Faculty of Computer Science

CSCI 6515 ML with Big Data

Project Report

Submitted by

Bhargav Dalal B00785773

Submitted to

Instructor: Dr. Stan Matwin

TA: Dr. Amilcar Soares

Submission Date

December 11, 2017

Data

The data set geolife is about the tracking of some of the users. The data set specifies the various transportation mode choose by the users such as bus, car, taxi, train, subway and walk. The data set is provided as raw data. Some of the statistics provided by Dr. Amilcar Soares are:

- bus (29%)
- car (11%)
- walk (36%)
- taxi (5%)
- subway (7%)
- train (12%)

Some other information shared is:

- Number of trajectory points: 4.485.796
- Number of classes: 6

Objective

The objective is to perform feature engineering on the above dataset and extract the useful information from it. As a part of the feature engineering, it is required to calculate the distance(m), speed(m/s), acceleration(m/s^2) and bearing. The features required to be calculated for per user and per day. Once the feature has been calculated, the statistics measure(mean, median, max, min, and std) is required to calculate for per class. This statistics measures calculated will be required to lot to find some similarity to form a hierarchical tree structure for the target variable. Furthermore, two classification models required to test on this hierarchical structure and the flat structure. The models will be compared with the help of t-test.

Task A - Feature Engineering

1. Group the trajectories by user id and day and compute the following point features: (i) distance traveled (e.g. haversine, in meters); (ii) speed (m/s); (iii) acceleration(m/s2); (iv) bearing (0 to 360 degrees).

The required features has been calculated as follows:

- **Distance:**

  The distance was calculated using haversine_distance method of package gpxpy.geo package. The method for calculating the distance is below:

```
def compute_distance(df):
    df['h_dt'] = [gp.haversine_distance(df.latitude[i], df.longitude[i], df.latitude[i+1], df.longitude[
    i +1]) if i != (len(df.t_user_id) - 1) else 0 for i in range(len(df.t_user_id))]
    df['h_dt'] = [df.h_dt[i] * df.msk[i] * df.dymask[i] for i in range(len(df.t_user_id))]
    return df
```

Fig. 1- Distance Calculation Method

- **Speed:**

  The formula of the speed is the ratio of the distance traveled and time take taken to cover the distance. Following is the method to calculate the time.

```
def compute_speed(df):
    df['speed'] = [df.h_dt[i] / (df.Tdelta[i]+ 0.1**10) if i != (len(df.collected_time) - 1) else 0 for i in range(len(df.collected_time))]
    df['speed'] = [df.speed[i] * df.msk[i] * df.dymask[i] for i in range(len(df.h_dt))]
    return df
```

Fig. 2- Speed Calculation Method

- **Acceleration:**

  The formula for acceleration is the ratio of the speed and the time difference. Both parameters have been calculated. Hence, the following is the method for acceleration:

```
def compute_acc(df):
    df['acc'] = [df.speed[i] / (df.Tdelta[i]+ 0.1**10) if i != (len(df.collected_time) - 1) else 0 for i in range(len(df.collected_time))]
    df['acc'] = [df.acc[i] * df.msk[i] * df.dymask[i] for i in range(len(df.speed))]
    return df
```

Fig.- Acceleration Calculation Method

- **Bearing:**

```
def set_bearing(df):
    df['bearing'] = [calculate_initial_compass_bearing(df.Tuple[i], df.Tuple[i+1]) * df.msk[i] * df.dymask[i] if i != (len(df.t_user_id) - 1) else 0 for i in range(len(df.t_user_id))]
```

Fig. 3- Bearing Calculation Method

One important thing to notice in calculation of each feature is, each feature is multiplied with the **msk[] and dymask[]**. This was performed as the feature was required to calculate daily bases for each user. The methods for obtaining the msk and dymask paramter are:

```
def usr_class_seperator(df):
    #usr = df.t_user_id.unique()
    usr_list = np.array(df.t_user_id)
    msk = [0 if i == (len(usr_list) - 1) or usr_list[i] != usr_list[i+1] else 1 for i in range(len(usr_list))]
    return msk

def day_class_seperator(df):
    dy_list = np.array(df.date)
    dymask = [0 if i == (len(dy_list) - 1) or dy_list[i] != dy_list[i+1] else 1 for i in range(len(dy_list))]
    return dymask
```

Fig. 4- User Separator and Day Separator Method

2. Create sub-trajectories by class using the daily trajectories and compute the trajectory features as follows:

- **Discard sub-trajectories with less than 10 trajectory points.**
- **For each point feature, compute the minimum, maximum, mean, median and standard deviation. Those 20 values (5 statistical measures x 4 point features) are the trajectory features you should use for classification.**

Now, all the features (distance, speed, acceleration and bearing) has been calculated. Further, it is required to compute the statistics measure about each feature for each class in the target variable. The following is the required code for it:

```
#Calculating 5 measures that is mean, min, max, median and standard deviation
data_Mn = data.groupby(['t_user_id', 'date', 'transportation_mode']).mean().reset_index()
data_Md = data.groupby(['t_user_id', 'date', 'transportation_mode']).median().reset_index()
data_Mnn = data.groupby(['t_user_id', 'date', 'transportation_mode']).min().reset_index()
data_Mx = data.groupby(['t_user_id', 'date', 'transportation_mode']).max().reset_index()
data_Sd = data.groupby(['t_user_id', 'date', 'transportation_mode']).std().reset_index()
```

Fig. 5- Computation for Statistic Measures.

