Expedia Hotel Recommendation

Desai, Kaushal
Dept. of Computer Science
NYU
NY, USA
kaushal.desai@nyu.edu

Kapoor, Sasha
Dept. of Computer Science
NYU
NY, USA
sk5820@nyu.edu

Singhal, Chirag
Dept. of Computer Science,
NYU
NY, USA
cs4536@nyu.edu

Abstract—

Expedia wants to take proverbial rabbit hole out of hotel search by providing personalized hotel recommendations to their users. Planning your dream vacation, or even a weekend escape, can be an overwhelming affair. With hundreds, even thousands of hotels to choose from at every destination, it will be difficult to know which will suit your personal preferences. To use the Data Science to provide a tool to build a reccomender system that will recommend hotels to the users can ease the work of Expedia to a great extent.

Keywords—Analytics, Recommnender, Spark, RapindMiner, Python, Scala, Classification, Machine Learning, MS Azure, Feature Selection, MlLib.

I. Introduction

Which hotel is children friendly? Which hotel will my family like? Will this hotel fit my pocket? Can I find a budget hotel of my choice? Finding a perfect hotel is tough. There is a need for a tool which can help in making such decisions.

. Expedia.com is a travel website owned by Expedia Inc. The website can be used to book airline tickets, hotel reservations, car rentals, cruises, vacation packages and various attractions via the internet. The site uses its own hotel reservation system for making hotel reservations.

Expedia uses search parameters their hotel recommendations, but there aren't enough customer specific data to personalize them for each user. We are planning to contextualize customer data and predict the likelihood that a user will stay at 100 different hotel groups.

Our task is to build a recommendation system based on the data provided by Expedia on Kaggle competitions. Our goal of this competition is to predict the booking outcome (hotel cluster) for a user event, based on their search and other attributes associated with that user event.

II. Process Overview

The Cross Industry Standard Process for Data Mining (CRISP-DM) was followed for this project as CRISP-DM is one of the top models for data mining process used to solve problems by experts. It provides a streamlined approach for the project development considering the iterative nature of data mining and machine learning projects.

CRISP-DM is based on the process flow shown in Figure 1.

Figure 1. CRISP-DM Model

The model proposes the following steps:

1. Business Understanding – to understand the rules and business objectives of the company.
2. Understanding Data – to collect and describe data.
3. Data Preparation – to prepare data for import into the software.
4. Modelling – to select the modelling technique to be used.
5. Evaluation – to evaluate the process to see if the technique solves the problem of modelling and creation of rules.
6. Deployment – to deploy the system and train its users.

The major steps involved in this process are described in the later sections.

III. Technical Specifications

- Programming Languages:
- Scala 1.6.0, Python 2.6.6
- Framework:
- Spark1.6.0, numpy, scipy, scikit
- Libraries:
- Spark MLlib 1.6
- Tools:
- Microsft Azure Machine Learning Studio
- RapidMiner 7.0
- Platforms Used:
- Windows PC
- Dumbo Cluster at NYU
- Intellij
- Dataset:
- Kaggle Expedia Hotel recommendation(3.9 GB)
- (https://www.kaggle.com/c/expedia-hotel-recommendations/data)

IV. Business Understanding

Currently, Expedia uses search parameters to adjust their hotel recommendations, but there aren't enough customer specific data to personalize them for each user. This creates a need to contextualize customer data and predict the likelihood a user will stay from among 100 different hotel groups with an aim to provide a personalized hotel recommendation to each of its millions of users.

The goal is to build two types of recommender systems, **Content-Based System and Collaborative-Based System.**

a. **Content-Based System:** Content-based systems examine properties of the items recommended. For instance, if a Netflix user has watched many cowboy movies, then recommend a movie classified in the database as having the “cowboy”.

b. **Collaborative-Based Systems:** Collaborative-Based Systems recommend items based on similarity measures between users and/or items. The items recommended to a user are those preferred by similar users.

V. Data Understanding

The initial data is collected and is verified for its quality. It is observed that our data constitutes combination of user features and hotel features, with user data portraying user demography and

search preference. The hotel features mainly constitute its geological location along with its proximity from the user. The description of the data is provided as follows:

Downloaded files:

File Name	File Size	File Description
Destination.csv	16.18 mb	hotel search latent attributes
Test.csv	65.92 mb	the test set
Train.csv	511.16 mb	the training set

The train and test datasets are split based on time: training data from 2013 and 2014, while test data are from 2015.

