Where-to-go

A Point-Of-Interest Recommendation System

Software Project Lab 3

Submitted by

Samima Aktar
BSSE 0841
Institute of Information Technology
University of Dhaka

Submitted to

Software Project Lab 3 Committee

Supervised By

Md. Nurul Ahad Tawhid
Assistant Professor
Institute of Information Technology
University of Dhaka

Submission Date: 17 December 2019

Letter of Transmittal

17 December 2019

Coordinator,

Software Project Lab 3,

Institute of Information Technology,

University of Dhaka.

Subject: Submission of technical report on SPL-3 project.

Sir,

With due respect, I am pleased to submit the technical report on **Where-to-go**, A point-of-interest recommendation system. Although this report may have shortcomings, I have tried my level best to produce an acceptable technical report. I would be highly obliged if you overlooked the mistakes and accepted the effort that has been put in this report.

Sincerely yours,	Approved by,
Samima Aktar (BSSE 0841) Institute of Information Technology, University of Dhaka.	Md. Nurul Ahad Tawhid Assistant Professor Institute of InformationTechnology, University of Dhaka.

Acknowledgement

I am highly indebted for getting such a tremendous opportunity to prepare the technical report on **Where-to-go** - a point-of-interest recommendation system. I would like to thank whole-heartedly my supervisor, Md. Nurul Ahad Tawhid, Assistant Professor, Institute of Information Technology, University of Dhaka, for giving me guidelines about how I can prepare this report.

Abstract

A point-of-interest (PoI) is a specific point location that someone may find useful or interesting. PoI recommendation system is beneficial as it helps users to quickly and effortlessly find interesting place or location. In Software Project Lab -III, I have intended to make a PoI recommendation system using user-based collaborative filtering along with considering social influence (Friend-based collaborative filtering) and geographical influence. Geographical influence deals with users' preference of nearby places over distant places. Geographical influence plays an important role in user check-in behaviors according to existing literature. Accordingly, I have implemented a collaborative recommendation algorithm using power distribution law based geographical influence. I have used large-scale data sets gathered from Gowalla, Also, I have conducted a detailed performance evaluation. Experimental results with real data sets show that the unified collaborative recommendation model significantly outperforms alternative approaches.

Table of Contents

Table of Figures

Chapter 1: Introduction

1.1 Purpose

This document briefly describes the overall system of proposed recommendation system **Where-to-go**. It contains functional, non-functional and supporting requirements and establishes a requirements baseline for the development of the system. The requirements contained in the technical report are independent, uniquely numbered and organized by topic. The technical report serves as an official means of communicating user requirements to the developer and provides a common reference point for both the developer team and the stakeholder community. The technical report will evolve over time as users and developers work together to validate, clarify and expand its contents. This overall document will help each person related to this project to have a better idea about the project.

1.2 Objective

The main objectives of this project are as follows:

- To complete the requirement of BSSE degree
- To learn recommendation system workflows
- To work with machine learning technique
- To deal with large dataset
- To implement recommendation model algorithm
- To incorporate requirement and design knowledge with the project

Chapter 2: Quality Function Deployment

2.1 Quality Function Deployment

Quality Function Deployment (QFD) is a technique that translates the needs of the customer into technical requirements for software. Ultimately the goal of QFD is to translate subjective quality criteria into objective ones that can be quantified and measured, and which can then be used to design and manufacture the product. It is a methodology that concentrates on maximizing customer satisfaction from the software engineering process. So, I have followed this methodology to identify the requirements for the project. The requirements, which are given below, are identified successfully by the QFD.

2.1.1 Normal Requirements

Normal requirements consist of objectives and goals that are stated during the meeting with the customers. Normal requirements of my project are:

- **Providing Point of Interest (PoI) recommendations system using user based collaborative filtering**: For example, Toru and Babui are two different users. They visited the same locations: L1 and L2. Toru visited different location like L3 and Babui visited different location like L4. Since Toru and Babui share common location, they are similar user. So, location L3 can be recommended for Babui, similarly location L4 can be recommended for Toru. This user-user similarity can be achieved by user-based collaborative filtering.
- **Incorporating social influence for recommendation**: For example, two friends can hang out to watch a movie together, or a client can go to a restaurant highly recommended by his family. All these possible reasons indicate that due to their potential associated check-in activity, friends may provide a good recommendation for a given user. This social influence might be beneficial for recommendations.
- **Incorporating geographical influence for recommendatio**n: For example, people tend to go to his/her nearest place according to the power low distribution. Geographical influence deals with focusing on the nearer locations or places for PoI recommendation.
- **Recommending PoI for users**: For example, user can provide his/her visited places and friend lists. According to that, PoI recommendation for new users has to be done.

2.1.2 Expected Requirements

These requirements are implicit to the system and may be so fundamental that the customer does not explicitly state them. Their absence will be a cause for dissatisfaction.

- Proper model training
- Identifying Location Address (details)

2.1.3 Exciting Requirements

These requirements are for features that go beyond the customer's expectations and prove to be very satisfying when present.

- Graphical user interface
- Showing location on map

2.2 Usage Scenario

A point-of-interest is a specific point location that someone may find useful or interesting. PoI recommendation system is advantageous as it helps users to quickly find useful place or location. I have intended to make a PoI recommendation system using user-based collaborative filtering along with considering social influence and geographical influence. In this system, a user has been given PoI recommendation according to his/her previous point-of-interests and friend-list.

2.2.1 PoI Recommendation Model:

At the very beginning, a PoI recommendation model will be generated according to a dataset -named Gowalla. This model will be generated using user-based collaborative filtering along with considering social influence and geographical influence.

2.2.2 PoI Recommendation System:

User can ask for PoI recommendation from the system. According to the user’s selected PoIs and friend list incorporated with pre-built PoI recommendation model, a list of PoIs will be generated. That list of PoIs is the recommendation given by our PoI recommendation system.

Chapter 3: Scenario Based Modeling

This chapter describes the Scenario Based Model for **Where-to-go**.

3.1 Introduction

Scenario based modelling is the first phase where the usage of product can be visualized. This model enables us to get a vivid idea how user will use the product. In the following, I describe how the user story and use case.

3.2 Definition of Use Case

A Use Case captures a contract that describes the system behavior under various conditions as the system responds to a request from one of its stakeholders. A Use Case tells a stylized story about how an end user interacts with the system under a specific set of circumstances. A Use Case diagram simply describes a story using corresponding actors who perform important roles in the story and makes the story understandable for the users.

The first step in writing a Use Case is to define that set of users that will be involved in the story. Users are the different people that use the system or product within the context of the function and behavior that is to be described. Users represent the roles that people play as the system operators. Every user has one or more goals when using system.

3.3 Use Case Diagram

Use Case diagrams give the non-technical view of overall system.

3.3.1 Level-0 Use Case Diagram- Where to go

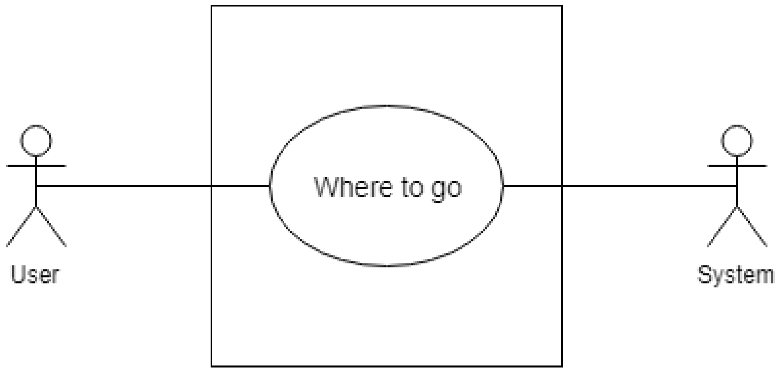


Figure 1:Level 0: Where to go

Name: where to go

ID: Level-0 Use Case

Primary Actors: User, System

Secondary Actors: None

Description of Level 0 Use Case Diagram:

After analyzing the user story I found that there are two actors who will use this system as a system operator.

