

M5 Forecasting - Accuracy

Estimate the unit sales of Walmart retail goods

CMPE257 – Machine Learning

Professor: Ming-Hwa Wang

Teng Gao, Huimin Li, Wenya Xie

San Jose State University, CA

M5 Forecasting - Accuracy

Estimate the unit sales of Walmart retail goods

Abstract	3
Introduction	4
1.1 Objective	4
1.2 What is the problem?	4
1.3 Why is this a project related to this class?	5
1.4 Why are other approaches not good?	5
1.5 Why do you think your approach is better?	6
1.6 Statement of the Problem	6
1.7 Area or Scope of Investigation	7
Theoretical Bases and Literature Review	8
2.1 Definition of the Problem	8
2.2 Theoretical Background of the Problem	8
Thoughts on Neural Network	12
Simple Regression Model	12
2.3 Related research to solve the problem	13
2.4 Advantage/disadvantage of those research	14

	2
2.5 Your solution to solve this problem	20
2.6 Where your solution different from others	21
2.7 Why your solution is better	21
Hypothesis	22
3.1 Single/multiple hypothesis	22
3.2 Positive Hypothesis	22
Methodology	23
4.1 How to generate/collect input data	23
4.2 How to solve the problem	24
4.3 How to generate output	27
4.4 How to test against hypotheses	28
4.5 How to proof correctness	29
Implementation	32
5.1 Code	32
5.2 Design Document and Flowchart	39
Data Analysis and Discussion	40

	3
6.1 Output Generation	40
6.2 Output Analysis	43
6.3 Compare Output Against Hypothesis	50
6.4 Abnormal Case Explanation	51
6.5 Static Regression	54
6.6 Discussion	54
Conclusion and Recommendation	56
7.1 Summary and Conclusions	56
7.2 Recommendation for Future Studies	56
Bibliography	57

Abstract

Department stores like Walmart have uncountable product and money transactions every day. Because of their rapid transaction rates, keeping a balance between the inventory and customer demand is the most important decision for the managers. Therefore, making an accurate sales prediction for different products becomes an essential need for stores to optimize their profits. Most of the existing sales predictions only depend on extrapolating the statistical trend. The previous studies on market sales prediction require a lot of extra information like customer and product analysis. A more simplified model is needed by the department store to predict the product sales based on only the historical sales record. The new emerging machine learning methods enable us to make more accurate predictions like that. Our data is from the Kaggle M5 Competition. We will use Support Vector Regression, Recurrent Neural Network, Simple Regression, and Neural Network to predict the next 28-day period of Walmart sales using the sales records, price and calendar information. Because making accurate predictions for each product on single days is almost impossible, this project will optimize the accuracy by all means for daily sales prediction.

Key Words: sales prediction, Walmart, Machine Learning, Neural Network

Introduction

1.1 Objective

The objective for this project is to estimate precisely for the product unit sales forecasting in the USA-based Walmart. In order to perform predictions on various products that are sold in Walmart, machine learning methods have been implemented along with the traditional methods to increase the precision. Three different machine learning models are used to forecast daily sales for following a 28-day period. The metric of evaluating the models is Root Mean Square Error(RMSE). The outcome from RMSE could support the stated hypothesis and assist the business analyst to improve planning on different aspects of the business level, for instance inventory distribution, distribution management, inventory storage solutions, product fulfillment, etc.

1.2 What is the problem?

The problem is misleading business forecasts on product sales could potentially cause opportunity and revenue loss for Walmart. For instance, if the analyst team tries to predict the weather for the following month at a specific area and the result of the model is that it would be sunny for most of the time. In consequence, for the inventory distribution stage of the management process, the retail stores of Walmart at that specific area would prepare less storage of umbrella because the prediction model proposes only a low quantity of umbrella is needed. However, the weather prediction is wrong that it rains for most of the time and a large quantity of umbrellas has been requested by the customers. As a result, the cost of shipment for allocating

the umbrella storage and the loss of opportunity for sales decrease the revenue for those retail stores; this will become beneficial for the competitors who share the same target market. Therefore, it is important to have accurate machine learning models to perform prediction on drawing business insights.

1.3 Why is this a project related to this class?

The project is related to this class because it requires machine learning models to compute the estimation on unit product sales for Walmart. All of the knowledge we learned in class, starting from database management to data exploration to robust machine learning models are expertise that we need for the project. Principally, in understanding the importance of manipulating and organizing the data to meet the threshold at the very first stage of data cleaning. Furthermore, the based learnings on machine learning support us to evaluate the significance and effects that different models could make, especially the advantages and disadvantages. In combination with the comprehension of the materials in this class, we can deliver a complete business proposal to the company to construct refinement.

1.4 Why are other approaches not good?

Other approaches without incorporating machine learning methods during analysis might reduce the precision of the outcome. For instance, the statistical models are unable to provide options for the analyst to choose the degree of interoperability. They could only enable lower perspective on prediction power. In addition, the models would be less robust that there would be no alternatives for selecting whether it is a supervised or unsupervised model. This significantly affects the repeatability of the function on prediction models. It further has an effect on the

schedule of the prediction process that instead of having the models to retrain on its own, the analyst has to repeat the process manually.

1.5 Why do you think your approach is better?

Our approach is better because it is more explicit. These models provide a deeper learning of the dataset instead of running the basic statistical analysis for drawing inference between variables. Machine learning methods require training (built based on mathematical algorithms) and testing for validation which increase the perfection of the model; models could be retrained during any stage of the running process for updated information to avoid inadequacy. In addition, features-based models take different variables into consideration for potential features and are useful when there are many layers of the parameters. Lastly, machine learning models are better alternatives for classification especially as there are many product categories that need to be applied.

1.6 Statement of the Problem

We want our outcomes from the machine learning models to provide the most accurate forecast for assisting the business analyst to deploy organization on inventory management. Today we have many approaches to solve the problem of predicting the product unit sales, but those methods are not providing the accurate forecast which potentially cause the revenue loss for Walmart. Therefore, we will be using three different machine learning models: Support Vector Regression, Long Short Term Memory Network, and Neural Network implemented along with a simple regression model in predicting the unit product daily sales for the following 28 days using product information dataset provided by Walmart.

1.7 Area or Scope of Investigation

- (1) Determine which machine learning models could be applied to solve the problem.
- (2) What are some solutions that others might have towards a similar problem/field of interest?
- (3) What are the incentives of solving the problem?
- (4) How could the solutions bring values to the company?
- (5) Determine some potential challenges that can possibly affect the research or the process of model testing.
- (6) Explore the metric of evaluation (RMSE).
- (7) What are methods to organize the data of the dataset? How to segment them into categories?

Theoretical Bases and Literature Review

2.1 Definition of the Problem

The historical daily unit sales data, price per product and store data was given by Kaggle.com, our task is to predict item sales at stores in various locations for the following 28-day time periods. The supplemental data was also given, which include the promotions, day of the week, and special events. These datasets can be used to improve the accuracy of prediction and drawing a closer insight on the performance of purchasing behaviors towards different products. If successful, our work will continue to advance the theory and practice of forecasting. The methods used can be applied in various business areas, such as setting up appropriate inventory or service levels.

2.2 Theoretical Background of the Problem

(1) SVR (Support Vector Regression)

SVR is a powerful machine-learning algorithm. Different from simple linear regressions whose goal is to minimize the sum of squared errors, the objective of SVR is to fit the best line within a predefined or threshold error. The basic formula of SVR can be expressed as following:

$$f(x) = (w \cdot \phi(x)) + b$$

where w is weight vector, b is bias, and $\phi(x)$ is a kernel function which could transform the nonlinear input into high-dimensional linear mode. There are many types of kernel such as

Polynomial Kernel, Gaussian Kernel, Sigmoid Kernel, etc. Vapnik (2000) introduced the ε -insensitivity loss function to SVR. It can be expressed as:

$$L_{\varepsilon}(f(x) - y) = \begin{cases} |f(x) - y| - \varepsilon & \text{if } |f(x) - y| \geq \varepsilon \\ 0 & \text{otherwise} \end{cases}$$

where y is the target output, ε (epsilon) is the maximum error permitted, and when the predicted value falls into the boundary, the loss is zero. On the other hand, if the predicted value is out of the boundary, the loss is equal to the difference between the predicted value and the margin.

The above model is correct only if we assume that the problem is feasible. If we want to allow some errors, we should introduce some slack-variables ($\xi_i + \xi_i^*$) to enlarge the tolerance of the machine. So, considering empirical risk and structure risk synchronously, the SVR model can be expressed to minimize:

$$\begin{aligned} \text{Min : } & \frac{1}{2} w^T w + C \sum_i (\xi_i + \xi_i^*) \\ \text{Subject to } & \begin{cases} y_i - w^T x_i - b \leq \varepsilon + \xi_i \\ w^T x_i + b - y_i \leq \varepsilon + \xi_i^* \\ \xi_i, \quad \xi_i^* \geq 0 \end{cases} \end{aligned}$$

where $i = 1, 2, \dots, n$ is the number of training data; ($\xi_i + \xi_i^*$) is the empirical risk; $1/2 w^T w$ is the structure risk preventing over-learning and lack of applied universality; C is the regularization parameter representing the trade-off between empirical risk and structure risk. After selecting proper regularization parameter (C), error margin (ε), and kernel function (K), we can find the best line to fit the input data. So, SVR gives us the flexibility to define how much

error is acceptable in our model and will find an appropriate line (or hyperplane in higher dimensions) to fit the data.

(2) LSTM Networks (Long Short Term Memory)

LSTM network is a special model that forms by the Recurrent Neural Network. It is capable of learning long-term dependencies. LSTM network provides feedback connections; it avoids dependency (long-term) problems and solves vanishing and exploding gradients. This method is widely used for classifying and forecasting on time series dataset because it gives us more controllability and in a time series there are important durations where unknown periods happen during significant events. LSTM network behaves similar to RNN (Recurrent Neural Network) that has a structure chain where modules are repeating but in different layers. Its unit has four components: a cell, an input gate, an output gate, and a forget gate. LSTM network allows operation of deletion or addition of information into the cell; the gates regulate the activity flow of information from the cell. The input gate determines the access of whether the updated information is allowed to enter the cell. In the input sequence, the cell's responsibility is to control the dependencies among elements. The forget gate decides if the value should remain or throw away in the cell. The output gate controls which value could be used to compute the output.

