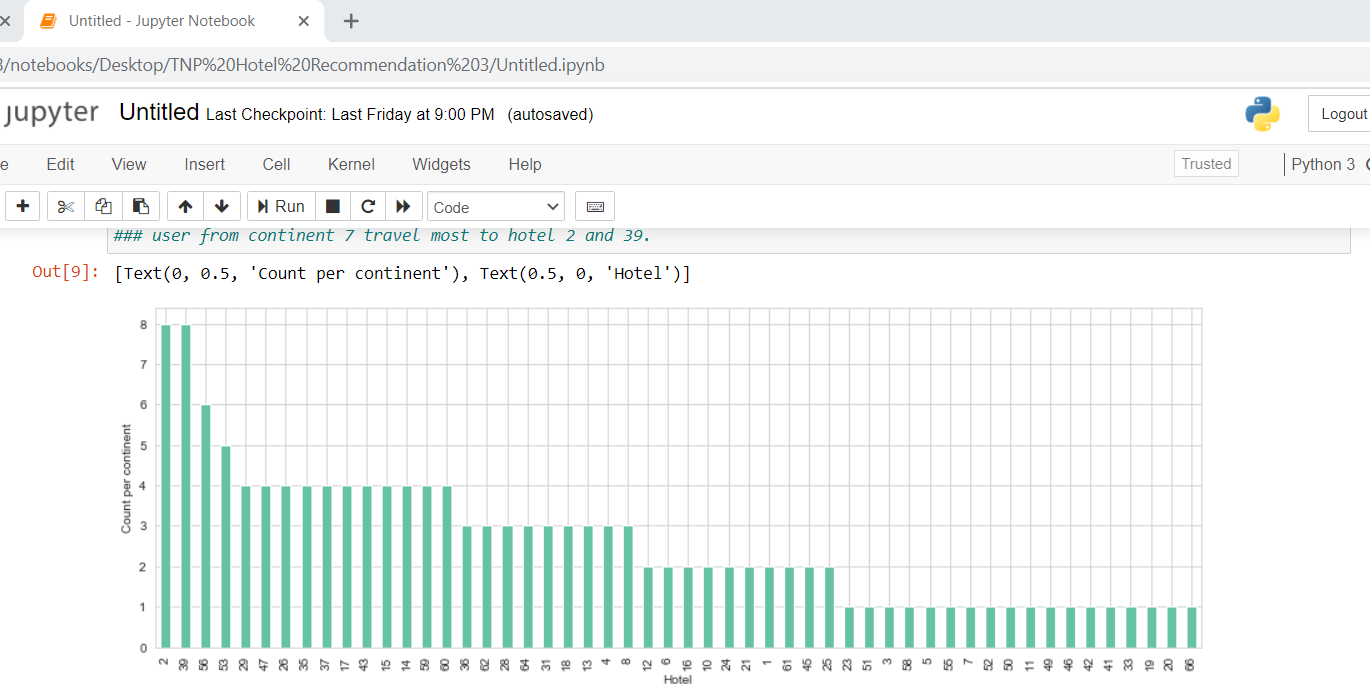


We can see that hotel 56 is the most popular one.