Each of the statistics measure has been calculated in separate data-frame. This all data-frame has been merged using the **concat** function of the pandas package in a new data-frame.

The class which has less than 10 entries was removed as required. Finally, the data is ready for the analysis.

3. Explore the data and compare the trajectory features values by class. Is it possible to detect similarities or significant differences between the classes? Provide plots to support the conclusions you make.

The following are the graph of speed vs transportation mode and bearing vs transportation mode.

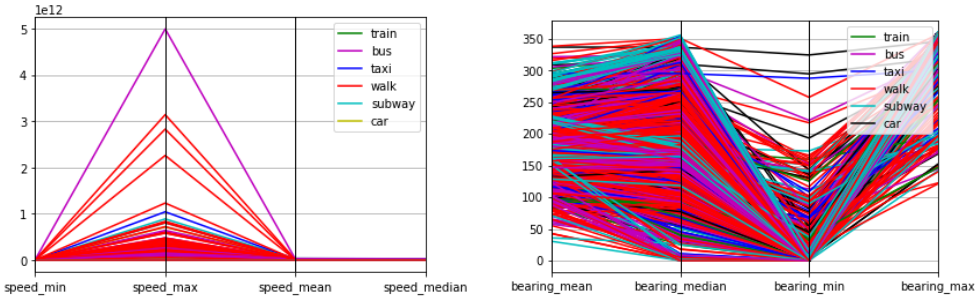

Fig. 6- Parallel Co-ordinates Graphs

From the above graphs, it is difficult to identify the similarity or difference between any class except walk. Thus, distance and bearing are not good feature to split on. However, the bar graph of transportation mode and maximum speed help to give some similarity.

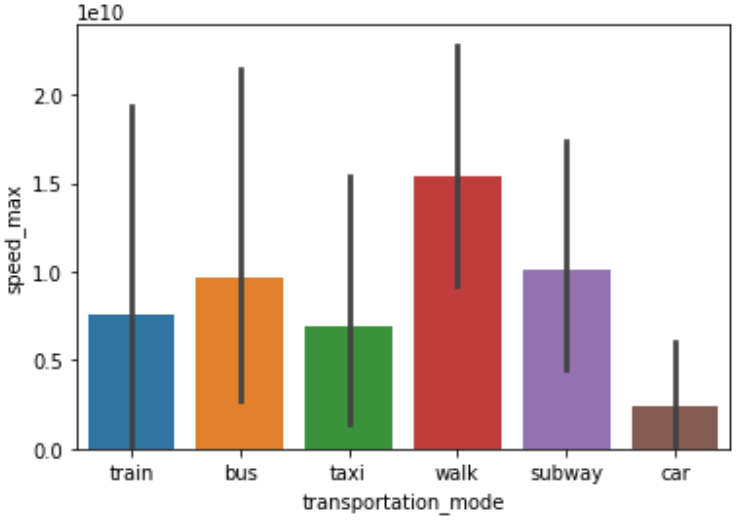

Fig. 7- Bar graph for transportation mode vs speed_max

From the above graph, we can derive the hierarchical structure for the transportation mode class. The feature chosen for splitting the data is speed_max. Now at first level, walk and other modes will be separated as walk is highest. For second level, other modes will be separated into ontracks and onroad mode as from the graph it is clear that we can group train and subway in ontrain while car,taxi and bus in onroad group. The speed of both train and subway is similar, so they are grouped in ontrack. Similarly, car, bus and taxi are grouped in thus onroad.

TASK B - Hierarchical Classification

1. After evaluating the trajectory features, propose a hierarchy to classify the data based in your conclusions. Provide an overview of the groups you merged on each layer.
2. Implement the structure you propose. Remember that in the final layer, the output must be the 6 classes from the original dataset.

The hierarchical structure for the target variable transportation mode is:

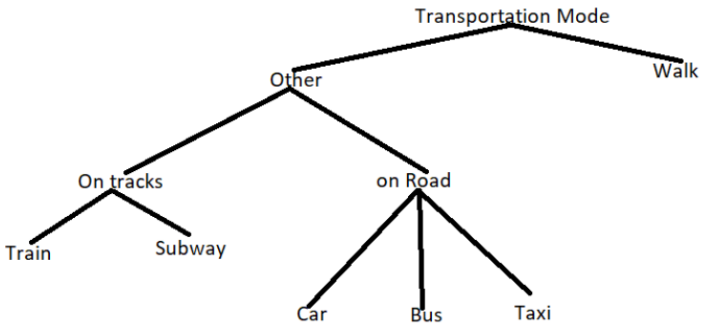

Fig. 8- Hierarchical tree structure

The hierarchical structure starts with splitting of **walk** and **others** in the first layer. The **others** split into **ontrack** and **onroad** in the second layer.The on-track is split further into train and subway, while the onroad splits into car, bus, and taxi in the final layer. The structure is also be justified in Fig 7.

