Datamining project: Taxi destination prediction

Béres Ferenc

January 21, 2017

1 Introduction

This project is related to the Discovery Challange of the ECML 2015 conference. The original contest was announced on Kaggle. I chose this dataset because I mainly work on projects related to classification, but this task is a regression problem as I have to predict the destinations of taxi routes.

1.1 ECML Challange 2015

In the original contest there were two tasks:

1. Taxi trajectory prediction
2. Taxi trip time prediction

The main motivation behind these tasks can be read on the Kaggle site:

"To improve the efficiency of electronic taxi dispatching systems it is important to be able to predict the final destination of a taxi while it is in service. Particularly during periods of high demand, there is often a taxi whose current ride will end near or exactly at a requested pick up location from a new rider. If a dispatcher knew approximately where their taxi drivers would be ending their current rides, they would be able to identify which taxi to assign to each pickup request."

1.2 Evaluation metric

The evaluation metric for this competition is the *Mean Haversine Distance*. The Haversine Distance is commonly used in navigation. It measures distances between two points on a sphere based on their latitude and longitude.

The Harvesine Distance between the two locations can be computed as follows

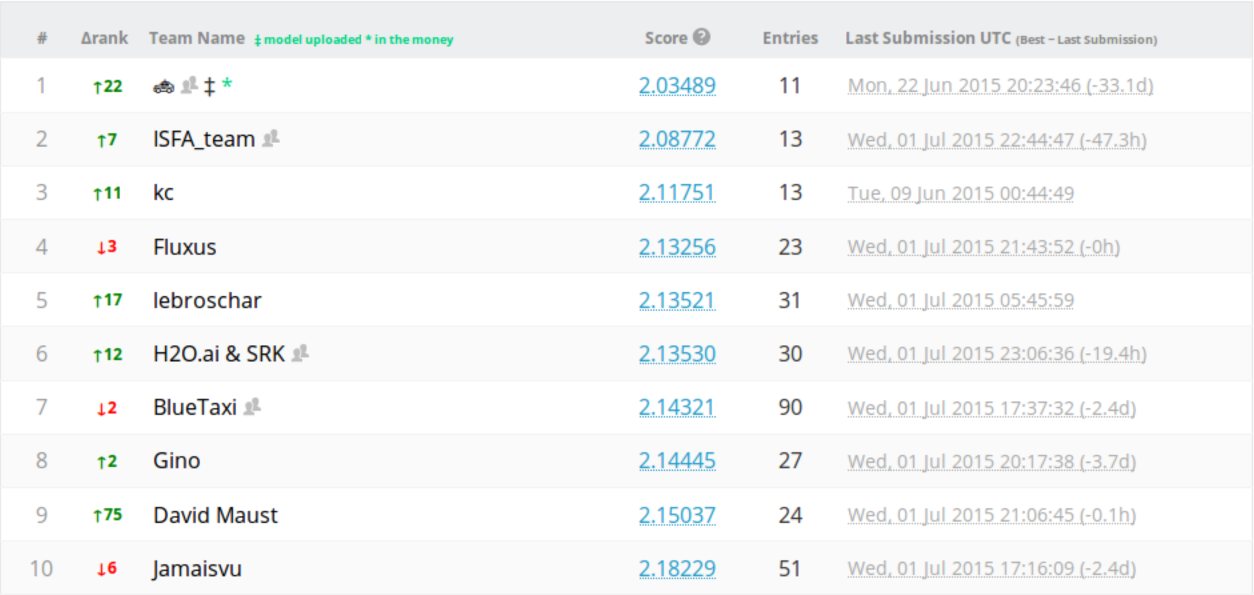
$$a = \sin^2 \frac{\phi_1 - \phi_2}{2} + \cos \phi_1 \cos \phi_2 \sin^2 \frac{\lambda_1 - \lambda_2}{2} \tag{1}$$

$$d = 2 \cdot r \cdot atan \sqrt{\frac{a}{1-a}} \tag{2}$$

where ϕ is the latitude, λ is the longitude, d is the distance between two points, and r is the sphere's radius,

In our case, it should be replaced by the Earth's radius in the desired metric (e.g., 6371 kilometers).

In the ECML contest the winner models had performance approximately 2 km which you can see on Figure 1.



