# Travel Planner Based on User Preferences

Atish Ghosh
IIITD
atish21018@iiitd.ac.in

Neha Rewari
IIITD
neha21055@iiitd.ac.in

Rishabh Gupta
IIITD
rishabh21070@iiitd.ac.in

Seema Anjum
IIITD
seema21078@iiitd.ac.in

Shrey Rastogi
IIITD
shrey21145@iiitd.ac.in

## 1 ABSTRACT

The objective of our project is to generate an intelligent travel plan for the user, that will include tourist locations based on user's city of choice, travel preferences, number of days and reviews of other tourists; and it will also provide him/her with an optimized schedule of visiting those places.For city and places mapping we are going to use Indian Places Reviews[1] data-set available on Kaggle, this data-set with city and corresponding places in city also include reviews by different user, with the help of NLP tasks we will map user preferences with reviews and then further process choices of places with k-means clustering algorithm, and then finally generate a day-wise itinerary for the user by using Traveling Salesman Algorithm on top of it.

## 2 INTRODUCTION

Tourists often consult with travel agencies or travel websites on the internet, while planning their trips. However, most of these sources provide recommendations based on old, static data. Our idea is to utilize the user's preferences and reviews of other tourists who recently visited those places, to make a curated list of tourist hotspots that fit the user's choice.

Also, travel agencies and websites usually provide a fixed schedule of the static travel plans to the user, but our schedule dynamically adapts to the travel preferences of the user, including the number of days he/she has planned the trip for; and this schedule also optimizes the time and distance parameters, and is not restricted to some static choices unlike in the former scenario.

## 3 LITERATURE REVIEW

### 3.1 Recommendation System Based on Tourist Attraction [P.A. Manjare et al., 2016][2]:

The research paper built a recommender system to list out city-wise tourist places and nearby hotels. Data Mining techniques were applied. Also, a Collaborative filtering technique was used to recommend tourist locations, based on user profiles.

### 3.2 Mining Social Media to Create Personalized Recommendations for Tourists' visit [Popesco Adrian et al., 2011][3]:

In this paper, the author used geotagged photos and data from Wikipedia articles. The personalization model has been designed from the user's tagging experience and also the experiences of similar users. Landmarks' popularity along with the above criteria have been carefully considered to generate recommendations based on the modified KNN algorithm.

### 3.3 A Travel Recommender System for Combining Multiple Travel Regions to a Composite Trip [Daniel Herzog and Wolfgang Worndl, 2014][4]:

This paper considers the travel recommendation task as an approximated Knapsack problem, where profit involves parameters like location ratings, activities etc. and the knapsack weight involves parameters like time and money. Then the tourist locations are suggested based on optimizing the Knapsack by getting the highest profit value.

### 3.4 From Photos to Travel Itinerary: A Tourism Recommender System for Smart Tourism Destination [Mickael Figueredo et al., 2018][5]:

This research paper built a module that can suggest tourist places on the basis of social media photos and recommends a set of tourist destinations using Convolution Neural Networks and Fuzzy logic on the tourists' visited locations.

### 3.5 User-based Collaborative Filtering for Tourist Attraction Recommendations [Zhiyang Jia et al., 2015][6]:

This paper analyzes the travel history of users and generates neighbors based on similarity with other users, using the collaborative filtering method. Based on this, whenever users login, tourist places are recommended based on the updated visit history of his/her neighbors.

, ,

ACM ISBN . . . $

### 3.6 ATIPS: Automatic Travel Itinerary Planning System for Domestic Areas [Yi-Ming Chang and Meng-Tze Tsai, 2016][7]:

In this paper Authors design a framework that takes departure and destination address as input and then framework generate nearby tourist spots, it uses "Travel scheduling algorithm" and "Tourist spot selection algorithm" with some greedy approaches to select tourist spot and then generate travel plan matching the preferences.

### 3.7 How Airbnb Tells You Will Enjoy Sunset Sailing in Barcelona? Recommendation in a Two-Sided Travel Marketplace [Liang Wu, Mihajlo Grbovic, 2020 Airbnb][8]

In this paper Authors proposes solution of host and guest side recommendation system through automatically knowledge graph building using web search trends based on location and user profiles, and recommend places with limited data availability ,Their approaches outperform traditional offline recommendation methods.

