EY NEXTWAVE DATA SCIENCE CHALLENGE 2019
Solution Report

Jin Cong Ho
University of Nottingham
jincongho@gmail.com

1 Summary

Our model is based on a gradient boosting tree (XGBoost [1]). The inputs of our model are the one-day journey of each device (concatenate all trajectories of each device into a single row). We also extracted features such as distance between entry and exit point. The output is whether the device is in the city centre between 15:00 and 16:00 (directly predicting true/false, instead of coordinate). We did no ensembling/stacking or any external data. Our solution achieved public LB f1 score is 0.88539, rank 3rd in the UK.

2 Feature Engineering

We extracted several features:

- *duration* (in seconds): *time_exit – time_entry*
- Transform *time_entry* into *entry_hour*, *entry_minute* and *entry_second* and *time_exit* into *exit_hour*, *exit_minute* and *exit*_second
- *entry_hour_quarter*: *entry_hour**4 + ceil(*entry_minute*/15), similar to *exit_hour quarter*
- *Distance* = Euclidean distance of *entry_point* and *exit_point*
- *Speed = Distance/Duration*
- Represent one day trip in a row: concatenate all trajectories of a device into a row;
- Extrapolate: if a device has less than 20 trajectories, then we pad the last entry point as the entry and exit point of all previous trajectories
- Standardize all feature by removing the mean and scaling to unit variance
- Classification target is whether the device is in the city centre between 15:00-16:00

Final featureset:

'distance_i', 'duration_i', 'entry_hour_i', 'entry_hour_quarter_i', 'entry_minute_i', 'entry_second_i', 'exit_hour_i', 'exit_hour_quarter_i', 'exit_minute_i', 'exit_second_i', 'speed_i', 'vmax_i', 'vmean_i', 'vmin_i', 'x_entry_i', 'x_exit_i', 'y_entry_i', 'y_exit_i'

where i = 0-19, represent each trajectory (we found that maximum length of trajectory in the dataset is 20, therefore use it). However, *distance, speed, vmax, vmean, vmin, x_exit* and *y_exit* from last trajectory were removed. Therefore, we have 18 (features)*19 (trajectory) + (18-7) features = 353 features for each row.

3 Modelling Techniques and Training

We first build a simple XGBoost Classifier. Then, use 4-folds cross-validation to do finetuning for the following parameters: *max_depth, min_child_weight, gamma, subsample, colsample_bytree, reg_alpha*. Finally, we used early stopping to find the optimal number of trees. Final parameters:

Learning_rate	0.03
Reg_alpha	0.05
Max_depth	10
Min_child_weight	5
Gamma	0.2
Subsample	0.87
Colsample_bytree	0.82
Estimators	255

Another reason we chose tree-based model is also of its explainability. After training the final model, we found top 10 most important features:

	Features	Importance
0	y_entry_19	0.116769
1	x_entry_19	0.041110
2	duration_19	0.025248
3	entry_hour_19	0.024366
4	y_entry_2	0.019652
5	y_entry_18	0.019642
6	y_exit_18	0.014287
7	y_exit_1	0.014023
8	vmax_9	0.010111
9	x_entry_2	0.009191

We can observe that last two trajectories (18 and 19) are very important.

4 Code Description

data/:

- raw/: data provided by organiser
- interim/: intermediate data processed
- submission/: submission files

models/: serialised models
notebooks/: exploration notebooks testing out ideas
reports/: documentation
*.ipynb: jupyter notebooks for the main working pipeline (last section in 3. Model Finetuning.ipynb shows how to run the serialised model)

5 References

[1] Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM, 2016.