**G6：Store Sales Analysis and Predicting using Multivariate Methods**[1]

Yijin Wang ([yijinw4@illinois.edu](mailto:yijinw4@illinois.edu)), Xin Fan ([xinfan3@illinois.edu](mailto:xinfan3@illinois.edu))

**Abstract**

Prediction of sales is an important field in the retail industry, and due to new technologies, it has recently gained broad attention on improving business operations and profitability.[2] Therefore, our project was created aiming to predict sales more accurately and reliably for the retail stores. We have constructed one statistical Time Series model and two other machine learning models - XGBoost and Long Short Term Memory (LSTM). The three methods mentioned were evaluated by measures like R2-square, RMSLE, and overall performances in predicting store sales. In a nutshell, our results show that the statistical model is not suitable in this case while the other two both perform well in general. Thus, we conclude that XGBoost and LSTM models are good enough for predicting store sales and could be a direction to do more research deeply in the future.

**Introduction**

Time series data widely exists in our lives. One of the most common types is store sales, whose forecasting is both important and complex for retailers. Forecasting is especially relevant to brick-and-mortar grocery stores, which must be flexible in terms of how much inventory is being purchased. More accurate forecasting systems will enhance supply chain operations and ensure customer demand. Unfortunately, grocers can easily run into problems with whether the sales are predicted a little over or under. Therefore, if more accurate forecasting methods are implemented, retailers can delight customers by offering the right quantity of products at the right time.

**Motivation**

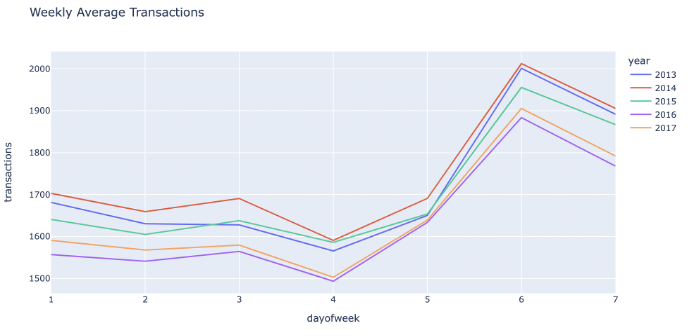
In the real world, forecasting has always been a mysterious but attractive field. However, the current subjective forecasting methods for retail stores are not able to realize automation and meet complex requirements regarding unique needs, new products, ever-transitioning seasonal tastes, and unpredictable product marketing. Therefore, we need to use statistical analysis and other related machine learning tools to fix the above problems and to make the forecasting more precise, in order to decrease the lost revenue and increase customer satisfaction.

**Related work**

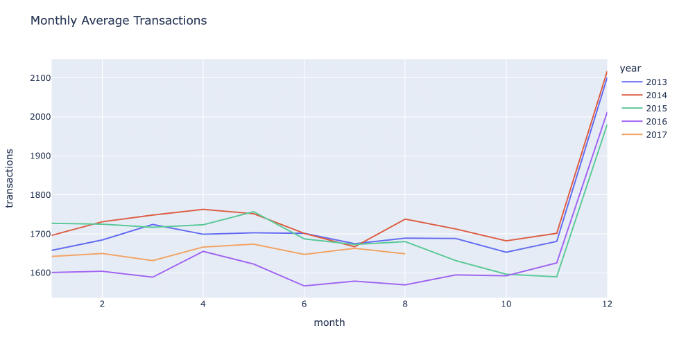
In most previous sales forecasting works, ARIMA[3] are basic and traditional models which focus on AR and MA terms of time series data. With in-depth research on machine learning and deep learning, new directions are available for time series forecasting. As a basic model, linear regression is used to predict the sales after features’ construction and selection.[4] XGBoost algorithm and feature engineering are combined to make predictions well.[5] Besides, some types of neural networks like RNN[6] show improvements over notable statistical methods.

**Methodology**

For the data visualization part, we customize different kinds of graphs according to the characteristics for each dataset. For example, line charts illustrate the pattern and trend of numerical variables. To investigate their features, these line charts are drawn on different time scales, like daily, weekly, and monthly (Figure 1, Figure 2). We draw bar plots to show the distribution of categorical variables and one tag cloud is constructed to display the popularity of sold products.