Those actors are given below:

- User
- System

3.3.2 Level-1 Use Case Diagram-Subsystems

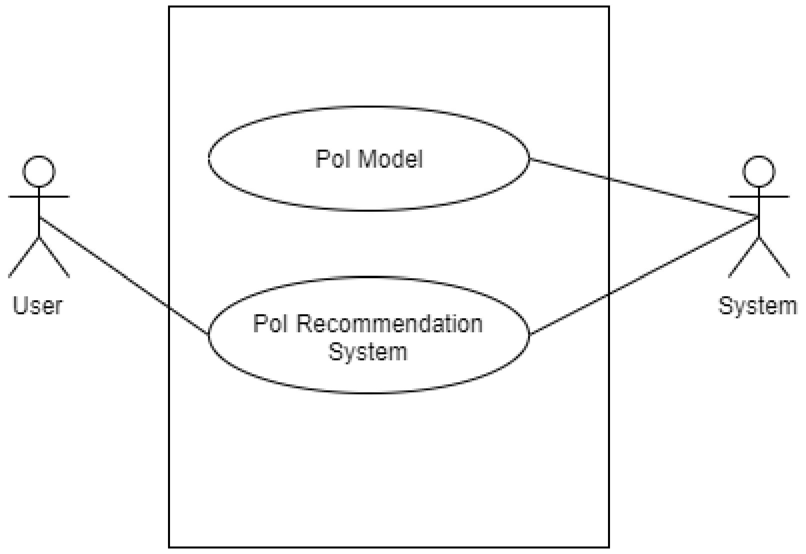


Figure 2: Level 1:Subsystems

Name: Subsystems of where to go

ID: Level-1 Use Case

Primary Actors: User, System

Secondary Actors: None

Description of Level 1 Use Case Diagram:

There are four subsystems:

- PoI model
- PoI recommendation system

3.3.3 Level-1.1 Use Case Diagram-PoI model

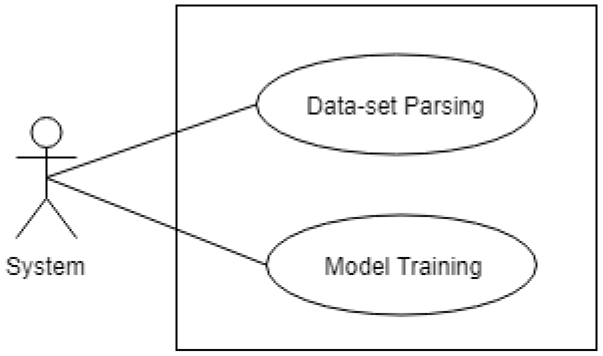


Figure 3: Level 1.1: PoI model

Name: PoI model

ID: Level-1.1 Use Case

Primary Actors: System

3.3.4 Level-1.2 Use Case Diagram-PoI recommendation system

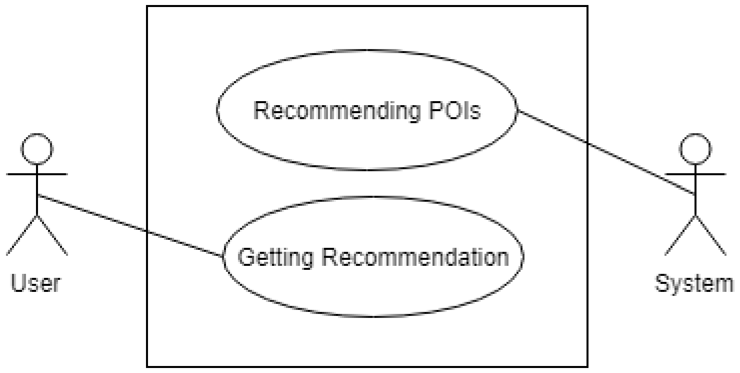


Figure 4: Level 1.2: PoI recommendation system

Name: PoI recommendation system

ID: Level-1.2 Use Case

Primary Actors: User, System

3.4 Activity Diagrams

3.4.1 Level-1.1 Activity Diagram- PoI model

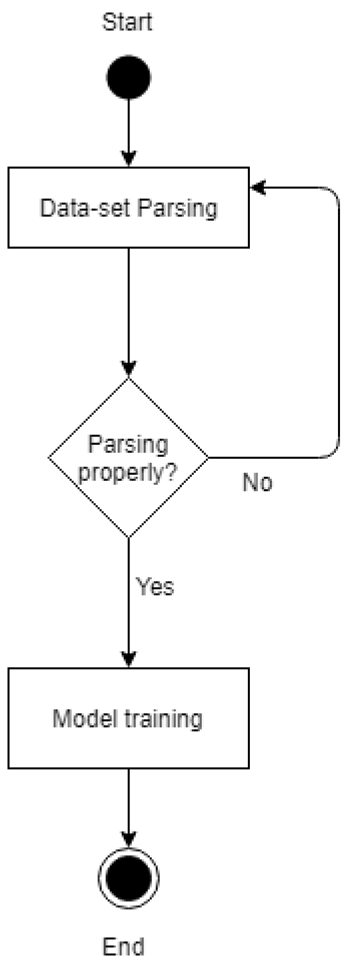


Figure 5: Level 1.1: PoI model

3.4.2 Level-1.2 Activity Diagram- PoIs Recommendation Model

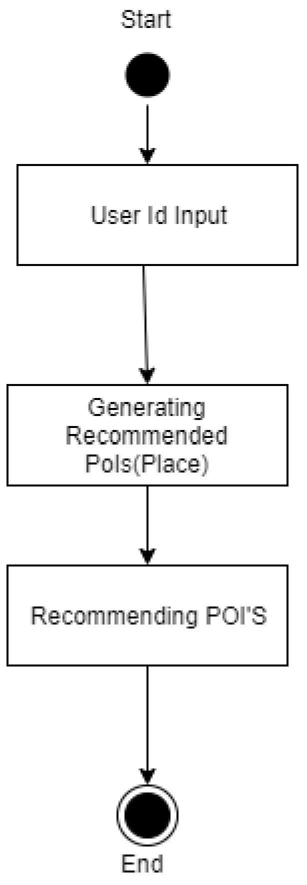


Figure 6: Level 1.2:PoIs Recommendation Model

Chapter 5: Methodology

5.1 Point-of-Interests

Point-of-interests (PoI) recommendation system is a system that recommends point-of-interest (possible location of interest or favor) of individual based on training data. PoI recommendation system mainly works using user-based collaborative filtering approach. For example, user 1 and user 2 have similar taste regarding point of interest:

- User1 check-ins at places: A, B, C, D & E
- User2 check-ins at places: A, B, C, E & F.

So according to user-based collaborative filtering, it is more likely that, user1 may prefer/like place F and user2 may prefer/like place D.

Most of the PoI recommendation system mainly works using user-based collaborative filtering approach. However, according to a state-of-the-art research [1], incorporating social influence (friend-based collaborative filtering) and geographical influence with the typical user-based collaborative filtering approach can improve the performance of PoI recommendation system.

How do social influence and geographical influence work? To understand those influences, Figure 1 might be helpful.

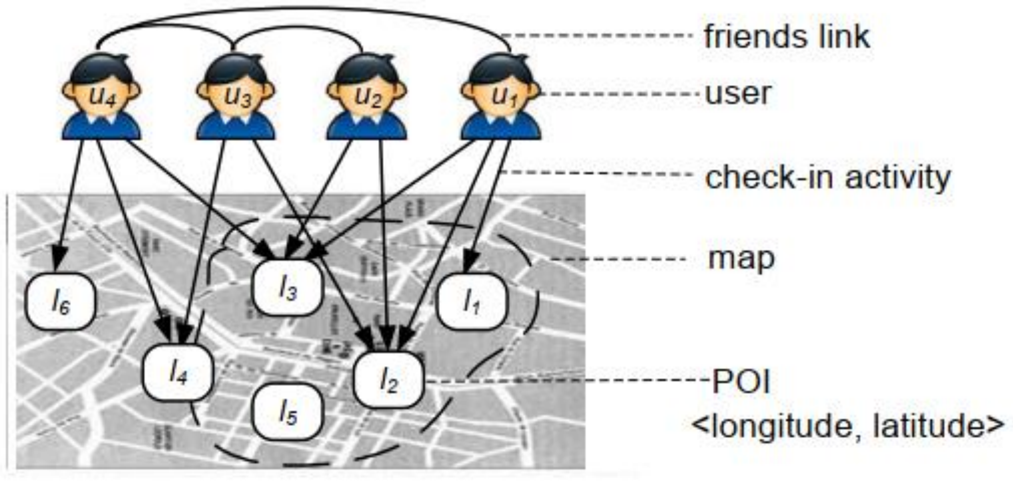


Figure 7: Overview of PoI recommendation

According to Figure 1, users u1 and u2 have common PoI - l2 and l3. They may be considered as similar users who are assumed to share similar check-in behaviors, i.e., preference of PoIs. As a result, l1 is a good candidate for recommendation to user u2 since u1 has visited this PoI before. On the other hand, social influence of friends can be incorporated in the recommendation process. For example, when considering l4 as a recommendation candidate for u1, the social influence of u4 on u1 may contribute to the decision making. Finally, the geographical influence of PoIs on nearby PoIs can be considered. As shown in the example, since u2 has visited l2 and l3 before, their nearby PoIs l1 and l5 may be considered positively due to the geographical influence. According to the research [1], the geographical influence among PoIs plays an important role in user check-in behaviors and model it by power law distribution.

In this project, I am going to make a PoI recommendation system based on the mentioned state-of-the-art research [1]. This system has three main components:

1. User-Based Collaborative Filtering
2. Friend-Based Collaborative Filtering

3. Geographical Influence

Finally, a unified PoI recommendation model will be implemented, which fuses user preference to a PoI with social influence and geographical influence. Here a large-scale public dataset named – Gowalla will be used for evaluation.

According to the recommendation model, I will recommend PoIs to the user based on their previous PoIs and friend list.