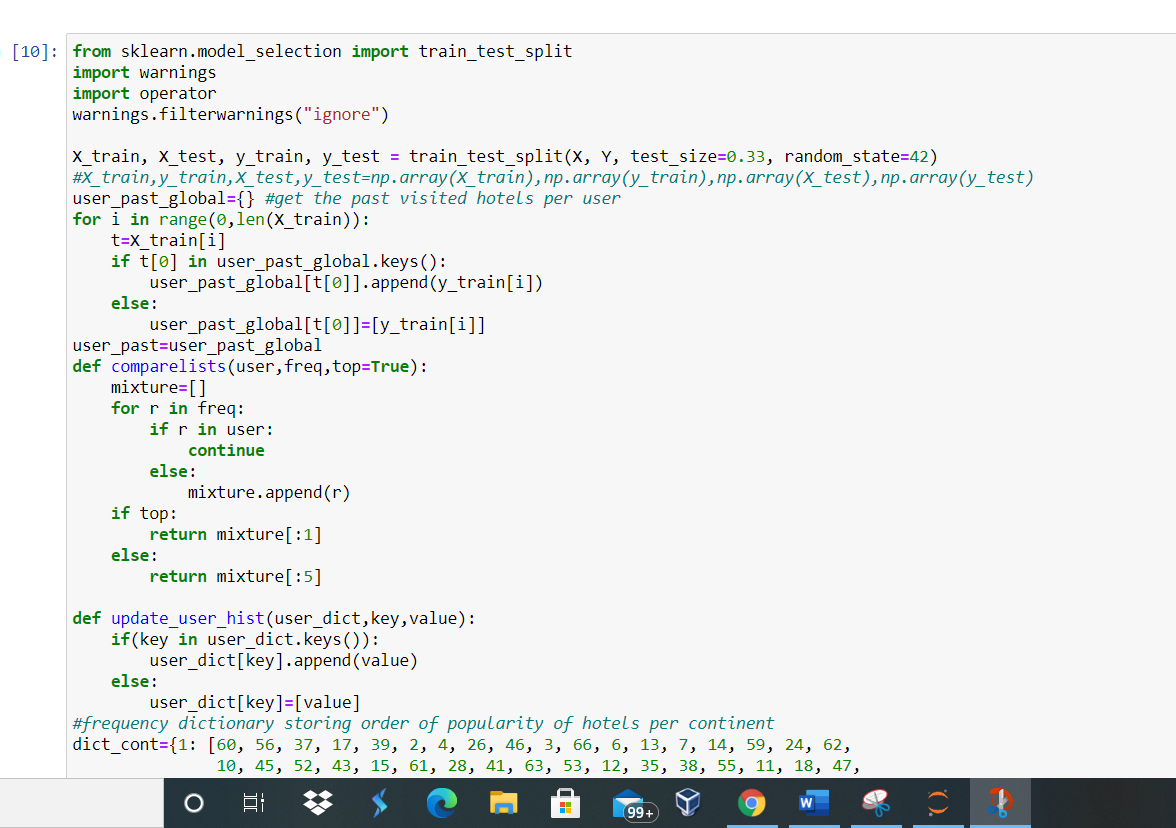


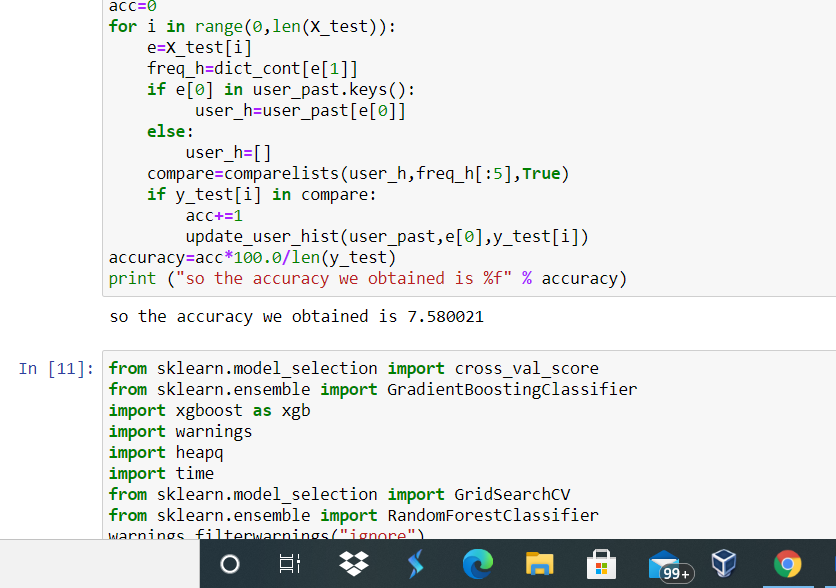


Now we will experiment this with different approaches. We will try two approaches :

