ISE 529_Final Project - Grupo Bimbo Inventory Demand - Wednesday

Roger Wang (5184990581), Danlei Zhang (9993088581)
Yunqing Ma (4552265616), Vic Wang (9785480181)

Overview

This project focused on the inventory demand of a Mexico company called Grupo Bimbo. Grupo Bimbo provided bakery products to its customers, and our objective was to predict the inventory demand based on the given historical data. In the given dataset, there were sales transactions data in nine weeks. We did a little data exploration first and followed the below steps to achieve our goal: 1) pre-processed data, 2) randomly selected a small train and test datasets from our big dataset, 3) applied grid-search with cross validation to find the best parameters for the XGBoost model, 4) fitted and trained our XGBoost model with the best parameters found, 5) ran feature selection to find the best ten features, and 6) fitted a new XGBoost model using ten features and made predictions on our test dataset. At last, we submitted our results to Kaggle.

Pre-processed Data

For feature engineering, we first removed redundant columns: ‘Venta_uni_hoy,’ ‘Venta_hoy,’ ‘Dev_uni_proxima,’ and ‘Dev_proxima’ which were highly correlated with Demanda_uni_equil (Adjusted Demand). We derived features: ‘Producto_ID,’ ‘Producto_name,’ ‘brand,’ ‘weight,’ ‘pieces,’ and ‘weight_per_piece’ from the original feature called ‘NombreProducto.’ We also obtained feature ‘Client_Type’ by clustering the original feature called ‘NombreCliente.’ We calculated the mean of weekly frequencies for the categorical features and saved as new features. Obviously, there would be a time series component to Adjusted Demand. We added lagged adjusted demand variables with different data periods. Thus, so far, we had 22 features. For numerical features, we used MinMaxScaler to normalize them. For categorical features, we converted them into integers. There existed NAs in ‘weight,’ ‘pieces,’ and ‘weight per pieces.’ We chose to replace them with the mean values of the corresponding features.

Randomly Selected a Small-size Train and Test Datasets

The size of raw data is greater than 40 million, and it is difficult to get results in a short time. In order to ensure the feasibility of the model, we obtained a subset of the data by random sampling. Simply used *random.sample()* function to generate n - 10,000 random numbers and used *skiprows* parameter in *pd.read_csv()* function to skip those n - 10,000 rows in our big dataset. Thus, what we had was a dataset with 10,000 rows. Finally, we used *train_test_split()* to obtain our small train(75%) and small test(25%) datasets.

Grid-search and Stratified K-fold Cross-Validation

We used grid-search with stratified K-fold cross-validation to find the best parameters for our XGBRegressor model. The objective used was 'reg:squaredlogerror' which happened to be our evaluation metric:

$$RMSLE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(ln(p_i + 1) - (ln(a_i + 1))^2}$$

where p_i and a_i stand for the prediction of demand and the actual demand. The parameters we tried were: n_estimators = [30, 40, 50], depth_value = [5, 15, 20], and learning_rate = [0.1, 0.3, 0.5]. It turned out that n_estimators = 30, depth_value = 15, and learning_rate = 0.1 were the best.

Fitted and Trained XGBoost Model

Using our best parameters obtained from the previous step, we fitted and trained our XGBRegressor model. We used this model to predict our small size test dataset. For any negative values in our prediction, we replaced it with zero since a negative demand means no demand for that product.

Feature Selection

We plotted the feature importance in *Fig 1*

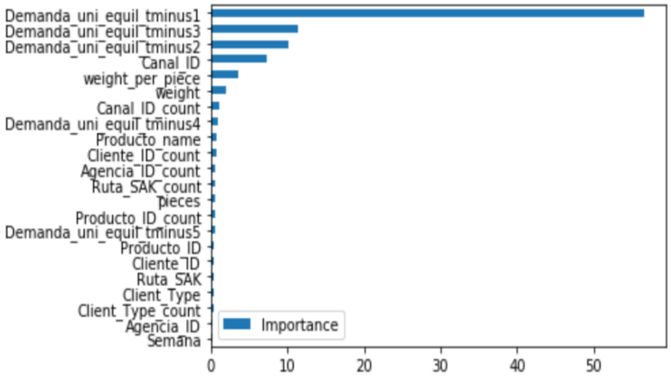


Fig 1. Feature Importance for Each Feature

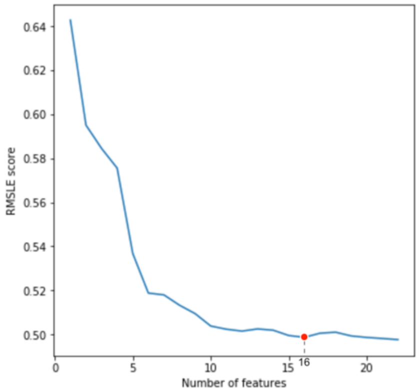


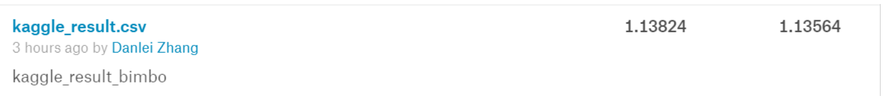
Fig 2. RMSLE score vs Number of Features Used

As the figure demonstrated, there were four features playing a significantly important role in the model. In order to figure out how many features used in the model returns the best prediction, we ran a for-loop and calculated the RMSLE score with the number of used features from one to twenty-two. We plotted our results in *Fig 2*.

Fitted a New XGBoost Model and Make Predictions

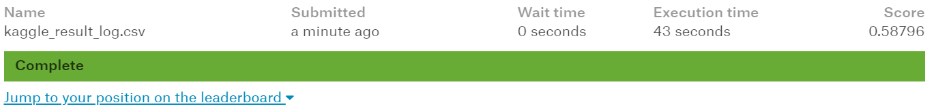
As shown in *Fig 2*, the best number of features that should be used in the model was sixteen. We then fitted a new model using the sixteen selected features and used that new model to predict our Kaggle test dataset. Lastly, we saved our results into a CSV file.

Kaggle Result



kaggle_result.csv 1.13824 1.13564
3 hours ago by Danlei Zhang
kaggle_result_bimbo

We submitted our results to the Kaggle, and the private score is 1.13564. We believe the major deviation was caused by the small size of our training set. Since our model well predicted our small-size test dataset. We believe if we have more time and more computing resources, it is possible to decrease our score by taking a bigger training set. Using Google Colab, we used the full train dataset to fit our model, and the Kaggle score decreased to 0.58796 as we expected.



Name	Submitted	Wait time	Execution time	Score
kaggle_result_log.csv	a minute ago	0 seconds	43 seconds	0.58796

Complete

Jump to your position on the leaderboard ▾