5.2 User-Based Collaborative Filtering

Based on collaborative filtering, users' implicit preference can be discovered by aggregating the behavior of similar users. Let U and L denote the user set and the PoI set in an LBSN, which keeps track of check-in activities in the system. The check-in activity a user ui ∈ U has at a PoI lj ∈ L is denoted as $c_{i,j}$ where $c_{i,j} = 1$ represents ui has a check-in at lj before and $c_{i,j} = 0$ means there is no record of ui visiting lj . These recorded user check-in activities at PoIs are thus used to discover a user's implicit preference of a PoI, which can be represented as a probability to predict how likely the user would like to have a check-in at an unvisited PoI. I denote this prediction by $c_{i,j}$ and obtain this predicted check-in probability of ui to lj as follows.

$$\widehat{\boldsymbol{c}_{i,j}} = \frac{\sum_{u_k} \boldsymbol{W}_{i,k} \cdot \boldsymbol{C}_{k,j}}{\sum_{u_k} \boldsymbol{W}_{i,k}}$$

where $w_{i,k}$ is the similarity weight between users u_i and u_k. To compute the similarity weights wi,k between users u_i and u_k, several similarity measures can be adopted, e.g., cosine similarity and Pearson correlation. In our study, I choose cosine similarity due to its simplicity. The cosine similarity weight between users ui and uk, denoted as $w^U_{i,k}$, is defined as follows.

$$W_{i,k} = \frac{\sum_{l_j \in L} C_{i,j} \cdot C_{k,j}}{\sqrt{\sum_{l_j \in L} C_{i,j}^2} \sqrt{\sum_{l_j \in L} C_{k,j}^2}}$$

5.3 Friend-based Collaborative Filtering

Friends tend to have similar behavior because they are friends and might share a lot of common interests, thus leading to correlated check-in behaviors. For example, two friends may hang out to see a movie together sometimes, or a user may go to a restaurant highly recommended by her friends. All those possible reasons suggest that friends might provide good recommendation for a given user due to their potential correlated check-in behavior. In other words, I can turn to user's friends for recommendation, and I call it recommendation based social influence from friends. PoI recommendations based on social influence can be realized by the friend-based collaborative filtering approach as described in [1].

$$\widehat{C_{i,j}} = \frac{\sum_{u_k \in F_i} SI_{k,i} \cdot C_{k,j}}{\sum_{u_k \in F_i} SI_{k,i}}$$

where $c_{i,j}$ is the predicted check-in probability of u_i at lj , Fi is the friends set of u_i, and $SI_{k,i}$ is directional social influence weight u_k has on u_i. On the one hand, I think friends who have closer social ties may have better trust in terms of their recommendation; on the other hand, friends who show more similar checkin behavior should have more similar tastes with the active user, thus suggestions from those friends are more worthy. Thus, in the following, I will introduce how to derive the social influence weight by combining the above two aspects. One way to derive the social influence weight between two friends is based on both of their social connections and similarity of their check-in activities [1].

5.4 Geographical Influence

As mentioned earlier, the check-in activities of users in LBSNs record their physical interactions (i.e., visits) at PoIs, Thus, I argue that the geographical proximities of PoIs have a significant influence on users' check-in behavior. Geographical influence which may be intuitively explained by the following tendencies: (1) people tend to visit PoIs close to their homes or offices; and (2) people may be interested in exploring nearby PoIs of a PoI that they are in favor of, even if it is far away from home. As a result, the PoIs visited by the same user tend to be clustered geographically. I believe that this geographical clustering phenomenon in user check-in activities can be exploited for PoI recommendations in LBSNs. Thus, in the following, I study and model this geographical influence on user check-in behavior at PoIs, aiming to utilize it in PoI recommendations. To achieve our goal, I would like to compute the likelihood that a user u_i would check in both PoI lj and l_k. I intuitively think the check-in probability may follow the power-law distribution. Generally speaking, the fact that a user's check-in PoIs tend to be in a short distance is confirmed in our data analysis. As mentioned earlier, nearby PoIs are more related to each other, which exhibits strong geographical influence [1].

5.5 Fusion Framework

As discussed, each factor, i.e., user preference, social influence or geographical influence, can be utilized to realize PoI recommendation. Thus, we intuitively can implement three different recommender systems. We propose to use a linear fusion framework to integrate ranked lists provided by the three above-mentioned recommenders into the final ranked list. By integrating multiple recommenders, top-ranked PoIs from each of the recommendation algorithms could increase both recall (due to the different highly ranked PoIs) and precision (giving that the recommender systems have a high density of user-preferred PoIs on top of the results lists.

Let $S_{i,j}$ denote the check-in probability score of user u_i at PoI lj , i.e., the more likely u_i has a check-in activity at l_j , the larger $S_{i,j}$ is. Let $S^u{}_{i,j}$, $S^s{}_{i,j}$ and $S^g{}_{i,j}$ denote the check in probability scores of user ui at PoI l_j , corresponding to recommender systems based on user preference, social influence and geographical influence, respectively. We have $S_{i,j}$ as follows.

$$S_{i,j} = (1 - \alpha - \beta)S^u_{i,j} + \alpha S^s_{i,j} + \beta S^g_{i,j}$$

Where the two weighting parameters α and β ($0 \leq \alpha+\beta \leq 1$) denote the relative importance of social influence and geographical influence comparing to user preference.

Chapter 6: Implementation Overview

The system is implemented according to the requirement specification and design. In this chapter I will discuss about the implementation details of the system.

6.1. Programming Languages

6.1.1 Python:

Python is a language used for many purposes that is extremely high programming. Since the Python 3 was released, it has become popular worldwide among the world's developers. It has a fantastic data city library, machine learning, profound learning, web scraping, creation of the internet, image processing, etc. Therefore, as a general-purpose library is generally accepted. In developing 'Where-to-go,' I have chosen Python 3. I have used open source libraries for implementation which are free to use and are supported by the python community. The following section contains a detailed list of libraries and a short description I used in my project:

6.2 Tools

6.2.1 Jupyter-notebook:

The Jupyter Notebook is a web-based, open-source application that allows you to develop and share live software, equations, visualizations and narrative text documents. Uses include: transformation and cleaning of data, simulation, mathematical modeling, visualization of information, machine learning and much more. All source codes of Python are written with the jupyter notebook.

6.2.2 Pycharm:

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as Data Science with Anaconda.
PyCharm is cross-platform, with Windows, macOS and Linux versions. The Community Edition is released under the Apache License [2].

6.3 Libraries

6.3.1 ipynb

ipynb is python library for handling jupyter source codes. it is used in my tool to importing notebook files and using their methods.

6.3.2 pickle

The Python object structure is used for serialization and degradation. Any python object may be picked up in order to save it on disk. There was a mistake. Beating is a means to transform an object python (List, Dict, etc.) into a stream of character [3].

6.3.3 numpy

NumPy is a Python programming language library, supporting large, multi-dimensional arrays and matrices, together with a wide range of high level maths for these arrays. I had to use extensive image processing tasks for my project, as I mentioned before. I know that image is nothing other than a vast array or matrix collection of pixels. So, every image processing task is a matrix collection. Therefore, Numpy was a must use package for my project to perform these matrix multiplication activities[4].

6.3.4 sklearn

Scikit-learn is a free-software learning library for the Python programming language and was previously known as scikit.learn and also known as sklearn. This is designed for interoperation with the Python numeric and scientific libraries NumPy and SciPy, with various algorithms of classification, regression and clustering including support vector machines, random forests, gradient boostings and k-means and DBSCAN. As my project involves using various machine learning algorithms like k nearest neighbor (knn), decision-making trees, random forests, vector support (svm), etc. That's why I used my project with Sklearn's library [5].

6.4 Framework

6.4.1 Django

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source [6].

Chapter 7: Source Code Description

The complete source code of the project can be found at this link:
https://github.com/SamimaAktar/Where-to-go-spl3
This section provides a brief overview of the dataset, source code files and their functionality.

7.1 Data Preprocessing

First, I collect Gowalla dataset which contains millions of data regarding point of interest. The dataset contains user check-ins, place id, place longitude-latitude, social relations etc. All of these were messed up because no serial is maintained, and a huge number of user-friend link and user-location link is missing. So, I need to preprocess the dataset.