Training data includes all the users in the logs, including both click events and booking events. **Test data** only includes booking events.

Destinations.csv data consists of features extracted from hotel reviews text.

Feature Descriptions

Train.csv/test.csv

Column name	Description	Data type
date_time	Timestamp	string
site_name	ID of the Expedia point of sale (i.e. Expedia.com, Expedia.co.uk, Expedia.co.jp, ...)	int
posa_continent	ID of continent associated with site_name	int
user_location_country	The ID of the country the customer is located	int
user_location_region	The ID of the region the customer is located	int
user_location_city	The ID of the city the customer is located	int
orig_destination_distance	Physical distance between a hotel and a customer at the time of search. A null means the distance could not be calculated	double
user_id	ID of user	int
is_mobile	1 when a user connected from a mobile device, 0 otherwise	tinyint
is_package	1 if the click/booking was generated as a part of a package (i.e. combined with a flight), 0 otherwise	int
channel	ID of a marketing channel	int
srch_ci	Checkin date	string
srch_co	Checkout date	string
srch_adults_cnt	The number of adults specified in the hotel room	int
srch_children_cnt	The number of (extra occupancy) children specified in the hotel room	int
srch_rm_cnt	The number of hotel rooms specified in the search	int
srch_destination_id	ID of the destination where the hotel search was performed	int
srch_destination_type_id	Type of destination	int
hotel_continent	Hotel continent	int
hotel_country	Hotel country	int
hotel_market	Hotel market	int
is_booking	1 if a booking, 0 if a click	tinyint
cnt	Numer of similar events in the context of the same user session	bigint
hotel_cluster	ID of a hotel cluster	int

Destination.csv

Column name	Description	Data type
srch_destination_id	ID of the destination where the hotel search was performed	int
d1-d149	latent description of search regions	double

VI. Data Preparation

We implemented various data cleaning, data reduction and feature selection techniques and their combinations to prepare our data.

1. **Replace Missing Values:** Data columns with missing values are replaced by specified

replacements such as median, minimum, maximum, zero, none, most frequent value of the attribute. The replacement option giving the highest accuracy is chosen for further data reduction. We chose zero or none to replace missing values in our data. The screenshot attached below shows how we used missing values in Microsoft Azure Machine Learning Studio.

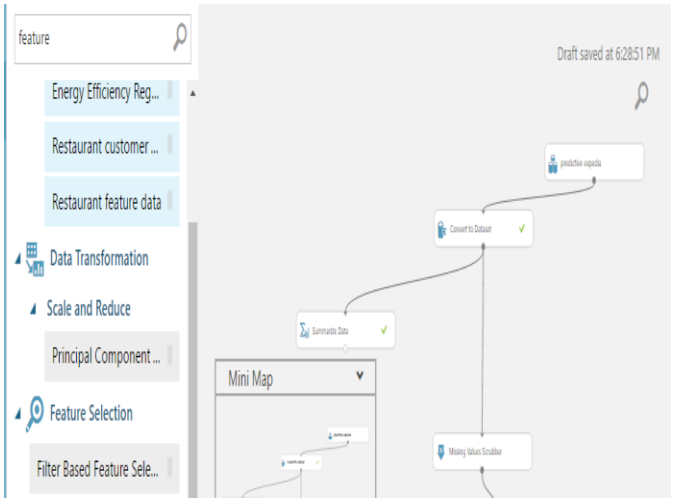


Figure 2. Data Preparation

2. **High Correlation Filter**: Data columns with very similar trends are also likely to carry very similar information. In this case, only one of them will suffice to feed our machine learning model. Pairs of columns with correlation coefficient higher than a threshold are reduced to only one.

3. **Low Variance Filter**: Data columns with little changes in the data carry little information. Thus all data columns with variance lower than a given threshold are removed.

4. **Missing Values Ratio**: Data columns with too many missing values are unlikely to carry much useful information. Thus data columns with number of missing values greater than a given threshold can be removed. The higher the threshold, the more aggressive the reduction. Here is a screenshot in which the highlighted column in blue shows the number of missing values some of the features have in our data.