#	Δrank	Team Name ‡ model uploaded * in the money	Score	Entries	Last Submission UTC (Best – Last Submission)
1	↑22	‡ *	2.03489	11	Mon, 22 Jun 2015 20:23:46 (-33.1d)
2	↑7	ISFA_team	2.08772	13	Wed, 01 Jul 2015 22:44:47 (-47.3h)
3	↑11	kc	2.11751	13	Tue, 09 Jun 2015 00:44:49
4	↓3	Fluxus	2.13256	23	Wed, 01 Jul 2015 21:43:52 (-0h)
5	↑17	lebroschar	2.13521	31	Wed, 01 Jul 2015 05:45:59
6	↑12	H2O.ai & SRK	2.13530	30	Wed, 01 Jul 2015 23:06:36 (-19.4h)
7	↓2	BlueTaxi	2.14321	90	Wed, 01 Jul 2015 17:37:32 (-2.4d)
8	↑2	Gino	2.14445	27	Wed, 01 Jul 2015 20:17:38 (-3.7d)
9	↑75	David Maust	2.15037	24	Wed, 01 Jul 2015 21:06:45 (-0.1h)
10	↓6	Jamaisvu	2.18229	51	Wed, 01 Jul 2015 17:16:09 (-2.4d)

Figure 1: Private leaderboard of the original challange. The score is the mean Haversine distance in kilometres.

2 Data preparation

The original dataset contains information about taxi rides in Porto, the capital of Portugal. In the challange there is a *training set* and a *testing set* provided for the participants. In the training set the full route trajectory is available but for the testing set only the first few locations.

2.1 Restrictions

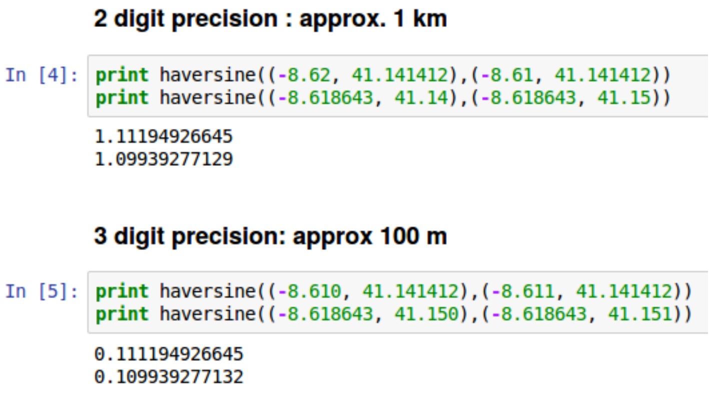
Due to the size of the original training set (1710670 records) I made some restrictions. I only used the first 50000 records from the original training set to build models. From the testing set I excluded 6 outlier records due to the fact that in this project I only used $\approx 3\%$ of the original training data.

	Train	**Test**
Average trajectory length	45.85	43.12
Number of records	50000	325

2.2 Feature engineering

2.2.1 Location precision

In the original data latitude and longitude coordinates are given with 10^{-6} precision. In order to use GPS information in features I chose to decrease this precision. Before generating new features I rounded latitude values to 10^{-3} (approx. 1km) precision and longitude values to 10^{-2}(approx. 100m) precision (see Figure 2). The reason why I chose to use more precision for the latitude was the localization of Porto. The city is along the coast of the atlantic ocean so it has great vertical expansion so latitude is more important than longitude. During model evaluation I use the original destination GPS coordinates, always with 10^{-6} precision. The table below contains the rounded features for latitude. Similar features are generated for the longitude as well.



2 digit precision : approx. 1 km

```
In [4]: print haversine((-8.62, 41.141412),(-8.61, 41.141412))
        print haversine((-8.618643, 41.14),(-8.618643, 41.15))

1.11194926645
1.09939277129
```

3 digit precision: approx 100 m

```
In [5]: print haversine((-8.610, 41.141412),(-8.611, 41.141412))
        print haversine((-8.618643, 41.150),(-8.618643, 41.151))

0.111194926645
0.109939277132
```

Figure 2: GPS location precision in distance for latitude and longitude

Name	**Description**
DEPARTURE_LAT	The rounded departure latitude coordinate
DESTINATION_LAT	The rounded destination latitude coordinate

2.2.2 Aggregation based features

I summarize each taxi route by aggregation based features. In the following table you can find the list of generated features related to trajectories.