The LSTM network formulas for gates could be expressed as:

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i)$$

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f)$$

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o)$$

Equation of Gates

Where i_t is the input gate; f_t is the forget gates; O_t is the output gate; σ is the sigmoid function; O_t is the weight for the respective gate(x) neurons; h_{t-1} is the output of the previous LSTM block at timestamp t-1; x_t is the input at current timestamp; b_t is biases for the respective gate(x).

The LSTM network formula for the final output could be expressed as:

$$\begin{aligned}\tilde{c}_t &= \tanh(w_c[h_{t-1}, x_t] + b_c) \\ c_t &= f_t * c_{t-1} + i_t * \tilde{c}_t \\ h_t &= o_t * \tanh(c^t)\end{aligned}$$

The first formula is the candidate for cell state at timestamp (t); c_t is the cell state(memory) at timestamp (t).

According to Gers & Schmidhuber (2000), they introduce the LSTM network with Peephole Connections. This model allows us to make decisions during the stages of input gate and forget gate together. Its formula can be expressed as follow:

$$\begin{aligned}f_t &= \sigma_g(W_f x_t + U_f c_{t-1} + b_f) \\ i_t &= \sigma_g(W_i x_t + U_i c_{t-1} + b_i) \\ o_t &= \sigma_g(W_o x_t + U_o c_{t-1} + b_o) \\ c_t &= f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + b_c) \\ h_t &= \sigma_h(o_t \circ c_t)\end{aligned}$$

Where i, o, f represent the peephole connections. Compared to the previous formulas, the only difference is that instead of h_{t-1} , it is c_{t-1} (which represents how the activation of memory cell c contributes to the timestamp of t-1).

(3) Simple Regression Model

A simple regression model will be applied to the data as a substitute of the other models. The calendar dataset includes the week of the day and holiday information of the dates of sales. The price data also have certain discounts on the product at some time. Therefore, we will make a regression model on different times and discount for different big categories. The model will be like the following.

$$S = w_0 + \sum_{i=1}^n w_i x_i$$

Where S is the sales prediction for the day. w_0 and w_i are different weights and will be adjusted by the historical data. x_i s are metrics that we considered for this model.

We will compare this model with different orders of the objective function to see if higher order is needed for this case. The best order will be selected according to the performance of each target function.

(4) Neural Network Model

Because of the limited information provided by the dataset, a neural network should be a good choice as it can auto select the hidden features and fit the data accordingly. The common simple multi-layer neural networks should be sufficient enough to output one value for each input feature. Although most of the neural networks are used in image or voice recognition, it will work for this project because each feature will be encoded into something without particular meaning.

We will use a simple 3 layer neural network for this project to predict the number of sales for each product. Sigmoid function is selected to be the activation function in the hidden layer because all our outputs are integers. Sigmoid function simply gives an output of 0 or 1 to help make a choice among output values.

2.3 Related research to solve the problem

In recent years, sales forecast models have been studied by a lot of scholars and experts.

1) Time series analysis

Na et al. (2019) used the Autoregressive Integrated Moving Average (ARIMA) model to forecast the sales volume of three convenience stores in Inner Mongolia. Based on the result of ACF and PACF, ARMA(1,1) model was chosen for the short-term sales forecast.

2) Machine Learning and Neural Network

Lingxian et al. (2019) proposed an integrated framework that combines traditional unsupervised learning model, K-mean clustering, and advanced artificial intelligence model of the Long Short Term Memory (LSTM) to predict future online retail sales dynamics.

Chen, I. and Lu, C. J. (2016) proposed six clustering-based forecasting models and compared their performance in sales prediction. The experimental results showed that the model combining the Growing Hierarchical Self-organizing Map GHSOM and Extreme Learning Machine (ELM), and the single Support Vector Regression (SVR) and ELM model displayed a better performance for computer retailing forecasting.

Di Pillo et al. described how to use Support Vector Machine (SVM) model to perform sales forecasting with predicting the daily sales of pasta in two retail stores.

Chen et al. (2018) propose a novel framework called TADA to carry out trend alignment with dual attention, multi-task RNNs for sales forecasting and divide the influential factors into internal features and external features. In 2019, they (Chen et al. 2019) enhanced this model with an online learning module based on their innovative similarity-based reservoir to improve the prediction accuracy for new data.

Moon compared Linear Model, SVM, and Artificial Neural Network(ANN) for the near-weekend movie box-office prediction (Moon, 2014). They further more analyzed the customer reviews to add complexity to their models for higher accuracy.

2.4 Advantage/disadvantage of those research

- The research “The empirical analysis of convenience store sales forecast based on multidimensional data in emerging markets”
 - Advantages:
 - (1) The ARIMA model used in this paper can well describe the changing trend of the sales situation and provide a basis for enterprises to formulate effective sales strategy and industrial layout.
 - (2) The geographical information was used in this research to analyze the future sales volume, which is more comprehensive.
 - (3) The ARIMA model used in this paper is flexible, since the parameters (p, d, q) could be modified to represent different types of time series.
 - (4) As a strictly statistical approach, the ARIMA method only requires the prior data of a time series to generalize the forecast.

- Disadvantages:

- (1) Pre-assumed linear relationship. ARIMA models assume that future values of a time series have a linear relationship with current and past values.

However, most of the questions in the real world are not linearly related.

So, the ARIMA models may not be a good choice for complex nonlinear real-world problems.

- (2) Only for univariate data forecasts. ARIMA models only use the historical value of the time series. It doesn't consider the influence of other predictors while forecasting, So, the forecast might not be accurate.

- The research “Online Retail Sales Prediction with Integrated Framework of K-mean and Neural Network”

- Advantages:

- (1) It proposed a new method to combined K-mean clustering model with LSTM model, in which K-mean method was used to label samples with respect to their overall characteristics, then LSTM) model of Recurrent Neural Network (RNN) was applied to learn the long-term and short-term information of time series and predict the sales dynamic.

- (2) This method overcomes the disadvantage of artificial intelligence method and traditional method. It solved problems with less time than an artificial intelligence method and made more accurate predictions than traditional methods.

- Disadvantages:

- (1) This paper only considered four factors (unit price, sales amount, total sales and time) to build a model, while other variables such as product descriptions, which contains the potential information of consumers' preferences for different products could also be used in the model.
 - (2) The geographical factor did not be taken into account in this model.
 - (3) The consumer behavior preferences factors as well as customers' personal information should be taken into account for building models, thus achieving sales prediction for each product.
- The project "Prediction the Near-Weekend Tickets Sales Using Web-based External Factors and Box-office Data"
 - Advantages:
 - (1) They applied multiple models for predicting the same results. Therefore, even if one of the models didn't work well, they can still optimize the result using other models.
 - (2) By comparing different models, researches can find which model will be the best to solve this problem. Sometimes simple models can work pretty well on particular problems. It is important to take them into account.
 - (3) Considering the external features is a perfect fit for this problem because movie box-office is strongly relying on people's comments. Finding the most influential external factor is a good thought while constructing the models.
 - Disadvantages:

- (1) This project only considered three separate models, while did not try to put them together. Ensemble is a good way to try while several models are built on the same dataset and their result did not agree that much. By using ensembles on those three models, they may find a better result.
 - (2) While conducting the comment rating analysis, they simply put all the comments into Hbase for the NLP analysis. On the other hand, not all the movie ratings and comments are valid form those websites. There are a lot of auto-generated and fake comments as well. They should do a sorting before putting those comments into the natural language processing.
- The research “Statistical and Machine Learning-based E-commerce Sales Forecasting”
 - Advantages:
 - (1) It proposed three different machine learning models: I-ARIMA, LSTM mode, and ANN model on sales prediction which provided various aspects on solving the same problem.
 - (2) They suggested normalization for potential features for creating new parameters when applying to the models, which reflected the impacts normalization could have on features and the relationship among variables.
 - (3) They analyzed the influence of feature changes and inspected the behaviors of the variables when one feature is fixed, which became good guidance when making market decisions.
 - (4) They performed tests on showing the impact of history data on accuracy to

reinforce their understanding of the model behavior that models behave better with historical data.

- (5) Included different types of features (intrinsic and extrinsic) for considerations which improved the models into a higher degree of interpretability.

- Disadvantages:

- (1) When they introduced the idea that using historical data when applying the Artificial Neural Network could possibly have a higher accuracy rate than I-ARIMA. However, they chose not to run a model to prove their idea but just to present the idea for future work.

- The research “Foreign Commodity Sales Forecast Based on Model Fusion”

- Advantages:

- (1) They differentiated two types of features, statistical and discrete, and applied sliding window sampling method and feature engineering for improving the accuracy of feature selection.
- (2) They applied three different models, from Linear Regression Model to XGBoost model to LightGBM model to prove multiple linear regression models could be used for data exploration, but might not be a sufficient method for making predictions compared with other two methods.
- (3) Clarification on the distinction between XGBoost model and LightGBM model that LightGBM is superior to XGBoost in that it requires less cost, high efficiency, and low memory consumption.

- Disadvantages:

- (1) Instead of using linear regression models as one of the machine learning models, they could have tested at least one more ML model for the dataset since linear regression models could only be used for learning the inference between variables for most of the time; they should consider other models with high complexity to incorporate with potential features.
- (2) They should include some background information about the dataset, for instance, the variables that the dataset contains because presenting features to the audience are important.
- (3) For the feature extraction part, they should clarify which features they are planning to use and what makes them become the potential features (include results if they have done some tests on the comparing the features).

2.5 Your solution to solve this problem

- The SVR model has been proved a economical and easy to implement model in time series prediction by some scholars. Compared with statistics methods, SVR model avoids overfitting and has good generality capability since it introduces the regularization parameters, which makes it being increasingly popular. Thus, we plan to use it in sales forecasting for Walmart.
- The LSTM (Long Short Term Memory) network will be applying to the sales dataset. The dataset contains thousands of columns of data with 1913 historical records. In other

words, even if the LSTM network requires high data demand for analysis, such a method will still be suitable for the dataset since it meets LSTM network's requirement.