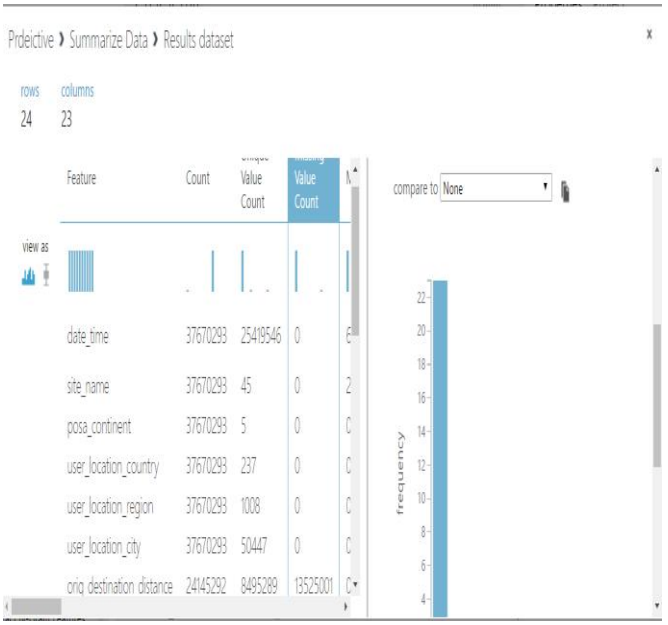


Figure 3. Imputing Missing Values

5. **Principal Component Analysis**

Principal Component Analysis (PCA) is a statistical procedure that orthogonally transforms the original *n* coordinates of a data set into a new set of *n* coordinates called principal components. As a result of the transformation, the first principal component has the largest possible variance; each succeeding component has the highest possible variance under the constraint that it is orthogonal to the preceding components. Keeping only the first $m < n$ components reduces the data dimensionality while retaining most of the data information, i.e. the variation in the data. Below is the screenshot attached, which shows how PCA is applied in Microsoft azure.

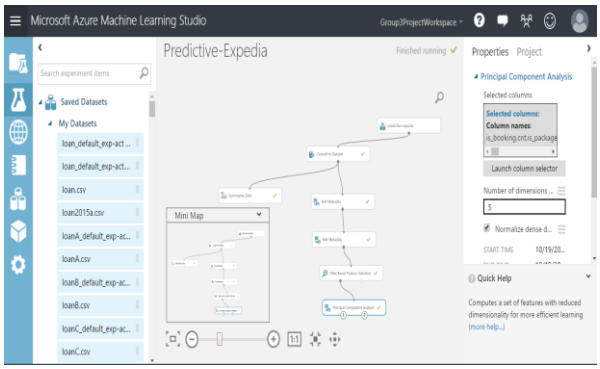


Figure 4. Dimensionality Reduction

6. **Filter Based Feature Selection**: Feature Selection is the process of selecting the attributes in the dataset that are most relevant to the predictive problem one is working on. Filter Based Feature Selection uses different statistical tests to determine the subset of features with the highest predictive power. We choose a statistical measure to apply, and the module calculates a score for each column that has been used as a feature. The features are then ranked by the score and the feature columns with the best scores are used in building the model, while others are kept in the dataset but not used in analysis. To use feature selection, one must choose an input dataset that contains atleast two columns that are candidates for use as features. The columns that one can analyze depend on the target column and the metric used to compute the scores. Filter Based Feature Selection provides a selection of widely used statistical tests for determining the subset of input columns that have that have the greatest predictive power. The statistical tests we used are:
    - **Pearson Correlation**: Pearson's correlation coefficient is also known in statistical models as the **r** value. For any two variables, it returns a value that indicates the strength of the correlation. Below is a screenshot which shows top features with highest predictive power that we got using Pearson Correlation.

Figure 5. Feature Selection

- **Fisher Score**: The Fisher Score is sometimes termed the information score, because it represents the amount of information that one variable provides about some unknown parameter on which it depends.