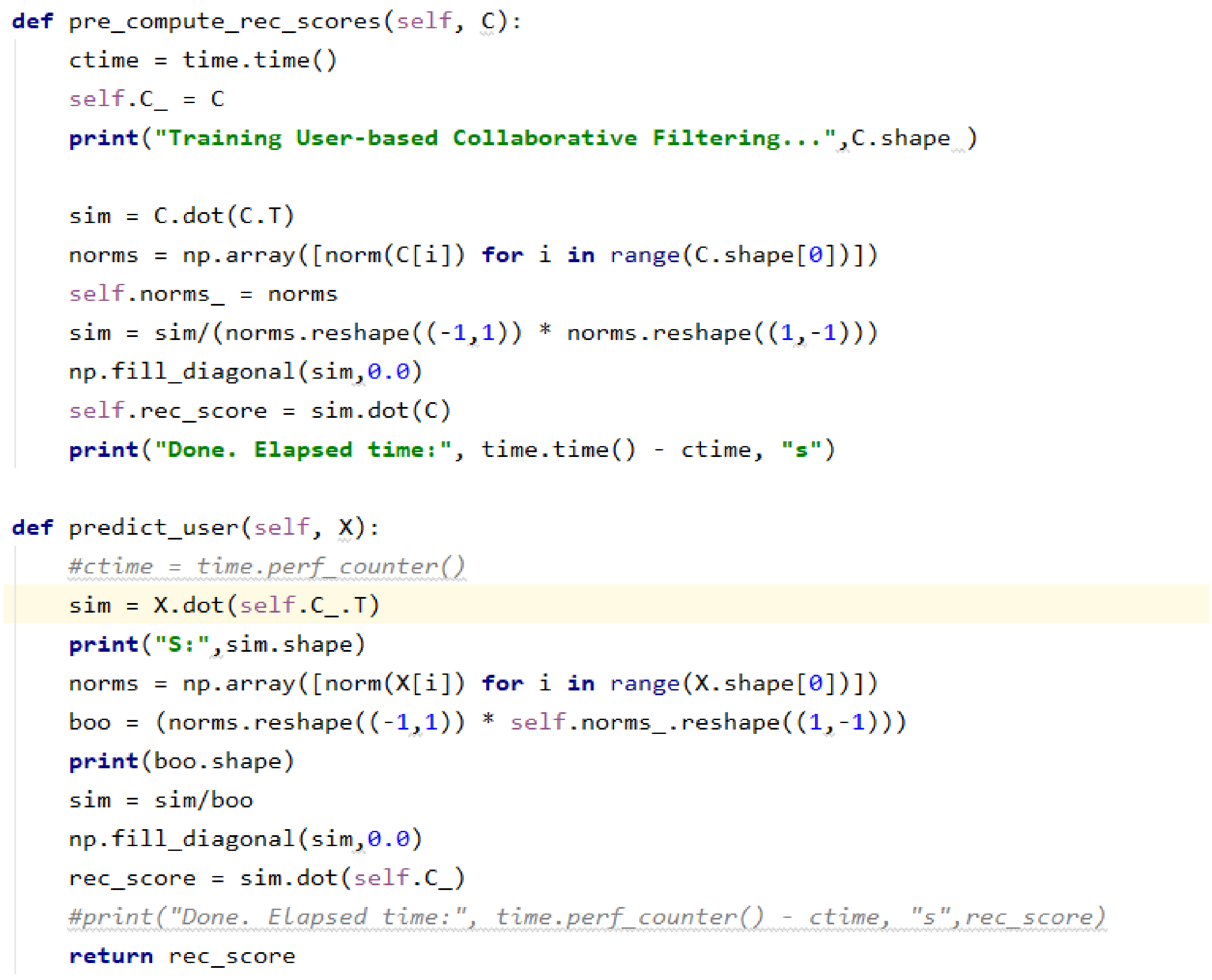
I have serialized the check in data, social relations, place ids and corresponding longitude latitude. Then I have filtered out duplicate data. I also have filtered out unlinked data from all of the files and synchronized the dataset. And I have divided into training and test data.

7.2 Model Building:

For three different influences (user, social, geographic) I have implemented three algorithms in the following files:

UserBasedCF.py:

In this file I have implemented the user-based collaborative filtering algorithm as described in the methodology. Cosine Similarity has to be performed for this task. The core algorithm is implemented as follows:

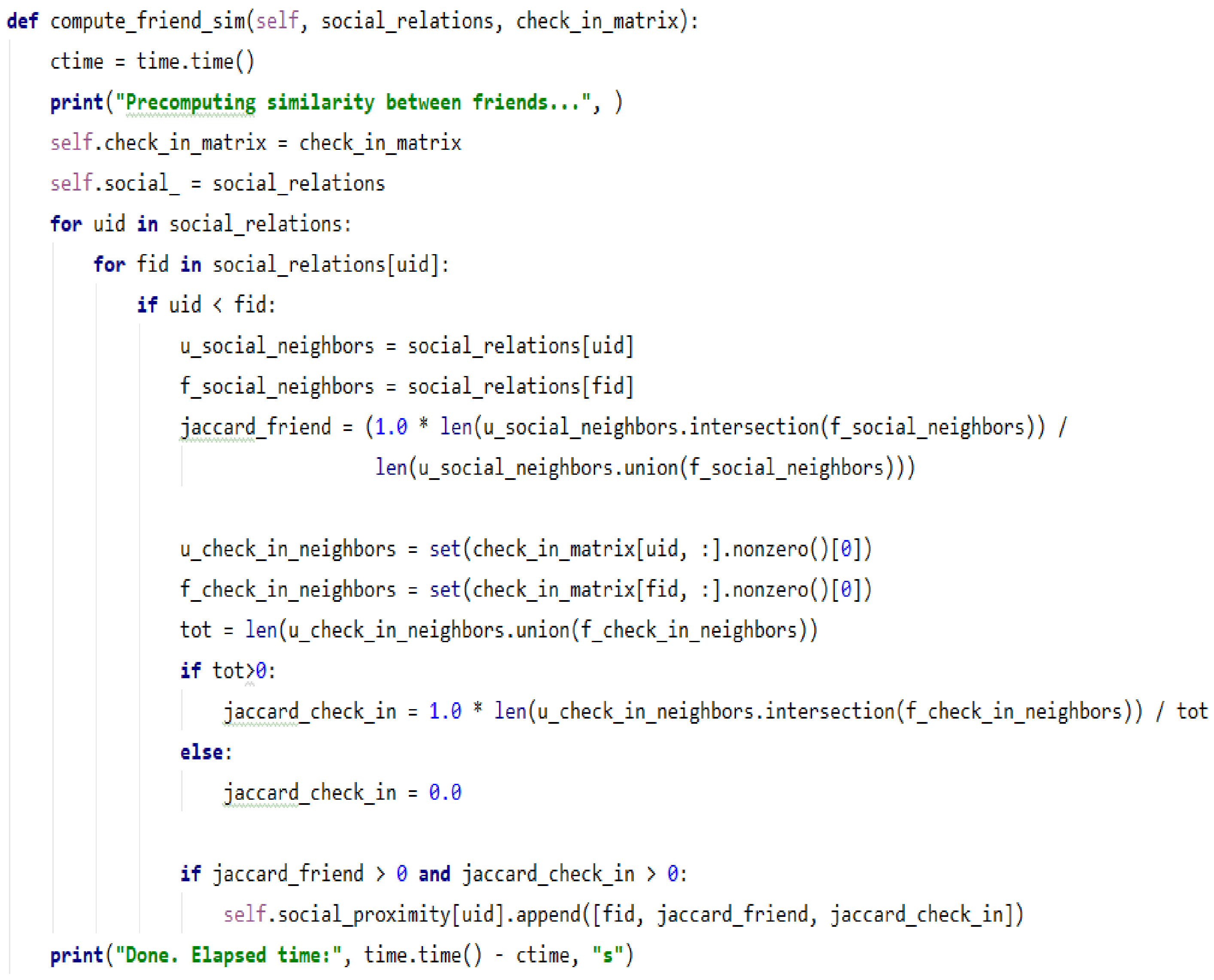


```python
def pre_compute_rec_scores(self, C):
    ctime = time.time()
    self.C_ = C
    print("Training User-based Collaborative Filtering...",C.shape )

    sim = C.dot(C.T)
    norms = np.array([norm(C[i]) for i in range(C.shape[0])])
    self.norms_ = norms
    sim = sim/(norms.reshape((-1,1)) * norms.reshape((1,-1)))
    np.fill_diagonal(sim,0.0)
    self.rec_score = sim.dot(C)
    print("Done. Elapsed time:", time.time() - ctime, "s")

def predict_user(self, X):
    #ctime = time.perf_counter()
    sim = X.dot(self.C_.T)
    print("S:",sim.shape)
    norms = np.array([norm(X[i]) for i in range(X.shape[0])])
    boo = (norms.reshape((-1,1)) * self.norms_.reshape((1,-1)))
    print(boo.shape)
    sim = sim/boo
    np.fill_diagonal(sim,0.0)
    rec_score = sim.dot(self.C_)
    #print("Done. Elapsed time:", time.perf_counter() - ctime, "s",rec_score)
    return rec_score
```

Figure 8:UserBasedCF.py

FriendBasedCF.py:

I have implemented the social influences on PoI described in the methodology. The core algorithm is implemented as follows:

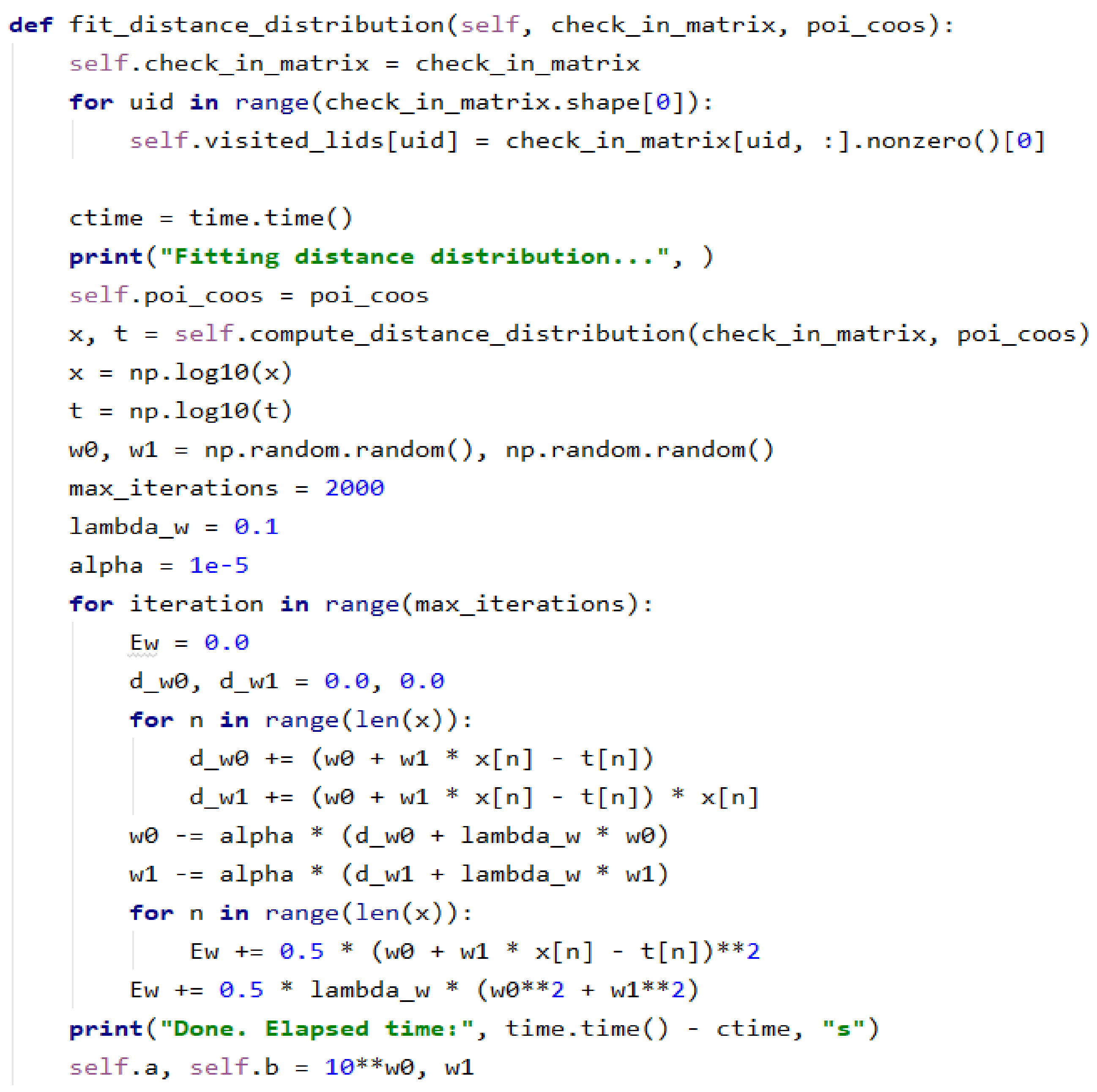


```
def compute_friend_sim(self, social_relations, check_in_matrix):
    ctime = time.time()
    print("Precomputing similarity between friends...", )
    self.check_in_matrix = check_in_matrix
    self.social_ = social_relations
    for uid in social_relations:
        for fid in social_relations[uid]:
            if uid < fid:
                u_social_neighbors = social_relations[uid]
                f_social_neighbors = social_relations[fid]
                jaccard_friend = (1.0 * len(u_social_neighbors.intersection(f_social_neighbors)) /
                                  len(u_social_neighbors.union(f_social_neighbors)))

                u_check_in_neighbors = set(check_in_matrix[uid, :].nonzero()[0])
                f_check_in_neighbors = set(check_in_matrix[fid, :].nonzero()[0])
                tot = len(u_check_in_neighbors.union(f_check_in_neighbors))
                if tot>0:
                    jaccard_check_in = 1.0 * len(u_check_in_neighbors.intersection(f_check_in_neighbors)) / tot
                else:
                    jaccard_check_in = 0.0

                if jaccard_friend > 0 and jaccard_check_in > 0:
                    self.social_proximity[uid].append([fid, jaccard_friend, jaccard_check_in])
    print("Done. Elapsed time:", time.time() - ctime, "s")
```

Figure 9: FriendBasedCF.py

PowerLaw.py:

I have incorporated the geographical influence as described in the methodology. Power law distribution has to be performed for this task. The core algorithm is implemented as follows:



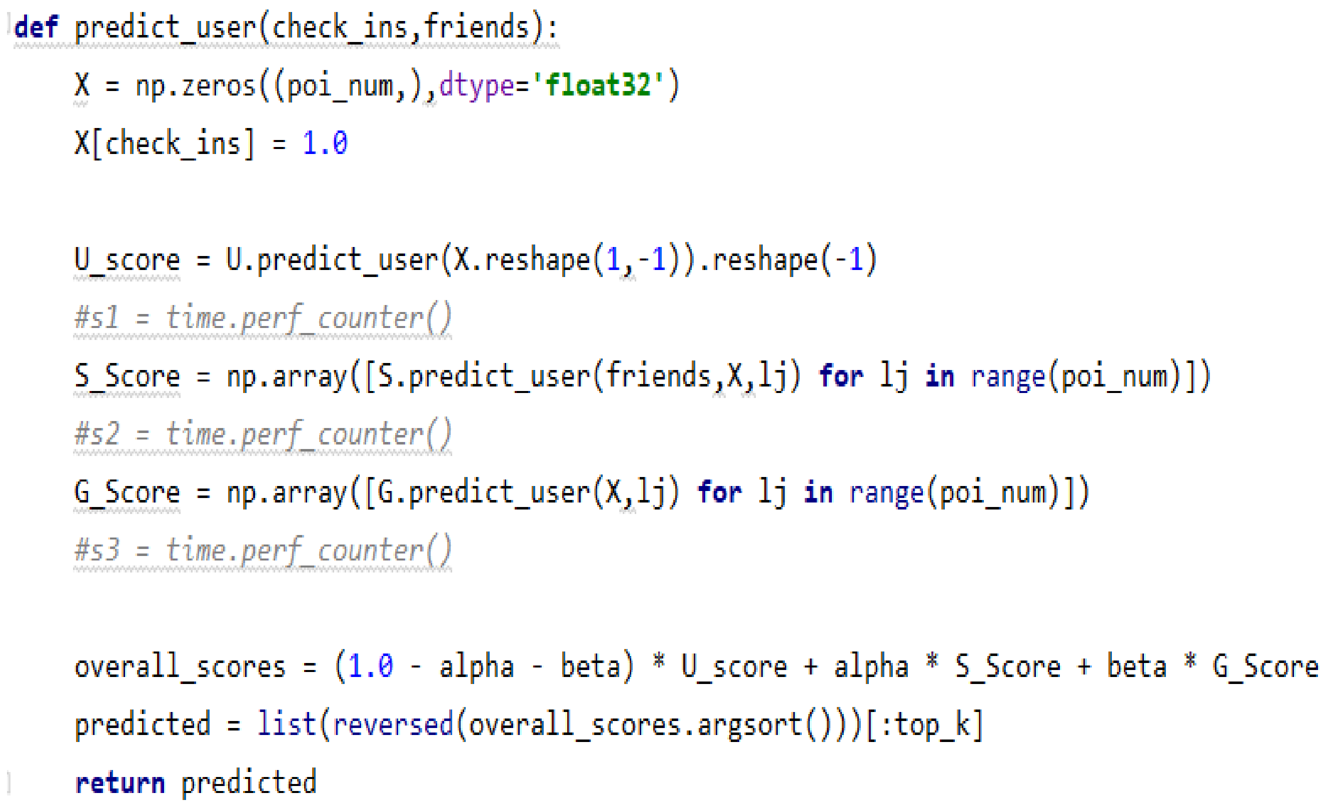
```python
def fit_distance_distribution(self, check_in_matrix, poi_coos):
    self.check_in_matrix = check_in_matrix
    for uid in range(check_in_matrix.shape[0]):
        self.visited_lids[uid] = check_in_matrix[uid, :].nonzero()[0]

    ctime = time.time()
    print("Fitting distance distribution...", )
    self.poi_coos = poi_coos
    x, t = self.compute_distance_distribution(check_in_matrix, poi_coos)
    x = np.log10(x)
    t = np.log10(t)
    w0, w1 = np.random.random(), np.random.random()
    max_iterations = 2000
    lambda_w = 0.1
    alpha = 1e-5
    for iteration in range(max_iterations):
        Ew = 0.0
        d_w0, d_w1 = 0.0, 0.0
        for n in range(len(x)):
            d_w0 += (w0 + w1 * x[n] - t[n])
            d_w1 += (w0 + w1 * x[n] - t[n]) * x[n]
        w0 -= alpha * (d_w0 + lambda_w * w0)
        w1 -= alpha * (d_w1 + lambda_w * w1)
        for n in range(len(x)):
            Ew += 0.5 * (w0 + w1 * x[n] - t[n])**2
        Ew += 0.5 * lambda_w * (w0**2 + w1**2)
    print("Done. Elapsed time:", time.time() - ctime, "s")
    self.a, self.b = 10**w0, w1
```