* 1. Rule – Based Approach - As, we could see, gender didn't play any role in determining the famous hotels. So, One experiment to test a frequency per continent results has also been done. Where, we return the most frequent hotel, as per continent, which is not present in user's history.
  2. Machine Learning Algorithms- As Data is low in parameters, plus the data entries is also not enough, its better to go for Non-Parametric Equation as parametric one might overfit. So, We can think of KNN, Decision Trees(Random Forest, XGBOOST). Upon Experimenting with KNN, I realised that it was giving good results, if there was no constraint on repetition. But not so well for non-repetition.

Lets try rule based approach first.





We can observe that accuracy upon applying the rule based approaches is 7.580021%. This is obtained when we are taking the most common values. But the accuracy will increase when we will consider the top 5 values. **One more Observation is that, we are able to get good accuracy, if we are not putting condition of non-repetitiveness of user.**

### Now we will try machine learning approach . We have used XG Boost, and Random Forest, both are Emsembles, making the accuracy of the model more accurate. Also, I have used 2 methods to return the hotel which is not present in user's history. First, which only consider Machine Learning Result. Second Hybrid Method which consider Both Frequency of Hotel as per continent, and Machine Learning Result. Upon Experimenting, I wasn't able to get good results with hybrid method, and decided to go with normal Machine Learning Approach. May be we need to determine proper weightage of what needs to given to frequency and Machine Learning Result.

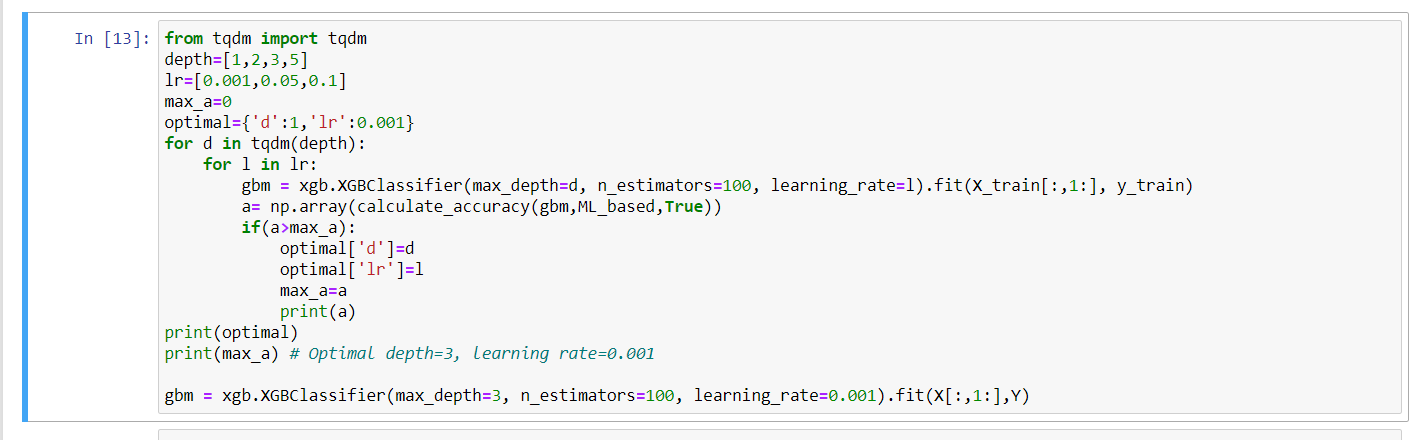
### 

### 

### 

### The accuracy using Random Forest method is 22.775237425255014%. So We can see that with Random Forest, with 100 Trees, and their obtained optimal paramteres via Grid Search CV. I have made 2 functions to get hotel predicition,

### first one is considering only results from Random Forest Algorithm. Second one considering the frequency of hotel too, its a mixture of hybrid.



