Data Scientist Report

Dan Golden

Assumptions

- Model build is focused on predictive accuracy rather than interpretability
- Missing data should be imputed
- All fields are numeric
- Target must be scaled between [300, 839]

Methodology and solution path

- Inspect data in RStudio
  - Check data type, range, and format of target
  - Confirm target is complete (no missing values)
  - Check percent missing among predictors
  - Check for perfectly correlated predictors
- Build baseline models in Python using PyCharm
  - Create XGB and LGB baseline models with no ETL
  - Optimize baseline models using GridSearch (objective = MSE)
- Build target rescaler using empirical CDF (ECDF)
  - Create ECDF and inverse ECD from training target
  - Create ECDF from training predictions
  - Rescale test predictions
    - Test prediction -> Training Predictions ECDF -> Training Target inverse ECDF
- Evaluate baseline models
  - Train / Test split using 80% / 20%
  - Calculate two required metrics using test data
- Iterate for model optimization
  - Tune LightGBM hyperparameters with GridSearch
    - Compare the same hyperparameters with LightGBM and XGB
  - Evaluate various ETL methods
    - Test feature reduction with PCA
    - Test feature scaling with standardization
    - Test imputation with mean / median / large value (999999)
  - Retrain on all data for final model
- Create prediction script
  - Saved model + ECDF + Inverse ECDF via joblib
  - Load saved model in prediction script
  - Generate output
  - Compare prediction script output to training script output

Algorithms and techniques used

- XGBoost
- LightGBM
- GridSearch
- Sklearn Pipelines
- Statsmodels ECDF / Inverse ECDF
- Model persistence

Tools and frameworks used

- RStudio
- Pycharm
- Conda
- Sklearn
- Statsmodels

Results and evaluation of models

Algorithm	GridSearch	RMSE (Train / Test)	Abs (pred - act) < 3
XGBoost	No	22.69 / 27.65	0.159 / 0.142
XGBoost	Yes	6.39 / 25.82	0.466 / 0.168
LightGBM	No	26.33 / 27.43	0.139 / 0.141
LightGBM	Yes	18.19 / 25.71	0.196 / 0.158

LightGBM trains significantly faster than XGBoost which allows for more expansive hyperparameter tuning. After hyperparameter tuning is complete with LightGBM the same hyperparameters can be tested with XGBoost. XGBoost provided better absolute accuracy and was saved for the final model.