Figure 10: PowerLaw.py

7.3 Recommendations:

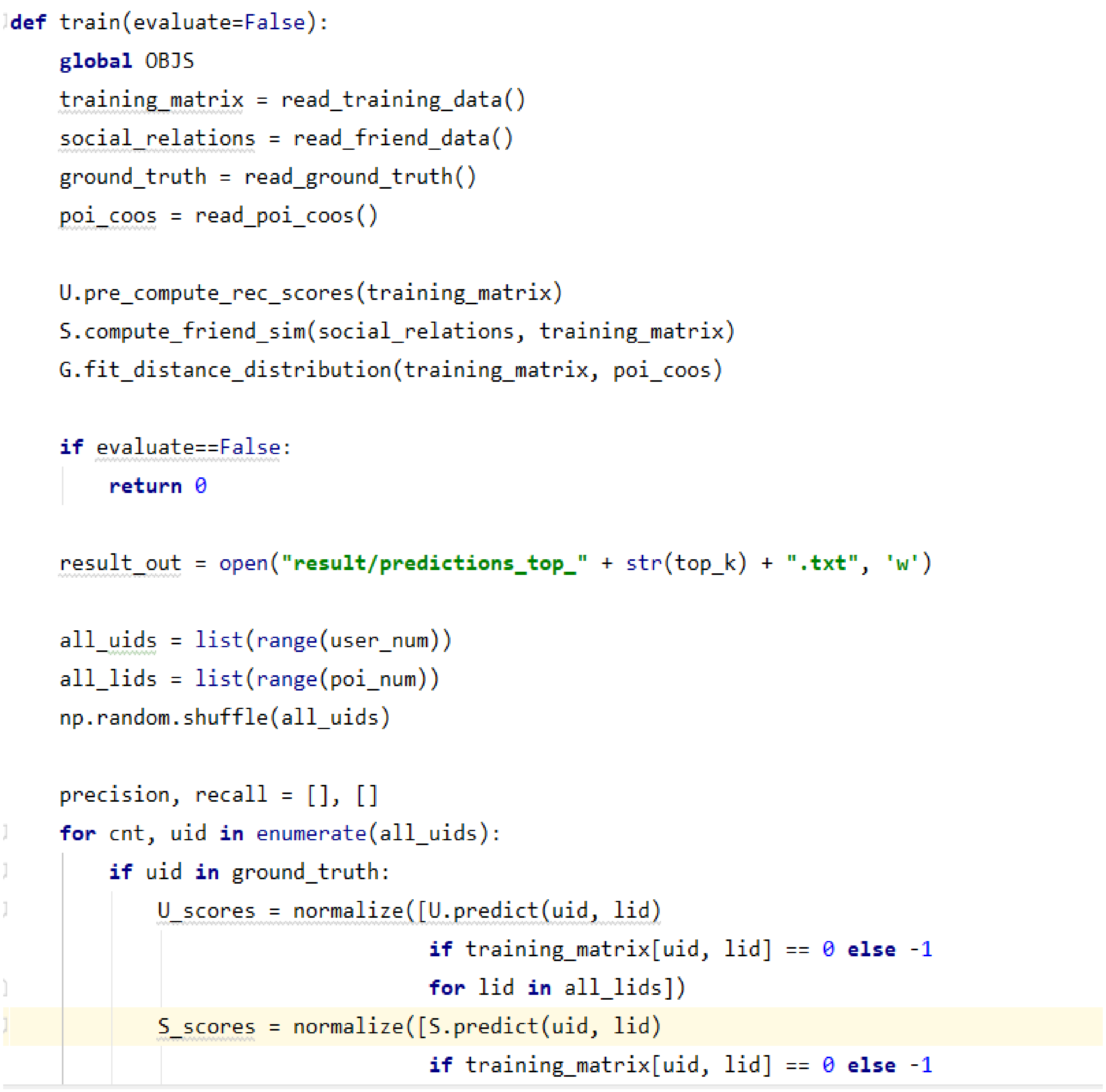
To implement the fusion framework, and to maintain and set parameters, tuning factors, Recommendation.py is responsible. It acts as the main file. All the code and user interactions is performed in this file.

Recommendation.py:



```python
def predict_user(check_ins,friends):
    X = np.zeros((poi_num,),dtype='float32')
    X[check_ins] = 1.0

    U_score = U.predict_user(X.reshape(1,-1)).reshape(-1)
    #s1 = time.perf_counter()
    S_Score = np.array([S.predict_user(friends,X,lj) for lj in range(poi_num)])
    #s2 = time.perf_counter()
    G_Score = np.array([G.predict_user(X,lj) for lj in range(poi_num)])
    #s3 = time.perf_counter()

    overall_scores = (1.0 - alpha - beta) * U_score + alpha * S_Score + beta * G_Score
    predicted = list(reversed(overall_scores.argsort()))[:top_k]
    return predicted
```

Figure 11: Recommendation.py [1]



```python
def train(evaluate=False):
    global OBJS
    training_matrix = read_training_data()
    social_relations = read_friend_data()
    ground_truth = read_ground_truth()
    poi_coos = read_poi_coos()

    U.pre_compute_rec_scores(training_matrix)
    S.compute_friend_sim(social_relations, training_matrix)
    G.fit_distance_distribution(training_matrix, poi_coos)

    if evaluate==False:
        return 0

    result_out = open("result/predictions_top_" + str(top_k) + ".txt", 'w')

    all_uids = list(range(user_num))
    all_lids = list(range(poi_num))
    np.random.shuffle(all_uids)

    precision, recall = [], []
    for cnt, uid in enumerate(all_uids):
        if uid in ground_truth:
            U_scores = normalize([U.predict(uid, lid)
                                  if training_matrix[uid, lid] == 0 else -1
                                  for lid in all_lids])
            S_scores = normalize([S.predict(uid, lid)
                                  if training_matrix[uid, lid] == 0 else -1
```

Figure 12: Recommendation.py [2]

The predict_user() is used for recommendation of new user. And train() is used for training model.

Chapter 8: Result Analysis

A benchmark dataset- Gowalla is used for evaluation. First, the dataset is loaded and trained according to the different strategy (user-based collaborative filtering, social influence, geographical influence) mentioned in the Methodology section. 10-cross validation is used for soundness.

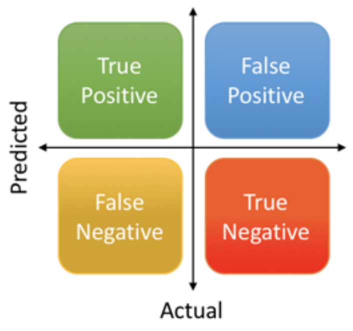
The result metrics of this work coherent and supported by an existing literature [1]. For different N (number of PoIs to be recommended) values - 5,10,20, result exhibits a pattern. For larger N, it is obvious that the number of relevant PoIs, which are recommended, increases as well as recall increases. Also, it is more likely that the number of PoIs, which are relevant, decreases for larger N. Accordingly, precision decreases larger N.

8.1 Precision & Recall

$$\textbf{Precision} = \frac{\text{True Positive}}{\text{Actual Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive + False Positive}}$$

$$\textbf{Recall} = \frac{\text{True Positive}}{\text{Predicted Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive + False Negative}}$$

$$\textbf{Accuracy} = \frac{\text{True Positive + True Negative}}{\text{Total}}$$



Precision and recall all imply template reliability. While this is very accurate, each of these words has a broader, distinct meaning. Precision is the percentage of my results that is significant. Recall, on the other hand, refers to the percentage of total relevant results that my algorithm correctly classifies.

Here, precision and recall are calculated for result evaluation. In this case, two important consideration is –

1. **How many recommended PoIs are relevant? -** Precision indicates that measure.
2. **How many relevant PoIs are recommended? –** Recall indicates that measure.

That is why precision and recall measurements are important and suitable in PoI recommendation.

8.1.1 Precision & Recall for N = 5, 10, 20

In general, precision increases when N (where n is the number of recommended PoIs) decreases. On the other hand, recall increases when N also increases.

