AI FOR S.E.A.

Grab AI Challenge: Safety

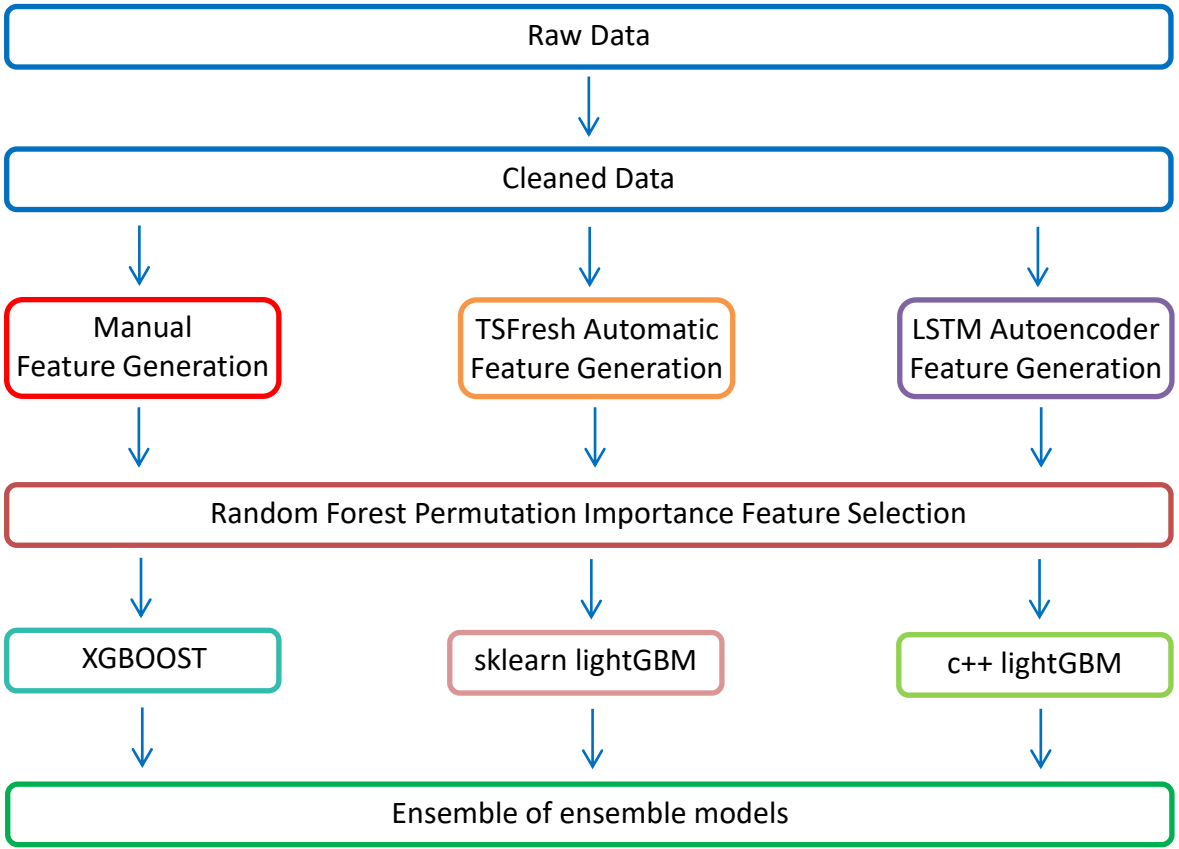
Lewis Lu Zhiping

17 June 2019

Introduction to Lewis

- Degree in Biological Sciences
- PhD in Computational and Structural Biology
- Currently working as Computational Bioinformatics Fellow (Data Analyst like role) researching on fatty liver disease in Singapore patients
- Started learning RL 3 years ago, then DL, then ML, feature engineering and then feature selection. Had the entire learning work flow backwards unfortunately.
- Interested in applying all faucets of ML to the classification of financial time series for trading

Strategy for Safety Challenge Prediction



Data Cleaning

- Data with second entry longer than 7 hours looked artificial in exploratory analysis and are removed

- Data with negative speed are imputed with value of either before or after that timestep depending on circumstances

- I am not sure if accuracy is an important factor so I am leaving it in the data set. If accuracy is not important for prediction of dangerous driving then it will be automatically filtered out in the feature selection step later. If it is predictive for some latent reason then it will help out with the final AUC score.

