Pattern Recognition and Machine Learning CS5691

Prediction Model for the Members of Bikers-Interest-Group to Predict the Biker's Tour Preferences

Data Contest Report

MM17B105

A RAMPRASAD

MM17B026

R ARUN PRAKASH

Project Report

Overview:

The aim is to build a prediction model for the members of Biker-Interest-Group to predict which bike-tours will be of interest to bikers. Different sets of data is provided, such as biker's previous interests, his/her demographic details, past tours, friend circle etc. This dataset is derived from a real world scenario; so take care of sanitizing/handling real data.

As it is a real-life data, cleaning and preprocessing the data is very important. Also, the data provided does not have a well-defined predictor variable (X) and a target variable (Y). So, the major and important task here is to form a supervised data and then implement different algorithms to find the best model for predicting the preference order of tours for different riders in test data.

Data:

There are different data files given apart from train and test files which we need to combine in order to obtain useful information for predictions. These are as follows:

- **tour_convoy.csv**:
  This data file contains information regarding the list of bikers who showed interest in a particular tour. It has a list of bikers with information regarding whether they are going, invited, maybe, not going to that tour.
- **bikers.csv:**
  This data file contains feature information about the bikers like his gender, location, age, area, language, time zone, membership period etc.
- **tours.csv:**
  This data file contains feature information about the tours like the biker who organised the tour, location of the tour, date on which tour was conducted, and the location description summarized in 100 most common words(w1, w2, ….. w100).

- **bikers_network.csv:**
  This data file contains information detailing the social networks of the bikers. This is derived from the group of bikers who know each other via some groups.

Finally, we must use the train data to generate a well-defined and structured supervised data to develop a model to use it on test data for prediction.

Approach for Data Generation and Prediction:

Our approach was a simple data compilation of different features from different data files and then to predict the output. The target variable in our case was Like/Dislike, in train data. Both the columns are combined into a single column by encoding as follows.

0 –> **NA** (Not Available)
1 –> **Dislike**
2 –> **Like**

So, our approach here is not to predict the value but to predict the probability of liking the tour (i.e. target value 2) and then using that probability to rank the tours for an individual biker on the test data.

Merging of Data:

First, we merged the **bikers.csv** data file to the train data and formed a new data file train_new.csv. Only matching riders in both train and bikers.csv will be merged, if there are no matching riders then it will be filled using NaN value. Next for tours information, we merged **tours.csv** to the train_new data. Because of this merging process, a lot of NaNs are created which handled later in the data processing step.

tours_convoy.csv and bikers_network.csv data files also contain useful information, which we need to extract in a useful manner and merge to the train data. **tours_convoy.csv** contains the tour and the biker going, invited, maybe, not going for that tour. Now, we added 4 separate columns which have the count of going, invited, maybe, not going respectively. Next, the **bikers_network.csv** file is directly merged with train data by matching bikers_id. Now, we have a raw X and Y data which we'll clean in further steps and then build the model.

Data Preprocessing:

The following data preprocessing was done on the dataset:

- NaN values:
  - Here we are not removing or imputing NaN, because they are large in number and LGB Algorithm is capable of handling NaNs.
  - NaN values in categorical data are also kept as it is, as LightGBM and XGBM are capable of handling them.
- Encoding categorical data:
  - In bikers.csv, language_id, location_id, gender and area are to be labelEncoded.
  - In tours.csv, city, state, pincode and country are to label encoded
  - In target variable, **0 – NA**; **1 – Dislike**; **2 – Like**

Feature Engineering:

We added a few important features after analyzing the data.

- **Membership duration** feature for every biker-tour pair is computed by subtracting member_since feature of the biker from tour_date feature of the tour. This feature provides information on how long the biker has been in the group.
- **Age on the tour date** feature computed by again subtracting bornIn feature of the biker from tour_date feature of the tour. This feature provides information on the age of the biker while going on a tour as depending on age, the biker may or may not go to certain tours.
- **Friends** who are **going**, **invited**, **not going** to a tour are **counted** and then added as features. This is done by finding the intersections between friends feature (in bikers_network file) and bikers going, invited, and not going on a particular route. This feature is useful as bikers usually tend to go to tours with their friends circle.

- To have some description of tours, **w1…..w100** feature can be used but as it has 100 features, **PCA** is performed on these 100 features and reduced to 3 features (**p1, p2, p3**). These features are merged to the final data.
- **Time available to prepare** for the tour of a biker is also computed as a feature by subtracting time_stamp feature from tour_date feature. This feature is useful in case if the biker is notified about a tour in short notice then he may prefer not to go to that tour and vice versa.
- Then to incorporate the distance, state, pincode data; we create a new feature which has the value of the first 2 characters of the tour_id (string). This feature can be useful as most of the tours have the first 2 letters in tour_id repeating, this may be because of certain common properties between the tours which might interest the bikers.

Model Building:

Before fitting any model, some of the features like biker_id, tour_id, organizer_id, friends, like, dislike, etc., features are removed from train_new as well as test data.

First, we tried different classification algorithms like Logistic Regression, KNN, SVM, Random Forest Classifier etc., but the validation score was not great. Then we thought of boosting using **LightGBM**, XGBoost; which eventually turned out to be the best performing model on the validation set.

LGBM Classifier: We tested this algorithm on the feature engineered data with random parameters and then **K-Fold validated** to get an accuracy of about 0.75 on validation dataset.

