Store Sales Final Project Report CSCI567 ID16

Hao Yang
University of Southern California
hyang800@usc.edu

Haoyu Guan
University of Southern California
haoyug@usc.edu

Shengyu Sun
University of Southern California
shengyus@usc.edu

Ke Wang
University of Southern California
wke45416@usc.edu

Abstract

Store Sales competition is about time series forecasting for store sales. There are totally 54 stores and 33 product families in the data. The time series starts from 2013-01-01 to 2017-08-31. The objective of this project is to predict the sales of each date, each store, and each family. This paper will mainly discuss about how we do the data exploration, feature engineering, model selection, how we approach to the final result and what techniques we use to optimize the kaggle score to the final score of 0.40042.

1 Data Exploration

This part of report will focus on the connection between data sets, visualization of original data, and what kind of observation and thoughts we have at this point.

1.1 Dataset Description

Figure 1 shows the relationship between data sets in the database

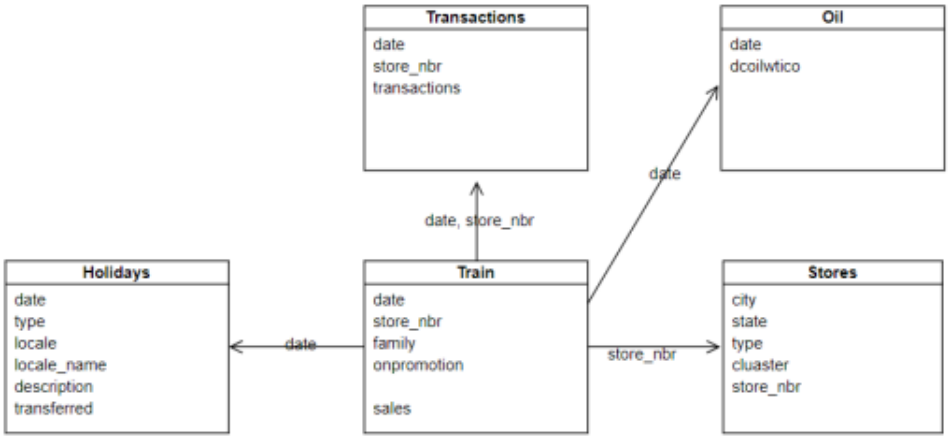

Figure 1: Data Connection

Preprint. Under review.

1.2 Daily Sales

Figure 2 shows the average sales prices of all kinds in all stores by date, we can see that the average sales increased with time. Although the trend of each year is not quite similar, but the total sales trend is going up with time. And also, the first few day of each year always has 0 sales and the sales of the last few day of each years always increase dramatically.

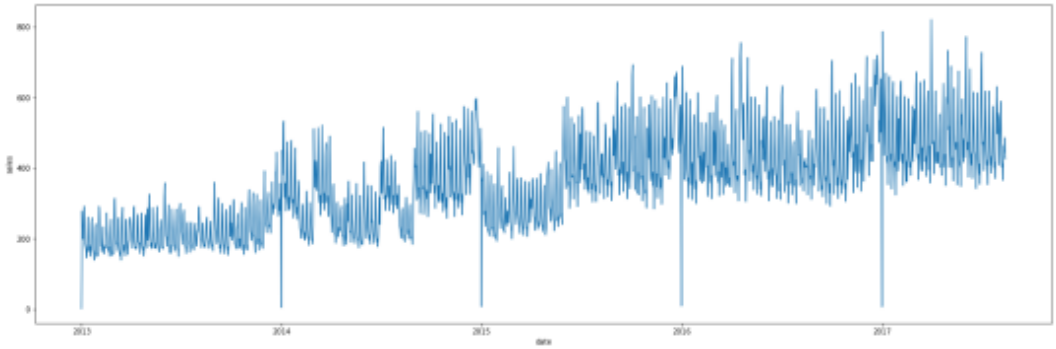

Figure 2: Daily sales plot

1.3 Daily transactions

Figure 3 shows the change of transactions by date, unlike the daily sales plot, the trend of daily transactions is more stable, it does not increase with time. And also, it has similar trend in each year. For example, the transactions of the last few day of each year will always increase dramatically. In the way of intuitive thinking, higher transactions will result in higher sales, but here, the sales keep increasing with time while transactions do not change that much. So, at this point, we can observe that there might be other factors that will affect sales like inflation or increase in the product price.

