SMART CITY TRAVELLER

[1]Harshil Joshi, [2]Shivani Chavan, [3]Rinkal Patel, [4]Abdullah Patel

B. tech Student, B. tech Student, B. tech Student, B. tech Student,

[1234]Computer Science & Engineering

[1234]Parul University, PIET, Vadodara, India

Abstract : Smart City Traveller by the name indicated smartly makes it way in analysing user's likes and dislikes and the time period the user is willing to explore a place and gives him with Amazing results in the form where utilisation of time is maximum. This system is basically used to help a traveller new to a city or anyone who wants to explore a city in the given time period, the system makes use of the preferences of the user to get all the locations and places with all their information to sort and give a plan to the user. Thus we have used certain algorithms and google maps API to create this application.

Index terms – travel planning, google maps API, user requirements

I. INTRODUCTION

Travelling is one of the most common way to spend the holiday. Information about the travel destination, infrastructure, facility, event and promotion should be enhanced and easy to be retrieved in order to help the traveller in preparing the enjoyable trip. It is now believed that daily lives of today's people are influenced by the interest and social media. When a person is planning a trip, especially for first time it is necessary to provide trustworthy information through by developing a web application. The proposed web application is responsible for collecting, processing, sharing, storing and analysing the tourist behaviour.

II. PROBLEM DEFINATION

The world is moving forward in the field of technology and yet it is slow in the tourism industry. The people still are unknown about the smart tourism web application. It is observed that people spend more time in travel planning and it becomes very difficult to organize a simple and enjoying trip. Furthermore it is quite difficult in decision making of travelling spots, hotels, exploring places. A person if he/she is travelling first time is difficult for him/her to know the exact about the place. Person have to totally dependent on the local people suggestions and directions given. In a developing country there will be need of smart tourism beneficial for everyone.

III. PROBLEM SOLUTION

The solutions for above mentioned problems can be achieved by developing an smart city traveler web application which be beneficial for user in travel planning, decision making. A user will select a location and has to answer a predefined set of questions. On the basis of answers a machine learning algorithms will automatically make a travel plan and will also notify the recent updates about the place. The machine learning algorithms will predict the outcome on the basis of input data. Although when the user reaches the destination he/she will have to submit feedback which will be useful for new user or the same person going to different place. The web application will be free of cost and will be very simple to use. The maintenance is very easy and it requires only to update location details. Thus this application will save the enormous time of user and he/she will be able to make enjoying destination planning.

IV. WHAT ARE RECOMMENDATION ENGINE

A recommendation engine filters the data using different algorithms and recommends the most relevant places to users. It first captures the initial behaviour of a user and based on that, recommends places which the users might like.

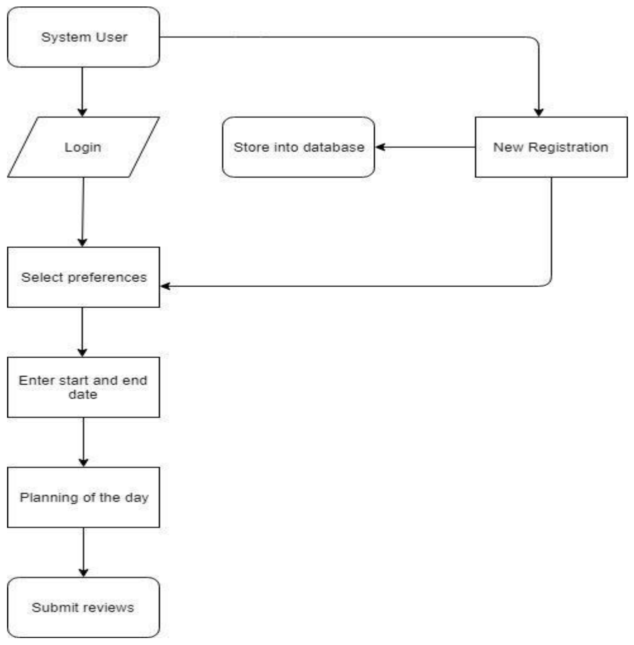
Before we deep dive into this topic, first we'll think of how we can recommend items to users:

- We can recommend places to a user which are most popular among all the users
- We can divide the users into multiple segments based on their preferences (user features) and recommend places to them based on the segment they belong to

Both of the above methods have their drawbacks. In the first case, the most popular places would be the same for each user so everybody will see the same recommendations. While in the second case, as the number of users increases, the number of features will also increase. So classifying the users into various segments will be a very difficult task.

