EVENT RECOMMENDATION USING DATA MINING

Introduction

Event Recommendation system is a system to predict whether a user is interested in any event or not. There are a lot of events happening all over the world in various fields. Some users might be interested in events based on their preferences or as suggested by their friends. In this project, we have to predict what events users will be interested, based on events they've responded to in the past, user demographic information, and what events they've seen and clicked on. This project is based on a competition posted on kaggle.com.

DataSet

Data for this project is taken from kaggle.com. There are six files in all: train.csv, test.csv, users.csv, user_friends.csv, events.csv, and event_attendees.csv.

events.csv This file contains all the information about every event present in the system. It has approx. 2 mil rows and 110 columns. Columns are:

- Event-id, unique id of event
- User-id, user who has created the event
- Start time, start time of event
- City, state, country and zip, are location of event(if known)
- Lat and lng, are latitude and longitude of the location of event
- Next 101 columns are for the 100 most common words throughout the application and count of each of 100 words in this event.

users.csv This file contains all the information about all users in the system. It has about 30000 rows and 7 columns.

- user_id, unique id of each user
- locale, string representing users locale
- birthyear, birth year of user
- gender, gender of user
- joinedAt, when user has joined the system
- location, user's location
- timezone, user's time-zone

user_friends.csv This file has just two columns user and friend. Friend is a list of friends of user which is space separated.

event_attendees.csv This file has information about which user attended which events.

- event_id, identifies the event
- yes, space separated list of user who indicated that they are going to event

- maybe, space separated list of user who indicated maybe
- invited, space separated list of user who indicated that they were invited
- no, space separated list of user who indicated that they are not going to event

train.csv, This file is used for training the model. It has six columns:

- user, id of user
- event, id of event
- invited, is a binary variable, whether the user is invited to event (1) or not (0)
- timestamp, when user saw the event
- interested, is a binary variable, if user is interested in event (1)
- not_interested, is a binary variable, if user is not interested in event (1)

test.csv, this file is used to test the data. Same as train.csv but we have to predict interested or not.

In this project, I have considered the target class as "interested" given a tuple of (event, user). Value of the interested, yes or no (1 or 0), is predicted based on relation between user and event using all the information provided in various files.

Data Preprocessing

In this dataset, a lot of preprocessing was needed to be done.

- I have loaded all the data into SQL database from the csv files, so that the data is easy and quick to access as compared to csv files. While loading data into the database file, I have encountered various problem with timestamp variable. The UTC version used in csv files of data were not compatible with the SQL versions. Hence SQL truncated the data based on the acceptable version.
- There was a lot of missing data about the location of event as well as the user, which was replaced by 0s in latitude and longitudes. Missing data in events file can't be just dropped because then the whole event will be erased from the system, which will affect all the files and it will create more irrelevant data in whole dataset.
- All the space separated list of users in all files was converted into the list of users and was stored as the list for ease of access. As seen below in fig, in user_friends dataframe now friends is a list of userids not space separated one string

```
: user_friend['friends'][:5]

: 0    [1346449342, 3873244116, 4226080662, 122290762...
  1    [1491560444, 395798035, 2036380346, 899375619,...
  2    [1484954627, 1950387873, 1652977611, 418596082...
  3    [83361640, 723814682, 557944478, 1724049724, 2...
  4    [4253303705, 2130310957, 1838389374, 392873576...
  Name: friends, dtype: object
```