### 3.8 Online Travel Planner using Sentiment Analysis [Shivam Singh., 2020][9]

In this paper Authors used Naïve Bayes classifier and evaluation with ten-fold cross validation on a small dataset to perform sentiment analysis to get the reviews of the hotels present in Chennai. Reviews were broadly divided into two categories 'Good' or 'Bad'.

## 4 DATASET CREATION

### 4.1 Training Data Pre-processing

- The size of the dataset is 1482466 rows × 7 columns. Columns are City, Place, Review, Raw Review, Rating, Name, Date. Name and Date are noise and are NAN values hence they are dropped. 'Raw Review' has users' reviews and the 'Review' column has tokens with stop words removed. The review column is also dropped as we would be performing our own preprocessing.

| | Place | City | Raw_Review | Rating | new_lat_long |
|---|---|---|---|---|---|
| | 3Cs Mall | New Delhi | ['better', 'ordered', 'gut', 'pre', 'strong', 'see', 'made', 'powder', 'idli', 'selling', 'indian', 'steel', 'billed', 'filter', 'jaws', 'making', 'idlies', 'south', 'coffee'] | 4.000000 | [28.5658122,77.2468466] |
| | Aai Museum | Mumbai | ['lovers', 'two', 'brave', 'nice', 'pilaji', 'sections', 'top', 'floor', 'art', 'jadhavrao', 'museum', 'oil', 'place', 'spot', 'ground', 'portrait', 'jewellery', 'one', 'four', 'first', 'maratha', 'lamps', 'galleries'] | 5.000000 | [19.0964017, 72.86293309999999] |

**Figure 1: Sample Data-Set**

- The total number of cities in the data has been found to be 1794 and the total unique places within those cities are 14494. Due to computational limitations, we have taken nine cities which are Delhi, Mumbai, Kolkata, Bengaluru, Hyderabad, Jaipur, Udaipur, Agra, and Pune. Moreover, we had to discard those places corresponding to these cities, whose latitude and longitude was not returned by the geopy module.

- Some of the unusually long reviews were shortened by using the pysummarization module.

- Figure 1 shows the sample dataset after preprocessing.

- The rows corresponding to each and every place of a particular city, are merged. So, we finally have 9 datasets pertaining to each city. Figure 2 describes the data corresponding to each city.

| Dataset belongs to this City | Rows (Unique places) | Columns |
|---|---|---|
| Delhi | 193 | 5 |
| Mumbai | 164 | 5 |
| Bengaluru | 118 | 5 |
| Pune | 83 | 5 |
| Hyderabad | 77 | 5 |
| Kolkata | 76 | 5 |
| Jaipur | 34 | 5 |
| Agra | 26 | 5 |
| Udaipur | 22 | 5 |

**Figure 2: City Dataset sizes**

## 5 PROPOSED METHODOLOGY

- The user selects the city he/she wishes to visit and enters a description of his travel preferences in a word-limited text field.

- The preferences entered by the user, are passed through a spell-checker (imported from TextBlob) to correct any spelling mistakes.

- Travel locations pertaining to the city preferred by the user, are selected from the datasets, and reviews corresponding to user preferences are filtered out.

- As of now, we have assumed that there are 10 places that the user wants to visit across input number of days.For clustering we have shown the results for top 25 relevant places.

- After getting these sets of relevant places we are performing clustering based on the latitude and longitude of the places so that our nearest places are grouped together.

- The shortest route-finding algorithm(Travelling Salesman Algorithm) is applied to the cluster centers to get the day-wise schedule of the places to be visited.