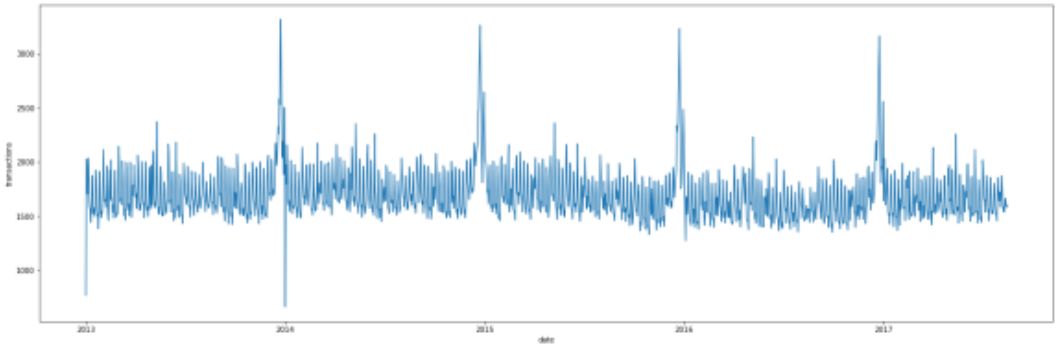

Figure 3: Daily transaction plot

2 Feature Engineering

Feature Engineering is one of the most important parts of the machine learning project. In this project, we are provided 4 extra datasets: holidays events.csv, oil.csv, stores.csv, and transactions.csv. Each of the datasets has some raw data related to the train and test dataset, and our job is to select, transform, extract, combine, and manipulate those raw data to generate the desired features to help our prediction.

2.1 Stores features

Our first approach is to do the data exploration, we plot the sales line plot for all stores and find that some stores open later than 2013-01-01 and some stores have temporarily closed during some periods.

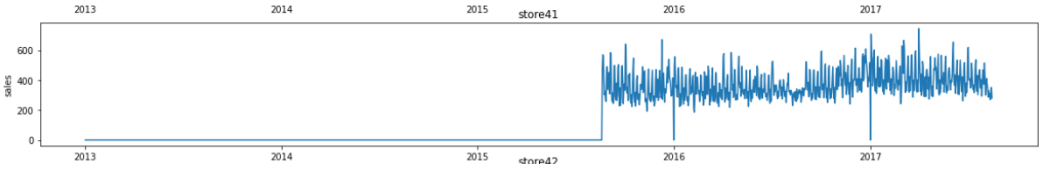

Figure 4: Stores that are opened late

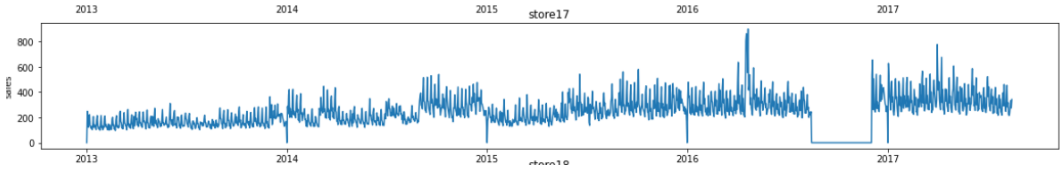

Figure 5: Stores that have temporarily closed

Also, we find that for some cities like Quito and Guayaquil, there are multiple stores in that city, and for some cities like Babahoyo and Cayambe, there is only one store per city. So, at this point, we decide to add three more features: **openlate**, **is unique**, and **is closed** to capture the underlying information of the dataset. Openlate is a boolean feature that represents whether the current store is opened later than 2013-01-01 or not. Is unique represents whether the current store is the unique store in that city or not. Is closed represents whether the current store is temporarily closed or not.

2.2 Holiday features

The objective of the holiday is to determine whether the current day will have a positive or negative impact on sales. More specifically, we can just determine whether the current day is a work day or not. We define these based on the following rule:

- Is a workday: Holiday but transferred is True, Work Day, events like World Cup, Mother's day, and Black Friday.
- Is not a work day: Holiday but transferred is False, Additional, Bridge, Transfer and events like Cyber Monday and earthquakes.

For the event type, we observe all the national events and find that there are only 5 types of events: World Cup, Mother's day, earthquakes, Black Friday, and Cyber Monday. Among these 5 types of events, we can see that World Cup, Mother's Day, and Black Friday have a positive impact on store sales, so we classify these events as a work day. On the contrary, Cyber Monday and earthquakes have a negative impact on store sales, so we classify these two events as not a work day.

