1. Abstract

2. Introduction

3. Related Works

4. Data Exploration

5. Methods

5.1. ARIMA and SARIMAX

5.2. Prophet

5.3. LightGBM

LightGBM (light gradient boosting machine) is gradient boosting based ensemble decision tree, where the term “light” indicates its faster speed and memory efficient operation compared to conventional gradient boosting techniques such as XGBoost [ref2] and pGBRT [ref3]. Hence, lightGBM is very effective in handling large datasets, which we are dealing here. In particular, lightGBM employs two techniques to significantly training time while handling large datasets with negligible compromise on accuracy. Namely, the two techniques are: (1) exclusive feature building (EFB) and (2) gradient-based one side sampling (GOSS). The EFB merges sparse features into one feature using a greedy algorithm to reduce feature space for a tree. On the other hand, the GOSS enables reducing the sample size by using large gradient samples and a fraction of lowest gradient samples which further multiplied by a constant to put more weight on the under-trained dataset. In addition, lightGBM also grows tree leaf-wise (grow leaf with maximum error) [ref4] rather than level-wise to achieve lower loss, which, in turn, helps to achieve higher accuracy with large datasets when handling the overfitting carefully using the depth parameter of the tree. Another notable feature of the lightGBM is that it can handle categorical features without doing one-hot encoding [ref5]. With all these features lightGBM tends to provide up to 20 times faster training time while achieving similar accuracy compared to conventional gradient boosting decision trees.

In this paper, we employ lightGBM predict the Walmart sales time series for the following 28 days (date range.) [ref6: Kaggle]. Here the use of lightGBM noteworthy due to the large size of the Walmart datasets, which can be easily trained by the lightGBM with a reasonable amount of time using a simple computer. Specifically, an implementation of the lightGBM regression tree with a 4.0 GHz Intel Corei7 CPU (no parallel processing) and system memory 16 GB took mention time in minutes to train and predict the Walmart sales of 28 days for 30,490 time series.

**Feature engineering for the lightGBM:**

Basic features will be described in the data exploration phase (or I will describe my sample feature engineering (e.g., merging and converting categorical features to numeric values) here)?

Show a figure after the basic feature creation.

We have created featues to capture the trend of the times series well by the lightGBM alogorithm. Fig. 1 provides a snapshot of the feature-engineered dataset that has been used to train the lightGBM algorithm. At first, we create two lags of the of the time series for capturing the weekly and approximately monthly trends, which are denoted by "lag\_7” and “lag\_28” in Fig. 1., respectively. We showed that by adding these two features with basic features we can improve the accuracy of the lightGBM, e.g., mention the improvement with the exact number. The above created daily lags are useful to learn history of the of sales on weekly and monthly basis. However, since these are daily sample values, they often fluctuate abruptly due to outliers and noise and unable to capture weekly and monthly statistical attributes of the sales. Hence, to capture the average weekly and monthly statistical patterns, we have created rolling means of “lag\_7” and “lag\_28” features to include four additional features in the dataset. Namely, “rmean\_7\_7” and “rmean\_28\_7” respectively denote rolling means of 7-days and 28-days of the “lag\_7” feature. Similarly, “rmean\_7\_28” and “rmean\_28\_28” denote rolling means of 7-days and 28-days of the “lag\_28” feature. In addition, since this is a time series some other time information is created to add some additional details about sale information. Particularly, week of the year, quarter of year, day of the month is created to incorporate time-based feature in the sales dataset.

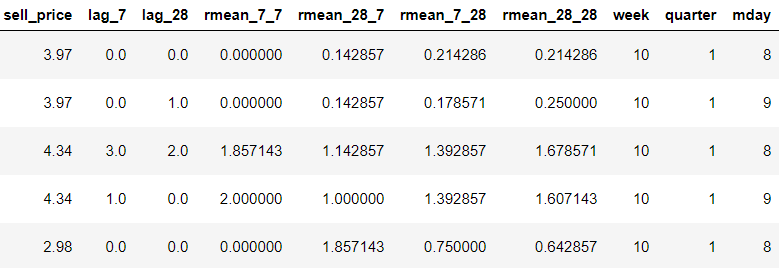


Fig. 1: A snapshot of the feature engineered dataset that has been used to train the lightGBM algorithm.

