Recommendation Systems for Tourists

1. Introduction & Motivation

What do you do when you need a break from your mundane life? – You go on a **Vacation!** Everybody loves going on vacations, but we need a considerable amount of time sitting and planning for one. We must go through every hotel, attraction, restaurant, their rating, reviews, attributes and choosing the set of things to do within our budget – The entire process is highly tedious. Planning for a vacation that takes into consideration all the travel preferences of an individual without having to look through at least hundreds of websites is close to impossible. Our motivation behind doing this project is to reduce the time spent on planning for the vacation and help travelers spend more time on a vacation that they will love. Our project provides a **Tailor-made Travel Plan** for travelers based on their travel details and preferences. We obtain information from users about the destination, travel dates, budget, amenities that they look for in hotels, categories of attractions that users would love to visit and their favorite cuisine type. Using these we generate a travel plan for the entire trip, **recommending** users where to stay, what to do at which part of the day and where to eat for different meals of a day. This project will be a one-stop tool for travelers planning their vacations, we achieve the same using Data Science tools and techniques which will be explained in detail in the report.

2. Problem Definition and Challenges

For the goals of the project, we have used 3 different recommendation models – ***2 Collaborative filtering techniques and 1 Hybrid technique***. The motivation behind this was to explore different models in the field of recommender systems and come up with an efficient solution. ***Restricted Boltzmann Machine (RBM)*** and ***Matrix Factorization using Alternating Least Squares*** technique were the Collaborative Filtering techniques used and ***Hybrid technique*** uses a combination of ***K-Means, a content-based filtering*** technique and ***K-Nearest Neighbours, a Memory-based Collaborative Filtering*** technique. The major modules of the project include:
(i) Data collection through web scraping, cleaning and integration **(ii)** Hotel recommendation based on user's amenity requirements **(iii)** Attraction recommendation for Morning and Evening using categories of tours **(iv)** Restaurant recommendation for different meals of the day using cuisine type and food preferences **(v)** Integration of all the above mentioned recommendations in-order to provide one-stop travel plan for each day of travel.

Major **challenges** in implementing the above modules were as follows:

(i) **Dataset:** The data required to provide the recommendations mentioned above was not available readily. Web scraping techniques were used to crawl through several attractions, hotels and their respective reviews on each item from relevant websites. (ii) **Categories of Recommendation:** Using different models for providing recommendations for Hotels, Attractions and Restaurants required different methods for data collection, pre-processing and cleaning, different techniques for user profiling and hence, thrice the work. (iii) **Evaluating** and improving the performance of the models to obtain relevant recommendations (iv) **Integration** of all the recommendations to get an efficient travel plan was a challenge. (v) **Scalability** of the recommendations provided.

3. Data Science Pipeline

3.1 Data Collection:

For the project requirements, we needed datasets containing information about details of attractions, hotels and restaurants and their corresponding reviews. There was no dataset readily available for Hotels and Attractions, thus making Data Collection a very important task for our project. To obtain these datasets, we crawled through thousands of attractions and hotels on **Tripadvisor**. We used ***lxml*** - a python library along with ***requests -*** another python library, for downloading and processing HTML files using ElementTree API. We had to crawl through hotels and attractions in a different manner due to unique HTML page structures of both the categories. Around ***3.5k attractions*** and ***35k hotels*** were collected through scraping tripadvisor. We also created a crawler for scraping the reviews for each attraction and hotel in the dataset and collected reviews close to ***33K*** for attractions and ***2.4M*** for hotels. Since the websites have a limit on the number of hits per day, we performed the data collection in batches. For the restaurant recommendation, we used data from **YELP dataset (original)**. This dataset consists of data related to users, businesses (not restricted to restaurants), reviews, check-in time, user tip reviews and photos of respective