Furthermore, LSTM network is mainly used as a time series model to discover the correlation between features, which we could use to analyze how unit product sales are relevant between days.

- The Recurrent Neural Network (RNN) model will only focus on the sales data. Therefore, the model itself is immunized to the external factors while the external features' data is unattainable for this project. Using RNN, we are able to find out the pattern of the sales behavior along with the time. The strength of RNN to memorize the pattern should be able to help us find out the relative trend of the Walmart sales in the next 28 days period.
- While the other models in this project are mostly focused on the trend of the sales, the simple regression model will consider the other factors provided by the dataset. This model will consider the distinct days of the week, holidays, discount on the product. This should be a compliment of the other models to see if this simple model will work out for sales prediction.
- Lastly, we will use the same feature selection as the regression model to the simple neural network mode. We will use a 3-layer neural network for classification to find out the best output for each product. Because after feature engineering, the features have no exact meaning, the neural network model should be a good fit to deal with this kind of problem.

We will eventually aggregate all our models together to try to get a more accurate result. We will put weights on the models and adjust weights according to the error of each model.

2.6 Where your solution different from others

Instead of using a unique model, we combine the machine learning techniques with a simple regression model. This could increase the accuracy of the forecast. In addition, because each of our models will have different focuses on the characteristics of the dataset, combining them together will result in a more comprehensive aggregated model.

2.7 Why your solution is better

Our method hybrid traditional regression model and clustering model with neural network models, and compare the performance of each model, aiming to find the optimal method for daily sales forecasting.

Hypothesis

3.1 Single/multiple hypothesis

We would like to provide a consistent model to predict the Walmart sales of each product for the next 28 days window based on the previous sales history. We will take the internal factors, which are the sales on each day, and the external factor, which are the date, special events and price of the product. We would like to test on several machine learning models to see how to make a more accurate forecast. Because making a prediction of each day is difficult based on the limited conditional data, we assume that the 28 days prediction for each product should have more than 60 % accuracy on average and RMSE around one. If the results meet our expectation, the models should be a success.

Our models are based on the assumption that, for each product, the sales in the Walmart market have a certain pattern along the timeline. Our model should be able to memorize the pattern and do the further prediction.

Our hypothesis is also based on the assumption that customers' demand and purchasing habits are also in some pattern along the time.

3.2 Positive Hypothesis

We will test the performance of our models by testing the rooted mean squared error (RMSE). Our model will also adjust themselves based on the error metrics in each train-test cycle. We will adjust our model to keep the RMSE as low as possible. The RMSE values should

be lower than 2, and the accuracy should be greater than 60% in testing to ensure the result is sounding.

Methodology

4.1 How to generate/collect input data

The data was provided by Kaggle website: M5 Forecasting - Accuracy, Estimate the unit sales of Walmart retail goods. The data covers stores in three US States (California, Texas, and Wisconsin) and includes item level, department, product categories, and store details. In addition, it has explanatory variables such as price, promotions, day of the week, and special events. Together, this robust dataset can be used to improve forecasting accuracy. It contains the following tables:

- Calendar: Contains information about the dates on which the products are sold and special events information, like national vacations and religious memorial days.

	date	wm_yr_wk	weekday	wday	month	year	d	event_name_1	event_type_1
0	2011-01-29	11101	Saturday	1	1	2011	d_1	NaN	NaN
1	2011-01-30	11101	Sunday	2	1	2011	d_2	NaN	NaN
2	2011-01-31	11101	Monday	3	1	2011	d_3	NaN	NaN
3	2011-02-01	11101	Tuesday	4	2	2011	d_4	NaN	NaN
4	2011-02-02	11101	Wednesday	5	2	2011	d_5	NaN	NaN

Figure 4-1. Calendar table

- Sales data: Contains the historical daily unit sales data per product and store from 2011/01/29 to 2016/06/14, which is 1913 days. The products are distributed in 3 categories: 'Hobbies', 'Household', 'Foods', and the department id only varies on the sequential numbers. Because of this limitation in the data, gathering external information for different products becomes almost impossible.

dept_id	cat_id	store_id	state_id	d_1	d_2	d_3	d_4	...	d_1904	d_1905	d_1906	d_1907	d_1908	d_1909	d_1910	d_1911	d_1912	d_1913
HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	...	1	3	0	1	1	1	3	0	1	1
HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	...	0	0	0	0	0	1	0	0	0	0
HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	...	2	1	2	1	1	1	0	1	1	1
HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	...	1	0	5	4	1	0	1	3	7	2
HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	...	2	1	1	0	1	1	2	2	2	4

Figure 4-2. Sales table

- **Sell price:** Contains information about the price of the products sold per store and date, which could indicate the promotion activity. The sell price of each product is marked on a weekly basis.

	store_id	item_id	wm_yr_wk	sell_price
0	CA_1	HOBBIES_1_001	11325	9.58
1	CA_1	HOBBIES_1_001	11326	9.58
2	CA_1	HOBBIES_1_001	11327	8.26
3	CA_1	HOBBIES_1_001	11328	8.26
4	CA_1	HOBBIES_1_001	11329	8.26

Figure 4-3. Sell price table

4.2 How to solve the problem

- **Algorithm Design**

The sales amounts of the first 1530 days (80 % of the total data) are used as the training sample, while the remaining 383 days (20 % of the total data) are used as the testing sample. We will experiment and compare four algorithms including simple regression, SVR, Neural Network and LSTM.

(1) SVR

- Load the dataset and implement feature scaling
 - Get the dates and sales data of each product preparing for the training model.
 - Transfer the data format into 1 dimensional vector.
- Split training/testing data
 - Split the dataset into a training and testing set with the ratio of 80:20.
- Select appropriate parameters for SVR model
 - Kernel function: the most widely used radial basis function (RBF) , the formula of RBF is as following:

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right)$$
 - We will perform a grid search to find the optimal C, ϵ and gamma (the radius of RBF) for the model. We can set the parameter-searching scope of the three variables from 2^{-5} to 2^5 . If the Root Mean Squared Error (RMSE) are not small enough, we can change the ranges used in the grid-search and further optimize the model to check if the model accuracy increases. In this step, we will select optimal parameters for each product.
- Evaluate the performance of the model with a test set and output the testing RMSE for each model. If the value of RMSE is not small enough, tune the model again until we find the best one.

(2) LSTM Network

- Transform time series to supervised learning
 - Shift the values in series down by 1 place for $t-1$ (using historical data to predict the current value)
- Transform time series to stationary
 - Remove dependency on time that subtract observation of the previous timestamp ($t-1$) from the current timestamp (t)
- Transform time series to scale
 - Transform the dataset into range $[-1,1]$ since for the activation function's (tanh) output is in between -1 and 1.
- Fitting a LSTM network model to the reshaped training data
 - Import LSTM from `keras.layer`
- Measure the performance of the LSTM network model on the testing data
- Conclude the forecast outcome of the prediction model

(3) Polynomial Regression Method

- Feature selection
 - Feature selection is the most important step in regression modeling. In this case, we will choose the characteristics of the time and discount rate as the features for regression analysis.
- Feature transformation
 - We use the Column Transformer and Onehotencoder in the Sklearn package from Python to transform the ordinal data into several columns with binary values. The ordinal values like days of the week, vacation, or

month do not contain specific relationship with the sales. Therefore, transferring each value into distinct columns as multiple features is very helpful for regression.

- Model selection and weight adjustment
 - We will start with the first order regression, checking the RMSE and error rate of the output. We will compare the error under different choices in order to see which polynomial model is the best fit for our case.

(4)Neural Network

- Use the same dataset as the simple regression method
- 3-layer shallow neural network performed on the data to classify each day
- Sigmoid function is used as the activation function
- Backpropagation is used as a way of adjusting weights on each hidden node based on the error on the output
- **Language used**

Language used in this project is Python.

- **Tools used**

Jupyter notebook, Python libraries, Tensorflow.

4.3 How to generate output

With the training data, trained models will be generated. Then the test data set is used to evaluate the accuracy of these models. We will use the model with the smallest RMSSE as the final model to predict the daily sales and the daily sales in the next 28 days of Walmart.

The output will be in table format. The amount of sales prediction will be generated for each day. For those 28 days predictions, we will make for each of the products in each store, as well as the aggregated amount in different states and different stores. The sample format of output is demonstrated in the following table (Figure. 4-4)

A	B	C	D	E	F	G	H	I	J	K
id	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10
Total_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
CA_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
TX_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
WI_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
CA_1_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
CA_2_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
CA_3_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
CA_4_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
TX_1_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
TX_2_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
TX_3_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
WI_1_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
WI_2_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
WI_3_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
FOODS_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
HOBBIES_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
HOUSEHOLD_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
FOODS_1_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
FOODS_2_X_0.005_validation	0	0	0	0	0	0	0	0	0	0
FOODS_3_X_0.005_validation	0	0	0	0	0	0	0	0	0	0

Figure. 4-4 Sample Output

4.4 How to test against hypotheses

If the accuracy of our output is less than 90% on average for each of the products in the testing set, our test may fail to prove enough strength on predicting the further 28-days prediction window. Also, the accuracy score feedback from Kaggle should be a valid qualifier of our models' performance.

Applying our model to any other store sales records should provide an accurate prediction output. Trying other similar sales records data on our models should also provide a successful result.

4.5 How to proof correctness

For each product, the sales record will be train-test split using cross validation. We will train the data with the training set and test with the testing set. An aggregated error will be shown after all the cross validation.

The Kaggle also has its own way of testing the correctness of the models. It will show a score once the model is implemented. We will upload the model on Kaggle to see the accuracy once the project is done.

An actual record of the 28 days sales records will also be provided by Kaggle on June 1 to test the correctness of our models.

