CS5691: Data Contest Report

JOSE MOTI CE17B118
RAMPRASAD RAKHONDE CE17B126

Team name: Something
Public Leaderboard → Rank 1
Private Leaderboard → Rank 3

PREPROCESSING

All the duplicate rows in the train and test dataset were dropped. This actually increased the validation score and leaderboard score by a small quantity. Some basic methods of filling nans were used and all the given datasets train.csv, bikers.csv, bikers_network.csv, tours.csv, tour_convoy.csv were merged together. All the features used are given below.

FEATURES USED

All the features that were used are given below. A total of 160 features were used.

From train.csv (8 features)

- timestamp - converted to pandas DateTime
- **month_query, day_query, hour_query, minute_query** - splitting timestamp
- **invited**
- **num_queries_count** - number of questions asked to the given biker. This is actually the number of rows having the same biker_id
- **num_bikers_asked_train, num_bikers_asked_total** - number of bikers in the train dataset/ train+test dataset who were asked about the given tour.

From bikers.csv (12 features)

- member_since - Nans filled with 31-12-2012, converted to pandas DateTime.
- **age** - nan filled with a mean age.
- **gender** - Nans filled with male gender as it is most common.
- **time_zone** - Nans filled with most occurring value (420)
- **join_query_difference** - the difference between timestamp and member_since
- **month_joined, day_joined** - splitting member_since
- **location_id, language_id**
- **biker_latitude** and **biker_longitude** from location_id mapping.
- **biker_timestamp_index** and **biker_member_since_index** - the index of tours based on timestamp and member_since for a given tour_id.

From tours.csv (101+10 features)

- **latitude** and **longitude**
- tour_date - converted to pandas dataframe
- **join_tour_difference** - the difference between member_since and tour_date in days
- **tour_query_difference** - the difference between timestamp and tour_date in days
- **year_tour, month_tour, day_tour** - splitting tour_date
- **timestamp_index** and **tour_date_index** - the index of tours based on timestamp and tour_date for a given biker.
- **geodesic_distance** - the distance between latitude, the longitude of the biker, and the tour in miles. Found with geopy.distance.geodesic.
- **w1, w2…, w99, w100, w_other.**

From bikers_network.csv and tour_convoy.csv (29 features)

- **num_friends** for each biker_id and **num_going, num_maybe, num_invited, num_not_going** for each tour_id
- **friends_going, friends_maybe, friends_invited, friends_not_going** for each biker_id
- **friend_organizer** (is the organizer of the tour a friend), **is_organizer** (is the given biker organizer of the tour).
- **percentage_going, percent_maybe, percent_invited, percent_not_going** - percentages with respect to num_friends
- **percentage_going_2, percent_maybe_2, percent_invited_2, percent_not_going_2** - percentages with respect to corresponding num_going, num_maybe, num_invited, num_not_going.
- **friends_loc, going_loc, maybe_loc, invited_loc, not_going_loc** - num bikers location same as that of the biker.
- **percent_friends_3, percent_going_3, percent_maybe_3, percent_invited_3, percent_not_going_3** - percentage of friends_loc, going_loc, maybe_loc, invited_loc, not_going_loc with respect to num_friends, num_going, num_maybe, num_invited, num_not_going.

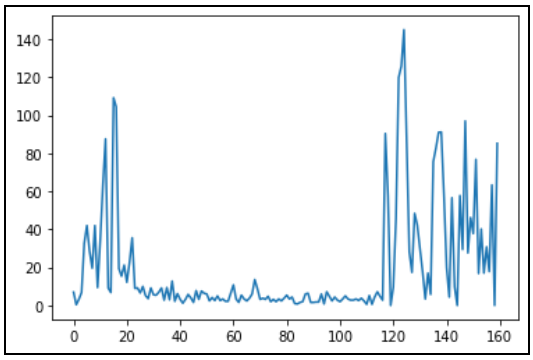
The top features used by the model as given by clf.feature_importance() for the LGB model are given below in that order. The corresponding score for the importance of the feature is also given

- num_queries_count, 145.0
- tour_query_difference, 125.8
- day_tour, 119.8
- biker_longitude, 109.2
- latitude, 104.4
- percent_maybe, 97.0
- num_maybe, 91.2
- num_going, 91.0
- w100, 90.4
- hour_query, 87.6
- timestamp_index, 87.0

The worst features and their corresponding scores are:

- w77, 0.8
- w95, 0.6
- w93, 0.6
- invited, 0.4
- join_tour_difference, 0.0
- friends_not_going, 0.0
- percent_friends_3, 0.0

Also below is the plot of feature importances of all the 160 features. The features w1, w2, ...etc has much lower importance compared to that of some meaningful features we created.