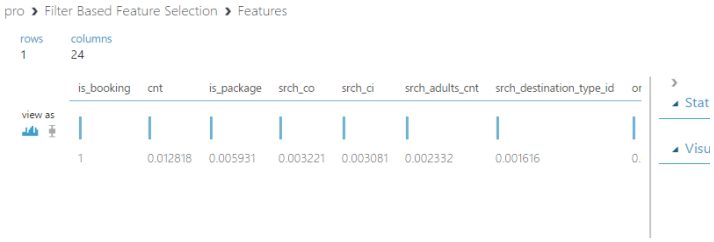


Figure 6. Feature Selection

- **Chi-Squared**: The two-way chi-squared test is a statistical method that measures how close expected values are to actual results.

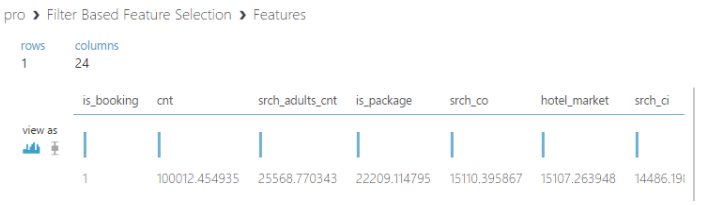


Figure 7. Feature Selection

- **Spearman's Correlation**: Spearman's coefficient is a nonparametric measure of statistical dependence between two variables. It expresses the degree to

which two variables are monotonically related.

The maximum accuracy was achieved by applying these pre-processing techniques.

Preprocessing Method	Accuracy	Notes
Replace Missing Values + Filter Based Feature Selection	83.3%	Using MS Azure

VII. MODELLING AND EVALUATION

Modelling process used to create, test and validate a model to best predict the probability of an outcome. Models help to derive new information about the data. Every model has its strengths and weakness and is best suited for different types of problem

Classification Modelling

A user may or may not book hotel for his/her stay. Classification modelling is an approach to identify whether a given user will book a hotel or not.

Classification Modelling was used to build models that would predict whether the user may or may not book hotel for stay. The process also helped us get insight about key features, to consider while building the recommender system.

The target variable user for classification modelling was 'is_booking'.

MS Azure was used to build the classification models.

Below mentioned are the classification algorithms we used for our project.

a. **Neural Networks:** Artificial neural networks are forecasting methods that are based on simple mathematical models of the brain. They allow complex nonlinear relationships between the response variable and its predictors. A neural network can be thought of as a network of "neurons" organized in layers. The predictors (or inputs) form the bottom layer, and the forecasts (or outputs) form the top layer. There may be intermediate layers containing "hidden neurons".

b. **Decision Tree:** A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm. Decision trees are commonly used in operations research, specifically in decision analysis, to help identify a strategy most likely to reach a goal, but are also a popular tool in machine learning.

c. **Support Vector Machines:** A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples.

d. **Logistic Regression:** In statistics, logistic regression, or logit regression, or logit mode is a regression model where the dependent variable (DV) is categorical. Logistic regression model is used for analyzing a dataset in which there are one or more independent variables that determine an outcome.

Classification Algorithm used	Recall = TP/TP + FN	AUC	Accuracy	Notes
Neural Networks	0.711	0.748	0.647	Microsoft Azure used, Threshold-0.1
Boosted Decision Tree	0.651	0.772	0.723	Microsoft Azure used, Threshold-0.1
Logistic Regression	0.713	0.735	0.627	Microsoft Azure used, Threshold-0.1
Support Vector Machine	0.576	0.702	0.673	Microsoft Azure used, Threshold-0.1

Figure 8. Confusion Matrix Result

As provided in the figure we evaluate our model based on three factors

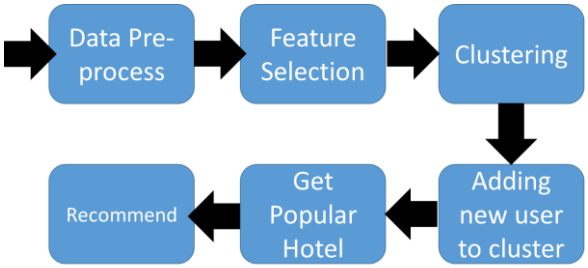
a. **Recall:** Recall also called Sensitivity is a True Positive rate. It is basically is the fraction of relevant instances that are retrieved.
b. **AUC:** The area under the curve (AUC) is the measure of hit rate to the false alarm rate. AUC has become a standard measure in tests of predictive modeling accuracy. The AUC is an estimate of the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance.
c. **Accuracy:** Accuracy is the starting point for analyzing the quality of a predictive model, and one of the important criterion for prediction. Accuracy measures the ratio of correct predictions to the total number of cases evaluated.