V. WORKING METHODOLOGY

Initially the user will have to register to the web application and after that he/she will have to enter his preferred location and answer some predefined set of questions which will be used by system for making a travel plan.



Flow chart

Let's focus on how a recommendation engine works by going through the following steps.

V.I DATA COLLECTION

This is the first and most crucial step for building a recommendation engine. The data can be collected by two means: explicitly and implicitly. Explicit data is information that is provided intentionally, i.e. input from the users such as movie ratings. Implicit data is information that is not provided intentionally but gathered from available data streams like search history, clicks, order history, etc.

V.II DATA STORAGE

The amount of data dictates how good the recommendations of the model can get. For example, in a movie recommendation system, the more ratings users give to movies, the better the recommendations get for other users. The type of data plays an important role in deciding the type of storage that has to be used. This type of storage could include a standard SQL database, a NoSQL database or some kind of object storage.

V.III FILTERING THE DATA

After collecting and storing the data, we have to filter it so as to extract the relevant information required to make the final recommendations. There are various algorithms that help us make the filtering process easier.

V.III.I CONTENT BASED FILTERING

This algorithm recommends products which are similar to the ones that a user has liked in the past. The content-based filtering algorithm finds the cosine of the angle between the profile vector and place vector, i.e. **cosine similarity**. Suppose A is the profile vector and B is the place vector, then the similarity between them can be calculated as:

$$sim(A,B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

Based on the cosine value, which ranges between -1 to 1, the places are arranged in descending order and one of the two below approaches is used for recommendations:

- **Top-n approach**: where the top n places are recommended (Here n can be decided by the business)
- **Rating scale approach**: Where a threshold is set and all the places above that threshold are recommended

Other methods that can be used to calculate the similarity are:

- **Euclidean Distance**: Similar places will lie in close proximity to each other if plotted in n-dimensional space. So, we can calculate the distance between items and based on that distance, recommend places to the user. The formula for the euclidean distance is given by:

$$\text{Euclidean Distance} = \sqrt{(x_1 - y_1)^2 + \ldots + (x_N - y_N)^2}$$

- **Pearson's Correlation**: It tells us how much two places are correlated. Higher the correlation, more will be the similarity. Pearson's correlation can be calculated using the following formula:

$$sim(u,v) = \frac{\sum(r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum(r_{ui} - \bar{r}_u)^2}\sqrt{\sum(r_{vi} - \bar{r}_v)^2}}$$

A major drawback of this algorithm is that it is limited to recommending places that are of the same type. It will never recommend products which the user has not bought or liked in the past. So if a user has visited only historic places in the past, the system will recommend only historic places. It's a very narrow way of building an engine.

To improve on this type of system, we need an algorithm that can recommend places not just based on the type, but the behaviour of users as well.

V.III.II COLLABORATIVE FILTERING

Let us understand this with an example. If person A likes 3 places, say Manali, Shimla and Meghalaya, and person B likes Shimla, Ladakh and Manali, then they have almost similar interests. We can say with some certainty that A should like Ladakh and B should like Meghalaya. The collaborative filtering algorithm uses "User Behaviour" for recommending places. This is one of the most commonly used algorithms as it is not dependent on any additional information. There are different types of collaborating filtering techniques and we shall look at them in detail below.

USER COLLABORATIVE FILTERING

This algorithm first finds the similarity score between users. Based on this similarity score, it then picks out the most similar users and recommends places which these similar users have liked previously. In terms of our places example from earlier, this algorithm finds the similarity between each user based on the ratings they have previously given to different places. The prediction of an place for a user *u* is calculated by computing the weighted sum of the user ratings given by other users to an place *i*.