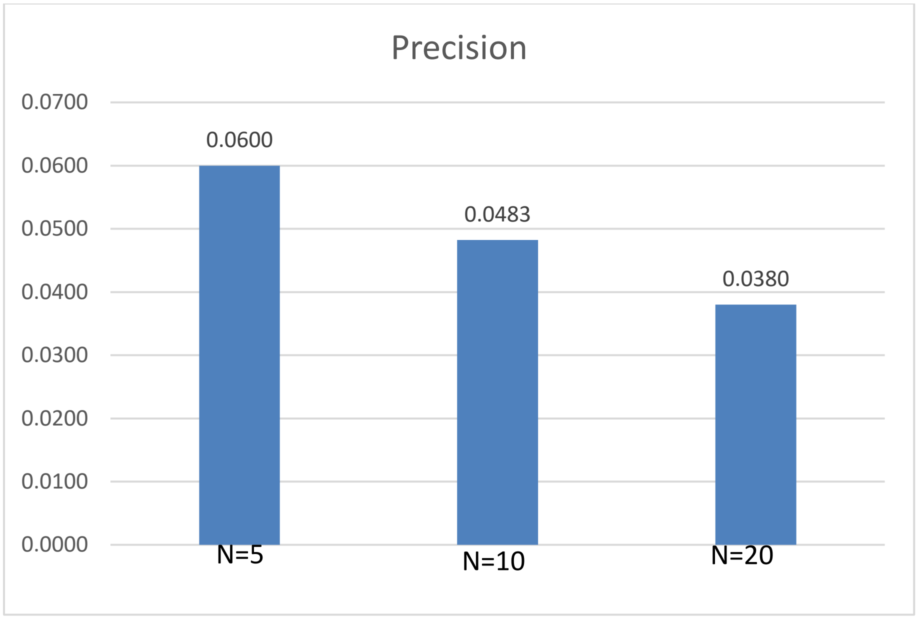


Figure 13: Precision for N=5, 10, and 20

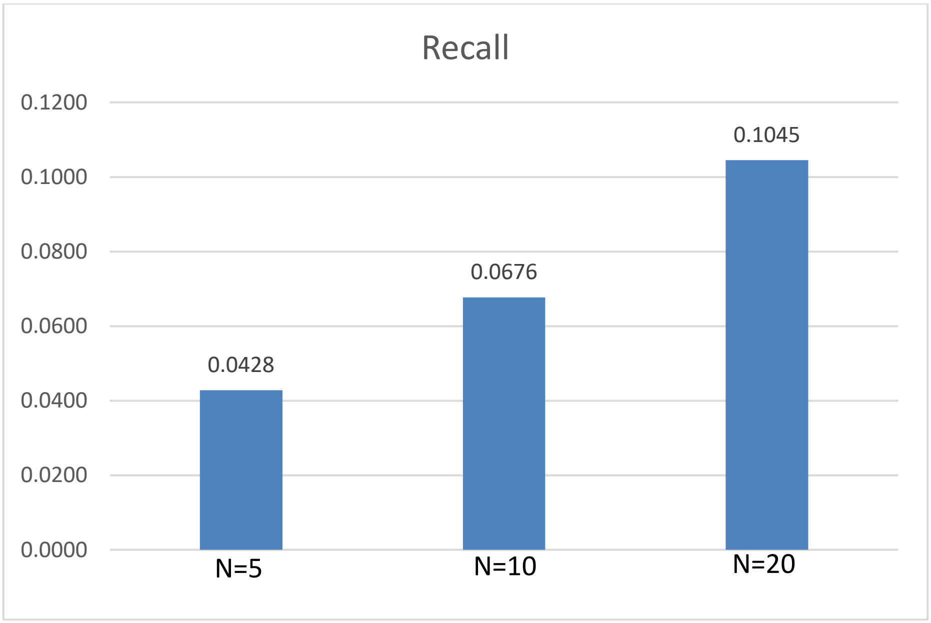


Figure 14: Recall for N=5, 10, and 20

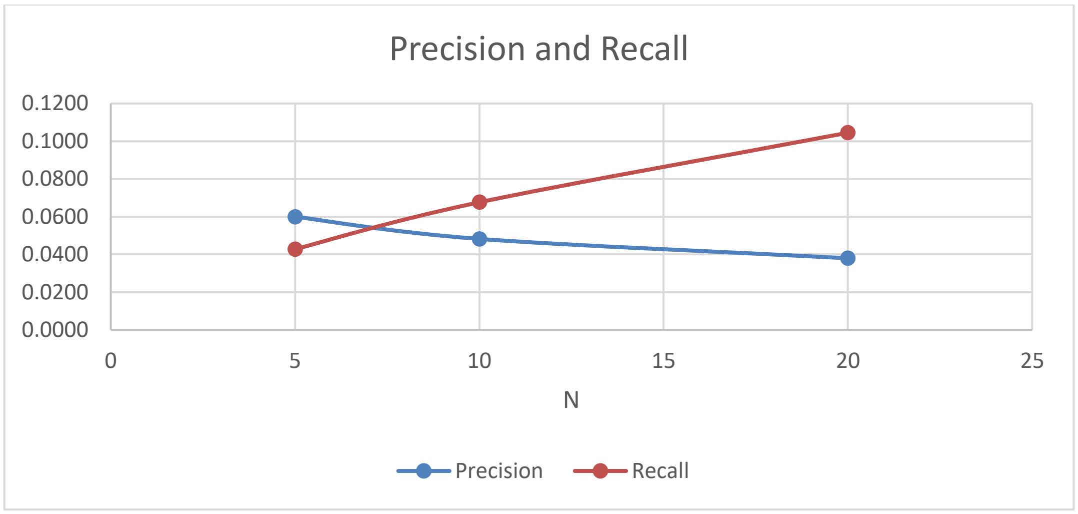


Figure 15: Precision and Recall for N=5, 10 and 20

8.1.2 Precision & Recall for different influences

Precision and recall are highest when all the three influences -USG (user-based collaborative filtering, social and geographical) are considered. Respectively, U performs significantly better than F or G exclusively.

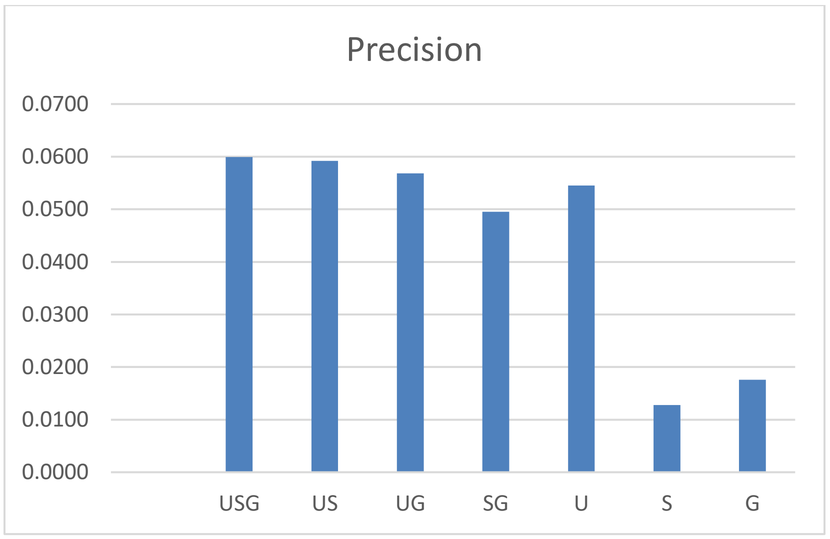


Figure 16: Precision for different combination

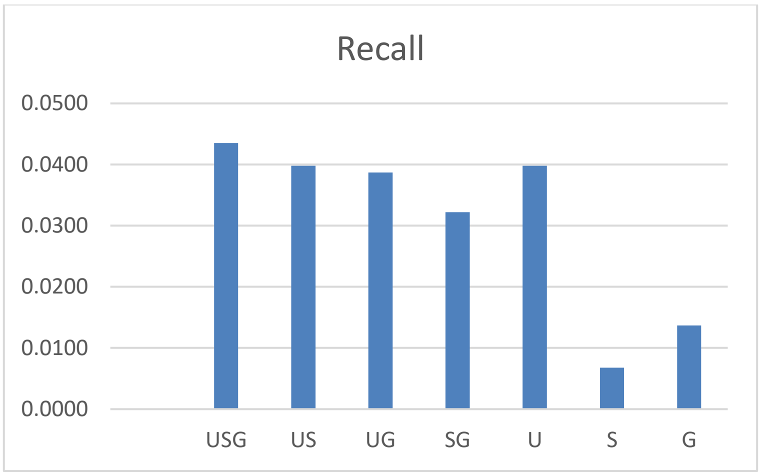


Figure 17: Recall for different combination

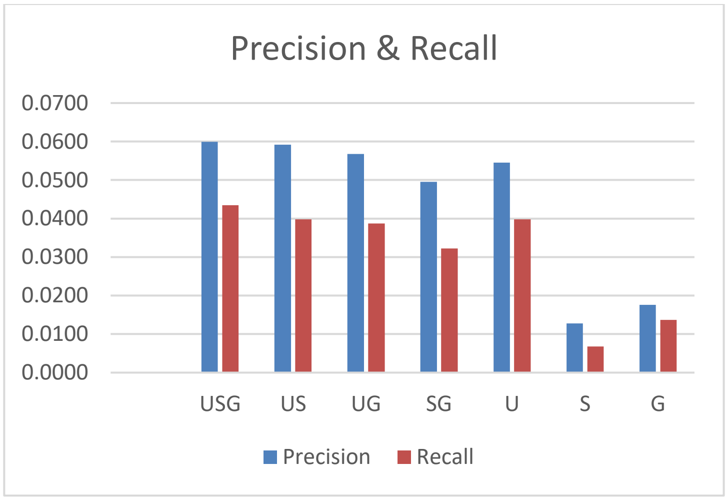


Figure 18: Precision & recall for different combination

Here,

USG means, considering user-based collaborative filtering, social influence and geographical influence

US means, considering user-based collaborative filtering and social influence

UG means, considering user-based collaborative filtering and geographical influence