Implementation

5.1 Code

- SVR Model

The process of SVR model building can be broken into 5 steps: loading dataset, getting product data, splitting dataset into training and testing set, training and tuning model and testing model. The code for SVR model is shown as following:

Step 1: load dataset.

```

import pandas as pd
import numpy as np

salesdata = pd.read_csv("sales_train_selected.csv")
salesdata

```

Step 2: Data transformation

In this step, we get the date and sales data for each product, and convert the data from Pandas dataframe to 1 dimensional vectors to make it fit the SVR model. At the end, this function returns the dates and sales by product.

```

# Create a product name list
def products_name(salesdata):
    products = []
    for p in salesdata['id']:
        products.append(p)
    return products

# Return the dates and sales of product
def ProductData(product_name):
    salesdata_p = salesdata.loc[salesdata['id'] == product_name]
    date = {'date': [i for i in range(1, 1914)]}
    datesdata = pd.DataFrame(date)
    dates = datesdata['date'].as_matrix()
    sales = salesdata_p.iloc[:, 6:].T.as_matrix()
    #Convert to 1d Vector
    dates = np.reshape(dates, (len(dates), 1))
    sales = np.reshape(sales, (len(sales), 1))

    return dates, sales

```

Step 3: Split training/testing set

We split each dataset into the training and testing set with a ratio of 80:20. The data of the former 1530 days was used as training samples and the other 383 days as testing samples.

Split training/testing set

```
def TrainTest (dates, sales):
    training_dates_80 = dates[0:1530]
    training_sales_80 = sales[0:1530]
    test_dates_80 = dates[1530:]
    test_sales_80 = sales[1530:]
    #print(len(training_dates_80), len(training_sales_80), len(test_dates_80), len(test_sales_80))
    return training_dates_80, training_sales_80, test_dates_80, test_sales_80
```

Step 4: Train the model

The sklearn library was imported to train SVR models. The most important phase in training SVR models is determining the kernel function and optimal parameters. We selected the most common RBF as the kernel function. Grid search was applied to find the optimal parameters: C and gamma. In the grid search process, we tried different ranges to tune models from 1e-6 to 10,000 and gradually narrowed it down to hit the best one for each model. Then we fit the optimal model to training samples. In this step, we could get the trained optimal model and the training RMSE.

Train model

```
from sklearn.svm import SVR
import warnings
warnings.filterwarnings('ignore')

import matplotlib.pyplot as plt
%matplotlib inline

from sklearn.metrics import mean_squared_error
import math
from math import sqrt
from sklearn.model_selection import GridSearchCV
from pprint import pprint
```

```

def svr_model_5(X, y):
    # Find the optimal parameters (C and gamma) using Grid Search with 5-fold cross validation.
    gsc = GridSearchCV(
        estimator=SVR(kernel='rbf'),
        param_grid={
            'C': [0.05, 0.1, 1, 2, 4, 8, 10],
            'gamma': [1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 5]
        },
        cv=5, scoring='neg_mean_squared_error', verbose=0, n_jobs=-1, return_train_score=True)

    grid_result = gsc.fit(X, y)
    best_params = grid_result.best_params_
    best_score = grid_result.best_score_
    #pprint(grid_result.cv_results_)
    print("The optimal parameters:", best_params, "| Best score:", best_score)

    best_svr = SVR(kernel='rbf', C=best_params["C"], gamma=best_params["gamma"])

    best_svr.fit(X, y)
    predicted_sales_train = best_svr.predict(X)
    #for d, i, j in zip(X, y, predicted_sales_train):
    #    print(d, i, j)

    tr_rmse = sqrt(mean_squared_error(y, predicted_sales_train))
    return best_svr, predicted_sales_train, tr_rmse

```

Step 5: test model

The data in the testing set will be used to evaluate the generalization capacity of our models and output the testing RMSE.

Test model

```

def Testing(X, y, best_svr):
    predicted_sales_test = best_svr.predict(X)
    #for d, i, j in zip(X, y, predicted_sales_test):
    #    print(d, i, j)
    test_rmse = sqrt(mean_squared_error(y, predicted_sales_test))

    return predicted_sales_test, test_rmse

```

- LSTM Model

The process of generating the outputs for the LSTM models has several important phases that are needed to be highlighted, which include loading all of the necessary libraries, establishing the predictor class function, rescaling the data, and defining the training model.

Importing the libraries:

```
import torch
from tqdm import tqdm
from pylab import rcParams
from torch import nn, optim
import torch.nn.functional as F
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
from statsmodels.tools.eval_measures import rmse
from sklearn.preprocessing import MinMaxScaler
from keras.preprocessing.sequence import TimeseriesGenerator
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout
import warnings
warnings.filterwarnings("ignore")
```

Predictor Class Function:

```
class M5Predictor(nn.Module):

    def __init__(self, input_dim, hidden_dim, seq_len, num_layers = 2):
        super(M5Predictor, self).__init__()

        self.input_dim = input_dim
        self.hidden_dim = hidden_dim
        self.seq_len = seq_len
        self.num_layers = num_layers

        self.lstm = nn.LSTM(
            input_size = input_dim,
            hidden_size = hidden_dim,
            num_layers = num_layers,
            dropout = 0.5
        )

        self.linear = nn.Linear(in_features=hidden_dim, out_features=1)

    def reset_hidden_state(self):
        self.hidden = (
            torch.zeros(self.num_layers, self.seq_len, self.hidden_dim),
            torch.zeros(self.num_layers, self.seq_len, self.hidden_dim)
        )

    def forward(self, input):
        lstm_out, _ = self.lstm(
            input.view(len(input), self.seq_len, -1),
            self.hidden
        )
        y_pred = self.linear(
            lstm_out.view(self.seq_len, len(input), self.hidden_dim)[-1]
        )
        return y_pred
```

Rescaling between 0 and 1:

```
scaler = MinMaxScaler()
scaler = scaler.fit(np.expand_dims(train_data, axis=1))
train_data = scaler.transform(np.expand_dims(train_data, axis=1))
test_data = scaler.transform(np.expand_dims(test_data, axis=1))
```

Training Model Based on Epoch 0-50:

```
def train_model(model, train_data, train_labels, test_data=None, test_labels=None):

    def RMSELoss(yhat,y):
        return torch.sqrt(torch.mean((yhat-y)**2))

    optimiser = optim.Adam(model.parameters(), lr=0.001)

    num_epochs = 60

    train_hist = np.zeros(num_epochs)
    test_hist = np.zeros(num_epochs)

    for t in range(num_epochs):
        model.reset_hidden_state()

        y_pred = model(X_train)

        loss = RMSELoss(y_pred.float(), Y_train)

        if test_data is not None:
            with torch.no_grad():
                y_test_pred = model(X_test)
                test_loss = RMSELoss(y_test_pred.float(), Y_test)
                test_hist[t]=test_loss.item()

            if t % 10==0:
                print(f'Epoch {t} train loss:{loss.item()} test loss: {test_loss.item()}')
            elif t % 10 ==0:
                print(f'Epoch {t} train loss:{loss.item()}')

        train_hist[t] = loss.item()

        optimiser.zero_grad()
        loss.backward()
        optimiser.step()
    return model.eval(), train_hist, test_hist
```

- Regression Model

Our regression and neural network model use the same processed data. The data processing involves data merging, filling, and encoding. The encoding process is especially important for those models because it minimizes the meaning of the date values, treating them as distinct values. For example, we transformed the days of week

feature from one column into seven columns, using 0s and 1s to indicate which day it is through the week. The packages we used are SimpleImputer, ColumnTransformer, and OneHotEncoder.

```

1 def data_pre(i):
2     df = product.iloc[i].iloc[6:]
3     df_n = df.reset_index()
4     df_n.columns = ['d', 'sales']
5     df_calendar = pd.merge(df_n, calendar)
6     price_item = price[price['item_id'] == product.iloc[i].iloc[1]]
7     price_itemstore = price_item[price_item['store_id'] == product.iloc[i].iloc[4]]
8     pd_n = pd.merge(df_calendar, price_itemstore)
9     pd_n['discount'] = pd_n['sell_price']/max(pd_n['sell_price'])
10    pd_nn = pd_n[['wday', 'month', 'year', 'event_type_1', 'discount', 'sales']]
11
12    pd_next = pd.merge(calendar[1913:(1913+28)], price_itemstore)
13    pd_next['discount'] = pd_n['sell_price']/max(pd_n['sell_price'])
14    pd_next_n = pd_next[['wday', 'month', 'year', 'event_type_1', 'discount']]
15    X_old = pd_nn.iloc[:, :-1]
16    X = X_old.append(pd_next_n).values
17    Y = pd_nn.iloc[:, -1].values
18    imputer = SimpleImputer(missing_values=np.nan, strategy='constant', fill_value = 'normal')
19    imputer.fit(X[:, 1:6])
20    X[:, 1:6] = imputer.transform(X[:, 1:6])
21
22    ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [3])], remainder='passthrough')
23    X = np.array(ct.fit_transform(X))
24    ct2 = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [5])], remainder='passthrough')
25    X = np.array(ct2.fit_transform(X))
26    ct3 = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [12])], remainder='passthrough')
27    X = np.array(ct3.fit_transform(X))
28    ct4 = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [24])], remainder='passthrough')
29    X = np.array(ct4.fit_transform(X))
30
31    X_new_train = X[:-28]
32    X_28 = X[-28:]
33
34    X_train, X_test, y_train, y_test = train_test_split(X_new_train, Y, test_size = 0.2, random_state = 1)
35
36    return X_train, X_test, y_train, y_test, X_28

```

For the regression model, we just used the simple linear regression package from sklearn and added polynomial features to it to enable polynomial regression.


```

1 def linear_regression(X_train, X_test, y_train, y_test):
2     regressor = LinearRegression()
3     regressor.fit(X_train, y_train)
4     y_pred = regressor.predict(X_test)
5
6     y_pred_int = []
7     for y in y_pred:
8         if y < 0:
9             y_pred_int.append(0)
10        else:
11            y_pred_int.append(round(y,0))
12
13    difference_array = np.subtract(np.array(y_pred_int), y_test)
14    squared_array = np.square(difference_array)
15    rmse = np.sqrt(np.sum(squared_array)/len(squared_array))
16
17    correct_count = 0
18    for i in range(len(y_pred)):
19        if (y_pred_int[i] == y_test[i]):
20            correct_count += 1
21    accuracy = correct_count/len(y_test)
22    return rmse, accuracy

```

```

1 def polynomial_regression(X_train, X_test, y_train, y_test, order):
2     poly = PolynomialFeatures(degree = order)
3     X_train_ = poly.fit_transform(X_train)
4     X_test_ = poly.fit_transform(X_test)
5
6     regressor2 = LinearRegression()
7     regressor2.fit(X_train_, y_train)
8     y_pred_ = regressor2.predict(X_test_)
9
10    y_pred_int_ = []
11    for y in y_pred_:
12        if y < 0:
13            y_pred_int_.append(0)
14        else:
15            y_pred_int_.append(round(y,0))
16
17    difference_array = np.subtract(np.array(y_pred_int_), y_test)
18    squared_array = np.square(difference_array)
19    rmse = np.sqrt(np.sum(squared_array)/len(squared_array))
20
21
22    correct_count = 0
23    for i in range(len(y_pred_)):
24        if (y_pred_int_[i] == y_test[i]):
25            correct_count += 1
26    accuracy = correct_count/len(y_test)
27    return rmse, accuracy

```

- Neural Network

The neural network model is implemented from the code for ‘Make Your Own Neural Network’. We can find the original code on

<https://github.com/makeyourownneuralnetwork/makeyourownneuralnetwork> (Rashid, 2016).