3. Choose two different classifiers and compare the results using your hierarchical structure with a flat structure (i.e. use the classifier with 6 classes).

1. **Perform a multiclass evaluation and a significance test (e.g. paired t-test)for each classifier. Use a ten-fold cross-validation with stratification.**

2. **Report your findings and discuss your results. You are free to use any type of plot that support your conclusions.**

This question required to apply the two classification models on the hierarchical structure and the flat structure. Random Forest and Decision Tree have been used as classification model. Once, the model is applied the accuracy of the overall hierarchical structure required to compare it with the accuracy of the flat structure.

The accuracies for it are as follows:

Structure	Model	Accuracy
Hierarchical	Random Forest	92.95
Flat	Random Forest	79.79

Table 2: Accuracy Table of Rain Forest Model

Structure	Model	Accuracy
Hierarchical	Decision Tree	92.95
Flat	Decision Tree	73.19

Table 2: Accuracy Table of Decision Tree Model

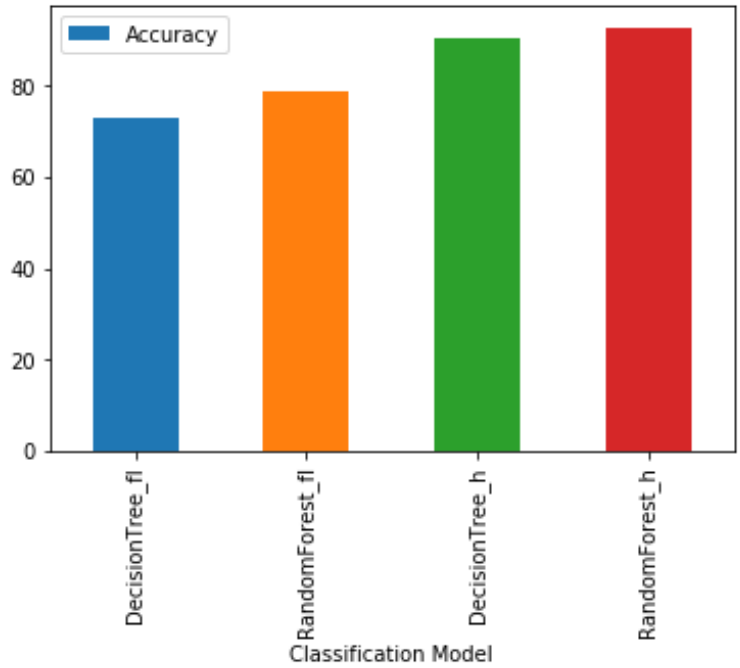

Fig. 8- Bar graph of classification model vs accuracies

It is evident from the accuracy table and graph that both of the classification models work well on the hierarchical structure. While, the accuracy of both of the models for the flat structure is less as compared to the hierarchical structure. Random Forest and Decision Tree tend to overfit if provided directly for classification in case of categorical variable [1]. Both of the models are biased in favor of the layer (hierarchical) structure. This exercise justifies it.

Paired T-Test

Student t-test is a test which compares two means of the example and tells how different they are from each other. Similarly, a paired t-test is a test where we choose two different measurements (Mean in our scenario) on two different samples (2 classifiers in our case) to tell us if they are statistically significant from each other or not. For example, you might test two separate groups of customer service associates on a business-related test or testing students from two universities on their English skills.

When we will apply the t-test, we will get two values, which is called as t-values and p-values.
t-values: It is the ratio between the difference of 2 groups (in our example 2 classification model) and the squared difference between 2 groups. So, if t-value is larger, more the difference from each other and if smaller, then it will be similar.
p-value: A p-value is a probability that the results from your sample data occurred by chance. Thus, the lower the p-value, the less chance of occurring the data by chance.
The results after applying the t-test are:

Structure	T-value	P-value	Result
Hierarchical	7.95136	$1.08*10^{-8}$	Not similar
Flat	6.04734	0.00019	Not similar

Table 3: Result of paired t-test on flat and hierarchical structure for both of the classification models

The t-value in both the scenario is big, while p-value is small. Thus, according to the hypothesis both of the classification are different from each other for both of the structure.

Conclusion

The classification models perform well if the categorical variable is provided in the hierarchical tree structure, rather than directly (flat structure) using the classification model. The problem with flat structure was the model overfits the data. The classification models are also more biased towards the tree structure in case of predicting the categorical variables. The classification model is also different from each other for both hierarchical and flat structure.

References

[1] Login - Dalhousie University. (n.d.). https://dal.brightspace.com/d2l/le/content/52772/viewContent (accessed December 11, 2017).
[2] Random Forests Leo Breiman and Adele Cutler, Random forests - classification description. (n.d.). https://www.stat.berkeley.edu/ breiman/RandomForests/cc_home.htm (accessed December 11, 2017).
[3] T Test (Student's T-Test): Definition and Examples, Statistics How To. (n.d.). http://www.statisticshowto.com/probability-and-statistics/t-test/ (accessed December 11, 2017).