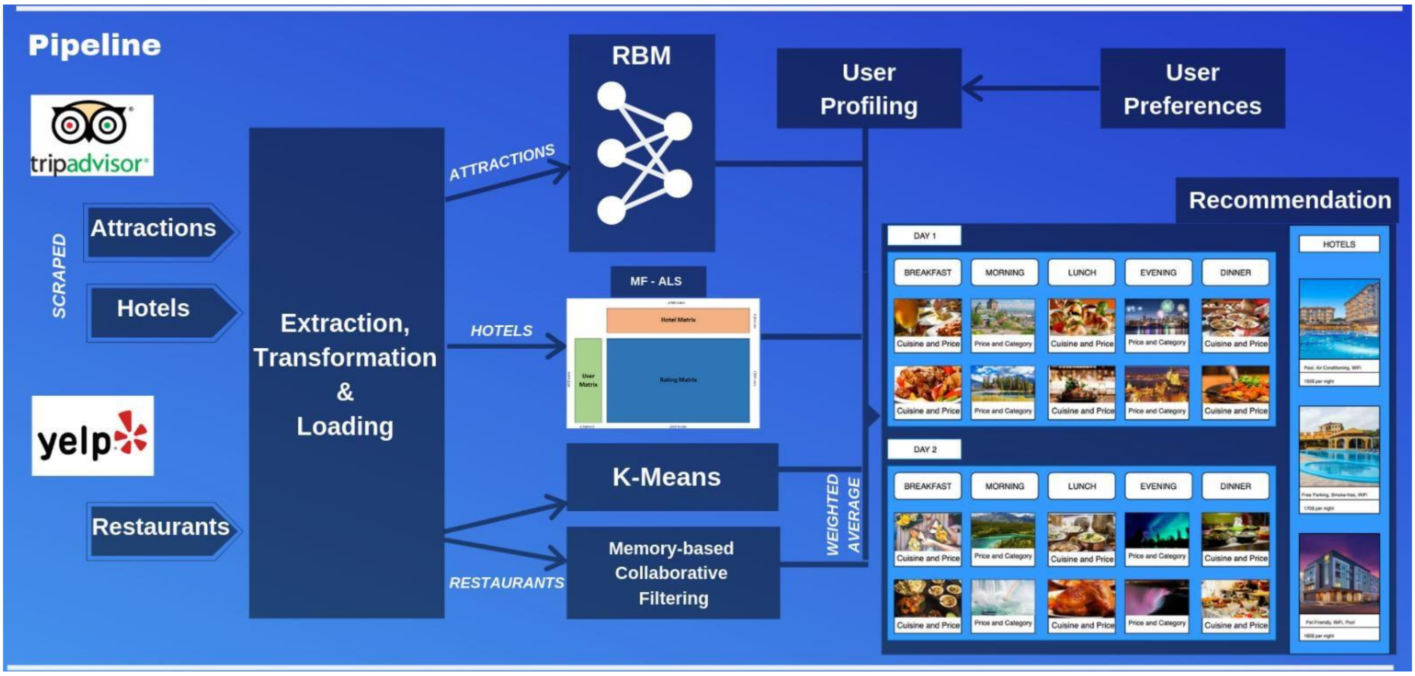


Figure 1 Complete technological pipeline of our project.

businesses. We used the yelp business, yelp reviews and the yelp photos dataset. The business dataset contained details of all the business on Yelp including restaurants, salons, gas stations, drugstores, etc., without any feature explicitly mentioning the categories of these businesses. The challenge was to extract only the restaurants from the dataset. The techniques used to do the same will be explained later in the report. We got ***12K*** restaurants and close to ***5M*** reviews on the yelp dataset for restaurants.

3.2 Data Collection and Integration:

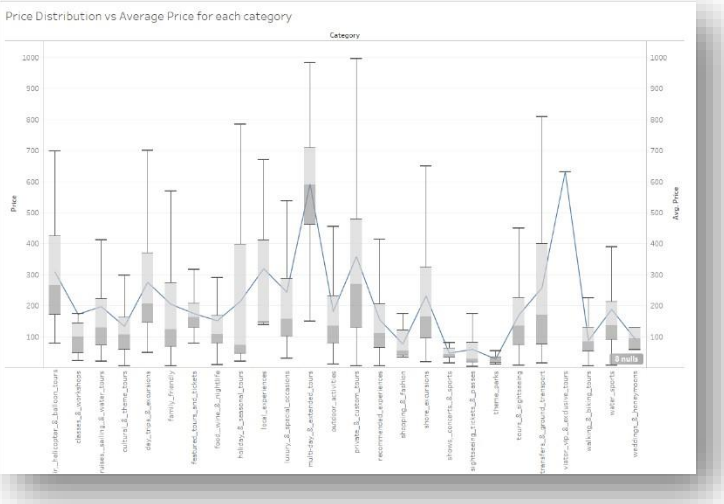
Having collected all the data mentioned above, next big task was cleaning the dataset and integrating them. Due to the inconsistencies in the Tripadvisor website, there were missing values for prices, ratings and details about the location for both the scraped datasets.

i. *Hotels*: For the hotel dataset, GeoPy - a python library to geolocate a query to an address or coordinates was used to obtain the latitude, longitude positions of a hotel from it's address. Values for which GeoPy wasn't able to output latitude, longitude coordinates from address, we tried to get coordinates based on its locality and further, based on its province. The missing prices were filled hierarchically with the average price of hotels present in a city, then province and finally the entire dataset based on availability of data. This was based on the assumption that hotels in same locality have similar pricing for competency in the market. Missing ratings were filled out in the same manner. Amenities was the most important feature in hotels dataset and hence, for interpretability, hotel amenities (string) was split, exploded, cleaned and converted into a list of amenities. Continuous unique IDs were provided to users and continuous hotel IDs were provided to hotels.

ii. *Attractions*: For attractions, only the priced tours and sightseeing were collected. The locations of the attractions could not be scraped from the website, and hence the address of the attraction was created from the information scraped from websites, and finally Google API was used to obtain the coordinates of an attraction. The missing coordinate values after this step was handled based on the assumption that similar categories of attractions will be present in the same locality. For example, two different water sports will be present next to each other in a city, there will be an area in the city where all the clubs will be present, etc. The location coordinates of an attraction were assumed to be the average location of attraction categories in a city. Missing prices and ratings were also averaged based on city and category of attraction.

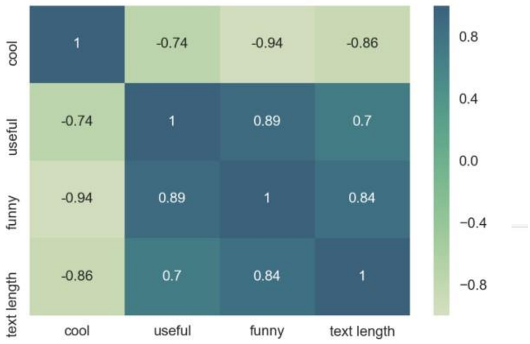
iii. *Restaurants*: The first level of Data cleaning required us to obtain from the humongous yelp dataset, businesses that pertain to 'restaurants' only. This was achieved using text filtering, we filtered out the businesses for which the 'categories' column of the business dataset had text values unrelated to the food/restaurants industry. Further, we cleaned up the reviews dataset to have only those businesses that are of interest to us (restaurants) and to be aligned with the those in the business dataset. Also, we have removed restaurants that have been marked as closed in the dataset, as recommending 'not-in business' restaurants doesn't make sense. The categories and attributes columns were of utmost important for us from the business dataset. Both of theses columns were textual features and hence, extensive cleaning had to be done to convert them- 'categories' to lists and 'attributes' to dict of dict. We also computed sentiment score, to find out how positive a review for a restaurant is, in-order to provide better personalized recommendations to our users, the details of how we achieved this will be talked about later in this report. Finally, we had to maintain the index (using reset_index) of the dataframe as we had used pandas for restaurants dataset- rather than assigning a continuous row ID for businesses and reviews dataset, we chose to exploit the in-built features of pandas.