The original code is used for hand-written number recognition. We used the same principle of classification in a multi-layer neural network and implemented it for sales number classification.

5.2 Design Document and Flowchart

We did our project in Google Colab and Jupyter Notebook based on Python 3. Sklearn, keras and TensorFlow libraries were used to train our models.

The process of this project is as the following flowchart:

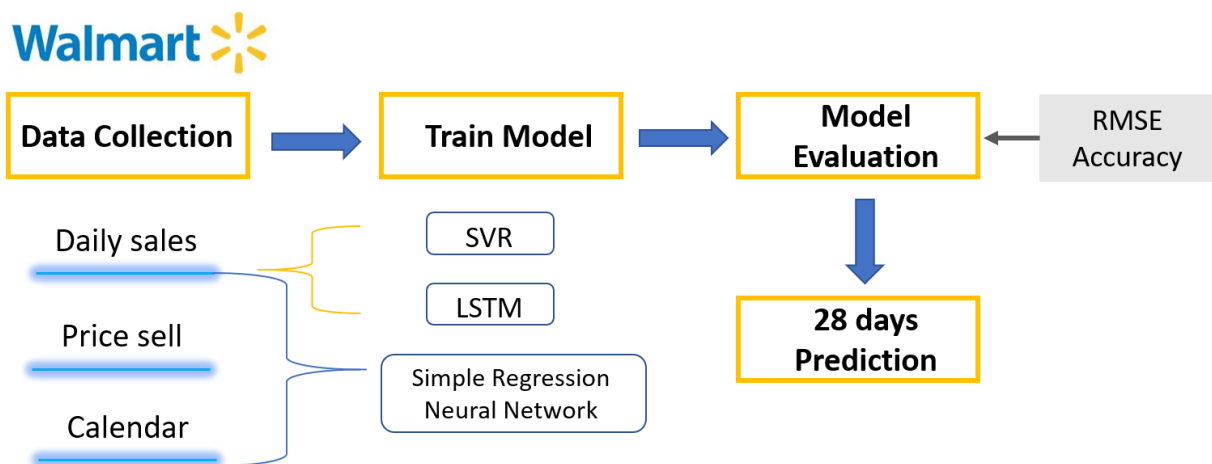


Figure 5-1 Flowchart

Data Analysis and Discussion

6.1 Output Generation

1. Testing RMSE

Firstly, we output the testing RMSE for each model in order to compare their prediction accuracy.

- SVR model

Table 6-1 Optimal Parameters and RMSE of SVR Model

ID	Optimal C	Optimal gamma	Testing RMSE
HOBBIES_2_029_CA_2	2	1	0.770
HOUSEHOLD_1_432_WI_3	2	1	0.823
FOODS_2_184_WI_3	0.1	1e-05	1.163
FOODS_1_185_TX_1	4	1	0.857
FOODS_2_386_TX_3	2	0.01	0.658
FOODS_2_263_CA_3	4	1	1.064
FOODS_3_275_WI_2	0.1	1e-05	1.548
HOBBIES_1_192_TX_2	4	0.01	0.315
FOODS_2_121_TX_1	10	0.0001	1.455
HOUSEHOLD_2_288_TX_3	1	1	0.595

- LSTM Model

Table 6-2 Training and Testing RMSE of LSTM Models with Epoch 50

ID	Epoch	Training RMSE	Testing RMSE
HOBBIES_2_029_CA_2	50	0.081	0.085
HOUSEHOLD_1_432_WI_3	50	0.079	0.070
FOODS_2_184_WI_3	50	0.147	0.091
FOODS_1_185_TX_1	50	0.058	0.041
FOODS_2_386_TX_3	50	0.118	0.129
FOODS_2_263_CA_3	50	0.160	0.117
FOODS_3_275_WI_2	50	0.115	0.156
HOBBIES_1_192_TX_2	50	0.102	0.106
FOODS_2_121_TX_1	50	0.082	0.080
HOUSEHOLD_2_288_TX_3	50	0.133	0.147

- Simple Regression and Neural Network Models

Table 6-3 Testing RMSE of Simple Regression and Neural Network Models

ID	Simple Regression Testing RMSE	Neural Network Testing RMSE
HOBBIES_2_029_CA_2	0.776	1.583
HOUSEHOLD_1_432_WI_3	0.932	1.563
FOODS_2_184_WI_3	1.081	1.698
FOODS_1_185_TX_1	1.175	1.738

FOODS_2_386_TX_3	1.067	1.659
FOODS_2_263_CA_3	1.105	1.710
FOODS_3_275_WI_2	1.172	1.845
HOBBIES_1_192_TX_2	1.069	1.759
FOODS_2_121_TX_1	1.108	1.794
HOUSEHOLD_2_288_TX_3	1.058	1.729

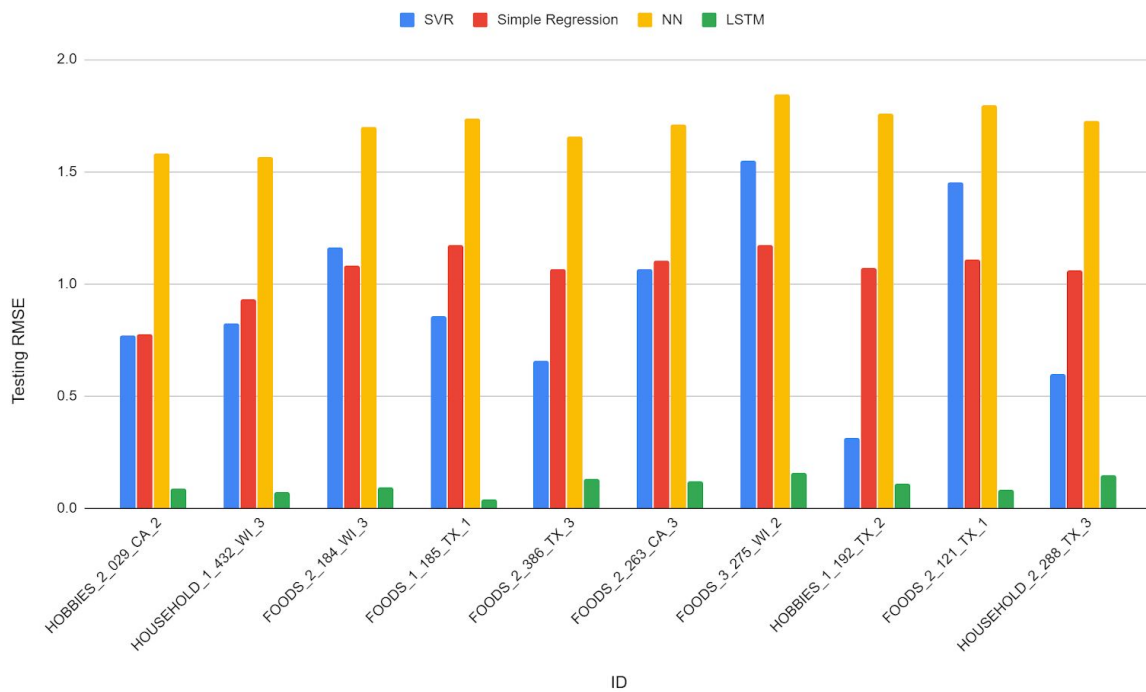


Figure 6-1 Testing RMSE of All of the Models

2. 28-day prediction

Our final 28-day sales prediction is aggregating all the outputs from our models, shown as Table 6-4. Because of the time limit of this project, we did not use ensemble

methods to take all the models into account. Instead, we calculated a simple weight based on the RMSE of each model on each product. The weight for each product is calculated by the inverse of its RMSE and we scale it to 0 -1 by comparing with other RMSE weights in other models. We sum all the products of the weight and the output values from our models to get an aggregated result. Since our LSTM model has relatively low RMSE, the result is very like the NN and LSTM model output.

Table 6-4 28 Days Prediction

ID	F 1	F 2	F 3	F 4	F 5	F 6	F 7	F 8	F 9	F 10	F 11	F 12	F 13	F 14	F 15	F 16	F 17	F 18	F 19	F 20	F 21	F 22	F 23	F 24	F 25	F 26	F 27	F 28
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.2 Output Analysis

1. Simple Regression Model

Surprisingly, the result of the simple regression model is not that bad. Data encoding should devote a lot to the accuracy of this model. We selected the linear regression model rather than other higher order polynomial regression models because the RMSE for the latter will always go very high because they tend to give extremely high output in certain cases. By comparing the RMSE and accuracy, as a result, we selected the linear regression model to keep the simplicity (Figure 6-2).

```

RMSE for Linear Regression is: 1.48583767745092
Accuracy of Linear Regression is: 0.47581484264830576
RMSE for polynomial Regression with order 2 is: 1259874992333.4854
Accuracy for polynomial Regression with order 2 is: 0.4671600748832761

```

Figure 6-2

From Figure 6-2 , by randomly selecting 100 products from the dataset, we have the average RMSE for those 100 samples as 1.486. If we only consider whether the prediction is the same as the test target, then the accuracy is 47.581 percent. Although the RMSE and accuracy may vary based on the randomly selected product, the RMSE is always around 1.5 and the accuracy is around 50%.

