Grab AI for S. E. A

Detect Dangerous Driving from Grab's Telematics Data

Wang Lu
17th June ,2019

1. Introduction

To facilitate a safer transportation system in South East Asia, Grab launched this challenge and shared telematics data of ~20,000 trips with all participants. Timeliness and reliability are two of the biggest concerns in this task. Therefore, both computational cost and model performance should be evaluated in finalising a solution from candidate models, and the best solution offering the optimum trade-off between cost and performance.

2. Defining the Task

There are 10 numerical features in original datasets, including acceleration and gyroscope on x, y, z directions, localization accuracy, and of special interest, the real-time speed and bearing apart from bookingID and label, Some initial engineering efforts were made through non-linear mathematical manipulation of original features, such as computing the change of bearing (derivative), accumulated acceleration on each directions (integral), bearing times speed (product), resultant of acceleration and so on. Even though bearing x speed and resultant of acceleration give most bonus on predicting power, they are not integrated into the current solution due to time constraints.

It is important to consider how dangerous driving is defined in the dataset. An article from Uber mentions user feedback is the main source for them to label dangerous driving **[1]**. In this case, driving style may not be the sole factor in determining dangerous driving, but also driver's attitude and personality, timestamp of the trip (passenger's mental sense of safety), area where the trip happen (congestion) and so on. Assuming a similar methodology employed here to generate labels, it is useful to treat abnormality in features such as extremely long trip, negative speed, extremely inaccurate localization as valuable information instead of being filtered as noise, as such abnormality potentially correlate with accident, unusual map or unusual driver behaviour which often led to passenger dissatisfaction.

3. Feature engineering

The task itself is by nature a supervised binary classification problem on imbalanced dataset (>70 % labelled as negative). However, it differs from many other classical classification problems in its time series characteristics, even though entries are disrupted and presented as snapshots in provided format. Common signal processing techniques such as Fourier Transform, Continuous Wavelet Transform are reasonable ways to handle them, with **FFT** looking at this dataset from frequency domain solely, and **CWT** taking care of time-frequency coefficients. For time domain, there are two methods explored in this work. One way is simply calculating central tendency measures, dispersion measures (mean, mode, STD, …), and absolute distribution features (by taking percentiles). (For convenience, they will be referred as **STATS** features later. Another way is to look at cross sectional distribution and statistics (referred later as **XSEC**), that is to say, taking a relative

perspective. This is justified by the intuition that understanding of dangerous driving would be enhanced by providing context where individual behaviour (or trip) is compared with peer behaviour. (Notice this method sometimes is impaired by extreme outliers, as upper limit for references taken to compute cross-sectional ratio is set by the 1st percentile value.)

Back to the concrete problem itself, detection of dangerous driving is an active research field. Some background research on academia and industry methodology reveals that mainstream idea is to extract individual events and use them as features to characterise dangerous driving **[1, 2, 3, 4].** Some simple examples are those rule-based algorithms. For instance, an abrupt braking event can be extracted by an accelerated velocity beyond 0.3G; an aggressive left turn can be characterised by big positive change in bearing that's greater than 30 degree/s. The more commonly used technique other than thresholding constraints is Dynamic Time Warping (**DTW**) **[2]**, which is based on pattern matching instead of pre-defined rules. It's one of the most popular time-series similarity metrics that utilise Euclidean distance between aligned time-series. However, it's difficult to scale it to this task with its quadratic time complexity O(MN), considering the upper limit running time for this task is only ~30mins. Therefore, DTW is not implemented.