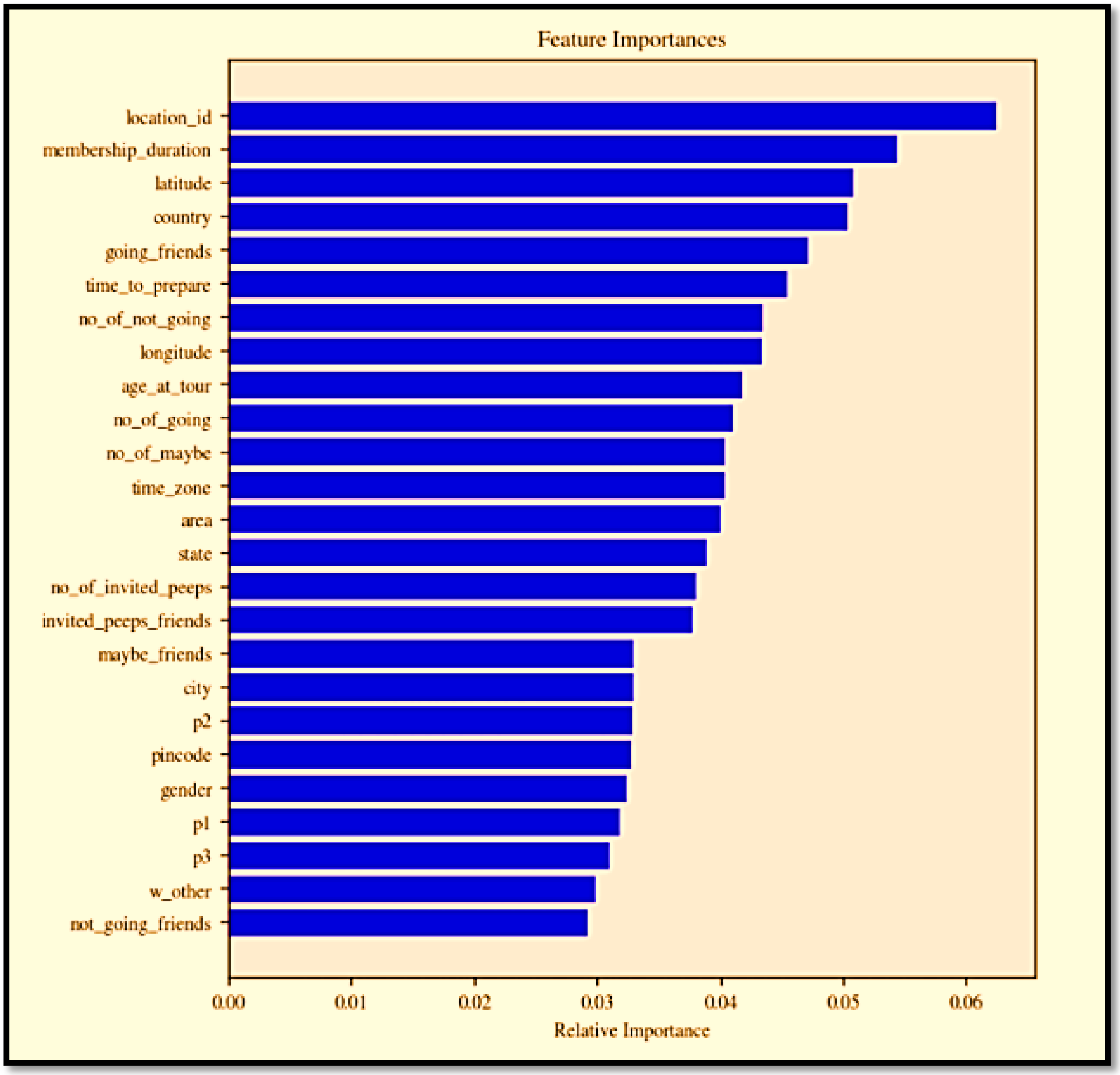
Parameter Tuning: Generally the default parameters give the best result most of the time. We tried to tune the 2 most important parameters in LGBM i.e. **the number of estimators** to use in LGBM and the **depth of each tree** in LGBM. We tuned these parameters using GridSearchCV to obtain valid parameters with best, which we later applied to the test.csv data to get the final predictions for submission.

Parameters: The final set of parameters obtained after Cross Validation was:
n_estimators = 300
depth = 10

Observations:

Feature Importance

- The following plot shows the relative feature importance of different features in the dataset obtained from the learnt model using feature_importances_ method.



- As can be seen from the plot, location_id has the highest feature importance along with other location features such as latitude, longitude, country etc.
- Apart from this, some of the features developed through feature engineering like **membership_duration**, **going_friends**, **time_to_prepare**, **age_at_tour**, **no_of_not_going** are some of the features with high importance.

- Most of the other features including those developed through feature engineering have almost equal feature importance in the range of 0.03-0.05.

Thus the features developed through **feature engineering** have helped the model to learn better and is one of the most important reasons behind the model's performance.

Also the selection of the model **LGBM** over **XGBoost** is due to the slightly better performance and the training time of the model. Otherwise, both models give almost equal performing model.

Overall, a **boosting algorithm** combined with good **data preprocessing** and most importantly good **feature engineering based** on the domain knowledge gives a high performance machine learning model.

Prediction:

The test.csv file is also merged together with train_new and preprocessed as discussed above. Then using the above fitted model and the tuned parameters, we get the **probability** of **tour liking** for each bikers tour and then using that probability, we **ranked the tours for each biker** and compiled the data file as asked in submission.csv.

This model gives a score of **0.75373** on the public test dataset and a score of **0.69642** on the private test dataset.