Feature Engineering 1 – Manual Feature Generation

- Manual feature engineering done on cleaned data

- Absolute, absolute square, log absolute and log absolute square were applied to all features

- Metric change between each time step were generated (acc_x(t) – acc_x(t-1)
- Metric change per second were calculated by taking metric change / second change
- Metric change per second is generated to detect sudden large changes in metric value (dangerous driving – sudden acceleration or sudden car turning)

- Metrics and metric changes were aggregated by taking sums, minimum, maximum, median, mean for each bookingID

- Addition, product and ratio feature interactions within and between acceleration and gyroscopic features were generated

- 1780 features were generated from 9 initial features using manual feature engineering

Feature Engineering 2 – TSFresh Automatic Feature Generation

- TSFresh - "Time Series Feature extraction based on scalable hypothesis tests“
- https://tsfresh.readthedocs.io/en/latest/index.html
- Automatic feature extraction tool for time series
- Has a great mixture of mathematical functions for feature generation
- 7146 features were generated from 9 initial features using TSFresh

Feature Engineering 3 – LSTM Autoencoder Feature Extraction

- LSTM autoencoder architecture for time series prediction is inspired from Uber's publication [*Nikolay Laptev et al, Time-series Extreme Event Forecasting with Neural Networks at Uber, ICML 2017 Time Series Workshop, Sydney, Australia*]
- 768 cell **1 layer LSTM autoencoder** is trained
- An array of 768 dimension features was obtained for each bookingID
- Max, mean, median and min for the array is extracted and concatenated to get encoded features
- 3072 encoded features were generated for each bookingID

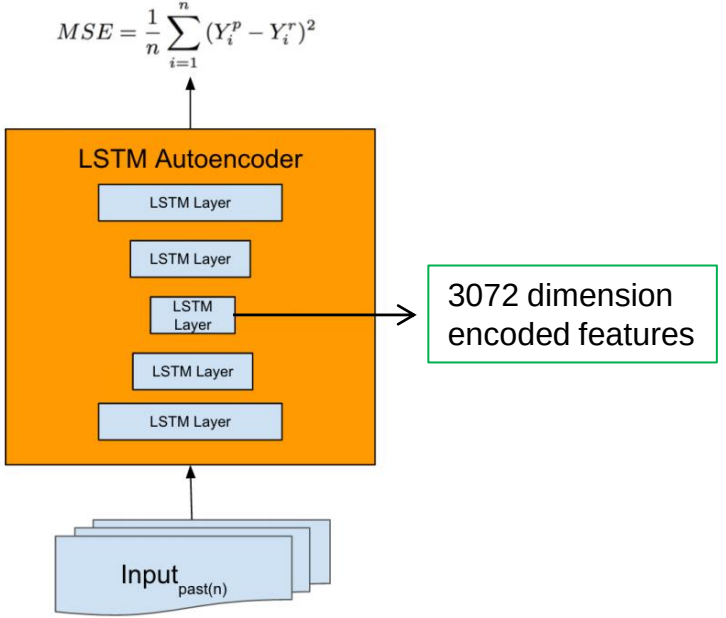


Figure modified from publication

Generated features are concatenated

- A total of 11998 features were generated
- Feature selection by random forest permutation importance is explained on the next slide

Random Forest Permutation Importance Feature Selection

- Feature importance deemed by mean decrease in impurity implemented in sklearn random forest are a quick way to determine feature importance most of the time but can give incorrect answers

- Terence Parr *et. al.* demonstrated that sklearn random forest gave incorrect feature importance in a small real world dataset involving New York housing price (https://explained.ai/rf-importance/index.html)

- The solution is to use permutation importance in determining feature importance
- Permutation importance shuffles a column, passes the input features through a model and then measure the change in prediction accuracy from that shuffling
- **If a feature is useless, shuffling it wont change the results at all**

- For this dataset, the python implementation of random forest permutation importance is employed for feature selection (https://github.com/parrt/random-forest-importances)

- 2307 features were selected from 11998 generated features

Training Ensemble of Ensemble Models

- 2307 selected features split into train and test set
- Many XGBOOST, lightGBM sklearn api and light GBM c++ api models were trained with shuffled train-validation split from train set
- Selection of ensemble models were done using maximum AUC on train data set
- **Only lightGBM models were selected in the end**
- **18** sklearn lightGBM api models and **41** c++ light GBM api models were chosen
- Incorporation of XGBOOST models decreased AUC (tuning can be done better)
- Arrays of predictions from ensemble of lightGBM models were averaged to get prediction probability and Test AUC value (0.7697)
- ROC AUC plot exported as jpeg image (not shown)

Improvements that could have been made

- AUC for LSTM encoded features is ~0.70. Deeper LSTM Autoencoder structures or longer latent encoding might create better features with higher AUC.

- A secondary LSTM classifier can be trained with the encoded latent feature representation to produce classification predictions just like in Uber's model. This increases model diversity which is good for stacking.

- Extracting out independent useful functions out of tsfresh can speed up computations and make it pandas independent

- A meta classifier (stacking) can be trained on top of those base classifiers instead of just merely averaging prediction probabilities

- **More time should be focused on understanding the data.** I had missed out on cleaning fake speed data (data with more than 400 km/h speed) because I do not understand the data well enough (have license, don't drive). I had learnt of this from another person github. Did not copy and incorporate into my workflow though.

Summary and Gratitude

- I want to thank Grab for the opportunity to challenge myself and take part in this very meaningful competition.

- I had learnt tremendously in the past 3 weeks

- My scripts wont be wasted. It will be put to good use in financial time series classification.

Testing instructions / Warnings

- LSTM Autoencoder encoding and tsFresh automatic feature extraction might take hours to run depending on the size of the dataset

- tsFresh required pandas of older version
- pip install tsfresh to downgrade the pandas version

Thank you