We realized that of the four models, boosted decision tree provides best insight as it predicts the data with highest level of accuracy of 72.30% and AUC of 77.20%. The result could be verified with the fact that Decision trees are able to predict categorical data with higher accuracy than other models, and as our model consist of categorical features, Decision tree model is more successful than other models.

d. Recommender System

a. **Content-Based System Modelling:** Used Apache Spark framework to cluster 'Train Data' based on user feature. Initially we used K-means for clustering. K-Means clustering intends to partition *n* objects into *k* clusters in which each object belongs to the cluster with the nearest mean. This method produces exactly *k* different clusters of greatest possible distinctions. This process was achieved by converting the categorical fields in one hot vector format. However, 'memory exception' error was obtained due to the large size of the data. To tackle this issue k-modes algorithm was used for clustering training data which converted our data into vector form. K-modes algorithm is basically used for clustering categorical variables. It defines clusters based on the number of matching categories between data points.
This trained model was used to map new users from Test data. Post clustering, we calculated popular 'Hotel Clusters' for each 'User Cluster'. We then recommended 'Hotel Clusters' for the Test User based on popular hotels of the Trained cluster.

The Content Based System thus created provided 45.6% accuracy in recommendation



Work Flow 1. Content Based Recommender

```
#!/usr/bin/env python

import numpy as np
from kmodes import kprototypes

# stocks with their market caps, sectors and countries
syms = np.genfromtxt('\home\chirag\Music\train.csv', dtype=str, delimiter=',')[:, 0:15]
X = np.genfromtxt('\home\chirag\Music\train.csv', dtype=object, delimiter=',')[:, 1:15]

kproto = kmodes.KModes(n_clusters=100, init='Cao', verbose=2)
clusters = kproto.fit_predict(X, categorical=[1, 2])

# Print cluster centroids of the trained model.
print(kproto.cluster_centroids_)
# Print training statistics
print(kproto.cost_)
print(kproto.n_iter_)
f = open('/home/chirag/Music/clustersKmodes','w')
f.write
```

Figure 7. Content Based Clustering Program Screenshot

b. Collaborative-Based System Modelling: We first identified that combination of 'user_location_city' and 'orig_destination_distance' accurately defines specific hotel. Based on this finding we built our recommender model using three cases.

- First we looked for leakage data - If there existed same combination of 'user_location_city' and 'orig_destination_distance' in 'Test Data' as in 'Train Data' we output same 'hotel_cluster' variable associated with 'Train Data'. Further, we ranked our data based on time of searches, to rank same 'hotel_cluster' having same features.

- We then calculated the most popular 'hotel_cluster' having same 'srch_destination_id', 'hotel_country', 'hotel_market' and 'book_year' to recommend popular hotels in the destination with high accuracy.
  The popularity was calculated by assigning weights to number of clicks and booking ratio. To assign weight to booking, we first calculated that for every 6 clicks there is one successful booking on average. Using this finding we calculated that booking constitutes 5.67 clicks.

- Finally, we found the most popular 'hotel_cluster' based on 'hotel_country' to determine popular hotels in the country to recommend best hotels overall.

The Collaborative Based System thus created provided 49.2% accuracy in recommendation

. Collaborative-Based Clustering Program Screenshot

```
import datetime
from heapq import nlargest
from operator import itemgetter
import math

def prepare_arrays_match():
    f = open("/home/chirag/Documents/train.csv", "r")
    f.readline()

    best_hotels_od_ulc = dict()
    best_hotels_uid_miss = dict()
    best_hotels_search_dest = dict()
    best_hotels_country = dict()
    popular_hotel_cluster = dict()
    best_s00 = dict()
    best_s01 = dict()
    total = 0
    count = 0

    # Calc counts
    while 1:
        count = count + 1
        line = f.readline().strip()
        total += 1

        if total % 2000000 == 0:
            print('Read lines...'+str(total))

        if line == '':
            break

        arr = line.split(",")
```