******

*Figure 1: Weekly Average Transactions*

******

*Figure 2: Monthly Average Transactions*

After the data visualization, we get some sense and basic understanding of the datasets. Our next step is dealing with the time-related data so that we could do the model construction stages later on. First of all, we convert the ‘date’ column in all datasets from object to datetime type. Categorical variables account for a large proportion in given sets, so we choose the label encoding method. There are some missing values in the oil price set, therefore a linear interpolation method is used to fill them.

Time Series Model:

From the statistical point of view, a time series can be viewed as an additive model, a combination of long-term trend and seasonality. This model focuses on the time series itself without any other added features. So, we need to design our time dummy variables, which are provided by the ‘DeterministicProcess’ function in the ‘statsmodel’ library. The ‘order’ parameter in this function is chosen as 1, a linear trend. For the seasonality, though it usually requires prior knowledge and is difficult for us to figure out, the fourier feature, ‘CalendarFourier’ function in python, is a way to capture periodic indicators. Then we combine the time dummy variables constructed from trend and seasonality to fit a linear regression model.

XGBoost Model:

In this model, we will consider more features than time-related ones. According to the given information, wages in the public sector are paid every two weeks on the 15th and on the last day of the month, so a ‘wage day’ feature is added. Another important dataset is ‘holidays\_events.csv’, which provides special days arrangement there. Based on different scopes of holiday celebrations, we construct ‘holiday\_national’, ’holiday\_regional’, ‘holiday\_local’ features. Similarly, we divide these events into three categories, ‘earthquake’, ’football’ (related to the world cup), ‘other events’. Adding time-related features, ‘year’, ‘month’, ’week’, ’day of week’, ’week of month’, we reach the final dataset. Then we use ‘XGBRegressor’ from the XGBoost library, which provides some parameters to avoid overfitting like n\_estimators and max\_depth. It is one of the fastest implementations of gradient boosted trees.

LSTM Model:

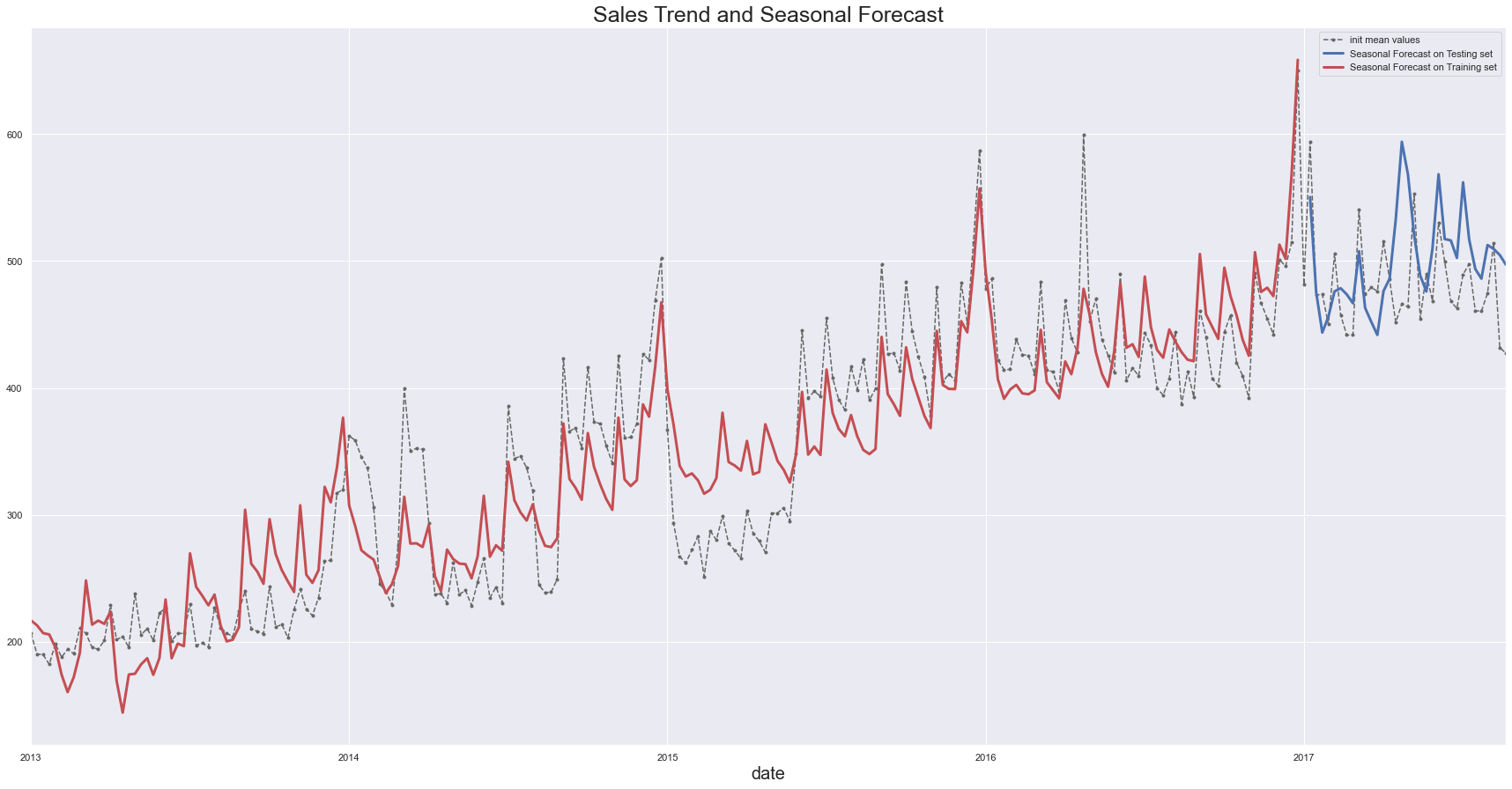
Now we come to the more advanced deep learning method - LSTM (long short-term memory networks). As we all know, LSTM is a special kind of RNN that is capable of learning long-term dependencies, which is especially useful in dealing with sequence prediction problems just as in our case, the store sales forecasting problems.