**Parameter optimization of the lightGBM:**

Table I shows the optimized (?) parameter settings for the lightGBM that has been used to predict to Walmart sales. The details about these features can be found in the lightGBM documentation website [ref7]. In short, the “learning rate” refers how fast the tree is growing, the smaller the learning rate the slower the learning but the learning accuracy is better given the sufficient number of iterations (“num\_iterations”) to learn. The “sub\_row” (describe all parameters in short?). Finally, the “min\_data\_in\_leaf" minimum data points a leaf requires to have.

In addition, the objective of the lightGBM is assumed to “Poisson”, since the Poisson distribution captures well number of counts, where our target variable is the number of sales (sales count). In addition, we use root mean squared error (RMSE) between the predicted sales and test sales as an metric to minimize the average error in sale’s prediction.

TABLE I: Parameter settings for the lightGBM regression

|  |  |
| --- | --- |
| lightGBM parameter names | Parameter values |
| “learning rate” | 0.075 |
| "sub\_row" | 0.75 |
| "bagging\_freq" | 1 |
| "lambda\_l2" | 0.1 |
| "nthread" | 4 |
| "sub\_feature" | 0.9 |
| “num\_iterations” | 1200 |
| “min\_data\_in\_leaf" | 100 |

**Feature importance:**

We have plotted feature importance of the lightGBM with respect to the number of times the feature is used as a splitting feature while growing the tree.

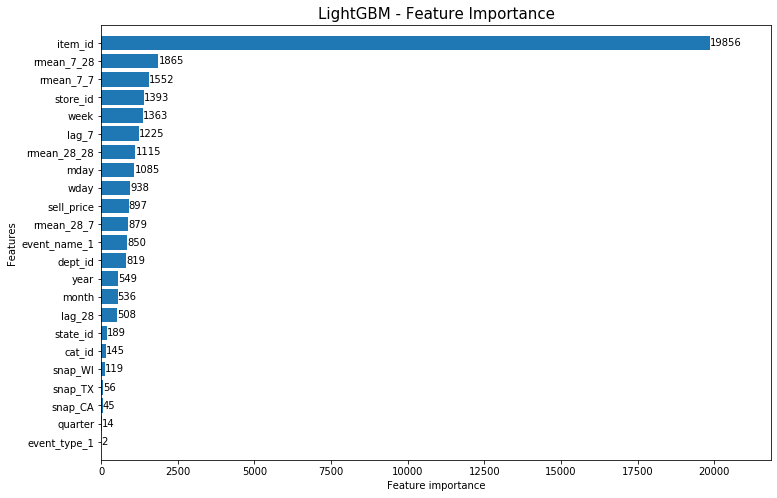


Fig. 2: Plotting feature importance of the Walmart sales dataset.

**Results:**

At first, we describe applying the lightGBM on predicting sales of a single product to compare the results with our previous methods (SARIMAX, Prophet). Fig. 3 shows sales prediction of 37 days of the HOBBIES\_1\_002\_CA\_1 product, which shows an RMSE of 0.481.

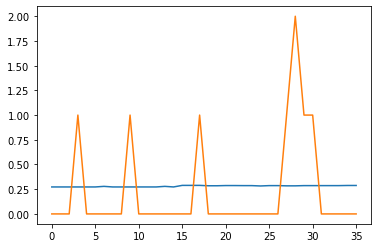


Fig. 3: Sales prediction of 37 days of the HOBBIES\_1\_002\_CA\_1 product.

Then we predict the sales of all 30,490 products for the 28 days and found an RMSE of 0.32. The reduction of the RMSE is noteworthy since the lightGBM works very well with large datasets.

6. Conclusion

7. References