### 5.1 Model 1: Clustering using Jaccard similarity

The first baseline model is K-means using Jaccard Coefficient. We have filtered out places based on the top scores computed. We have taken out the top 10 places. For performance evaluation, we have prepared a benchmark dataset,

considering Google's recommendation. We have taken the top 10 places based on Google's recommendations in the benchmark dataset.
Metric Used is Precision. For calculating precision, we have taken out common places between benchmark values and predicted values and divided them by 10. The average precision was found to be 10. These places are then clustered using the K-means clustering algorithm as the task which we will be performing is unsupervised learning. The no. of days are same as the no of the cluster centroids, with relevant places as cluster data points. Architecture is shown in Figure 3. Scatter plot for model 1 cluster is shown in Figure 4. The day-wise sequence of clusters have been shown in Figure 5, which is computed based on the Traveling Salesman Algorithm. Locations corresponding to each cluster, in the same sequence as Figure-5, is shown in Figure 6.

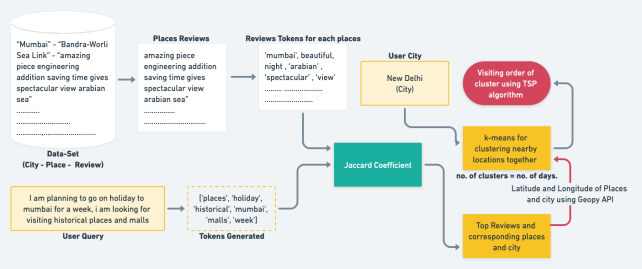


**Figure 3: Model-1 : Architecture**

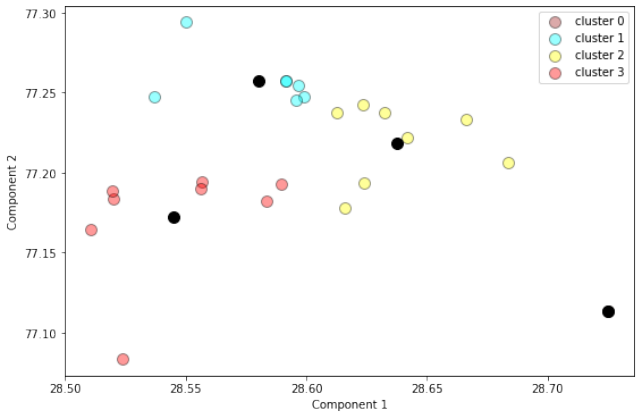


**Figure 4: K-Means clustering for Model-1**

```
[0, 2, 1, 3]
```

**Figure 5: Day-wise sequence of clusters for Model-1**

```
Travel locations for day 1
['Adventure Island' 'Metro Walk Mall']
Travel locations for day 2
['Ajmeri Gate' 'Buddha Jayanti Park' 'Delhi University Sports Complex'
 'Major Dhyan Chand National Stadium' 'Mutiny Memorial'
 "National Children's Museum" 'Pragati Maidan' 'Talkatora Garden']
Travel locations for day 3
['Chittaranjan Park Kali Mandir' 'Delhi Rides' 'Millennium Park'
 'National Zoological Park of Delhi' 'Sunder Nursery'
 'Waste to Wonder Park' 'Waste to Wonders']
Travel locations for day 4
["Children's Park" 'Deer Park' "Fun 'n' Food Village"
 'Hauz Khas District Park' 'Nehru Park' 'Rajon Ki Baoli'
 'Swarn Jayanti Park' 'Tomb of Balban']
```

**Figure 6: Travel locations corresponding to the day-wise sequence of clusters for Model-1**

## 5.2 Model 2: Clustering using Cosine Similarity

The second baseline model is the TF-IDF model with 5 different weighing schemes. Then we have taken out cosine similarity with every row of the matrix and the top 10 places with the highest similarity scores are filtered out. Comparison with the benchmark dataset is done in the same way as in case of model 1. Architecture is shown in Figure 7.
Metric Used is Precision. For calculating precision, we have taken out common places between benchmark values and predicted values and divided them by 10. The average precision was found out to be 20.
Clusters with different weighing schemes have been shown in Figure 8. Traveling Salesman Problem algorithm was used to find the sequence of the clusters, which has been shown in Figure 9. Locations corresponding to each cluster, in the same sequence corresponding to Figure-9, is shown in Figure 10.

