Documentation

June 16, 2019

1 Introduction

A training dataset of 16,135,561 rows and 20,000 labels (equivalently, bookings, rides, or trips) were provided. The features include the speed, acceleration and gyroscope readings along the x/y/z-axis, at typically, but not always, 1-second intervals. The objective is to predict if a ride was safe or unsafe.

2 Cleaning data

An exploratory data analysis was conducted. It turned out there were 18 bookings that had multiple labels. Since it was not possible to tell which label was the correct one, all bookings that had multiple labels were removed. There was a booking that took 600 million seconds, which is unrealistic, and so a cutoff of 1000 seconds was imposed, such that any bookings lasting longer than that were removed on the basis that the data might be incorrect. There were also some speeds that were incredibly high, e.g. 148 m/s (530 km/h), which is far greater than typical expressway limits in any country. Looking at the histogram of speeds, a cutoff of 45 m/s (162 km/h) was imposed such that any bookings with a speed higher than the cutoff were removed.

The dataset was imbalanced at a 3:1 ratio (safe:unsafe), however, this can be handled by weighting the data accordingly during model training.

3 Feature engineering

Several features were created, on the basis of some physical explanation. The max and min of acceleration along the x/y/z-axis were computed, on the hypothesis that perhaps passengers felt unsafe if the car was moving with too great an acceleration in either direction along any axis. The max speed was also computed on the hypothesis that speeding cars made their passengers feel unsafe.

To quantify the change in acceleration, the jerk (second-order derivative of the rate of change of velocity) is typically approximated as follows, where Δt is the time interval between the readings of acceleration:

$$\frac{d^2x}{dt^2} \approx \frac{\frac{dx}{dt}}{\Delta t} \tag{1}$$

However, another approximation of the jerk where the standard deviation σ of the acceleration was found to be more successful by increasing the model AUC by about 5%:

$$\frac{d^2x}{dt^2} \approx \frac{\sigma_x}{\Delta t} \tag{2}$$

Then, the max, mean, standard deviation of the jerk along the x/y/z-axis were computed to provide some baseline measurements. The 80th, 85th, 90th and 95th percentiles were calculated for speed, and the acceleration and jerk along the x/y/z/-axis, thinking that perhaps at some percentile, the speed, acceleration or jerk become noticeable for the passenger to report a ride as unsafe.

The number of times that the jerk along the x/y/z-axis was greater or equal to 2, 3, 4, 5 and 6 standard deviations was also computed, based on the intuition that when a jerk occurs at greater standard deviations, this is more likely to be a significant unsafe event; and when it occurs frequently, the passenger is more likely to report the unsafe event rather than treating it as a one-off event that can be overlooked.

4 Model training

Multiple models were tried: decision trees, random forests, support vector machines, XGBoost and a neural network. XGBoost was selected as it yielded good results relative to the others, and was easier to tweak its hyperparameters.

The hyperparameters of the XGBoost model were varied, and a $max_depth =$ 2, binary logistic objective function, L_1 regularization term of 0.9 and L_2 regularization term of 0.9 were found to produce a model with the highest AUC at 70.6%.

After training the model, the various features were sorted by the gain in the model. The top 12 features then used to train the model again with minimal loss in AUC. This is due to a trade-off between computation required to regenerate all features and performance of the model,

5 Making predictions

To run the Python notebook code, please first check that the numpy, pandas and xgboost libraries have been installed. In the second cell, the path to the test data will also need to be updated. The assumption in the code is that the test data is in the form of CSV files with the same headers as the training data. If the file format is different, the all_files command in the second cell should be modified accordingly.

The XGBoost model has been provided in the repository, and needs to be loaded. The 12 features that were used in the training of the final model then need to be generated for the test data. This step may take a while.

The XGBoost model is used to perform predictions, and the output will be saved to "predictions.csv".