Figure 9. Collaborative-Based Clustering Program Screenshot

VIII. DEPLOYMENT

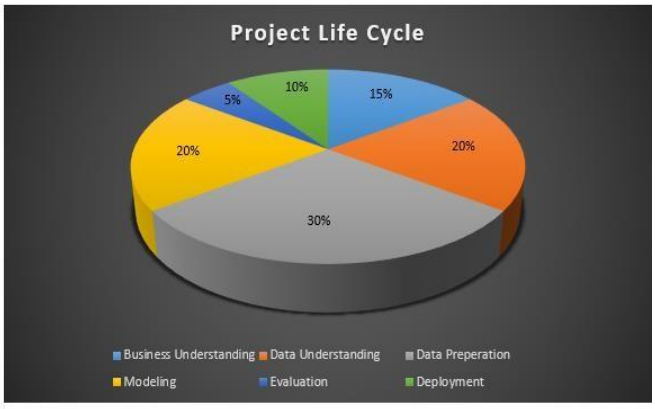
We made two recommender systems and we know the strength of each of them. We checked the accuracy of both the models by uploading our results on kaggle and getting a rank for both the recommender systems. So in deployment phase we select the recommender system which can most benefit the users by recommending them a hotel of their choice. Also it is required that we must not forget to create artifacts in each preceding phase of the process so as to conduct training sessions with the users of the system.

Information that is true today may not be tomorrow, since the data are very volatile and different types of fault are always expected. So it is necessary that the model is flexible enough to incorporate these changes. We can implement our project in such a way such that it can handle more data for our training model which

adds the possibility of flexibility in our project with new features and testing those features.

IX. Experiment & Results

Finally, all the phases of CRISP-DM Model were completed. Our efforts can be summed as below:

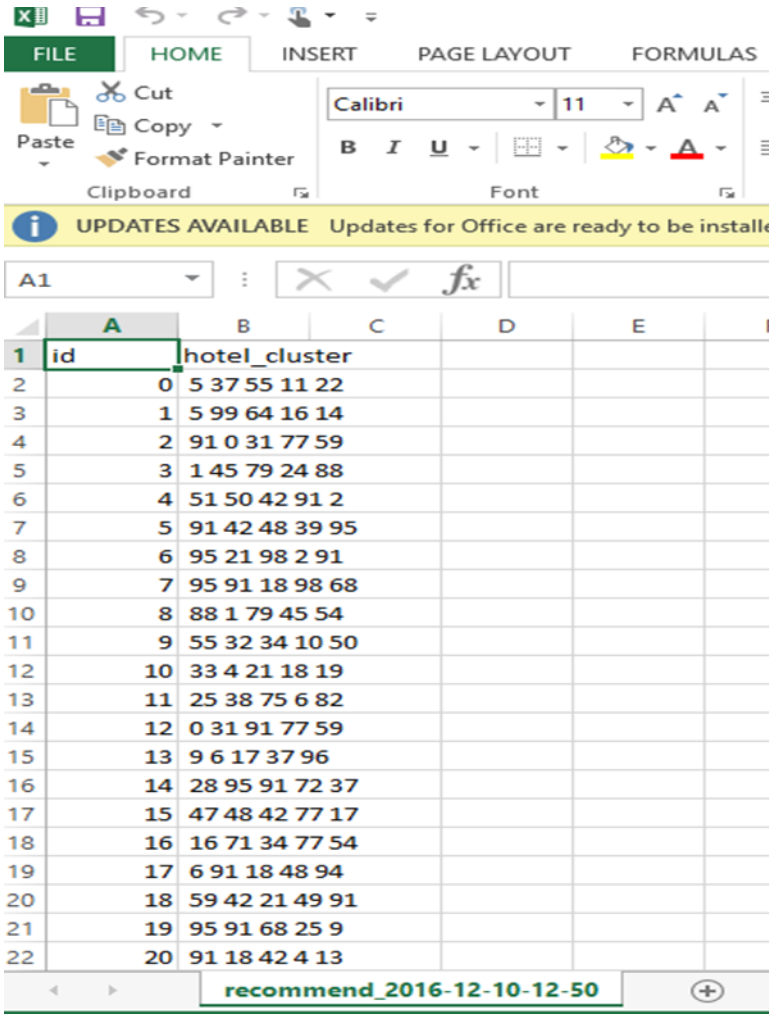


Graph 1. Crisp-DM Model Effort

We can see that data preparation and modelling took most of the time as these were the most time consuming phases.