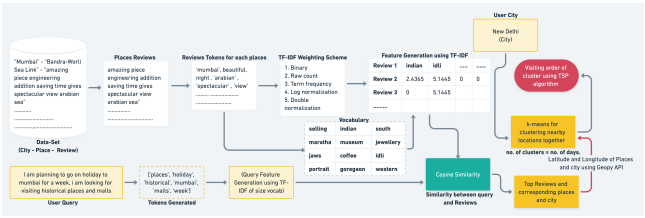


**Figure 7: Model-2: Architecture**

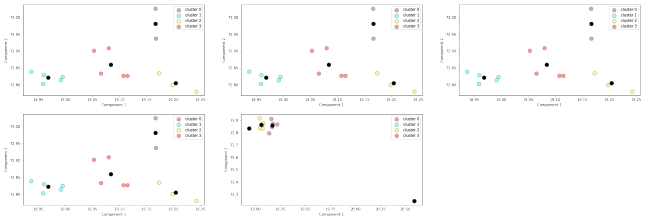

**Figure 8: K-means clustering for Model-2**

```
Visting order of clusters for 1 is [0, 2, 1, 3]
```

**Figure 9: Day-wise sequence of clusters for Model-2**

```
Travel locations for day 1
['Nirmal Lifestyle Mall' 'Prime Mall']
Travel locations for day 2
['Gorai Beach' 'Inorbit Mall' 'Madh Island']
Travel locations for day 3
['Atria Mall' 'Ballard Estate' 'Chor Bazaar' 'High Street Phoenix'
 'Malabar Hill']
Travel locations for day 4
['Dynamix Mall' 'Juhu' 'K Star Mall' 'Link Square Mall' 'R Odeon Mall']
```

**Figure 10: Travel locations corresponding to the day-wise sequence of clusters for Model-2**

## 5.3 Other models: BERT using Cosine similarity

BERT(Bidirectional Encoder Representations from Transformers ) is a pre-trained language model developed by Google which uses the context of words used in train data, to generate embeddings for text data. We have used the bert-base-uncased and bert-large-uncased models. bert-base-uncased uses a vector of size 768 and bert-large-uncased uses a vector of size 1024, to generate the embeddings. For bert-base-uncased, the embedding size corresponding to each city was (no. of places * 768). For bert-large-uncased, the embedding size corresponding to each city was (no. of places * 1024). Similarly, the embedding of an input query of size 768 and 1024 has been calculated. To find the places corresponding to the given query, the similarity between the input query embedding and the matrix rows' embeddings have been calculated using cosine similarity and the most similar places have been suggested. BERT is not giving desired output in our case.

## 5.4 Final Model: Vectorization using GloVe embedding

To train our final model we have used GloVe embeddings, GloVe takes overall global context into account to form word embeddings. The text summarizer has been used on the data to shorten the length of the reviews. We have tried two variants of GloVe. The first model is 'Document Pool Embeddings' and another model is 'Document RNN Embedding'. Document Pool Embedding takes the mean of pooling operation over all the words in a sentence.

We have taken an embedding of size 100 for each review. The matrix corresponding to each city has been calculated and stored. This matrix has been deployed to the server to manage the latency. At runtime, the embedding of size 100 will be generated from the User's query and the cosine similarity will be calculated with the matrix, to find the most relevant places corresponding to the user query. The Average of the final model on the test data we have used earlier is found to be 32.5. The final model architecture has been shown in Figure 11, its k-mean clustering has been shown in Figure 12, day-wise sequence for the final model has been shown in Figure 13 and the corresponding travel locations have been shown in Figure 14.