Due to the limited time we have, we decide to only consider national holiday, which means we do not need to deal with those local and regional holidays. Then we create a feature called **wd** to represent whether the current day is a workday or not, if the current day is a holiday, the value of wd feature will be false, otherwise will be true. For the problem of one day may have multiple holidays, we just simply use the groupby function to group the data by date and then always take the first holiday(or event) into consideration, since if we only consider national holiday, there are only 6 days that have multiple holidays at the same day. And among these 6 days, most of them are just a mixture of Bridge holiday and Additional holiday, which means both will lead to a False wd value. So there is no difference in which one you choose for aggregation.

2.3 Oil features

We had done some feature engineering about oil in HW4, at this point, we just add some features like the moving average of the past 3 days, the moving average of the past 14 days to try to capture more underlying information. Also, we use the shift method in pandas to create some lag oil prices features to help us to observe and compare the percentage of change in oil prices.

2.4 Transactions features

This is the most tricky parts of our feature engineering. Based on the figure below, We find that transactions is highly correlated with sales, and this intuitively makes sense because more transactions will definitely result in increased sales.

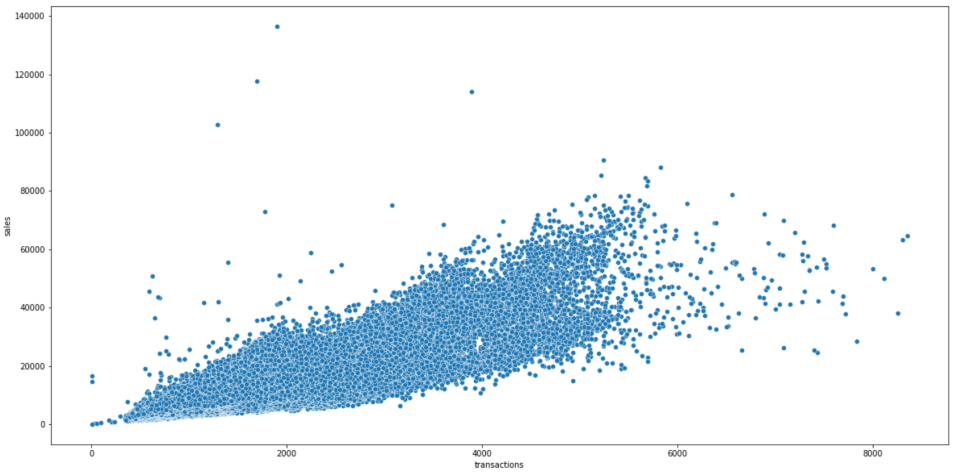

Figure 6: sales vs. transactions

But the problem is that the transactions data is missing from 2017-08-16 to 2017-08-31, which is the date range of our test dataset. Normally, we will not use such features for our prediction but due to its highly correlated relationship to sales, we do not want to waste such powerful features. Then, we try several techniques to handle the missing data: filling with the moving average, replacing it with mean, mode, and median, and using the pandas interpolate method to fill the missing value linearly. But all of the above approaches do not work really well, so finally, we decide to discard this feature.

2.5 Time Series features

Besides the trend feature we had done in HW4, we add a lot of time series features in this project like year, quarter, dayofweek, isweekend, and some fourier series deterministic terms based on the calendar time to try to capture more underlying information. Also, we create a feature called **is school** that represents whether the current month is in school season or not. Because based on the historical data, we observe that the sales of school and office supplies family dramatically increase in April, May, August, and September. And the date range of our test dataset is in August, after adding this feature, there is a huge improvement in our model performance.

3 Model Selection

3.1 Overview

Since the score of HW4 is already better than most of the public score, our first approach is just simply replace linear regression by other model with some advanced feature engineering. At this point, we tried Neural Networks like LSTM and Tree-based models like XGBoost. Then we set the size of training dataset a little bit larger to solve the problem of overfitting. And the best score at this point is **0.43802** which is obtained by XGBoost Regressor with feature engineering and hyperparameter tuning. Then, we got inspired by the notebook: Sales_ Using_LR_And_RF and we try to fit a model for each of the 54 stores. And also, instead of using one model to do the prediction, we determine to use multiple models and averages the prediction of each model to form the final prediction. With this approach, we obtain the final score of **0.40042**.

3.2 Single model selection

We used algorithms of these three type of models: Linear models, Neural Networks and Tree-based models. In this section, we will introduce the best algorithm for each of the three types of models.

3.2.1 Linear regression

Linear regression is the best algorithm of linear models. It has the advantage of high comprehensibility and easy to produce. According to HW4, we completed the construction of feature engineering and linear regression model, and finally got a score of **0.45807** on Kaggle.