The prediction *Pu,i* is given by:

$$P_{u,i} = \frac{\sum_v (r_{v,i} * s_{u,v})}{\sum_v s_{u,v}}$$

Here,

- *Pu,i* is the prediction of an place
- *Rv,i* is the rating given by a user *v* to a place *i*
- *Su,v* is the similarity between users

Now, we have the ratings for users in profile vector and based on that we have to predict the ratings for other users. Following steps are followed to do so:

1. For predictions we need the similarity between the user u and v. We can make use of Pearson correlation.
2. First we find the places rated by both the users and based on the ratings, correlation between the users is calculated.
3. The predictions can be calculated using the similarity values. This algorithm, first of all calculates the similarity between each user and then based on each similarity calculates the predictions. **Users having higher correlation will tend to be similar.**
4. Based on these prediction values, recommendations are made.

This algorithm is useful when the number of users is less. It's not effective when there are a large number of users as it will take a lot of time to compute the similarity between all user pairs. This leads us to item-item collaborative filtering, which is effective when the number of users is more than the items being recommended.

ITEM ITEM COLLABORATIVE FILTERING

In this algorithm, we compute the similarity between each pair of places. So in our case we will find the similarity between each place pair and based on that, we will recommend similar places which are liked by the users in the past. This algorithm works similar to user-user collaborative filtering with just a little change – instead of taking the weighted sum of ratings of "user-neighbors", we take the weighted sum of ratings of "places-neighbors". The prediction is given by:

$$P_{u,i} = \frac{\sum_N (s_{i,N} * R_{u,N})}{\sum_N (|s_{i,N}|)}$$

Now we will find the similarity between places.

$$sim(i,j) = cos(\vec{i}, \vec{j}) = \frac{\vec{i} \cdot \vec{j}}{||\vec{i}||_2 * ||\vec{j}||_2}$$

Now, as we have the similarity between each place and the ratings, predictions are made and based on those predictions, similar places are recommended.

What will happen if a new user or a new place is added in the dataset? It is called a **Cold Start**. There can be two types of cold start:

1. Visitor Cold Start
2. Product Cold Start

Visitor Cold Start means that a new user is introduced in the dataset. Since there is no history of that user, the system does not know the preferences of that user. It becomes harder to recommend places to that user. So, how can we solve this problem? One basic approach could be to apply a popularity based strategy, i.e. recommend the most popular places. These can be determined by what has been popular recently overall or regionally. Once we know the preferences of the user, recommending places will be easier.

Product Cold Start is not applicable in the proposed web application.

VI. RESULTS AND DISCUSSION

The results prove that it opens the ground on how to transpose smartness to tourism and destination levels. System will also collect feedback from different tourists and will be helpful for users planning a trip. Application will also provide Push notification for events and latest news for selected place. These mentioned features will save enormous time of user and will be helpful in organizing a destination place. As a result we witness the dawn of an age of smart tourism. The future scope of this application will be users might be fully dependent on this type of application while planning a trip as time will be most concerned factor in individual's life. Also it will be responsive and more dynamic with machine learning concepts.

VII. ACKNOWLEGMENT

Behind any major work undertaken by group there lies the contribution of the people who helped them to cross all the hurdles to achieve his goal. It gives us the immense pleasure to express our sense of sincere gratitude towards our respected guide Ms. Rasika Thakare, Assistance Professor for her persistent, outstanding, invaluable co-operation and guidance. She is a constant source of encouragement and momentum that any intricacy becomes simple. We gained a lot of invaluable guidance and prompt suggestions from her during entire project work. I will be indebted of her forever and we take pride to work under her. We also express my deep sense of regards and thanks to Mr. Harshal Shah, Professor and Head of Computer Engineering Department. We feel very privileged to have had their precious advices, guidance and leadership. Last but not the least, our humble thanks to the Almighty God.

REFERENCES

[1]P. Resnick, N. Iakovou, M. Sushak, P. Bergstrom, and J. Riedl, "GroupLens: an open architecture for collaborative filtering of netnews," in Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW '94), pp. 175–186, Chapel Hill, NC, USA, 1994. View at Publisher · View at Google Scholar
[2]U. Shardanand and P. Maes, "Social information filtering: algorithms for automating 'Word of Mouth'," in Proceedings of the Human Factors in Computing Systems Conference, pp. 210–217, May 1995. View at Scopus
[3]R. Ghani and A. Fano, "Building recommender systems using a knowledge base of product semantics," in Proceedings of the Workshop on Recommendation and Personalization in eCommerce at the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH '02), Malaga, Spain, 2002.