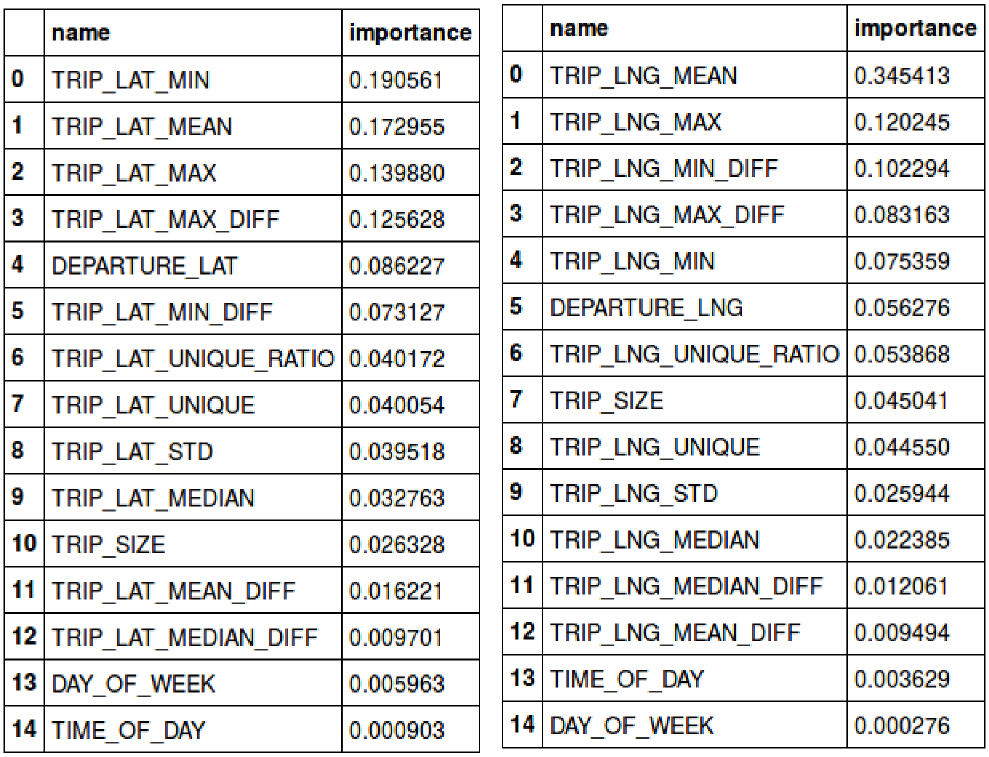
Name	**Description**
TRIP_SIZE	The number of locations in a trajectory
TRIP_LAT_UNIQUE	The number of unique latitude values in a trajectory
TRIP_LAT_UNIQUE_RATIO	TRIP_LAT_UNIQUE / TRIP_SIZE
TRIP_LAT_MIN	The minimum latitude value in a trajectory
TRIP_LAT_MAX	The maximum latitude value in a trajectory
TRIP_LAT_MEAN	The mean of latitude values in a trajectory
TRIP_LAT_MEDIAN	The median of latitude values in a trajectory
TRIP_LAT_STD	The standard deviation of latitude values in a trajectory
TRIP_LAT_MIN_DIFF	DEPARTURE_LAT - TRIP_LAT_MIN
TRIP_LAT_MAX_DIFF	DEPARTURE_LAT - TRIP_LAT_MAX
TRIP_LAT_MEAN_DIFF	DEPARTURE_LAT - TRIP_LAT_MEAN
TRIP_LAT_MEDIAN_DIFF	DEPARTURE_LAT - TRIP_LAT_MEDIAN

Similar features are generated for the longitude as well.

2.2.3 Time based features

Originally, only TIMESTAMP is given. From this I generate the following features:

Name	**Description**
DAY_OF_WEEK	Index of the day of the week
TIME_OF_DAY	HOUR // 6 (integer division)



	name	importance
0	TRIP_LAT_MIN	0.190561
1	TRIP_LAT_MEAN	0.172955
2	TRIP_LAT_MAX	0.139880
3	TRIP_LAT_MAX_DIFF	0.125628
4	DEPARTURE_LAT	0.086227
5	TRIP_LAT_MIN_DIFF	0.073127
6	TRIP_LAT_UNIQUE_RATIO	0.040172
7	TRIP_LAT_UNIQUE	0.040054
8	TRIP_LAT_STD	0.039518
9	TRIP_LAT_MEDIAN	0.032763
10	TRIP_SIZE	0.026328
11	TRIP_LAT_MEAN_DIFF	0.016221
12	TRIP_LAT_MEDIAN_DIFF	0.009701
13	DAY_OF_WEEK	0.005963
14	TIME_OF_DAY	0.000903