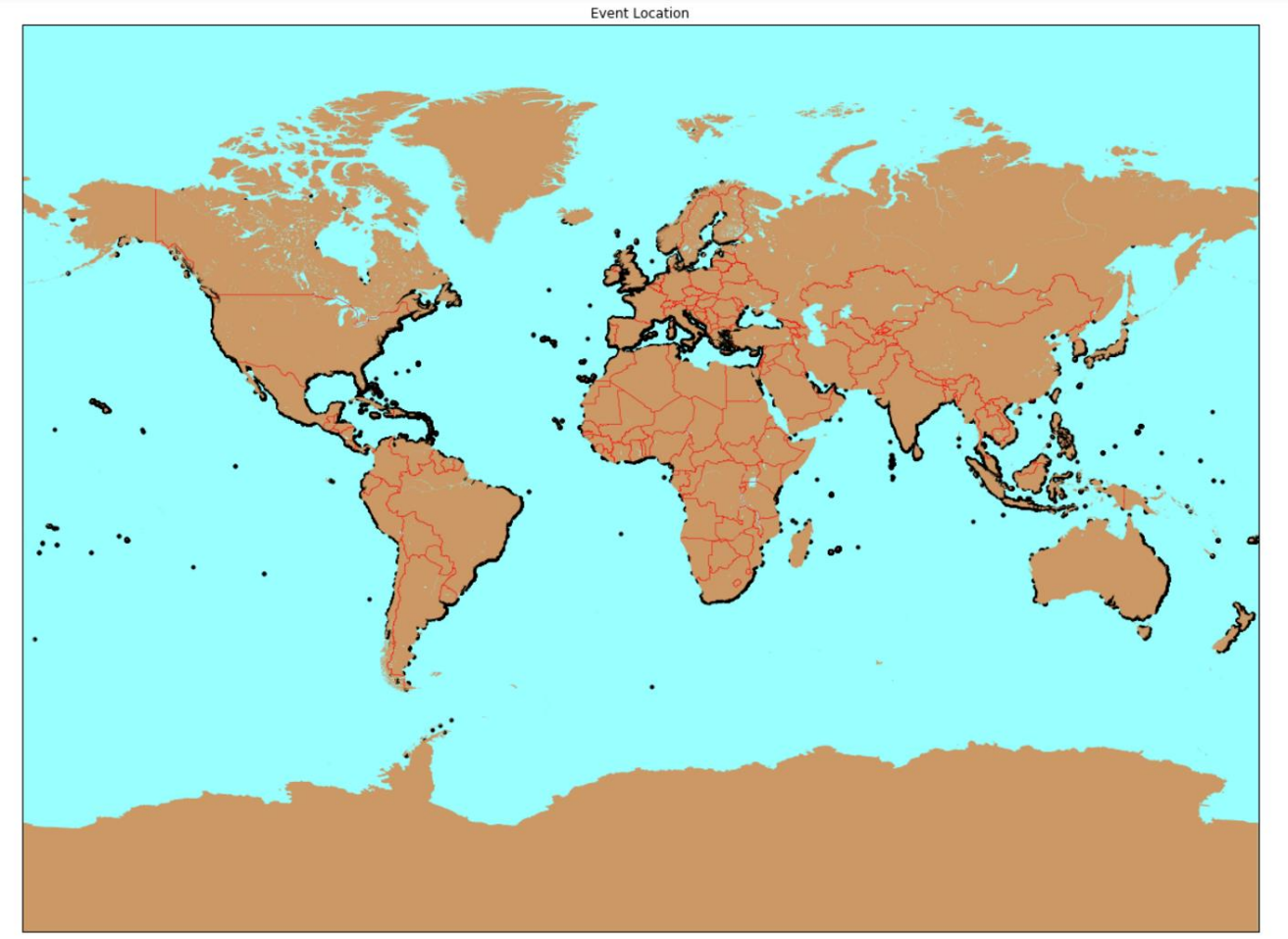
- Location of users were of the form string they needed to be converted into latitudes and longitudes. If the location of user is unknown, it is stored as zero. They were converted using geopy geocoders module using the below code:

```
users['latitude'] = [geolocator.geocode(str(s)).latitude if
geolocator.geocode(str(s)) is not None else 0.0 for s in users['location']]
users['longitude'] = [geolocator.geocode(str(s)).longitude if
geolocator.geocode(str(s)) is not None else 0.0 for s in users['location']]
```

Implementation

For implementation, first I analyzed the location of events. On plotting on map all the events I found out that most of the events are along the coast lines. Many events location were false values as they lie out in the ocean, middle of no where.

Map was plotted using the Basemap module from the matplotlib toolkit.



On analyzing the location of events, it seems that location of events is covering only the coastal regions. According to the graph plotted above, there are no events that took place in inner region of any country.

For predicting the interest of user in any event, I have considered a few things:

- Is user invited to the event?
- Is event created by friend of user?
- Has user shown any interest in the event? If yes, how?
- Relation between location of event and user?
- Day and time of the event?
- Is event popular enough?

To answer all these questions, we have to look into the dataset provided to us.

Is user invited to the event?

Answer to this question is already present in our training data file. So this can directly be taken from data set.

Is event created by friend of user?

For this question, first we have to find the creator of event from the events dataframe and then we have to check if user is friends with creator of event from the user_friend dataframe. We will add a new column to the train and test data set "isfriend" and value will be zero or one based on the answer of above question.