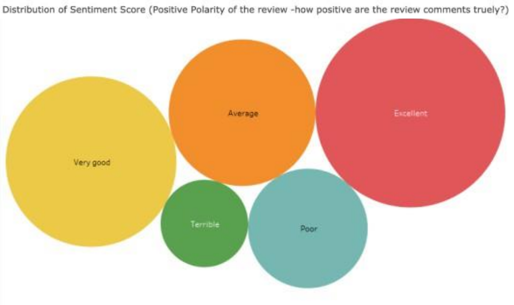
3.3 Exploratory Data Analysis (EDA):

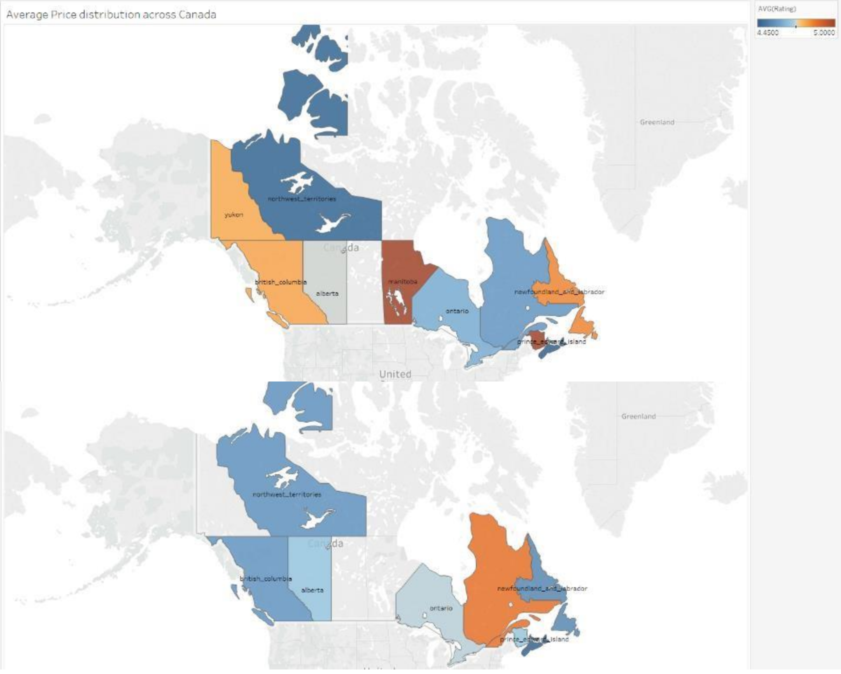

Price Distribution vs Average Price for each category

In order to have a better understanding of the data collected and integrated for recommendations, we performed extensive exploratory data analysis on various features of each dataset. This step was highly significant in helping us understand the distribution of prices and ratings across different categories and other attributes in the dataset. EDA helped us arrive at the decision for using specific attributes from each dataset for user profiling by explaining variance in the data with respect to different attributes of the dataset. The following is the list of some of the important analysis that led us to the decisions we made:

(i) Average rating for different categories of attraction **(ii)** Rating distribution for attraction categories **(iii)** Average Price and rating distribution across different provinces of Canada **(iv)** Price distribution vs Average price for attraction categories **(v)** Province-wise and city-wise number of attractions **(vi)** Distribution of poorly and highly rated restaurants province-wise **(vii)** Rating distribution for restaurants province-wise **(viii)** Number of restaurants province-wise and city-wise **(ix)** ***Sentiment Analysis on user reviews and user-tip reviews*** was performed to better understand the underlying restaurants dataset. We were able to identify that the dataset was highly skewed towards positive reviews. So, we computed the positive to negative ratio and normalized the score across the entire dataset, in-order to understand how truly positive a user review is indeed. The challenge here was that reviews and tips were sometimes written in French. Thus, we first used google translator API to convert them to english sentences before feeding them into ***VADER sentiment analyzer***, which is a rule-based system that is explicitly designed for English language. We have used the ***compound score*** metric from VADER to compute the positive-negative ratio.




Distribution of Sentiment Score (Positive Polarity of the review -how positive are the review comments truely?)


Average Price distribution across Canada

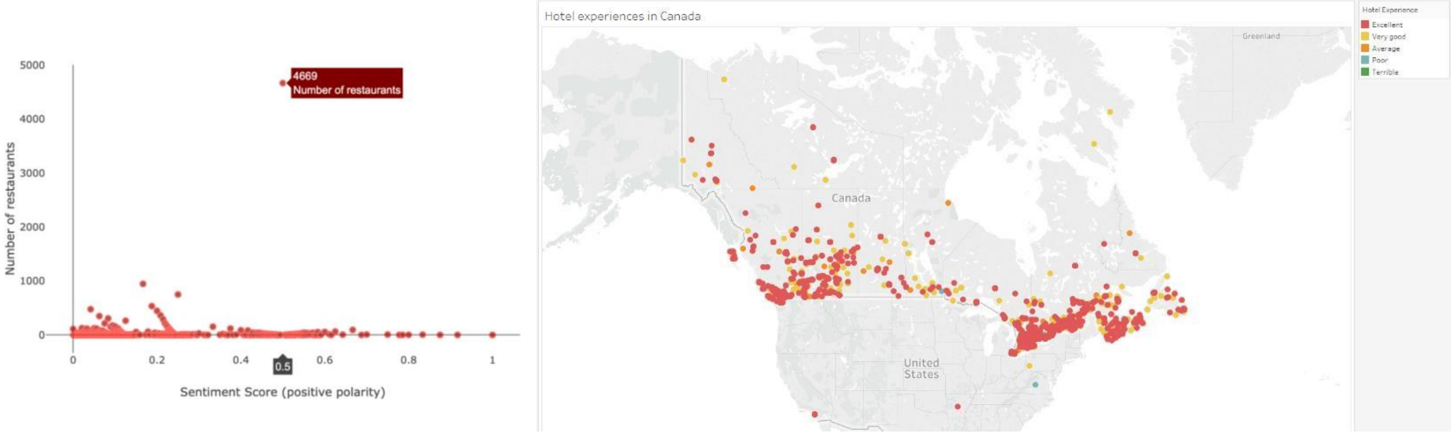
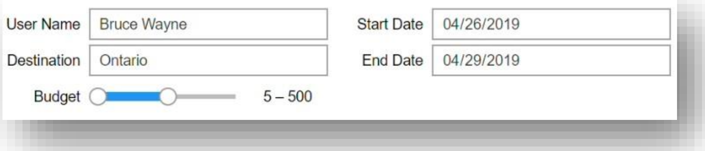