	name	importance
0	TRIP_LNG_MEAN	0.345413
1	TRIP_LNG_MAX	0.120245
2	TRIP_LNG_MIN_DIFF	0.102294
3	TRIP_LNG_MAX_DIFF	0.083163
4	TRIP_LNG_MIN	0.075359
5	DEPARTURE_LNG	0.056276
6	TRIP_LNG_UNIQUE_RATIO	0.053868
7	TRIP_SIZE	0.045041
8	TRIP_LNG_UNIQUE	0.044550
9	TRIP_LNG_STD	0.025944
10	TRIP_LNG_MEDIAN	0.022385
11	TRIP_LNG_MEDIAN_DIFF	0.012061
12	TRIP_LNG_MEAN_DIFF	0.009494
13	TIME_OF_DAY	0.003629
14	DAY_OF_WEEK	0.000276

Figure 3: **Left:** feature importance of the latitude GBT model, **Right:** feature importance of the longitude GBT model.

3 Model selection

In this project I propose a two-phase model setting for solving the problem of taxi route destination prediction. The first step is a GBT regression model which predicts approximate destination using only rounded GPS coordinates. Then based on the predictions of the GBT, I specify the final destination with k-NN models. I also suggest a baseline model which performs well but it is rather theoretical than practical.

3.1 Gradient Boosted Trees for regression

I trained two regression models. One for the latitude coordinate and another for the longitude coordinate. Both models were trained only for the rounded destination coordinates (DESTINATION_LAT, DESTINATION_LNG) as I only would like to get approximate locations from GBT. I experimented with different model parameters, from which 40 tree and 5 depth seemed to be the most promising. You can see the mean Haversine distance for the testing set regarding GBT model parameters on the table below.

Number of Trees	50	40	30	30	30
Maximum Depth	5	5	5	4	3
Mean Haversine	0.708	**0.726**	0.791	0.847	0.938

Figure 3 shows feature importances of the GBT models using 40 tree with 5 depth each.

3.2 Baseline mean predictor

As GBT models were trained for only approximate destination coordinates, they cannot achieve such a good performance. In order to improve the model quality I compute the mean destination in each rounded cell only for the training set. Then my final prediction for a test record r is the following:

1. Predict an approximate destination for route r with GBT (x_{gbt}, y_{gbt}).
2. Identify the location cell C_i that contains (x_{gbt}, y_{gbt}).
3. Predict the mean of destinations of the training set in C_i for the route r as destination.

Although this modification improves the model significantly, there is a major disadvantage. In a given location cell a specific location, the mean destination, is predicted for all routes. That is why I chose this model as a baseline because I think it is rather a theoretical model. In a real application we would like different locations to be assigned for different routes.

3.3 Specifying destinations with k-NN

In order to predict more diverse destinations I use k-NN models. First, I train k-NN models for each location cell in the training set. After the models were trained I predict final destination in the following way:

1. Predict an approximate destination for route r with GBT (x_{gbt}, y_{gbt}).
2. Identify the location cell C_i that contains (x_{gbt}, y_{gbt}).
3. Predict the mean of the k nearest neighbors, with the k-NN model trained on cell C_i, for the route r as destination. In case there are less than k records in C_i then the mean of those locations is proposed.

k-NN models use only a subset of the generated features (DEPARTURE_LAT, TRIP_LAT_MEAN, TRIP_LAT_MIN, TRIP_LAT_MAX, TRIP_LAT_MEDIAN, DEPARTURE_LNG, TRIP_LNG_MEAN, TRIP_LNG_MIN, TRIP_LNG_MAX, TRIP_LNG_MEDIAN, DESTINATION_LAT_FULL, DESTINATION_LNG_FULL). These features were selected based on their feature importance in GBT models. At first I wanted to define a custome weighted distance for k-NN but scikit learn could not integrate it. But the results are still good using only euclidean distance.

4 Results

Unfortunately, my results are hard to compare with results from the private leaderboard due to two reasons:

- In the original contest the leaderboard (public or private) are based only on 50-50% of the testing set. While I use the full testing set for evaluation
- From the testing set I excluded 6 outlier destinations.

On the other hand I could achieve similar results with the k-NN based model to the baseline model. It is good because the baseline model has good performance but it is impractical. The following table show the mean Haversine ditance for the models examined in this project.

	GBT (40 tree, 5 depth)	**Mean predictor**	**k-NN predictor**
Mean Haversine	0.726317	0.172401	**0.173984**

We can see that the mean predictor and the k-NN predictor have similar performances, and they outperform GBT model. But altogether GBT was good enough to predict and approximation location cell for the route destination. Figure 4 shows that the predictions for the k-NN model are indeed more diverse than for the baseline which is a big advantage in this application.