3.2.2 LSTM

LSTM is the best algorithm of Neural Network. It is a variant of RNN. Its core concept is cell state and "gate" structure. The cell state is equivalent to the path of information transmission, allowing information to be transmitted in a sequence. In this case, the influence of short-term memory is overcome.

Since this prediction is in time-series type, we believe that LSTM will have a good prediction result. However, when we input the characteristics of building 2016/08/15-2016/08/31 into the model and fix the “gate” structure at 7 days (which means that the data of 7 days will affect the prediction result of the eighth day), we did not get a good prediction effect, and the score on Kaggle was only **0.82764**. Compared with the following tree based models, we believe that the reason for the poor performance of LSTM is the construction of network structure and the inapplicability of feature engineering (the main reason). Therefore, we discarded this model in the subsequent optimization process.

3.2.3 XGBoost Regressor

XGBoost Regressor is a model based on the Gradient Boosting structure, which is composed of multiple CART Regressor trees. Compared with GBDT, XGBoost optimizes the loss function, adds a regular term and adds a column sampling process in the training process.

In order to make the model get better prediction effect, we choose RandomSearchCV to get better hyperparameters. Then adjust the parameters of {n_estimators, gamma, sample, max_depth, colsample_bytree, reg_alpha, reg_lambda, min_child_weight}.These hyperparameters represent the number of XGBoost Regenerator subtrees, the minimum value of subtree node splitting loss, the proportion of subsampling, the maximum subtree depth, the proportion of feature random sampling, L1 regularization, L2 regularization, and the weight of the minimum subtree.
Finally, we get {’subsample ’: 0.8,’ reg_lambda ’: 100,’ reg_alpha ’: 1,’ n_estimators’: 100, ’min_child_weight’: 5, ’max_depth’: 5, ’learning_rate’: 0.5, ’gamma’: 1.5, ’colsample_bytree’: 0.8} with the score on Kaggle **0.43802**.

3.3 Multiple model selection(Voting Regressor)

To improve the effect of the model, we refer to the store on Kaggle: Sales_ Using_LR_And_RF, it is found that the prediction accuracy rate of two models without adjusting the hyperparameters after equal weighting will be higher than that of a single model. That's probably because averaging the prediction of multiple models will give us a less biased estimation of the output value. To make the prediction more accurate, we build 54 different models for each store as well, which means for each store, we train some unique models for that store, then we use voting regressor to average the predicted values of all the models of that store. So, finally we choose **Linear Regression + Extra Tree**.

Table 1: Multiple Model and Results

Multiple Model	Results
Linear Regression + Extra Tree	0.40042
Linear Regression + Random Forests + LightGBM + Extra Tree	0.40496
Linear Regression + Random Forests + LightGBM	0.40729
Random Forest + Extra Tree	0.40848
Linear Regression + Random Forests	0.42877

4 Other approaches that works

4.1 Training start date

We set the training start date to be 2016-06-01, there are several reasons: first, the trend of sales of 2013-2015 is different from the trend of sales of 2016-2017(we had tried to train the whole dataset but result in lower score). Second, some stores did not open during 2013-2015. Third, we want to save the data of Aug-Sep of previous year to help us to predict the SCHOOL AND OFFICE SUPPLIES family more precisely.

4.2 Log transformation

We perform log transformation to our y, this is inspired by the notebook: Store Sales: Using the average of the last 16 days. According to the notebook, RMSLE will penalize much more the underestimated predictions than the overestimated predictions. And also, log transformation can make the distribution of y look more like Gaussian and make the models better fit to the data.

4.3 Sample weighted

We add more weight to the data after 2017-07-1, that's because this is time series forecasting task, we want those data that are close to the test data to have larger weight during fitting.

References

[1] Manav, T. (2022) *Multivariate Time Series*, https://www.kaggle.com/code/manavtrivedi/multivariate-time-series.

[2] ARTEMCHISTYAKOV(2022) *Store Sales - EDA + RF*, https://www.kaggle.com/code/artemchistyakov/store-sales-eda-rf.

[3] M.R.0024 (2022) *Store Sales Using LR And RF*, https://www.kaggle.com/code/mr0024/store-sales-using-lr-and-rf.

[4] Rizky, A.A. (2022) *Hyperparamaters*, https://www.kaggle.com/code/rizkykiky/hyperparamaters.

[5] Carl, M.E. (2022) *Using the average of the last 16 days*, https://www.kaggle.com/code/carlmcbrideellis/store-sales-using-the-average-of-the-last-16-days.

[6] George. (2022) *Per Family with Store Multioutput*, https://www.kaggle.com/code/mscgeorges/per-family-with-store-multioutput.