Code for adding "isfriend" column:

```
train['isfriend'] = pd.Series(np.random.randn(len(train['event'])), index=train.index)
i=0
for user,event in zip(train['user'],train['event']):
    createdby = events[events['event_id'] ==event]
    #print(createdby)
    if (str(createdby['user_id']) in (user_friend[user_friend['user']==user]['friends']).tolist()[0]):
        train['isfriend'][i] = 1
    else:
        train['isfriend'][i] = 0
    i=i+1
```

Has user shown any interest in the event? If yes, how?

For answer of this question we need to check the event_attendees data frame. From that dataframe we will take the user and check how the user reacted to the event. Whether he

mentioned “yes”, “no” or “maybe”. A new column, “reaction”, is added to the train data set similar in the previous question. Values of train are 1 if “yes”, 0.5 if “maybe” and 0 if “no”.

Relation between location of event and user?

This question tells us about the distance of physical location of events and user, if known. Theoretically, we would take the location of user and event in form of latitude and longitude and find the distance between the two. If the distance is reasonable we can add a column saying user can reach the event or not. But due to too many missing values and misleading data in latitude and longitudes fields, I didn’t get any success in using this relation in my model.

Day and time of the event?

For this Question, we check the timestamp of the event from the events dataframe and if the event is on Friday, Saturday or Sunday we can give it higher number (5) and if is on weekday we can give it a lower number(2) to show that people are less likely to go to the event.

Is event popular enough?

This question deals with the popularity of event, if event is more popular more and more users will tend to go to the event. For this we count “yes” from the event attendees dataframe and if the number of people attending the event is greater than a fix value (50) we can mark it as popular otherwise unpopular.

We will do the same process in test file and then we will put the values of our test file in the multiple regression model that we implemented.

```
from sklearn import linear_model

# Create linear regression object
reg = linear_model.LinearRegression()
reg.fit(train.ix[:,[0,1,2,6,7,8,9]], train['interested'].reshape(-1,1))
print('Columns: \n', train.columns)
# The coefficients
print('Coefficients: \n', reg.coef_)
print('Intercept: \n', reg.intercept_)
test['interested'] = reg.predict(test.ix[:,0:3])
print(test['interested'].max(),test['interested'].min())
j=0;
for i in test['interested']:
    if i>0.23:
        test['interested'][j]=1
    else:
        test['interested'][j]=0
    j=j+1
```

```
Columns:
Index(['user', 'event', 'invited', 'timestamp', 'interested','not_interested'
       'isfriend', 'reaction', 'day', 'popular'],
```

```
      dtype='object')

Coefficients:

 [[ -8.75769458e-13  -9.47836566e-12  -7.94554891e-05 -6.12546889e-03 -4.95567321e-02
-9.54937621e-03 -5.45342166e-02]]

Intercept:

 [ 0.29311704]
```

When implemented multiple regression model, I learnt that I am getting a range of values instead of just 0 and 1. So I just took middle of the range and if less than that I predicted as “not interested” else “interested”. Some of the results are:

```
38    1.0
39    1.0
40    1.0
41    1.0
42    1.0
43    1.0
44    1.0
45    1.0
46    1.0
47    1.0
48    1.0
49    0.0
Name: interested, dtype: float64
```

Result

In prediction of “interested” and “not interested”, I am getting about 70% users as interested for the given event in test file.

These results can be compared by implementing the same dataset with different model like k-nearest neighbors or support vector machines.

Future Improvements

Some more points that can increase the accuracy of prediction can be:

- Try to implement the data with other models and then take the majority vote and then decide the value for target class
- Location of user and event can be considered and that will give more accurate results.
- Similarity between users and event can be calculated using collaborative based filtering and that similarity score can be used to predict the interest of user.

Conclusion

While implementing the project, I learnt many new techniques to handle data in python. My biggest challenge was to manage this amount of data. To resolve this problem, I learnt how to integrate the database queries with python.

Next challenge was to get the latitudes and longitudes for each location of the event and user. This task took about 2 weeks to complete but I was unable to use that in my model because of missing data. However, I learnt a lot about conversion of string and addresses to latitude and longitudes.

In this project, I have learnt a lot about the pre-processing of data. Pre-processing of data has taken about 60% time of the whole project. Another 10% (approx) of time went in execution of code. As all the dataframes have large amounts of data so each block of code took a long time to execute.

Overall in this project, I predicted the interest of user in any event based on multiple regression model and I think that it could be done better with other data mining models.

References

- Kaggle.com
- https://pypi.python.org/pypi/geopy
- https://matplotlib.org/basemap/
- Programming Collective Intelligence, O'Reilly
- Data Science From Scratch, Joel Grus
- Fundamentals of Machine learning, John D. Kelleher, Brian Mac Namee, Aoife D'Arch