Figure 2 Few Visualizations we created during EDA

3.4 Input and User Profiling:



User Name: Bruce Wayne

Destination: Ontario

Budget: 5 – 500

Start Date: 04/26/2019

End Date: 04/29/2019

Figure 3Travel Details of user

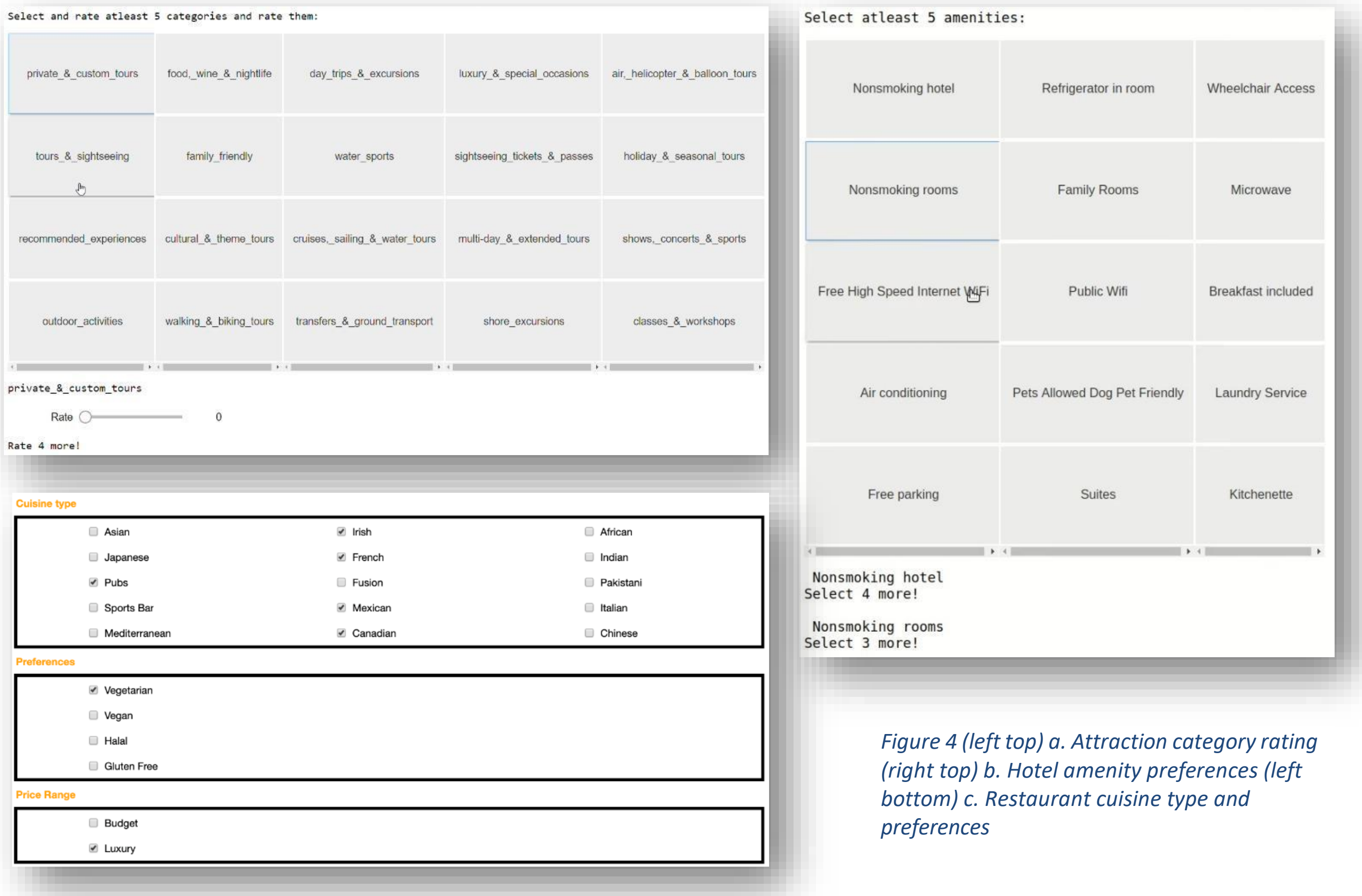
To provide the travel plan to the user, we obtain information about user's travel dates, their budget and their attraction, hotel and restaurant preferences. One of the major challenges in our project was to create a profile for user based on his preferences, A very common problem in providing recommendations to users is the **cold-start problem**. That is, providing recommendation to a new user who does not exist in the database. For a new user, there is no information on what he likes to provide him with recommendations. If we have a website, we can take into account the user's interaction with the website to provide implicit recommendations, but even then, there is no possibility of providing him recommendations he will like before his interactions. In order to solve this problem, we need to profile the user understanding his preferences. Since we have 3 categories of recommendations and we used 3 different for the same, we had to use different user profiling techniques:

i. *Hotels*: From EDA, we understood that hotel amenities is the major feature to consider for providing recommendations. We exploded the list containing amenities of a hotel and found top 15 amenities that most hotels have. The user was provided with these amenities to choose from and based on the amenities he chose, hotels were rated on a scale of 5 with rating 5 indicating the hotel containing maximum of user's required amenities. A 10% of this dataset was randomly sampled and combined with the original dataset containing ratings and reviews on hotels. The combined dataset was then passed to the model for training. This user profiling enabled us to match the new user with the closest user in the existing database who had liked similar hotel preferences.

ii. *Attractions*: Travelers have unique preferences in visiting different categories of attractions. For example, a travel group might like to visit beaches, water and adventure sports. Another group of travelers might like to visit museums, parks and sightseeing. This observation was leveraged to perform user profiling for Attraction recommendations. 20 Most frequent categories of tours were taken and provided as options to users. User was asked to select and rate at least five of the provided categories. This information then used to match the new user with a user in existing database sharing similar preferences, in essence, rated attracted categories in similar fashion. The recommendation obtained for the matched user in the database was provided to the new user as his attraction recommendations.

iii. *Restaurants*: People differ widely! For restaurant based user-profiling, we consider the cuisine type, budget and food preferences (like- vegan, gluten-free, etc,.) of the users. We give them a wide variety of popular cuisine types to choose from (15 different options). This information is then used to build the user-profile for an unknown user and three different personalized recommendations lists are provided, one each for each meal of the day. The closest item (restaurants) match to the attributes provided by the user is recommended.

