Winning Model Documentation

Name: Xiaozhou WANG, Qian QIAN, Kele XU
Location: Canada, China, China
Email: xiaozhou@ualberta.ca, qqgeogor@gmail.com, kelele.xu@gmail.com
Competition: Porto Seguro's Safe Driver Prediction

Background

1. *What is your academic/professional background?*
    a. Kele Xu is currently an assistant professor. He finished his Ph.D. in University of Pierre and Marie Curie.
    b. Xiaozhou Wang currently works as Chief Data Scientist at Quartic.ai.
    c. Qian Qian works as Data Scientist at Moseeker inc.

2. *Did you have any prior experience that helped you succeed in this competition?*
    a. We have participated in various Kaggle competitions before this one so we are all fairly experienced Kagglers.

Model Summary

Two most important models we built are LightGBM and Neural Networks. We did feature engineering for each model separately and in the end, just averaging these two models can give us 0.29387 on Private LB (2nd place). We were able to push the performance a bit further (0.29413) but that requires substantially more models, which we think is not worth it for this competition. We mostly trusted our Cross Validation and didn't fully trust public LB for our averaged models so we successfully avoided overfitting in the end.

Feature Engineering

- Calculated features were all removed because we didn't find them useful for models
- Categorical features were one hot encoded for LightGBM, and were fed into embedding layers for Neural Nets
- Count of categorical features were also included in models
- Engineered features:
    - Total number of missing values per row
    - All individual features were combined into a new feature and its count was included as a feature
    - Additional features engineered for NN:
        - Feature multiplication and division of important features (e.g. ps_car_13, ps_ind_03, ps_reg_03, ...)

    - Xgboost predictions: divide feature sets into three groups (car, ind, reg) and then use two group as features and the other group as target, train a xgboost model on it, and use prediction as features
    - feature aggregation: pick two features (e.g. ps_car_13, ps_ind_03), and then use one as group variable, the other as value variable, do mean, std, max, min, median. Only important features are picked.

Model Training

- Both LightGBM and NN were trained and averaged across different random seeds because we found that stabilizes model performance quite a bit.
- Nonlinear ensemble methods were tried without success so we only used weighted average.

Simple Features and Methods

Both LightGBM and NN are fairly easy models and can be trained quickly. We would recommend LightGBM as the better/simplified model for production, and it alone can score in top 50 in the competition.

Appendix

A1. Model Execution Time

- What software did you use for training and prediction?
    - LightGBM, Keras with Python wrapper
- What hardware (CPUS spec, number of CPU cores, memory)?
    - We used 8 core CPU with 64GB memory and one GTX 1080 GPU. However, we think that the minimum configs needed is probably 2 core cpu with 16GB memory and a GTX 1050 GPU.
- How long does it take to train your model?
    - For bagged LightGBM it can only take about half an hour and for bagged NN it should take about 2 - 3 hours.
- How long does it take to generate predictions using your model?
    - The prediction file can be generated in a few seconds.

A2. Dependencies

older or newer version of below packages should theoretically work fine

Python 2.7
numpy 1.13.3
pandas 0.20.3
sklearn 0.19.1
keras 2.1.1
tensorflow 1.4.0

lightgbm 2.0.10
xgboost 0.6

A3. How To Generate the Solution (aka README file)

Please refer to README.md for better formatted steps.

- Put unzipped data in `input`
- `cd code`
- `python fea_eng0.py`
- `python nn_model290.py` to get a nn model that scores 0.290X
- `python gbm_model291.py` to get a gbm model that scores 0.291X
- `python simple_average.py` and then you can find the submission file at `../model/simple_average.csv`