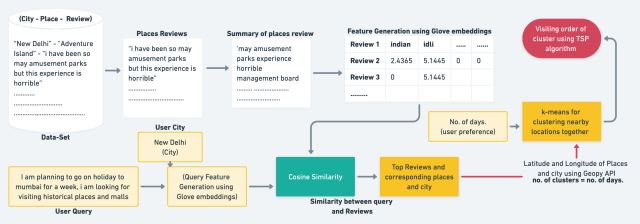


**Figure 11: Final-Model: Architecture**

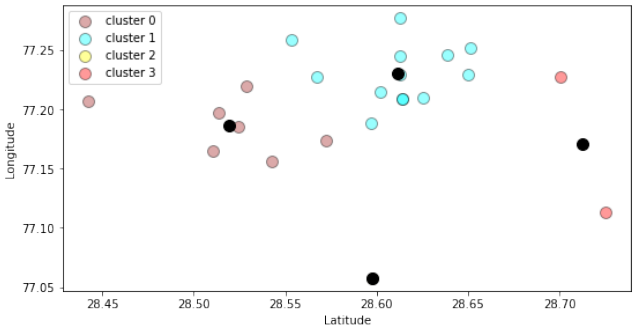


**Figure 12: k-means clustering for Final model**

```
Visting order of clusters for 2 is [[0, 2, 3, 1]]
Total distance travelled is [0.5312532806709462]
```

**Figure 13: Day-wise sequence of places for Final model**

```
Travel locations for day 1
['DLF Promenade Mall' 'Bade Mandir' "Children's Park"
 'Garden of Five Senses' 'Qutub Minar' 'Sangam Courtyard'
 'Select Citywalk']
Travel locations for day 2
['Castle And king']
Travel locations for day 3
['Adventure Island' 'Majnu ka Tilla']
Travel locations for day 4
['Art of Living Center' 'Chawri  Bazaar' 'Gandhi Smriti'
 'Gurudwara Bangla Sahib' 'India Gate' 'Lotus Temple'
 'National Gandhi Museum' 'National Science Centre' 'Raj Ghat'
 'Sai Baba Temple' 'Swaminarayan Akshardham' 'The Chanakya' 'Tumble House']
```

**Figure 14: Travel locations corresponding to the day-wise sequence for Final-Model**

## 6 LIMITATIONS

- The dataset was noisy and contained individual reviews of respective places, we transformed the data to the proper format.
- We tried to get locations based on 9 filtered cities' data using Google API, but as this was a paid API, we used Geopy API. This API doesn't provide locations of all the places within these cities. Some of the locations received were "None", also some of the latitudes and longitudes were incorrect. We have manually removed such noisy data for every city.
- Due to computational limitations, reviews corresponding to 9 cities have been taken, and the time to create a matrix corresponding to these 9 cities took around 50-60 minutes to complete.
- We tried to deploy our models corresponding to each city to the Heroku server, but due to the limited free size of (500 MB), we have switched to the Localhost. Our application was exceeding the free permitted size.

## 7 CONCLUSION

Among the 4 models built i.e is Jaccard Coefficient, TF-IDF, BERT, and the GloVe model, the GloVe model turned out to be the best model and avg precision was 32.5. For other models, avg precision was less than 30. For calculating precision, we have generated a test set of 4 queries that contain recommendations based on a google search. The GloVe model performed better than other models, since GloVe captures the context of queries and reviews. BERT and other models were not able to capture the context and gave less precision. The reason for low precision is due to the unavailability of latitudes and longitudes of various places present in source data so those places were not included in the dataset and thus are not a part of the recommendation. Coming to the second part of our project, i.e suggesting a Day-wise travel itinerary, this part remained the same among all the models. Clustering worked well for all the models, nearest places were clustered to be travelled in a single day. And TSP gives the order to travel these clusters.

## 8 REFERENCES

[1] https://www.kaggle.com/shravanijadhav23/indian-places-reviews/data
[2] RecommendationSystemBasedonTouristAttraction
[3] https://www.researchgate.net/publication/221449162_Mining_social_media_to_create_personalized_recommendations_for_tourist_visits
[4] https://www.researchgate.net/publication/281655048_A_travel_recommender_system_for_combining_multiple_travel_regions_to_a_composite_trip
[5] https://www.researchgate.net/publication/326277455_From_Photos_to_Travel_Itinerary_A_Tourism_Recommender_System_for_Smart_Tourism_Destination
[6] https://www.researchgate.net/publication/281698410_User-Based_Collaborative_Filtering_for_Tourist_Attraction_Recommendations
[7] https://www.hindawi.com/journals/cin/2016/1281379/#related-works
[8] https://dl.acm.org/doi/10.1145/3397271.3401444
[9] https://www.indusedu.org/pdfs/IJREISS/IJREISS_3663_39823.pdf