After User profiling, the models are trained and evaluated to obtain best performing models. Then the recommendations are filtered and integrated into an **Interactive IPython Notebook**.



```
Select and rate atleast 5 categories and rate them:
```

private_&_custom_tours	food,_wine_&_nightlife	day_trips_&_excursions	luxury_&_special_occasions	air,_helicopter_&_balloon_tours
tours_&_sightseeing	family_friendly	water_sports	sightseeing_tickets_&_passes	holiday_&_seasonal_tours
recommended_experiences	cultural_&_theme_tours	cruises,_sailing_&_water_tours	multi-day_&_extended_tours	shows,_concerts_&_sports
outdoor_activities	walking_&_biking_tours	transfers_&_ground_transport	shore_excursions	classes_&_workshops

```
private_&_custom_tours
```

Rate 0

```
Rate 4 more!
```

```
Select atleast 5 amenities:
```

Nonsmoking hotel	Refrigerator in room	Wheelchair Access
Nonsmoking rooms	Family Rooms	Microwave
Free High Speed Internet WiFi	Public Wifi	Breakfast included
Air conditioning	Pets Allowed Dog Pet Friendly	Laundry Service
Free parking	Suites	Kitchenette

```
 Nonsmoking hotel
Select 4 more!

 Nonsmoking rooms
Select 3 more!
```

Cuisine type

□ Asian	☑ Irish	□ African
□ Japanese	☑ French	□ Indian
☑ Pubs	□ Fusion	□ Pakistani
□ Sports Bar	☑ Mexican	□ Italian
□ Mediterranean	☑ Canadian	□ Chinese

Preferences

☑ Vegetarian
□ Vegan
□ Halal
□ Gluten Free

Price Range

□ Budget
☑ Luxury

Figure 4 (left top) a. Attraction category rating (right top) b. Hotel amenity preferences (left bottom) c. Restaurant cuisine type and preferences

4. Model Training Methodology and Evaluation:

As mentioned earlier, 3 different models were used to obtain the recommendations. In this section, we will discuss about the models used in detail, the input for the model, training and evaluation of the model.

i. Matrix Factorization using Alternating least Squares technique (MF-ALS):

Matrix factorization is one of the most used Collaborative Filtering technique after Netflix Prize Challenge. Matrix Factorization uses the user-item interaction matrix to learn the embedding among the interaction to predict the ratings that will be given by users to unseen/ unused items. Alternating Least Squares (ALS), a matrix factorization algorithm, runs in a parallel fashion. It runs the gradient descent of multiple partitions of training data in parallel. It is also highly scalable and thereby used for large collaborative filtering tasks. It is simple and deals with solving the sparseness issue of the ratings data. Also, ALS can be used in distributed file systems.

INPUT: The hotel data containing information about user, hotel and user's rating for the hotel was used as input for training the MF-ALS model. The dataset also contains 10% sample of the new user data created using his preferences of hotel amenities. The remaining 90% sample used to obtain recommendations for new user.

PARAMETERS: The ALS model was trained with three different ranks (i.e.) 4, 8, 12. Ranks are also known as the latent factors. Latent factors are the features projected from user-item interaction in the lower dimension latent space. To represent user preferences in lower dimension space, latent factors are used. The ALS model was trained with three different ranks, maximum number of iterations as 5, regularization parameter as 0.01 and three attributes namely user id, hotel id and the rating given by user for the hotel.

EVALUATION: Root Mean Square Error (RMSE) was calculated for each case with different ranks and the model with lowest RMSE was used for recommendations. RMSE is calculated by finding the difference between predicted and existing rating, squaring them and then finding the square root of that value. The model with least RMSE denotes that the user-item interaction embedding has been learnt better and the model is able to predict user ratings to unseen items more accurately.

RECOMMENDATION: The saved model was used for predictions. Based on user preference of amenities, ratings were provided to hotels. 10% of the sample data containing user preferences was combined with the actual training data. Remaining 90% of the sample data was used for predictions. The data used for predictions contains the hotels not visited by the user. The trained ALS model is used to predict the ratings that the user will give for the hotels they haven't visited.

ii. Restricted Boltzmann Machine (RBM):

Boltzmann machines (BM) are stochastic and generative neural networks capable of learning internal representations and are able to represent and solve difficult combinatorial problems. RBM's are non-deterministic Deep Learning models with only two types of nodes - **visible** and **hidden** nodes. A regular Boltzmann machine has connections between all hidden nodes and visible nodes and amongst themselves. **Restricted Boltzmann Machine** is a special class of BMs and is restricted in terms of the connection within the same category of nodes and is fully bipartite. There are two other layers of bias units (hidden bias and visible bias) in an RBM. The hidden bias of RBM produces the activation on the forward pass and the visible bias helps RBM to reconstruct the input during a backward pass. The reconstructed input is always different from the actual input as there are no connections among the visible units and therefore, no way of transferring information among themselves. The model is trained to achieve the input after the reconstruction phase.

The inputs are multiplied by weights and hidden bias is added to them. This activation is then passed into a **sigmoid** activation function. This is the **forward-pass**. During the **backward-pass** or **reconstruction phase**, the activation function outputs act as the new input, multiplied by weights, added to the visible bias and then passed through the sigmoid activation function. The output of the reconstruction phase is then compared with the input to the forward-pass to calculate reconstruction error. The model is trained by reducing the reconstruction error using a gradient-based **Contrastive Divergence** algorithm. Contrastive Divergence is the difference between two Kullback-Leibler divergences of the input and reconstructed output. The algorithm tries to bridge the gap between the input and the reconstructed output while updating the weights to do so.