We first deal with the day shifts. Since we do not want to throw away the covariates and also know the LSTM only supports the future-known covariates, we shift the sales and transactions data forward 16 days into the future so that we could also use the data from our current solely past-known covariates. After dealing with this shift, we train the model with the rest of the datasets and make a forecast based on this 16 days test set. We split the data in training, validation, and testing sets for future tuning and validation processes with the 16 days shift manually. Next, we built the LSTM model with RNNModel() function from darts.models library and with parameter (model = “LSTM”). After that, we trained the model and reloaded the best model over the course of training. Finally, we perform the evaluation process by using metrics mean RMSLE (Root Mean Squared Logarithmic Error) from darts.metrics library to see how the model we constructed performed. We also constructed a simple LSTM model with LSTM function from the tensorflow.keras.layers library and without the shift, and we finally got some results as well.

**Results**

Time Series Model + Evaluation work:

From figure 3, the dashed line is the true value, the red and blue lines are predictions on train and test datasets. It shows the store sales continue to go up in general as time goes by. Though some kind of function of sales is predicted, it does not perform well on train sets compared with true value. Besides, since time series forecasting needs consecutive periods, we can only predict daily average sales as a whole. Although we can apply this model to each item in each store separately, it is time-consuming and not functional enough. So, this model is not suitable for this case.



*Figure 3: Sales Forecasting by Trend and Seasonality*

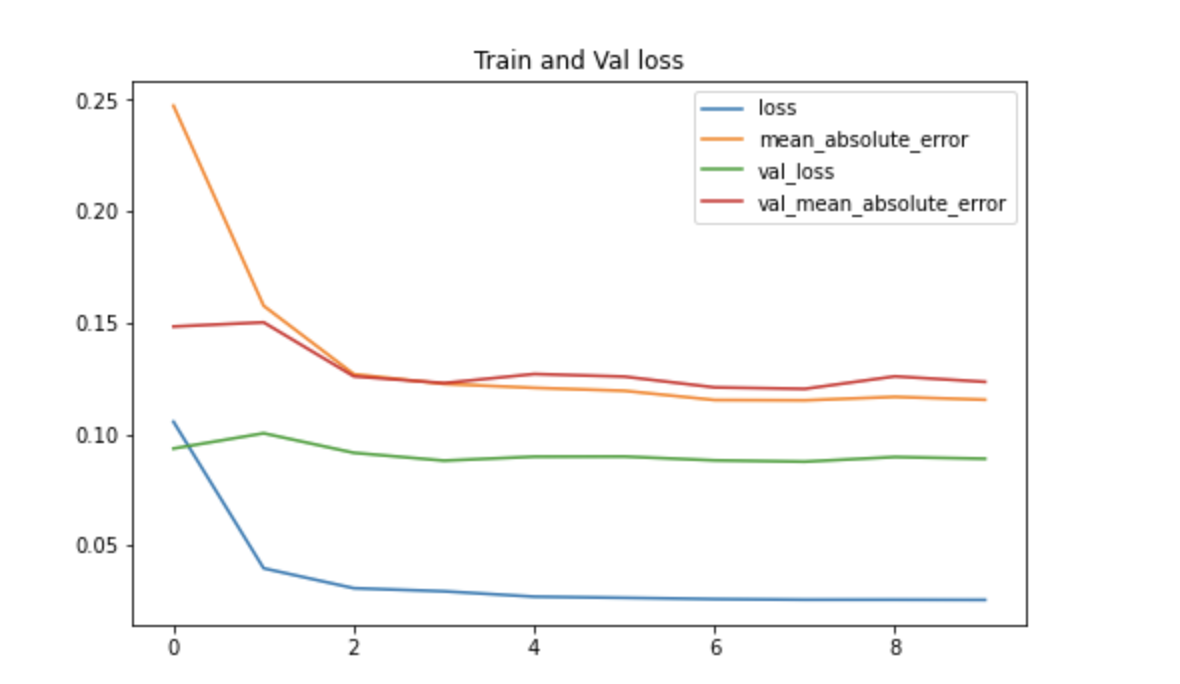
XGBoost + Evaluation work:

Since it is time series data, we cannot split the train and test set randomly. So, the data with ‘date’ before ‘2017-01-01’ are chosen as train and data with ‘date’ between ‘2017-01-01’ and ‘2017-08-15’ as test set. We set the n\_estimators=20 and max\_depth=6. The maximum tree depth represents each individual tree can grow to. The default value of 6 is a good starting point, and I haven’t found a need to go beyond a max\_depth of 6. After fitting the model, ‘model.score’ function, actually the R2-square, evaluates the performance of the model as 0.908, which seems a good model for this data set.

LSTM + Evaluation work:

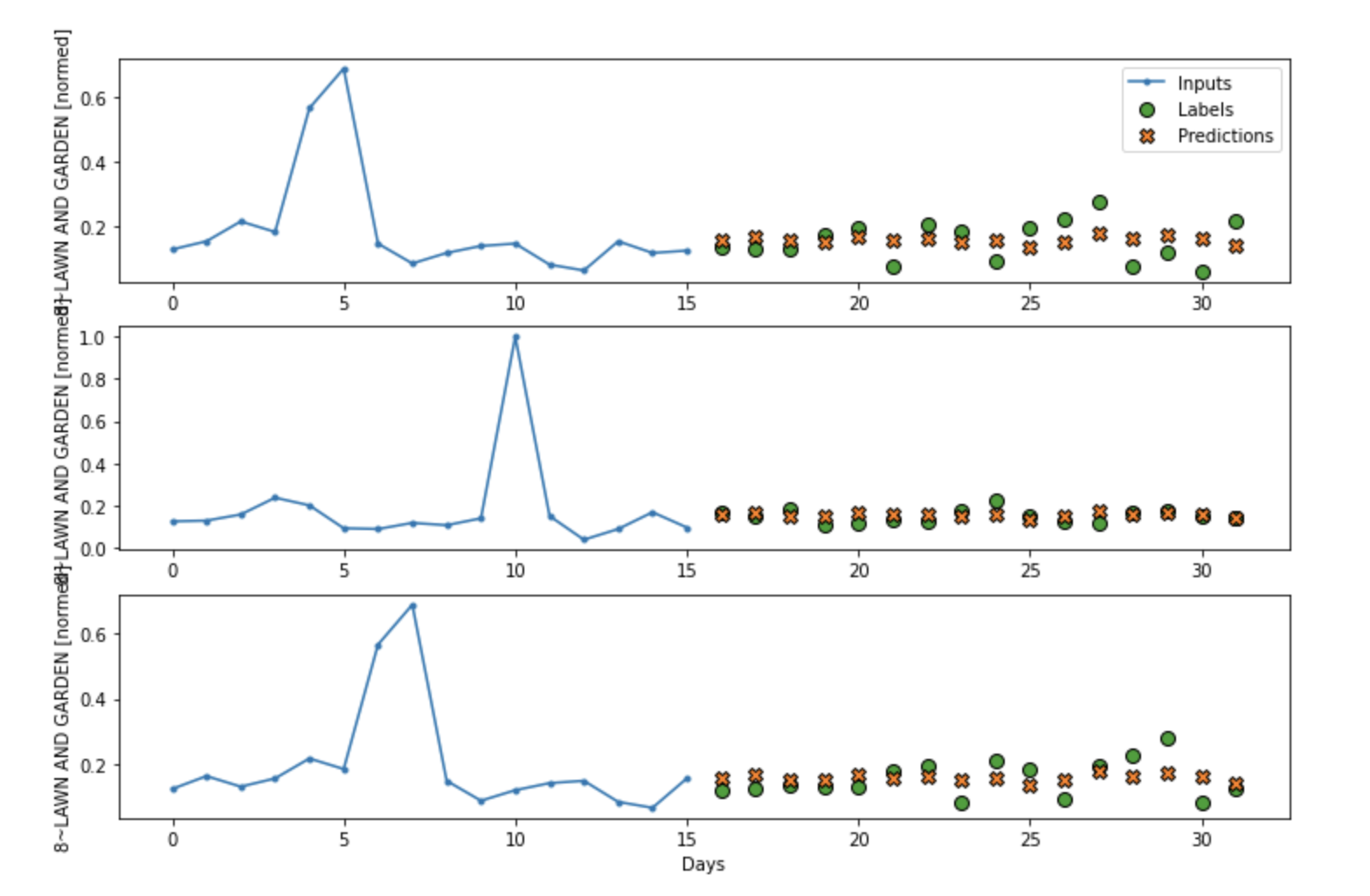
During the LSTM model construction, we are worried about the trade-off between loss and learning rate. Since when the learning rate is large, it's possible for loss to be nan; while we could also have the loss to be nan when we have large enough loss. Therefore we used data regulation ReLU with parameters (activation='ReLU') to decrease the loss. After the train - test splitting that we did manually mentioned in the Methodology section, we did the hyperparameters tuning. For the hyperparameters tuning of our 16-days shifted model, we used 131 to be the input chunk length, 39 hidden dimensions, 3 RNN layers, and about 0.002 learning rate to make the model perform the best. And we used the mean RMSLE as an evaluation measure or metric of this model. After computation, we got the mean RMSLE value to be around 0.55, which is a good score indicating the model is well-modeled based on the dataset.

