Instacart Data science challenge

Sai Harshith Reddy Gaddam

Objective: Build a predictive model to predict the shoppers time in fulfilling a customer's requirement

- The code is written in the form of a class and a instance of the class has been created to run the model and score the model.

Brief outline of the steps taken to arrive at the predictions

Preprocessing Steps:

- Merged train_trips and items_trip csv files to create a data at a trip id level
- Checked for missing values and outliers

Feature engineering Steps done:

I created the following features from the data fields described above:

- Sum of products bought in a trip
- Number of unique depts visted in a trip
- Item counts by department
- Average number of items purchased per dept in a trip
- Sum of quantity by department and total quantity purchased
- Extracted "pickup_wkday", 'pickup_weekofyear', 'pickup_hr' to capture time trends

To do feature engineering

1. See if there is any information obtained from shopper ids by clustering . They may be proxy to the city they live in
2. Use t-SNE to cluster trips and extract more features based on the clustering behavior

Model Building:

- Considered 2 Highly Non-linear tree based Ensemble models

a) Random Forest from Scikit-learn package

b) Xtreme Gardient Boosting machines from xgboost package

- Split the Data into Train and Validation: 70-30 split
- Defined the Parameter search space for each algorithm
- Leveraged Hyperopt package to do Hyper-parameter tuning on the search space defined
- The Loss function to be minimized to choose the best set of parameters for each model is defined as follows
- Loss Function=|Train_RMSE-Validation_RMSE| * Validation_RMSE
- This ensures that the best model is the one, which is not completely tuned to the Validation data but is a generalizable model, which can extrapolate well on the unseen data

Models and the search space for the hyper parameters using Hyperopt

Parameter Space for Random Forest

- **n_estimators:** # Trees to be built [100-500] with step 1
- **Criterion:** The evaluation metric on the Out of Bag samples ('MSE') -Only Mean Squared Error is available as part of Regression

- **Max_depth:** The maximum depth to grow each tree [5-20]
- **Max_features:** Maximum features to look at when splitting a node [50%-100%, step: 5%]
- **n_jobs:** Number of cores to use for Parallelization (-1: indicates, use the available number of cores)
- **random_state:** seed for reproducibility of a model (123)

Parameter Space for xgboost

n_estimators: # Trees to be built or # of rounds of Boosting [100-500, step:1]

eta: learning rate for the gradient descent [0.025-0.5, step:0.025]

max_depth: The maximum depth to grow each tree [5-20]

‘min_child_weight’: Minimum children at each leaf node [5-100, step:5]

Subsample: % of records to consider to build each tree[50%-100%, step: 5%]

Colsample_bytree: % of attributes to consider to build each tree [50%-100%, step:5%]

gamma: Minimum loss reduction to make a split [0.5-1,step:0.05]

Objective: Loss function to be minimized -‘reg:linear’

eval_metric: Evaluation metric to be used ‘rmse’

nthread: # cores for parallelization-4

random_state: seed for reproducibility of a model (123)

Xgboost best result

Parameters

```
Training with params :
{'colsample_bytree': 0.6000000000000001, 'silent': 1, 'eval_metric': 'rmse', 'eta': 0.275, 'nthread':
4, 'min_child_weight': 20.0, 'n_estimators': 144.0, 'subsample': 0.9500000000000001,
'random_state': 123, 'objective': 'reg:linear', 'max_depth': 17.0, 'gamma': 0.6000000000000001}
loss:644435.377677
Train Error:649.044
Validation Error:1190.403
```

Random forest best result

Parameters

```
Training with params :
{'n_estimators': 179.0, 'n_jobs': -1, 'criterion': 'mse', 'verbose': 0, 'max_features': 0.7000000000000001,
'random_state': 123, 'max_depth': 18.0}
loss:436137.85557
Train Error:763.899
Validation Error:1144.854
```

Ensembling

After getting both the predictions, I have given 0.6 and 0.4 weightage to xgboost and random forest model

Xgboost-Pros:

- It is a very fast Ensemble algorithm
- It is a parallel implementation of Gradient Boosting machines with Regularization
- Efficient pruning, as it builds a tree to max depth and start removing splits beyond which there is no positive gain

- It has built in cross-validation

Xgboost-cons:

- Many parameters to tune
- Care should be taken to avoid overfitting

Random Forest-pros:

- It is a fast Ensemble algorithm
- It is parallelizable
- It is good at dealing with the variance
- It reports OOB rate (Out-of-bag error), which is a good estimate of the error on unseen data

Random Forest-cons:

- Prone to over fit when data is noisy

Future work given more time and computing resources

- Run Neural networks which will automatically create features and use them in the xgboost model
- Feature scaling which prevents the model from giving greater weightage to certain attributes as compared to others and feature selection to pruning them to reduce noise/redundancy
- Cross validation: Implement k-fold validation with 5 folds
- Ensemble of various models with different random seeds and use stacking to ensemble the models
- Getting more data regarding traffic, weather and customer information, demographics will help us in having a better understanding of the prediction