INPUT: The attractions dataset was used to create the user-item interaction matrix, also known as **rating matrix.** The rating matrix contains *n* rows for users and *m* columns for each attraction present in the dataset. The ratings provided by the user *i* for attraction *j* was filled at (*i,j*) position of the matrix. The positions without any user-item interaction was filled using a 0 rating. To avoid overfitting, the dataset was split for training and validation and each row of the matrix was then passed to the visible layer of the RBM model for training with weights, hidden bias and visible bias initialized randomly.

PARAMETERS: The model was trained for different combinations of parameters including number of epochs, learning rates, batch size, number of rows for training and number of hidden units. Trained best model for every combination tried was store for future usage.

EVALUATION: The model performance was evaluated using the Contrastive Divergence algorithm and the model with the best performance, i.e., least divergence between input and reconstructed output at visible layer was saved. The model with lesser divergence from output suggests that RBM has learnt well the embeddings between user and the attraction interactions so that it can replicate the input row.

RECOMMENDATION: The saved model was used to obtain recommendations for all the users in the existing dataset. Then the traveler is matched with the closest user in the dataset based on attraction category preferences and the recommendations are provided obtained for the traveler.

iii. K-Means + Memory based Collaborative Filtering Hybrid technique:

We have used K-Means as content based filtering technique and K-Nearest Neighbors as a memory based Collaborative Filtering technique **.K-Means** is a clustering technique that uses Euclidean distance in vector-space to measure the proximity of two points (in our case two different restaurants). The algorithm learns to classify the features in vector-space based on the similarity of input features into different clusters, which will be a M x N matrix, where M denotes the number of items and N denotes the number of features. It's an unsupervised machine learning technique as the algorithm does not have truth values (labels) to learn from and correct itself. Being a Content Based

Filtering Technique, K-Means does not suffer from cold start problem, but is highly sensitive to outliers in the dataset and dimensionality reduction techniques can help make the algorithm more efficient. Thus, we carefully performed EDA to analyze the restaurants dataset, we computed Positive-Negative sentiment ratio per restaurant based on user reviews. This helped weigh down the fact that the dataset was highly skewed towards positive reviews and tips. Also, we performed Principal Component Analysis to reduce the dimensionality, however, we did not scale our features as our input features did not have a linear relationship- scaling these would result in losing important semantic similarities in words. In-order to find the number of components apt for our dataset using PCA, we plotted the variance in features with respect to the number of components and chose the smallest number of components for which the variance was not less that 95%.

KNN, for our project, is used as a classifier that helps classify the user-item interaction matrix based on ratings provided. But, unlike the other two techniques, this is an Item-Based collaborative filtering technique as it predicts ratings for unknown users based on items that have obtained similar ratings to the one user is looking for rather than finding a user who is a close match to the current user. This is advantageous as it scales much better than the User-based Collaborative Filtering technique. It scales better because Item features tend to be more stable over user features, user taste changes more dynamically and hence, is very hard to rely on. But, a drawback is that Item-Based techniques will suffer due to the Curse of Dimensionality. Also, these do not suffer from cold-start problem like User-based Collaborative Filtering techniques.

INPUT:

For content based approach-the restaurants are filtered based on destination before running K-Means on the dataset. Categories, Attributes, Sentiment score, Star rating, City and State were used as the input features for K-Means algorithm. The Word2Vec model was trained for the vocabulary of words we have obtained after feature extraction and was saved for future use. Also, we had concatenated the input features into a single sentence, thus we had to average out the Word2vec vector obtained per word for each sentence (one sentence per restaurant in the dataset). In-order to obtain three different recommendations for different meals of the day, we obtained predictions for three different inputs from K-Means simultaneously with appropriate keywords to indicate the meal of the day(breakfast, lunch or dinner) for which we are seeking the recommendation.

For KNN algorithm, the restaurants dataset was used to create the user-item interaction matrix, also known as **rating matrix.** The rating matrix contains *n* rows for users and *m* columns for each restaurant present in the dataset. The ratings provided by the user *i* for attraction *j* was filled at (*i,j*) position of the matrix. The positions without any user-item interaction was filled using a 0 rating. An inference data was created using the user preferences aka item attributes provided to us. The ratings used here is not the actual user rating, it is a weighted sum of user ratings and user defined labels for the item (is the rating useful? is it funny? is it cool?).

PARAMETERS:

The hyper-parameter K for K-means algorithm represents the optimal number of clusters into which the dataset could be divided into. In-order to find the best K value, we iteratively ran K-Means on the entire dataset for two to fifteen times and each time calculated the inertia and silhoutte score. The inertia is used to plot the Elbow Curve in the Elbow method while Silhoutte score can help us find the best K via the Silhoutte analysis. They are explained in the next section.

KNN takes in the input sample and labels to train on with a parameter to choose the weights you want to assign during prediction to the points and a parameter to say how many recommendations (K) aka nearest neighbors of a point you want the algorithm to predict. Also, you can choose the algorithm and metric that you want to use. We used brute force KNN and cosine similarity as the metric respectively.

EVALUATION: For evaluating K-Means both the Elbow Method and Silhoutte Analysis were performed. The best K value (number of clusters) was obtained based on these metrics.

Elbow Method:

The Elbow curve at a point will have a sharp change, that is the point in the curve (in increasing k value) after which the decrease in Inertia with respect to K isn't substantial gives us the optimal K value. At this optimal value, the curve looks to have an Elbow shape. The inertia basically gives us the intra cluster point density, that is how closely knit is a cluster. So smaller the inertia value the better, as it indicates that the clusters are well formed and are tightly packed with points aligned close to each other within the cluster.