We built two recommender systems, one based on collaborative filtering and other based on content based filtering. Both the recommender systems, however, recommended five hotel clusters for each user. Our recommender systems recommended hotel clusters for 800,000 users. Here is a screenshot of the output we got from the recommender systems where id is the user_id and hotel_cluster is the hotel_cluster_id.

Recommender System	Accuracy %
Collaborative Based	49.90%
Content Based	45.6%



	A	B
1	id	hotel_cluster
2	0	5 37 55 11 22
3	1	5 99 64 16 14
4	2	91 0 31 77 59
5	3	1 45 79 24 88
6	4	51 50 42 91 2
7	5	91 42 48 39 95
8	6	95 21 98 2 91
9	7	95 91 18 98 68
10	8	88 1 79 45 54
11	9	55 32 34 10 50
12	10	33 4 21 18 19
13	11	25 38 75 6 82
14	12	0 31 91 77 59
15	13	9 6 17 37 96
16	14	28 95 91 72 37
17	15	47 48 42 77 17
18	16	16 71 34 77 54
19	17	6 91 18 48 94
20	18	59 42 21 49 91
21	19	95 91 68 25 9
22	20	91 18 42 4 13

Figure 10. Recommender Output

We also secured a rank of 595 out of 3000 teams with an accuracy of 0.49904 for collaborative based recommender system. Whereas, for content based recommender system, our accuracy was 0.402. If we were able to increase our accuracy for collaborative based recommender system by 0.1 , we would have been in top 20 in the kaggle competition.

Figure11. Kaggle Ranking

Thus, in classifying whether a user will book a hotel or not, we got best results by using Boosted Decision tree. And in totality, Collaborative Based Recommender System gave us better results than Content Based Recommender System.

X. CHALLENGES FACED

Some of the challenges we faced during the course of the project were-

- Since our data was large, we couldn't use tools like Knime, Rapid Miner etc.
- Setting up of NYU HPC account for the first time.
- MS Azure had a limit of 10 GB per account. So, we had to use multiple azure accounts.
- The Recommender system available in Spark's MLlib recommends on the basis of single feature. We had to built our custom recommender system that could recommend on the basis of multiple features.

XI. CONCLUSION

Thus, we implemented the project on recommending hotel to users using Microsoft Azure Machine Learning Studio and Spark (MlLib).

For Data preparation we tried different preprocessing methods on our data and choose the best performing methods. Then, we implemented different classification models namely Boosted Decision Tree, Neural Networks, Logistic Regression and SVM Model. Then we built two recommender systems(collaborative and content based) using languages like python and scala and spark framework. Lastly, we compared the accuracies of the models and analyzed the result.

This project gave us a better understanding of Data Science concepts particularly the Crisp-DM model, Data Preprocessing methods, Data Classification and validation.

XII. FUTURE SCOPE

This project shows how the power of Data Science can be used in a real life scenario. We have shown that this tool can help expedia recommend hotels to users. If we could increase the accuracy of our model, it can become one of the recommender systems that expedia used in its day to day operations for recommending hotels to users. Also, there are several other AI Algorithms like Fuzzy Algorithms & Genetic Algorithms, which are still not supported by Spark's MlLib Library which might result in better Accuracy.

XIII. REFERENCES

[1]http://spark.apache.org/
[2]http://spark.apache.org/docs/latest/mllib-guide.html
[3]https://msdn.microsoft.com/enus/library/azure/dn905854.aspx
[4] http://www.cis.upenn.edu/~ungar/CF/
[5]https://en.wikipedia.org/wiki/Recommender_system
[6] https://studio.azureml.net/Home
[7] http://recommender-systems.org/collaborative-filtering/
[8]http://dca.iupr.com/lectures/lecture08-classification-with-neural-networks
[9] https://docs.microsoft.com/en-us/azure/fundamentals-introduction-to-azure