MODEL SETUP

- **5 fold cross-validation** - Cross-validation was done on the basis of biker id. All bikers in train data were split into 5 and all the rows corresponding to the given bikers form the group. This was to prevent leakage of data while using training and a validation dataset. The bikers in the training dataset are never repeated in the test dataset. So the bikers in the train-split dataset should never be repeated in the validation-split dataset. This is to make sure the validation score gives a rough average of the test score. The training was done in 5 rounds. In each round, a group is taken as a validation dataset, and the remaining 4 taken as a training dataset. This gave a total of 5 models and the average of the predictions of these models are taken in the end for predicting the test dataset.
- **Competition metric implemented** - It involves taking average precision for each biker_id set and averaging the result. This metric was used for selecting the best models by finding the scores on the validation dataset.
- **Trying different models** - The decision trees were found to work better than SVM, logistic regression. In decision trees, gradient boosting models like Light Gradient Boosting (LGB) and Catboost were giving the best validation scores. Num boosting rounds were determined by the AUC score of the validation dataset. Early stopping was used. This means when the AUC score of the validation dataset was not found to increase for 250 iterations the boosting stops and the model till the iteration where it gives the best validation score is taken. LGB gave the best cross-validation score of 0.735 and the leaderboard score of 0.798.

- **Multiple seeds for cv split** - So the dataset was split into 5 folds so that now there are 5 models when the training is completed. This process was repeated several times but this time the different seed was used for the K-Fold cross-validation split so that new 5 sets of training data were created and thus new 5 models. This was repeated multiple times. So using a total of 25 different seeds means a total of 25*5=125 different models. This is similar to the bagging technique. Different combinations were used as training data for each of the models and adding these helps to decrease the variance of a single model and make the cross-validation scores more trustworthy. The cross-validation score increased from 0.735-0.745 after moving from 5 to 125 models. Cross-validation score was taken as a more trustworthy score compared to that of leaderboard score on 30% of test data.

Extra parameters (may or may not be used for different experiments - least importance)

- **Ordinal/ One-Hot Encoding** - The location_id and language_id features are encoded using either Ordinal Encoder or OneHot Encoder. Ordinal encoding was found to give much better validation scores.
- **Standard Scaler** - It is used to scale each feature to a distribution with mean 0 and variance 1. This is not very much important since we are using decision trees and decision trees do not need scaling to take place.
- **Training label** - Regarding the label for training, it can be either binary or multiclass. If binary, only the like column is passed as the label. The prediction probability of label 1 is taken as the score of the tour for that given biker. If multiclass like is taken as 2, dislike taken as 0 and neutral (both like and dislike column as 0) is taken as 1. The prediction_proba will now be a 3-dimensional vector say A for each row and the score of the biker for that given tour is given as A[2]-A[0]. The multiclass training actually does not improve the score. This might be because the number of disliked rows are very low compared to liked rows or neutral rows.
- **Balancing the dataset** - The given dataset is actually an imbalanced dataset. Only 3370 out of 13866 has a like label as 1. So in order to make it balanced the rows with like the label as 1 were duplicated 2 more times in order to have the almost same number of 1s and 0s.
- **Postprocessing1** - Here for each row we find the most similar row from the training dataset using cosine distances. Then some of the features from both these rows were used to predict the label. This included the label of the similar row and prediction_scores of the current biker_id. This increased the cross-validation score as well as the leaderboard score by a small quantity.
- **Postprocessing2** - Some of the tour_id in the training dataset also comes under the test dataset. So the tours that are repeated 4 or more times are taken and some features like count of the given tour, an average of the like column for the given tour were calculated. These then served as features to another model that predicts the score for biker_ids having these given tours. This does not increase the validation score. So this was not used in the final submission models.
- **Bayesian optimization** - Bayesian optimization was tried out for finding the right hyperparameters for models. For example, if we are using LGB we have a lot of

hyperparameters to tune like num_leaves, max_depth, feature_fraction, bagging_fraction, etc. Instead of doing the manual tuning, we could give these hyperparameters and the range to tune on as input and it would try to increase the validation scores by predicting the correct set of hyperparameters based on previous scores and probability. This helps in much faster hyperparameter tuning. But on our dataset, the difference in scores after hyperparameter tuning was not that high. So we went with default parameters for LGB and CATBOOST models.

Other things tried

- Tried to reduce features using PCA - decreased the score.
- Tried using tour_id as a categorical feature. It actually decreased the score so not used in the final models.
- Tried to train on the full dataset without having a validation dataset. This was a bad idea since there was no check on overfitting on the training dataset.
- Regression was used instead of binary classification for LGB. Actually gave similar scores.

PREDICTION

- Preprocessing done in the same way as the training dataset.
- We passed all the trained models to a list. So we iterate the list, get predictions of each row, add it for each model and get the average in the end.
- Postprocessing was done if required.
- The tours of the biker arranged in descending order of the score and made to the required format for submission.

OUR SUBMITTED MODELS

Since we were allowed two 2 submissions. 1 - we submitted the highest cross-validation score we got. 2- we submitted the highest public leaderboard score we got. Even though they were not the best in terms of private scores, the scores were not bad.

- Submission 1 - LGB with 5 fold CV using 25 different seeds (seed number from 101 to 125, 125 models in total), using ordinal encoder and no standard scalar, default hyperparameters for LGB, binary classification problem, no postprocessing was done.
    - Cross-validation score → **0.7439**
    - Public score → 0.7909
    - Private score → **0.7381**
- Submission 2 - LGB with 5 fold CV using seed 101 (5 models in total), using ordinal encoder and standard scalar, default hyperparameters for LGB, balancing dataset, binary classification problem, postprocessing1 was done.
    - Cross-validation score → **0.7351**
    - Public score → 0.7993
    - Private score → **0.7362**