Silhoutte Analysis:

The value ranges between -1 to 1. The higher the value the better it is. The Silhoutte score essentially tells us the inter-cluster separation area or distance. Thus, a higher value means better closely knit clusters have been formed as the inter-cluster distance is significantly high, which in turn says we have an optimal K value(cluster split).

RECOMMENDATION: For predictions on new points (new users), we use K-means to assign a cluster to the new point based on the user preference. Then, we find the points within the assigned cluster that are closest to our new point using 'Cosine Similarity'. We go for cosine over euclidean as the input features in our case are word embeddings obtained from Word2Vec, whose semantics will be better captured by using cosine similarity (euclidean will calculate the meare difference in letters of the words).

The final recommendation list is a weighted average (based on rating)of the recommendations obtained from the content-based (K-Means) technique and the Memory-Based Collaborative Filtering technique(K-Nearest Neighbors).

5. Output and Data Product

The data product for the project was an Interactive IPython Notebook created using **ipywidgets** – an Ipython library for creating widgets in a notebook. Inputs from traveler containing travel details, budget, their hotel, attraction and restaurant preferences were obtained through interactive widgets as shown in figures 4a, 4b, 4c. The final recommendation for each category of recommendation was handled differently and finally integrated into one single **Data Product**.

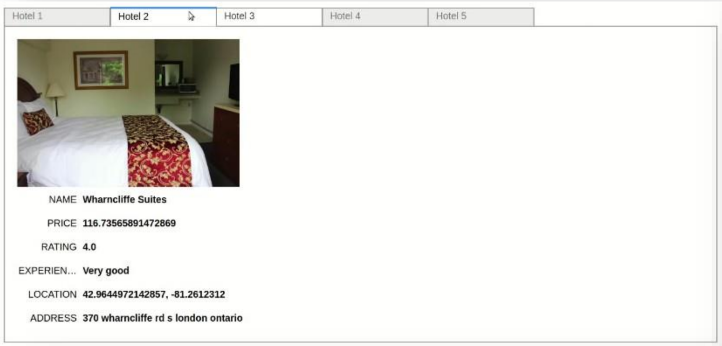
i. *Hotels*: After obtaining recommendations from the Matrix Factorization model, the result was filtered based on budget and location. The price was multiplied for the number of days of stay and result was sorted and filtered. Details of 5 hotels with top recommendation scores were provided to the traveler as Hotel recommendations to choose from. The result was consistent with the user's amenity requirements. The final output for hotel recommendation is shown in figure 5a.

ii. *Attractions*: The result from RBM was filtered based on location and budget. Challenges in providing this recommendation was taking into account distance between attractions and the time of day to visit the attraction. Few attractions and tours will be better for morning visits and few tours like night trekking or watching a sunset would be better to go for in the evening. To solve this problem, NLTK's Wordnet was used to find the synonyms of evening and night, and these keywords were checked for in the name of the attraction. This enabled us to distinguish **Morning** and **Evening** attractions. Secondly, a person going to an attraction in North Vancouver would not want to travel all the way to Surrey to visit the second attraction. Hence, distance between consecutive visits for attractions had to be considered. The top recommendation with highest score was chosen, distance was calculated between every attraction to our selection and the closest attraction with next highest recommendation score was chosen for next recommendation. The list of recommendations provided were kept track of so that the recommendations were not repeated. This process repeats for all the days for which the traveler will be staying at the destination and the final output will be provided as shown in figure 5b.

iii. *Restaurants*: After obtaining recommendations (weighted average of the both the algorithms output) using the Hybrid technique, the results are de-duplicated to hold only distinct restaurants for the three different meals (Breakfast, Lunch and Dinner) of a day for the entire duration of travel. It's also sorted based on the star rating of the recommended restaurant. We recommend two restaurants per meal per day as in figure 5c.

Major Challenges in building the Data product:

(i) The images of hotels and restaurants provided in the final output could not be scraped from the website. Hence, google_images_download – a python library for downloading google images was used to obtain images for recommendations based on their name. **(ii)** Using ipywidgets for interactive output in jupyter notebook and to parse HTML/CSS commands to provide final output. **(iii)** Efficiently integrating all three recommendations to provide a single final **Data Product.**



NAME Wharncliffe Suites

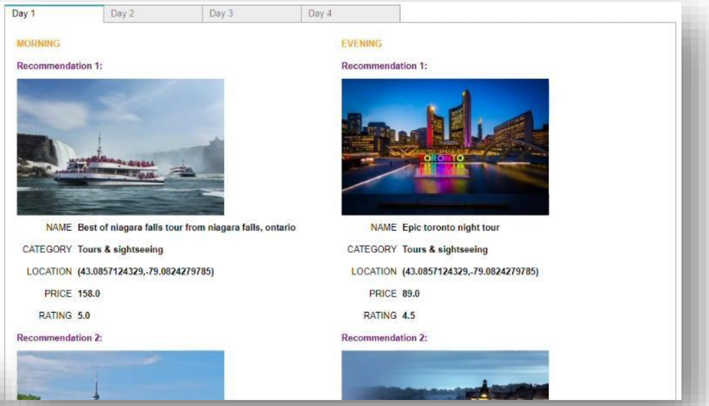
PRICE 116.73565891472869

RATING 4.0

EXPERIEN... Very good

LOCATION 42.9644972142857, -81.2612312

ADDRESS 370 wharncliffe rd s london ontario



MORNING

Recommendation 1:

NAME Best of niagara falls tour from niagara falls, ontario

CATEGORY Tours & sightseeing

LOCATION (43.0857124329,-79.0824279785)

PRICE 158.0

RATING 5.0

Recommendation 2:

EVENING

Recommendation 1:

NAME Epic toronto night tour

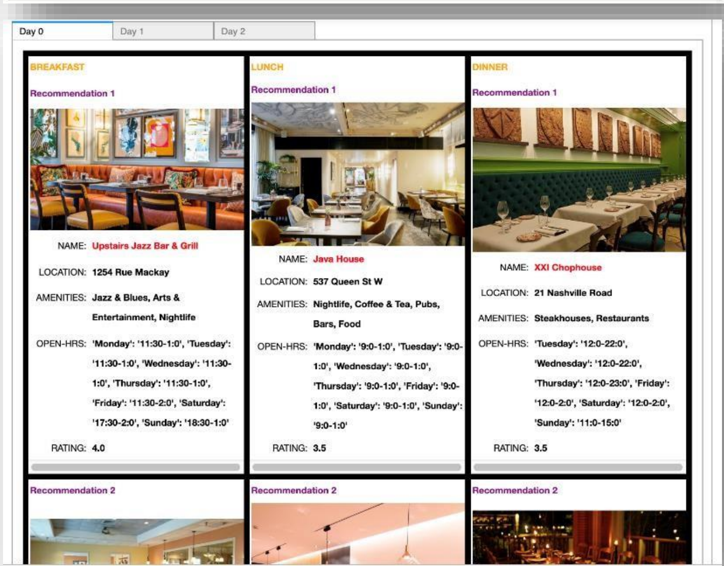
CATEGORY Tours & sightseeing

LOCATION (43.0857124329,-79.0824279785)

PRICE 89.0

RATING 4.5

Recommendation 2:



BREAKFAST

Recommendation 1

NAME: Upstairs Jazz Bar & Grill

LOCATION: 1254 Rue Mackay

AMENITIES: Jazz & Blues, Arts & Entertainment, Nightlife

OPEN-HRS: 'Monday': '11:30-1:0', 'Tuesday': '11:30-1:0', 'Wednesday': '11:30-1:0', 'Thursday': '11:30-1:0', 'Friday': '11:30-2:0', 'Saturday': '17:30-2:0', 'Sunday': '18:30-1:0'

RATING: 4.0

Recommendation 2

LUNCH

Recommendation 1

NAME: Java House

LOCATION: 537 Queen St W

AMENITIES: Nightlife, Coffee & Tea, Pubs, Bars, Food

OPEN-HRS: 'Monday': '9:0-1:0', 'Tuesday': '9:0-1:0', 'Wednesday': '9:0-1:0', 'Thursday': '9:0-1:0', 'Friday': '9:0-1:0', 'Saturday': '9:0-1:0', 'Sunday': '9:0-1:0'

RATING: 3.5

Recommendation 2

DINNER

Recommendation 1

NAME: XXI Chophouse

LOCATION: 21 Nashville Road

AMENITIES: Steakhouses, Restaurants

OPEN-HRS: 'Tuesday': '12:0-22:0', 'Wednesday': '12:0-22:0', 'Thursday': '12:0-23:0', 'Friday': '12:0-2:0', 'Saturday': '12:0-2:0', 'Sunday': '11:0-15:0'

RATING: 3.5

Recommendation 2

Figure 5(left top) a. Hotel Recommendations (right top) b. Attraction recommendations (left bottom) c. Restaurant Recommendations

6. Learnings

i. Gained experience in learning and implementation of 3 different recommendation algorithms – RBM, Matrix Factorization using ALS and a hybrid recommendation algorithm using K-means and memory-based Collaborative Filtering (KNN).

ii. Understanding data requirements of the project and collecting the data required to create the recommender system.

iii. Importance of Exploratory Data Analysis for understanding the data was realised and carefully performed.

iv. Explored options to create a User Interface using Jupyter Notebook. Explored ipywidgets, an IPython library for creating interactive widgets in Jupyter Notebook.

v. How to communicate our findings and project to non-technical people as well through verbal, written and video communication.

7. Conclusion and Future Work

The project's aim was to provide tailor-made travel plans for users based on their travel details and preferences. The aim of the project was successfully accomplished by exploiting various Data Science tools and techniques and the recommendations were provided to the user. The scope of the project is really wide and hence there is a lot of room for improvement and future work. Few suggestions are as follows:

i. The data currently collected for the project was restricted to Canadian hotels, attractions and restaurants and recommendations are provided for only Canadian vacations. This can be scaled for multiple travel destinations around the world and recommendations can be provided for various parts of the world.

ii. The data was scraped from tripadvisor website. There are several other websites that can contain more relevant information on the items and hence can be used.

iii. The project can be extended by creating an interactive web application to provide recommendations instead of using interactive ipython notebook to get information from user. Using the web application, interactions of users can be collected to improve the user experience and provide implicit recommendations.

iv. The recommendations provided can also be improved by taking into account the number (solo or a group), type (professionals, friends, family with or without kids). For example, having kids during the travel, we can eliminate under 18 recommendations and emphasize on suggesting kids-friendly places like theme parks.

v. In our project, three recommendation algorithms were used. Various other recommendation algorithms can be used and fine-tuned to improve the accuracy of the recommendations provided. For instance, Deep Belief Networks or Auto-encoders can be used instead of RBM to improve the recommendations. There are carious ensemble recommendation algorithms that can be exploited to provide highly accurate recommendations.

Project Summary

CATEGORY	WHAT WE DID	TECHNOLOGIES USED
Problem Statement	A highly significant real-life problem was taken into account and a very useful solution was provided.	
Data Collection	YELP dataset used for restaurants data, TripAdvisor website was scraped for hotels and attractions data.	**lxml, requests, GeoPy, Google API**
Bigness	34K reviews for 3.5K attractions, 2.4M reviews for 35K hotels, 5M reviews for 12K restaurants	
EDA & Visualization	Performed EDA to understand distributions and correlations between different attributes in data.	**Plotly, Tableau, Seaborn**
ETL	Cleaned and integrated data from different sources to transform the data into more usable form.	**Spark, Pandas**
Algorithmic work	Performed NLP operations on reviews, efficiently combined the results of k-means and KNN algorithms to provide a final (weighted average of both the techniques output) list of restaurant recommendations, used spark's MLlib to train MF-ALS, Tensorflow to train RBM and get required results	**Sklearn, Spacy, NLTK, Tensorflow, Spark MlLib**
User Interface & Output	The final data product was a jupyter notebook output with interactive widgets and output	**IPython, IPywidget, HTML, google_download_images, CSS**