SG means, considering social influence and geographical influence

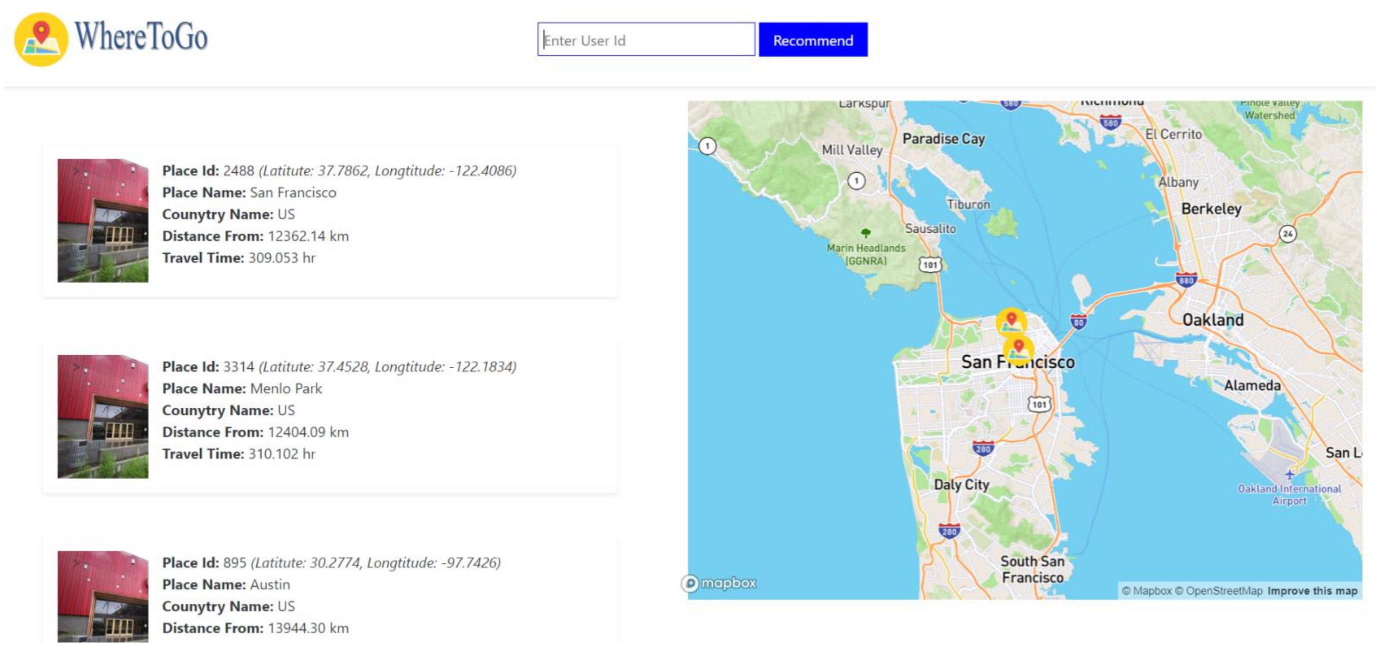
U means, considering only user-based collaborative filtering only

S means, considering social influence only

G means, considering geographical influence only

Chapter 9: User Manual

In this project, the recommendation for new users is one of the requirements. Here one user can get new PoI recommendation according to the model.



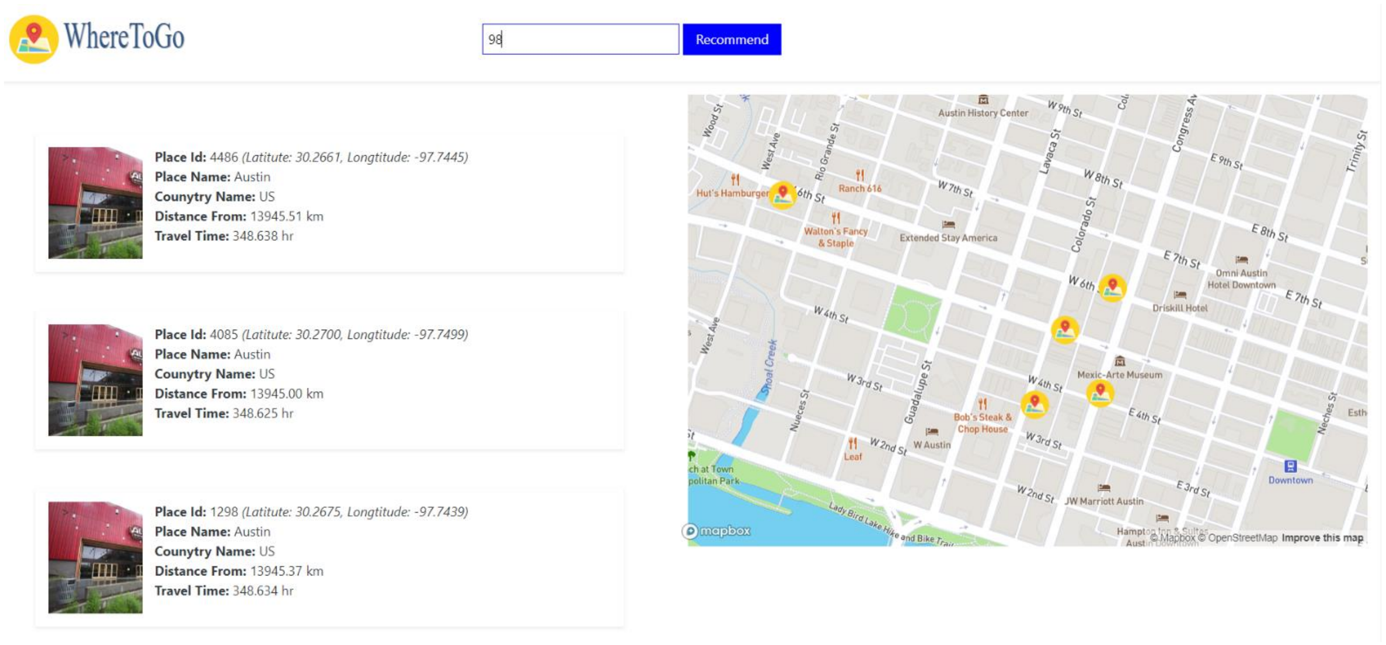
Place Id: 2488 *(Latitute: 37.7862, Longtitude: -122.4086)*
Place Name: San Francisco
Counytry Name: US
Distance From: 12362.14 km
Travel Time: 309.053 hr

Place Id: 3314 *(Latitute: 37.4528, Longtitude: -122.1834)*
Place Name: Menlo Park
Counytry Name: US
Distance From: 12404.09 km
Travel Time: 310.102 hr

Place Id: 895 *(Latitute: 30.2774, Longtitude: -97.7426)*
Place Name: Austin
Counytry Name: US
Distance From: 13944.30 km

Figure 19: User Manual: Intro

After starting the recommendation module, a user can see the following user interface. At first, user has to provide his user Id into the text input area. Then, he has to click the 'Recommend' button to get recommendation.



Place Id: 4486 *(Latitute: 30.2661, Longtitude: -97.7445)*
Place Name: Austin
Counytry Name: US
Distance From: 13945.51 km
Travel Time: 348.638 hr

Place Id: 4085 *(Latitute: 30.2700, Longtitude: -97.7499)*
Place Name: Austin
Counytry Name: US
Distance From: 13945.00 km
Travel Time: 348.625 hr

Place Id: 1298 *(Latitute: 30.2675, Longtitude: -97.7439)*
Place Name: Austin
Counytry Name: US
Distance From: 13945.37 km
Travel Time: 348.634 hr

Figure 20: User Manual: Recommendation

According to user's input, user can see top five recommended places. He can see the place's longitude, latitude, country name, distance form current location and travel time. Besides, the recommended locations are pinned in map for user convenience. The locations in the map can zoomed in and zoomed out by scrolling the mouse.

Chapter 10: Conclusion

This technical report draws a line between the implementation and the underlying details. With the help of this technical report, any developer can understand the whole system requirement and design. So, it acts as a guideline for the developers who will use it in practice.

This document has pointed out every necessary point which is necessary for the next development phase. For better understanding, different figure and the table have shown in this document. All the steps for developing the software and both high-level and low-level categories have been shown in brief in the usage scenario point. The mockup UI design for the tool has also been shown for the front-end development. Besides, this will help to understand the workflow of the total system. Besides, this technical report also includes methodology, implementation, and source code description and result analysis.

Finally, with this document, I have tried to minimize all the ambiguity of the development. I hope this report can be used effectively to maintain the software development cycle. It will be very easy to conduct the whole project using this technical report.

Reference

[1] Ye, M., Yin, P., Lee, W. C., & Lee, D. L. (2011, July). Exploiting geographical influence for collaborative Point-of-interest recommendation. In Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval (pp. 325-334). ACM.

[2] <https://en.wikipedia.org/wiki/PyCharm> [Accessed October 22, 2019]

[3]< https://docs.python.org/3/library/pickle.html> [Accessed October 22, 2019]

[4] <https://numpy.org/>[Accessed October 22, 2019]

[5]< https://en.wikipedia.org/wiki/Scikit-learn>[Accessed November 22, 2019]

[6] < https://www.djangoproject.com > [Accessed December 03, 2019]