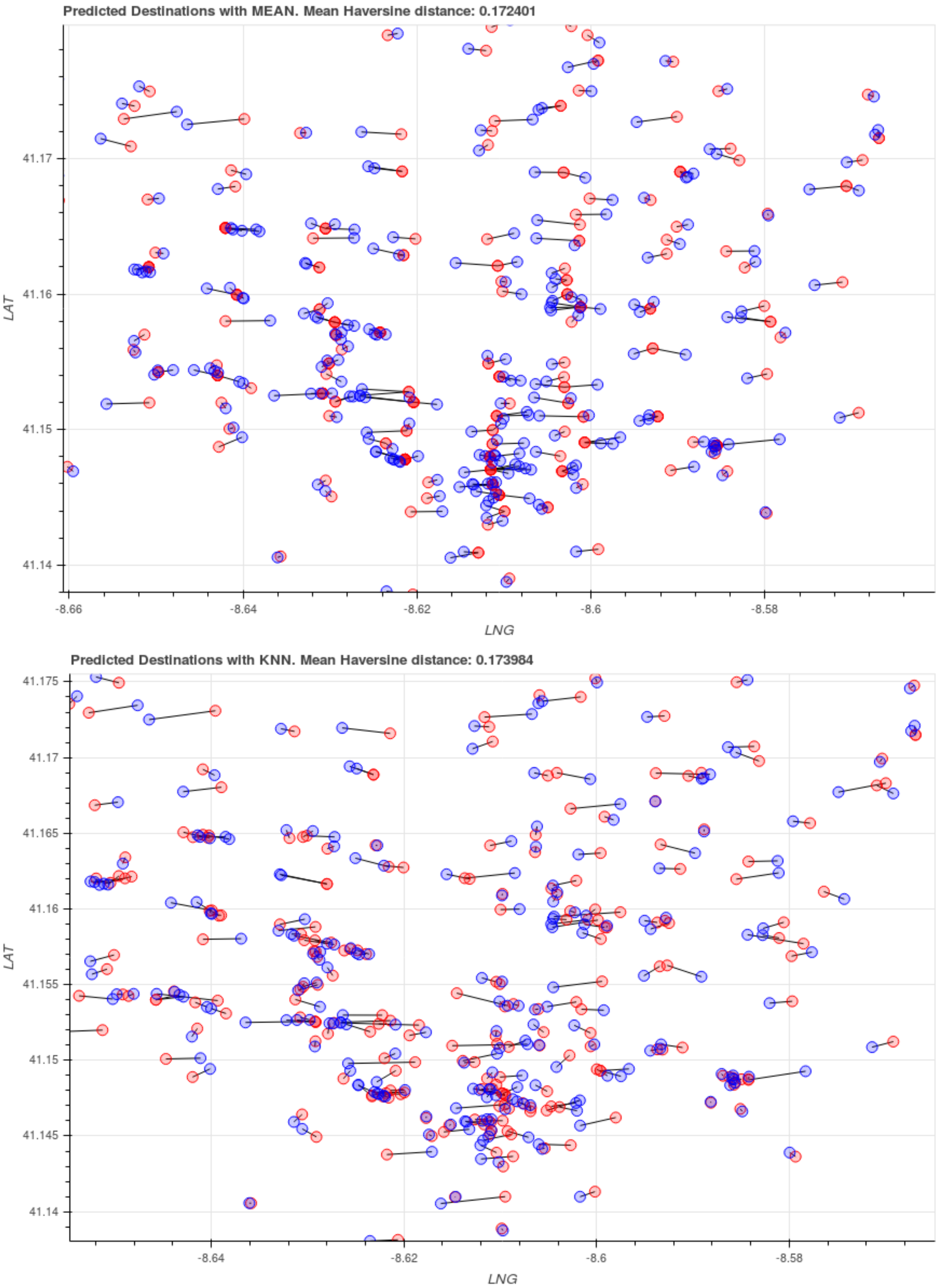


Figure 4: Predictions of the baseline (top) and k-NN predictor (bottom). models in the city centre of Porto. Blue circles are the real destination, while red ones are the predictions. Black lines represent the error.

5 Summary

In this project I solved the problem of taxi route destination prediction. I propose a two-phase modeling framework. The first model is a GBT which use rounded GPS locations. In the second phase I use several 2-NN models to predict the final destination based on the output of the GBT model. With this setting I could achieve approximately 173 meter mean error for Haversine distance.