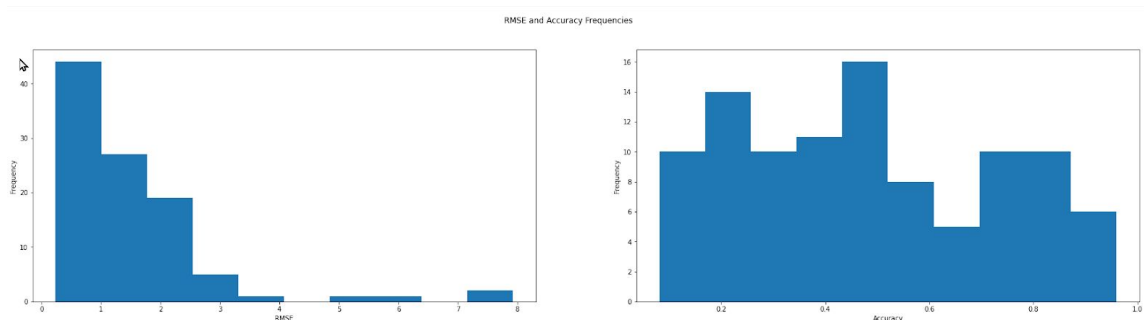


Figure 6-3

By looking at the RMSE (left) and accuracy (right) distribution (Figure 6-3) among the 100 samples, we can see that most of the RMSE shows exponential distribution and accuracy shows uniform distribution. We can see that linear regression does tend to minimize the RMSE. While comparing the prediction outputs with the correct sample targets, we find that the linear regression model will omit some large values because there are a lot more smaller values. Looking at the accuracy graph, we can find that the overall accuracy of this linear regression model is not good for all products.

2. Neural Network

The neural network model is another way using the same encoded feature data. From the test performance result (Figure 6-4), we see that the RMSE for the neural network model is high. However, the average accuracy for this model is a lot better and meets our expectations.

```
RMSE for Neural Network is: 8.010384649479144
Accuracy for Neural Network is: 0.6217278012982411
```

Figure 6-4

By looking at the distribution of RMSE and accuracy(Figure 6-5), we find out that the RMSE for the neural network model is very low for the majority. There is a super high RMSE that leads to the high average RMSE in our test case. We will explain the result further in session 6.4. The accuracy of the neural network model looks much better than the linear regression model, as the majority of accuracy tends to be at the higher end.

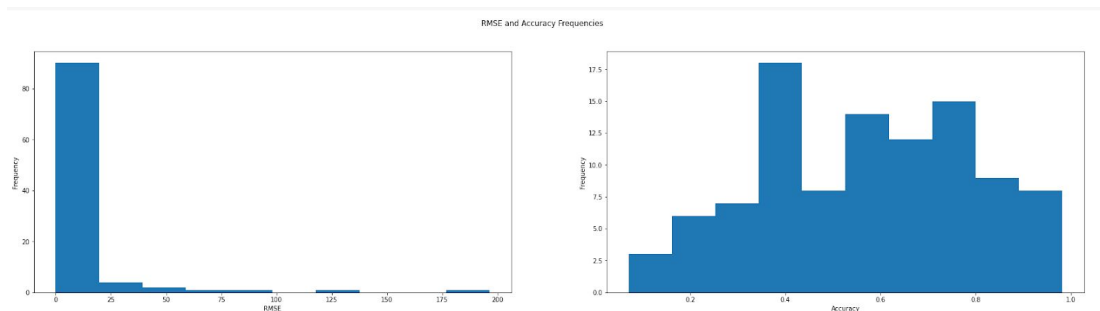


Figure 6-5

Only looking at the RMSE, the neural network model could be a bad option for this project. However, without looking at the magnitude of the error made on the predictions, the accuracy of this model is actually good for most of the products.

3. Support Vector Regression(SVR)

During the tuning of the SVR model, grid search was used to find the optimal parameters of the rbf kernel SVR: C and gamma. C is a regularization parameter. A smaller C indicates a simpler decision function, at the cost of training accuracy but could prevent overfitting. 'Gamma' defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. And the regression with lower gamma values will be more smoother, such as products FOODS_2_184_WI_3 and FOODS_2_121_TX_1 (Figure 6-*). After many attempts, we finally set the search range at [0.05, 0.1, 1, 2, 4, 8, 10] for C and [1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 5] for gamma. In grid search, the 5-fold cross validation was used and parameters with the highest scores will be chosen as the optimal parameters. The results of grid search for each product were shown in table 6-1.

The testing RMSE of the SVR model (Table 6-1) shows an unstable prediction capacity for different time serieses with the testing RMSE values ranging from 0.315 to 1.455 . We find that for majority of products whose sales data is stable and shows some seasonality, this model can well simulate the trend in dataset during training phases, such as product HOBBIES_2_029_CA_2 and HOUSEHOLD_1_432_WI_3 (Figure 6-6). However, for dataset which has greater variations, such as product FOODS_2_184_WI_3 and FOODS_2_121_TX_1 (Figure 6-6) and a lot of continuous 0s, such as product FOODS_2_121_TX_1, the model displayed a poor performance.

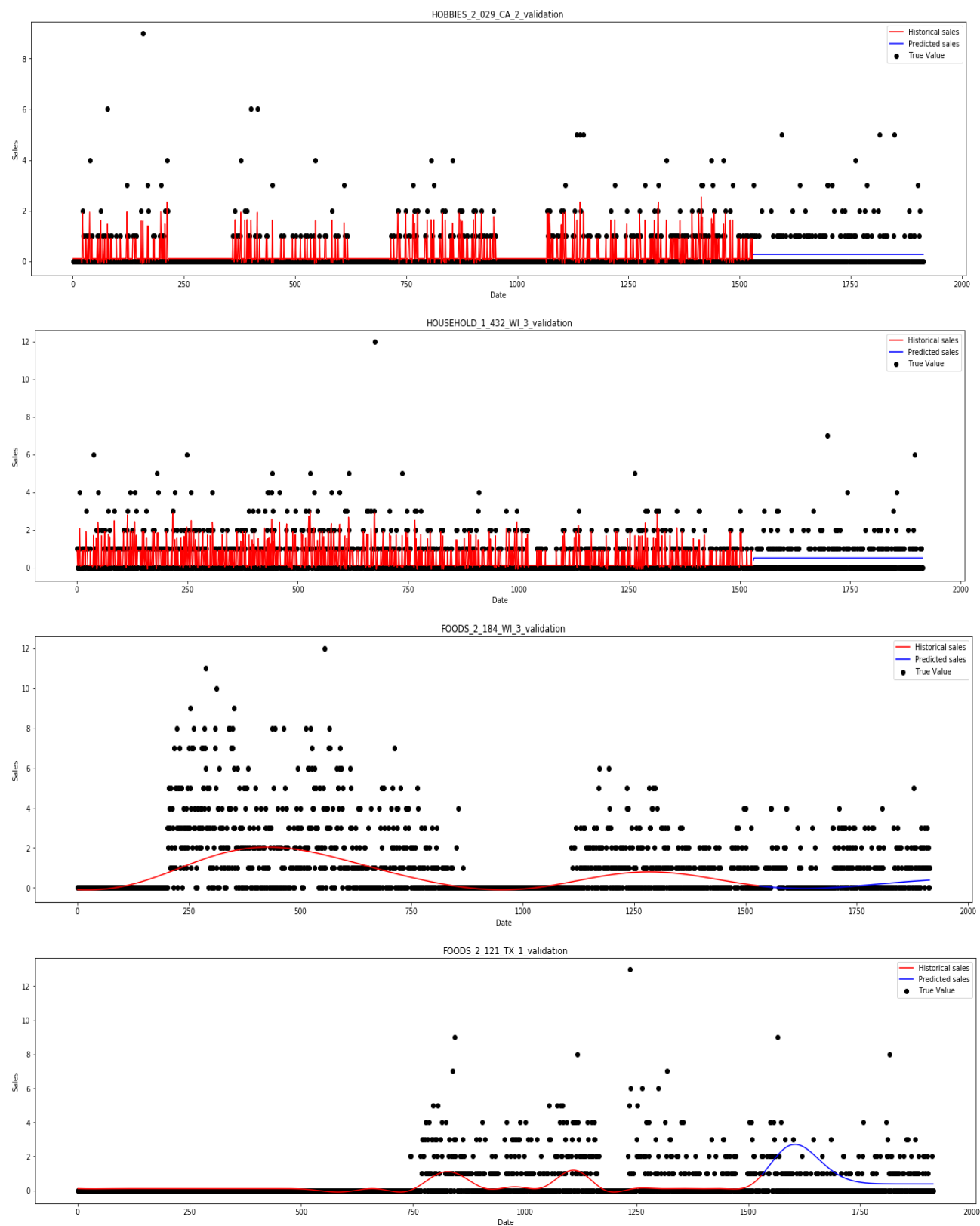


Figure 6-6 Historical Sales and Predicted Sales

In addition, one phenomena we can easily see in the prediction is that, for the majority of products, the prediction values for each day are very close, which is a blue line in Figure 6-6. This is because small sales values, such as 0 and 1, dominate our datasets. Thus the prediction results tend to be lower and flatter. Thus we will be reluctant to use this model for every-day sales forecasting.