#### In Above we have used XGBoost, and tested with same hotel predicting functions.

1. When We are using 1 prediction per user, and pure Machine Learning Prediction, we are getting around 23.25% Accuracy.
2. When We are using 5 prediction per user, and pure Machine Learning Prediction, we are getting around 62.46% Accuracy.
3. When We are using 5 prediction per user, and pure Hybrid Prediction Method, we are getting around 22.24% Accuracy.
4. When We are using 1 prediction per user, and pure Hybrid Prediction Method, we are getting around 11.22% Accuracy.

### So As we can see that Both Algorithms are providing somewhat same result, a little bit of randomness could be cause of random picking up test set. Secondly, we have observed that Best result is obtained via Machine Learning Method and when 5 cases are considered.

### This is the Case because we don't have enough features, if we would have enough features, then we could have obtained more efficient results, but as we can see, that the real output exists in top 5 predicted values.

### Finally I decided to choose XGBoost? Why XGBoost?

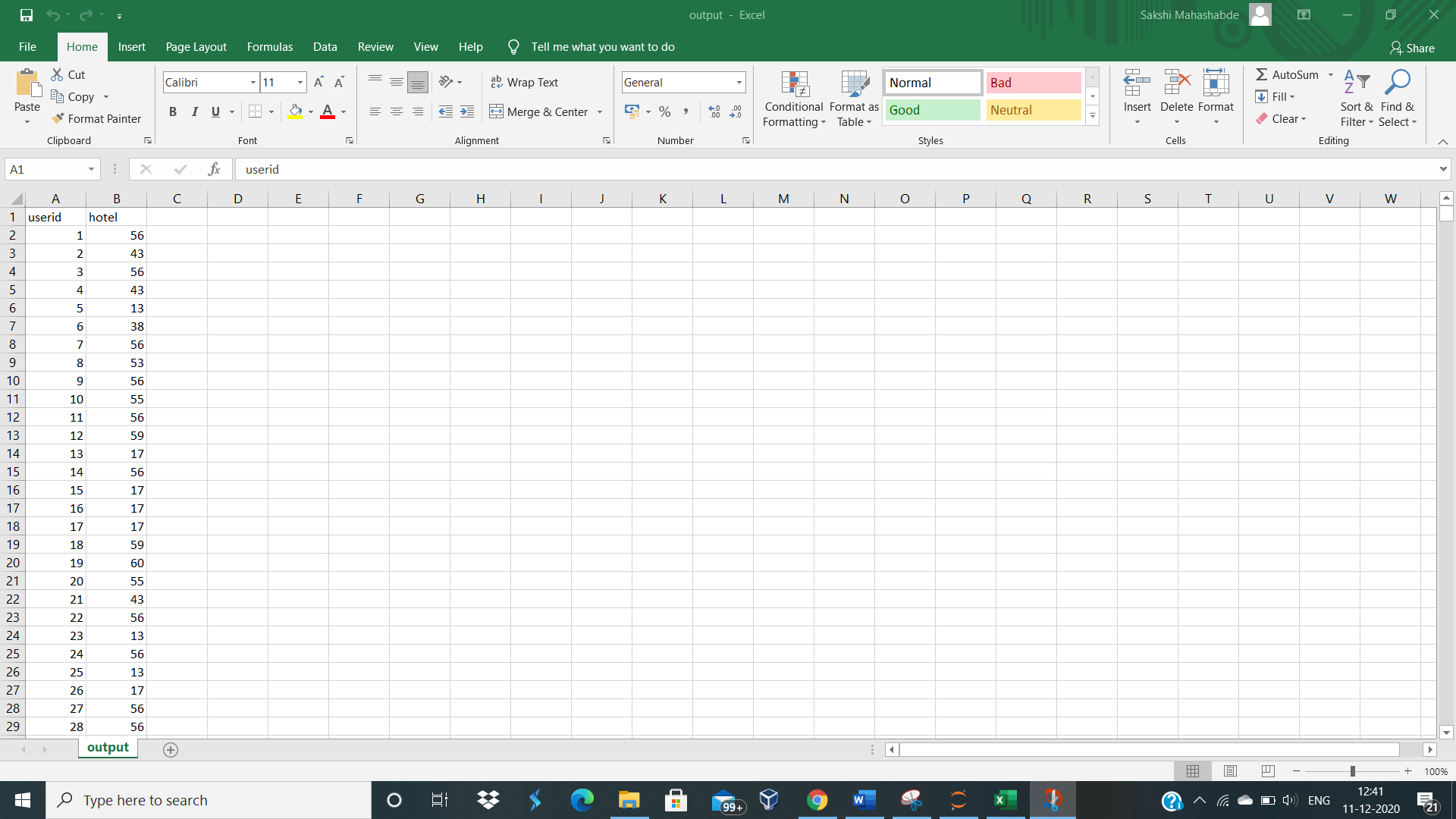
I chose XGBoost as it was giving better results by some points, plus, Boosting is based on weak learners (high bias, low variance). Boosting reduces error mainly by reducing bias (and to some extent variance, by aggregating the output from many models). On the other hand, Random Forest uses fully grown decision trees (low bias, high variance). It tackles the error reduction task in the opposite way: by reducing variance, and for given dataset we need to reduce bias.