There is in fact a method in this work which is directly inspired by the idea of rule-based algorithm, where peak statistics such as STD, frequency and intensity (assuming every set of peaks is a group of notable events) are calculated, at evenly spaced thresholds instead of empirically predefined constraints (referred later as **PKF**). Similar to XSEC features mentioned before, this group of features can be sparse, as threshold range is diluted by occasional outliers. Besides, an initial exploration shows the feature `**bookingID**` does have some predicting power on label at least for this provided training dataset, though relatively vague. This most likely due to some unusual distribution of positive labels, for example, bookingID from 1709396983812 to 1709396983975 (about 20 consecutive ID) and from 17179869191 to 17179869363 (about 40 consecutive ID) are all labelled as positive, this is vey unusual considering rareness of positive labels. Assuming homogeneity between provided dataset and dataset to be tested, bookingID is also used as features in this model. When paired with XSEC features (~200 features), ID features boost model performance to a level that comparable the all-inclusive model (863 features) which saves half of the running time, making this combination the most cost-effective solution.

4. Pre-processing

Pre-processing is carried out but not intensively. As mentioned before, abnormality in features was treated as information. Scaling is therefore not implemented. Since tree algorithms (DecisionTree, GradientBoosting) are especially good at handling dirty features, they are chosen in both base estimators and final estimator for ensemble (GradientBoosting). Faulty entries in labels are detected, where 18 trips are labelled as both positive and negative. The second entries are simply ignored. PCA is used to compress CWT features into 1D. Otherwise all extracted features are equally fed in to base estimators to form predictions, which used by GradientBoosting later for final decision.

5. Feature selection

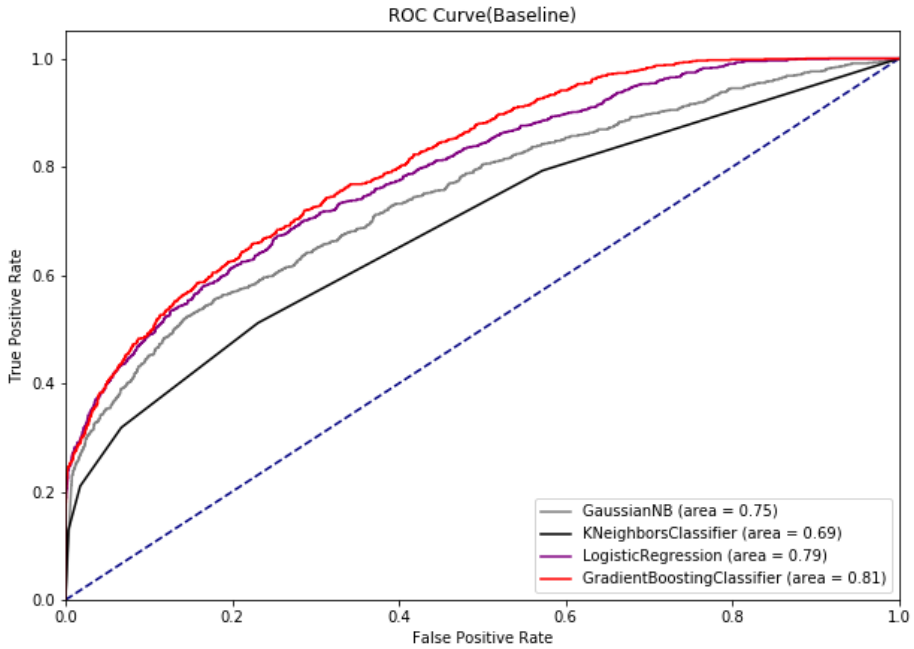
Except for bookingID features, FFT, CWT, STATS, XSEC, PKF features take roughly similar time to extract, and composed of 100~200 features each. STATS features are most

informative standalone, and bookingID the least. However, their performance in a combination shows that **bookingID** shows most compensating power. When combined with **XSEC** features, ensemble model achieves 0.81 AUC with the chosen 213 features, very close to 0.82 for all 863 features, indicating information in bookingID overlaps the least with XSEC features.