4. Long Short Term Memory (LSTM)

For the process of training the LSTM model, there are few stages that are significant to reinforce. Starting from the beginning that the dataset has been splitted into training and testing dataset with 80% versus 20%. Then, they are rescaled based on the training data using the MinMaxScaler. With the use of the sliding windows on the daily sales of the product, a loop is generated based on the sequence length that characterize how long each sequence should be and goes to the next number of data. For the predictor class function, there are two layers: the LSTM layer which basically can be stacked of multiple layers and take the LSTM output into a linear layer; the output layer is within the linear layer. The forward method within the function takes the inputs, which are all of the sequence and pass out the final time stamp of the LSTM layer to the linear layer. Thus, the final prediction of the value would only be a single output. For the train model function, this phase invokes the backward propagation of the errors from the model and using the `model.eval` to prevent the final model from evoking itself using a dropout rate of 0.

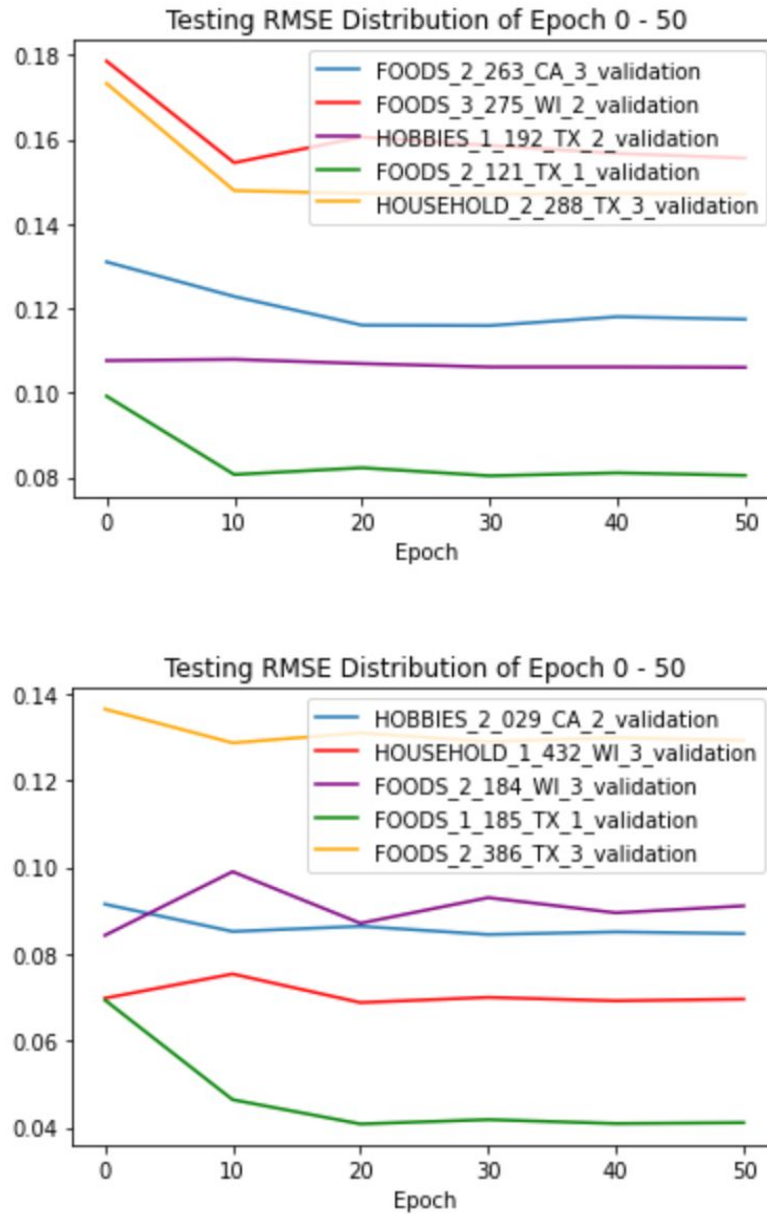


Figure 6-7 Testing RMSE Distribution of Different Level of Epoch

In Figure 6-7, the two charts display the outputs of RMSE from using different levels of epoch. In the train model function, the number of epochs is set to 60, therefore the levels of epoch are varied from 0 to 50. The higher the number of epochs, the deeper the LSTM model could get; since the number of epochs reflects the amount of time that

the dataset is passed forward and backward through the model. As displayed above, the line graphs illustrate an decrease of the RMSE when the epoch increases. Especially FOOD_1_185_TX_1_validation and FOOD_2_184_WI_3_validation, there are tremendous decreases of testing RMSE from epoch 10-20 and becomes static gradually. This concludes that the highest level of epoch might result with the lowest RMSE, therefore when comparing the RMSE results with different models in Figure 6-1, the testing RMSE from the epoch 50 is used instead of 0-40. We can see from the comparison that the RMSE of LSTM model is significantly lower than the other three model, which means it has the best prediction performance among them.

6.3 Compare output against hypothesis

In the previous hypothesis, we assumed that the 28 days prediction for each product should have more than 60 % accuracy on average or the RMSE around one. The evaluation results for these four models indicate that we have achieved this goal. In our project, the prediction accuracy of the Recurrent Neural Network model is 62.2%, and the RMSE values are less than 1.0 for majority of products and the RMSE values of LSTM models are even around 0.1, which is such a surprise for us. Thus, the results meet our expectations and the LSTM model is the optimal model among these four models for sales forecasting at Walmart. We tried to aggregate the result for the four models based on the RMSE output for each product. However, since the RMSE of the LSTM model is so low, the final result is mostly like the LSTM model output.

The results also show that the LSTM models have the ability to memorize the time series patterns and could be used to do the further prediction. Therefore, the LSTM is a relatively more suitable model for projects like this.

6.4 Abnormal case explanation

In session 6.2, we found that the extreme high values in the neural network prediction are causing higher average RMSE in average of all the samples. We went to check the actual RMSE values for the samples. We find the top three RMSE are 196.36, 119.45, and 96.79. By looking at the target values, we find out that there is one large sales in the training data for those products, resulting in more output nodes in the neural network (because we set the number of output nodes to be the highest number of sales in the training set). This offers the possibility in the testing stage that the prediction could be made very big while the actual target is low, or vice versa. This will result in super high RMSE when we square the error in that case. Therefore, I think RMSE may not be the best quantifier for testing the accuracy of the prediction. A simple accuracy factor could be a better quantifier for this model.

In addition, we find that the SVR model performed extremely poorly for three products, FOODS_2_184_WI_3, FOODS_3_275_WI_2 and FOODS_2_121_TX_1, with RMSE values even higher than simple regression model. So, we dug into the reasons behind this. First, we assumed that the continuous “0”s decrease the prediction accuracy, so we eliminated these “0”s and retrained models. The results showed that the model indeed improved after removing these continuous “0”s (Figure 6-8), with RMSE value for product FOODS_3_275_WI_2 changed from 1.548 to 1.396, and RMSE value for product FOODS_2_121_TX_1 changed from 1.455 to

0.975. Second, we find that the model does not have the capacity to deal with datasets with greater variations and less seasonality with FOODS_2_184_WI_3 as an example. The model still can not well predict the high sales in the testing set, which caused the higher RMSE on products with greater variations. So, just use the daily sales to predict the daily sales is not sufficient to get a satisfactory result, more variables should be taken into consideration, such as customer purchasing behavior, discount and holidays.

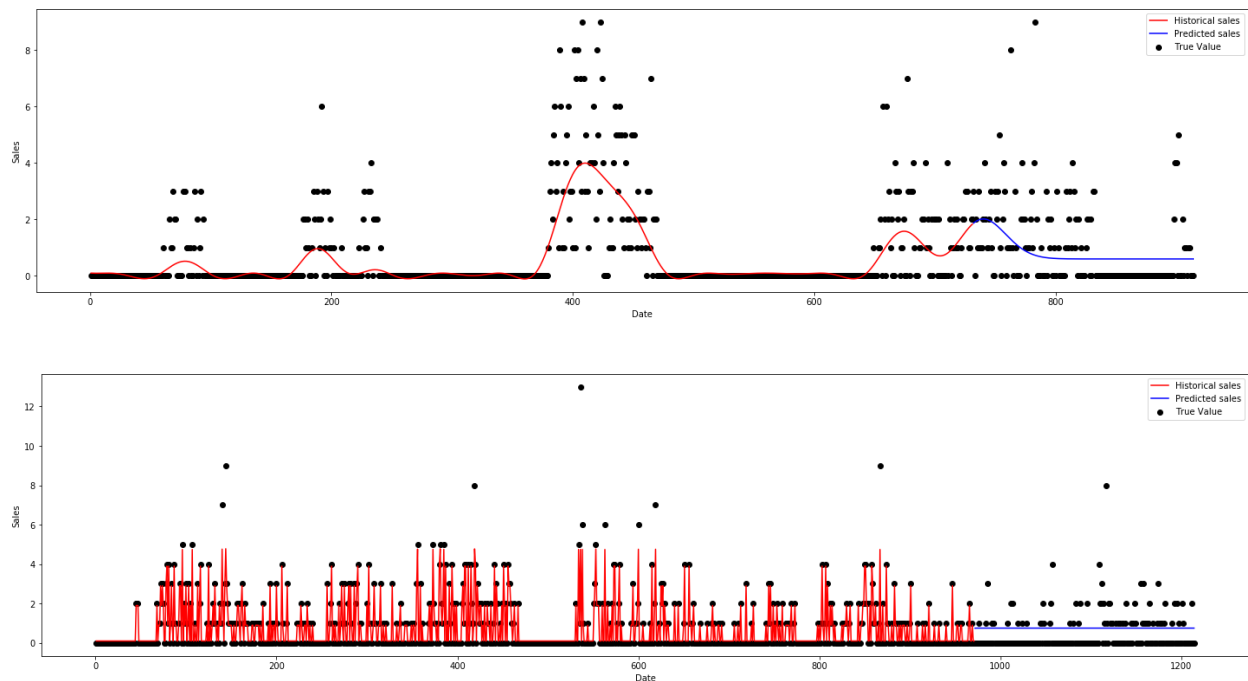


Figure 6-8 Retrined Model with Removing Continuous “0”s at the Beginning

Furthermore, even the LSTM model is the best model for forecasting the 28-days sales prediction, it still contains some irregularities. In Figure 6-9, the training loss and testing loss eventually will converge as illustrated, which reflects the problem with overfitting could possibly avoid and the datasets are adequate for making good estimates. However, looking at the time

series chart (Figure 6-10), the daily sales prediction of HOBBIES_2_029 that the model makes is definitely different from the actual daily sales. This is quite strange because LSTM generates the lowest RMSE, therefore the prediction and the actual daily sales shouldn't have such a significant difference. However, by going over the dataset, HOBBIES_2_029 has varied daily sales for the past 1530 days. Even this product contains a larger volume of the daily sales during certain days of the week, but the record shows that most of them were 0. Therefore, just using the historical daily sales record to forecast the daily sales is not sufficient to produce an accurate mode. In the future when handling a similar type of datasets, other factors should be taken into consideration when running the LSTM model to prevent misleading predictions.