Our basic LSTM model without the shift is constructed similarly with the tf.keras.layers.LSTM() function. We did not do any hyperparameter tuning for this model. For this model, we used loss, mean\_absolute\_error,val\_loss, and val\_mean\_absolute\_error as evaluation metrics. Here are some plots based on the model:



*Figure 4: Train and Validation Loss*

Here’s a graph representing loss and error for the training and validation set. As we can see all the errors and loss are decreasing as we fit in more data, which means the model is working well with the errors.



*Figure 5: Sales Prediction by LSTM*

Here’s a graph showing the prediction we had after the LSTM model training. The graph shows that our model fitted data pretty well, especially in the middle graph, where the labels and predictions almost overlap together.

Therefore, overall, we say both LSTM models we constructed worked well with these datasets.

**Conclusions**

In the project, we have successfully performed three methods - Time Series, XGBoost, and LSTM. Combined with their predicting performances and evaluation scores, we get some notable results. When the response variable, sales, are affected by many external factors, the statistical model’s predictive ability is limited. XGBoost and LSTM, built from different perspectives, both have an excellent performance on sales forecasting. Based on our current results, people should dive into these two kinds of machine learning models more deeply regarding their forecasting abilities to store sales in the future research work. We think store sales prediction is an interesting and worthy field for people to study, and we believe store sales will become more accurate and precise over time.

**References**

[1] [Store Sales - Time Series Forecasting | Kaggle](https://www.kaggle.com/competitions/store-sales-time-series-forecasting/overview)

[2] [Predicting sales in a food store department using machine learning](https://www.diva-portal.org/smash/get/diva2:1108597/FULLTEXT01.pdf)

[3] Jakaša T, Andročec I, Sprčić P. Electricity price forecasting—ARIMA model approach[C] //2011 8th International Conference on the European Energy Market (EEM). IEEE, 2011: 222 -225.

[4] Kausthub, Kodamagulla. (2021). COMMERCIALS SALES PREDICTION USING MULTIPLE LINEAR REGRESSION.

[5] Niu Y. Walmart Sales Forecasting using XGBoost algorithm and Feature engineering[C] //2020 International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE). IEEE, 2020: 458-461.

[6] Schmidt A, Kabir M W U, Hoque M T. Machine Learning Based Restaurant Sales Forecasting[J]. Machine Learning and Knowledge Extraction, 2022, 4(1): 105-130.