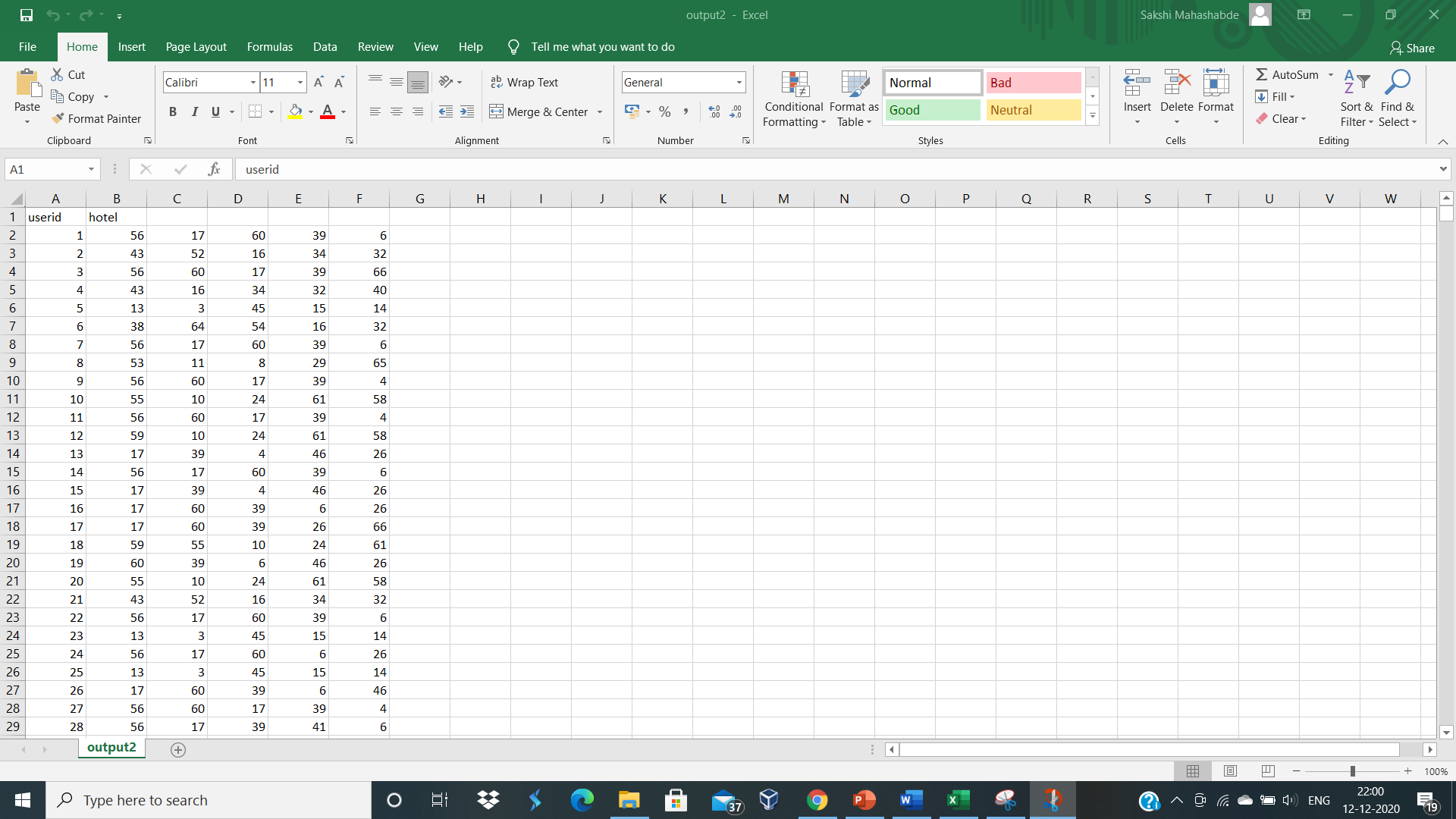
Next step is to generate the results in tab-delimited file as asked. I have produced two types of output files:

1. just predicting the next hotel, which is not in user's history.
2. predicting next hotels from obtained 5 probabilistic outputs, which are not in user's history.



This code would generate the output in the form of csv file.





In the above result we have seen 5 different hotel clusters for each user has been generated.

**8.LIMITATIONS AND CONCLUSION**

**8.1 Limitations**

Sometimes the dataset becomes very lengthy and memory intensive due to which there are lot of issues in front of us in getting the proper results for the particular dataset. Also the version of Python differs every one or two days the technology is enhancing due to which every one or other day the versions of python is also changing. This can also be an issue in getting the proper result. We need to resolve these bugs to get the accurate result. One should have knowledge to resolve the issues completely.

**8.2 Conclusion and future work**

We have implemented the Hotel Recommendation Using Past Entries using Random Forest Algorithm and XG boost algorithms. On testing the dataset with both the algorithms we came to a conclusion that we got the near about same results on testing it with both the algorithms but we chose XG boost as our algorithm. Also non-repetitiveness gave better results than repeating records.

This again reaffirms the fact that combining several weak learners has a synergistic affect. Most of our methods involved ranking of clusters by their predicted class probabilities which seems fair. Due to the volume of data, we could not replicate Due to the volume of data, we could not replicate the distance matrix completion techniques employed by the first rank solution. This leaves a room for future improvement, wherein we can try replicating the existing code in conjunction with our features, data leak solution and ensemble learning to achieve an even better result. Another direction for future work is to try different ranking methods like Rank SVM and Rank Boost.

XG Boost has a performance comparative to that of Random Forest but slightly better. This is because both these methods work in a similar manner, handling both categorical features and missing values efficiently. XG Boost automatically learns what is the best direction to go when a value is missing. Its learning mechanism gives a slightly better solution than random forests.

### 

### **REFERENCES**

G. Huming and L. Weili. A hotel recommendation system based on collaborative filtering and rankboost algorithm. In 2010 Second International Conference on Multimedia and Information Technology, April 2010.

[2] A. Levi, O. Mokryn, C. Diot, and N. Taft. Finding a needle in a haystack of reviews: Cold start contextbased hotel recommender system. In Proceedings of the Sixth ACM Conference on Recommender Systems.