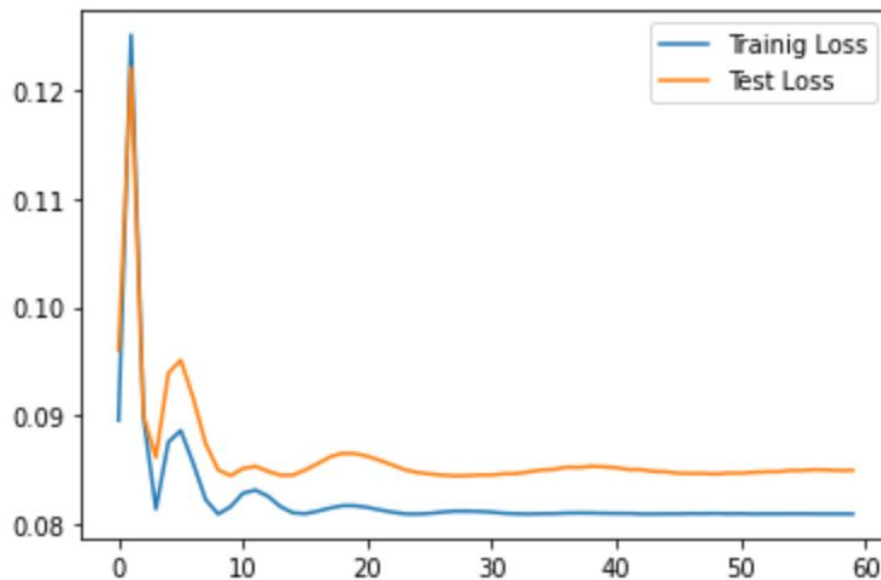


Figure 6-9 Loss Function of Product HOBBIES_2_029

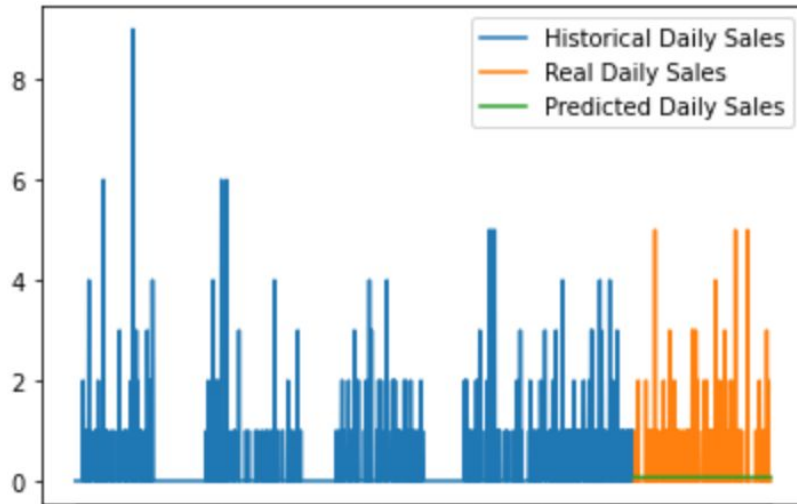


Figure 6-10 Sales Distributions of Product HOBBIES_2_029

6.5 Static regression

We chose the linear regression model as one of our output models. We encoded the features into 28 distinct columns and generated the testing performance and result. The performance of this model is not that bad. As we can see from Figure 6-2, the RMSE is around 1.5. However, compared with RNN and LSTM, the error of this model seems still too high. Our model result is not shown in the report because for each product we have a curve to fit and the coefficient in front of each variable will be different. Also, the new encoded features do not have specific meaning. Showing the fitted curve will not be helpful in modeling for this project.

From the result of the 100 randomly selected products from our project (Figure 6-3), we see that the regression model works well in some of the products but badly on the others. This is saying that this model may not be suitable for predicting all the products in department stores. The performance of such a model highly depends on the stability of the product. Products in department stores, however, do not always show such good stability.

The possible ways to improve the regression model is to collect more data on the product feature so that the model can take considerations of the quality of the products. This could reduce the effect of uncertainty that caused by the product to our regression model.

6.6 Discussion

As our goal for this project is to find the optimal method for daily sales forecasting, we trained four widely used models with limited variables. Comparing the performance of these four techniques, each model has shown their pros and cons.

For simple regression models, as a traditional method, although it does not provide a satisfactory accuracy, it's the most time-saving among them and it also considers other variables during model training, such as price, holidays and promotion activities. The SVR model was easy to implement and could overcome the overfitting effect, but it showed an unstable prediction capacity in this project. The SVR can not correctly predict large values when the dataset was dominated with small values. The performance of Neural Network was similar to SVR, with the RMSE around 1.0 for the majority of products. And it was able to simulate the trend of the Walmart daily sales. The LSTM model had the best prediction accuracy among all of these four methods, with the average RMSE around 0.1. It is the most time-consuming, and complex one since each epoch that the training model runs more than thirty minutes. However, considering its highest accuracy and the depth that the model gets into the dataset, we still recommend it as the optimal model for daily sales forecasting or even time-series forecasting. Of course, we will still spend time in modifying our models, as well as looking for more time-saving approaches to make our prediction more accurate.

Conclusion and Recommendation

7.1 Summary and Conclusions

For this Walmart sales prediction, we used four different Machine Learning techniques: Support Vector Regression, simple regression, Neural Network, and Long Short Term Memory. We used different feature selection of various models and output RMSE and 28-days predictions for each product in the dataset. Comparing the model performance of the four, LSTM model has the highest accuracy so that our final output is mostly based on that model.

Our result meets our expectation that the RMSE for LSTM is about 0.1. From the Kaggle, of other user's submission, we see that most of the overall weighted RMSE is about 0.4. Therefore, we expect our RMSE for all the products will be around that number as well.

We are going to compare our output with the actual data in June when Kaggle posts the 28-day prediction correct targets. After that, we can see how well our model is for predicting the sales value.

7.2 Recommendation for Future Studies

It seems that based on the limited data, LSTM is the best model to make predictions. For better performance of the other models, more data on the products and customer behaviors should be collected and analyzed. Having more features, more precise models can be built and as accompaniments to ensemble with the LSTM model for better aggregated results.

Bibliography

- Chen, I. F., & Lu, C. J. (2017). Sales forecasting by combining clustering and machine-learning techniques for computer retailing. *Neural Computing and Applications*, 28(9), 2633-2647.
- Chen, T., Yin, H., Chen, H., Wu, L., Wang, H., Zhou, X., & Li, X.. (2018). Tada: trend alignment with dual-attention multi-task recurrent neural networks for sales prediction. *IEEE International Conference on Data Mining (ICDM)*, Singapore, 2018, 49-58.
- Chen, T., Yin, H., Chen, H. et al. (2019). Online sales prediction via trend alignment-based multitask recurrent neural networks. *Knowledge and Information Systems*. 4(13), 1-29.
- Di Pillo, G. , Latorre, V. , Lucidi, S. , & Procacci, E. . (2016). An application of support vector machines to sales forecasting under promotions. *4OR*, 14(3), 309-325.
- Doudou Dong, Rui He, and Guixi Xiong. "Foreign Commodity Sales Forecast Based on Model Fusion". 2019. *In Proceedings of the 2019 4th International Conference on Big Data and Computing(ICBDC 2019)*. Association for Computing Machinery, New York, NY, USA, 185–188. DOI:<https://doi.org/10.1145/3335484.3335507>
- F. A. Gers and J. Schmidhuber. "Recurrent nets that time and count," *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000*. Neural Computing: New Challenges and Perspectives for the New Millennium, Como, Italy, 2000, pp. 189-194 vol.3.

- Li, N., Cai, Y., & Bi, D.. (2019). The empirical analysis of convenience store sales forecasts based on multidimensional data in emerging markets. ICBIM '19: Proceedings of the 3rd International Conference on Business and Information Management. 32-36.
- Moon, S., S. Bae and S. Kim, "Predicting the Near-Weekend Ticket Sales Using Web-Based External Factors and Box-Office Data," 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), Warsaw, 2014, pp. 312-318.
- Rashid, T. (2016). *Make Your Own Neural Network Tariq Rashid*. Erscheinungsort nicht ermittelbar: CreateSpace Independent Publishing Platform.
- Valkov, Venelin. "LSTM Time Series Prediction Tutorial Using PyTorch in Python | Coronavirus Daily Cases Forecasting." *LSTM Time Series Prediction Tutorial Using PyTorch in Python | Coronavirus Daily Cases Forecasting*, YouTube, 3 Mar. 2020, youtu.be/8A6TEjG2DNw.
- Wenxiang Dong, Qingming Li, and H. Vicky Zhao. "Statistical and Machine Learning-based E-commerce Sales Forecasting," 2019. *In Proceedings of the 4th International Conference on Crowd Science and Engineering (ICCSE 19)*. Association for Computing Machinery, New York, NY, USA, 110–117.
DOI:<https://doi.org/10.1145/3371238.3371256>
- You, L., Kou J., & Wang S.. (2019). Online Retail Sales Prediction with Integrated Framework of K-mean and Neural Network. In Proceedings of the 2019 10th International Conference on E-business, Management and Economics (ICEME 2019). Association for Computing Machinery, New York, NY, USA, 115–118.