It is to be noted when computing XSEC and PKF features, reference information containing max, min, 1st percentile and 99th percentile values of each original 10 features is needed for setting upper and lower limits of thresholding, from which outliers can impose negative impact on the quality of features extracted by diluting the output. Quality of FFT and CWT features could be impaired by missing time points. Though a minority, there are actually some trips missing a large chunk of data, techniques like FFT cannot handle this very well; consistent with previous observation, central tendency features or averaged features in STATS and PKF have less predictive power than distribution features.

6. Finalized model

The finalized model is the best trade-off between cost and performance. In this work, the one-layer ensemble model composed of 6 base estimators on 2 categories of features bookingID and cross-sectional ratios. Without ensemble, a default GradientBoost model built from 213 raw features from get_idf (bookingID) and get_xsec (XSEC features) can give 0.75 AUC. When ensemble is enabled, GB can achieve 0.81 AUC on 6*2 prediction features, which takes 1min for data pre-processing, 6mins to extract raw features and 8mins for individual base estimators to make prediction on a 400MB testing dataset. A GB ensemble model from 863 features have 0.005 performance bonus at the expense of extra 15mins.



7. Further optimization

It should be noted that major efforts are made on feature engineering and optimization is solely done to select appropriate base estimator depth for the ensemble GradientBoosting estimator (though the depth to achieve optimal model complexity is found actually near default depth). To make the implementation as simple as possible, Grid search and other hyperparameter tuning methods are also not carried out. However, further optimization to improve quality of raw feature extracted is promising, such as empirically choosing bounds/ choosing a different percentile or differently space the thresholds for XSEC and PKF to avoid too sparse features. All base estimators are used in their default setting and there is no incentive to tune them. But a pre-processing to better cater vastly different dimensionality and scaling aversion of different base estimators would definitely improve the quality of prediction features.

8. Discussion on model strengthens and limitations

First, this model can be quickly implemented in business scenarios. Besides, thresholds can be more carefully chosen from bigger training set beforehand which improves model without additional running time cost. Secondly and more importantly, feature importance from this model can be used to guide future safety improvement program. For example, if a high percentage of acceleration beyond 0.3G during a trip is highly correlated with positive label, we know that an above peer/chosen standard tendency of strong acceleration should be notified and educated.

There are also two main limitations on this model. Firstly, real-time monitoring at small window size is difficult using XSEC features, as to account additional input of an ongoing trip, all previously computed ratios need to be recalculated and predictions need to be remade. However, setting a bigger monitoring interval such as every 5 mins would make things easier. The second possible limitation is the lack of information in bookingID. Predicting power of bookingID must come from underlying algorithms generating the ID, which may be traced back to the time, or driverID/passengerID of that trip, anything that not random in the algorithm. To use it right, the source of information has to be identified.

9. Conclusion

In general, anomaly is valuable information and helpful in evaluating overall danger experienced by passenger. An ensemble GradientBoosting model can be fast implemented, from sorting, to extract ~200 raw features, to generate predictions and ensemble, cost around 15 mins on 400 MB data. The model can be further and inexpensively improved by choosing better thresholding constraints to avoid sparsity and dimensionality adjustment for base estimators. It is also easier to interpret and helpful with future decision making. However, to monitor an ongoing trip in practice, the time window needs to be set relatively large. Identifying the source of predictive power of bookingID is needed. Very likely, adding timestamp, driverID, passengerID of a trip besides telematics data would vastly improve the model.

[1] How Uber Engineering Increases Safe Driving with Telematics: https://eng.uber.com/telematics/
[2] Johnson, D. A., & Trivedi, M. M. (2011, October). Driving style recognition using a smartphone as a sensor platform. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)* (pp. 1609-1615). IEEE.
[3] Eren, H., Makinist, S., Akin, E., & Yilmaz, A. (2012, June). Estimating driving behavior by a smartphone. In *2012 IEEE Intelligent Vehicles Symposium* (pp. 234-239). IEEE.
[4] Wang, W., Xi, J., & Li, X. (2016). Statistical pattern recognition for driving styles based on Bayesian probability and kernel density estimation. *arXiv preprint arXiv:1606.01284*.