Department Store Sales Forecasting

CS 487/519 - Applied Machine Learning, PS4

Robert Selje
Klipsh School of Electrical and Computer Engineering
New Mexico State University
Las Cruces, NM
rseljeii@ nmsu.edu

Brock Middleton
Computer Science
New Mexico State University
Las Cruces, NM
bromid@nmsu.edu

Jason Joe
Computer Science
New Mexico State University
Las Cruces, NM
jaysonrj@nmsu.edu

ABSTRACT

Businesses rely on forecasting sales to make effective business decisions while mitigating risk. Sales forecasting allows businesses to predict future growth and manage their cash flow to achieve sales targets. If a business can accurately model future sales, it can purchase inventory, make practical staffing decisions, and allocate resources. We look to apply our knowledge and skills from the topic discussed in class to find a model that accurately predicts weekly sales given data obtained from major department stores. We will implement various machine learning algorithms and analyze their results.

1 Motivations

Forecasting sales is a valuable undertaking that allows companies to make effective business decisions while mitigating risk. Businesses can purchase inventory, make practical staffing decisions, and allocate resources by accurately modeling future sales. Sales forecasting allows businesses to predict future growth and manage their cash flow to achieve sales targets for the month, quarter, or year and build revenue. Sales forecasts allow you to spot potential issues while there's still time to avoid or mitigate them. Forecasting sales is a foundational building block for other more challenging problems, including stock and currency predictions, market analysis, and political forecasting. In this project, we look to predict future sales for Walmart using data from Walmart stores all across the US, as well as data from Rossmann stores in Germany and several other European countries.

2 Related Works

An individual by the name of Aslan Ahmedov had done a project he called ‘Walmart Sales Forecasting’ where he built a machine learning model to predict the weekly sales by department for Walmart stores. They pulled open source data that included features on: store number, calendar week, temperature, fuel price of local region, markdowns, consumer price index, unemployment rate, and holiday week.

3 Problem definition

Predicting weekly sales for a given department store such as: Walmart or Rossmann. Potential features to make the prediction are: the calendar date, holiday week, temperature, consumer price index, national unemployment rate, number of employees the store has, fuel price of the region, and national inflation. Holiday week is a binary case where the week contains a U.S. holiday or not. And temperature refers to average temperature for the given day. All sales will be reported in the American Dollar. Datasets that use other currencies will be converted with the appropriate exchange rate at the given point in time.

4 Specific Analysis Tasks

Our primary task is to predict the weekly department store sales with given datasets using supervised learning methods. We will compare the analysis of regression Machine Learning models: Linear Regression and RandomForestRegressor. The goal is to accurately predict the weekly sales of a given store. Furthermore, we will also use this data on a ML model of our own design, and compare the

performance of said model with our previous results. As we are pulling datasets from different stores we will compare how our models perform depending on the dataset used, for example how our model performs with Walmart sales predicting and with Rossmann sales predicting.

5 Datasets

Our first dataset is pulled from a competition project hosted on Kaggle. We will build a prediction model for Walmart sales. This dataset contains features on: the weekly sales, temperature, fuel price of local region, promotions, and whether the week contained a holiday. This data is from Walmart stores based in the United States. The sales are reported in the American Dollar. This dataset and its analysis code are viewable in the "data_explr_walmart.ipynb" file.

Our second dataset is pulled from a competition about predicting sales for a European company: Rossmann. This dataset provides us with weekly sales, number of customers, holiday, and promotion days. Furthermore this dataset also provides distance between stores, this is a feature that is not provided by our first dataset. Rossmann is a European company with stores across 9 countries. The sales are reported in Euros. We will need to convert these sales into the American Dollar with the appropriate exchange rate at the given date to properly compare this dataset with our first dataset. This dataset and its analysis code are viewable in the "data_explr_rossman.ipynb" file.

6 Initial Solution

Sales forecasting can be described as using historical data over time to predict future sales. A neural network that works perfectly with such data sets is a Recurrent Neural Network (RNN), which utilizes historical data in a time series to predict future time series. An RNN employs Long Short-Term Memory (LSTM) which maintains only relevant information and passes on the previous hidden state to the following sequence. A model is trained by grouping a series of weeks (four weeks, five weeks, eight weeks, ten weeks, twelve weeks, etc.) and passing the data through multiple LSTM layers. RNN is a supervised learning algorithm that takes in the data series and the series of the true sales values. We can then make a prediction of the next week's sales based on the previous weeks' data.

We would also like to consider the Convolutional Neural Network (CNN) in addition to the RNN. CNN is more of an instantaneous prediction than a time series; only one week's data will be used at a time. Unlike RNN, CNN employs a network of perceptrons to learn a model. A CNN is trained through supervised learning by taking each week's data and the true sales value. The learning algorithm uses backpropagation to update the weights. A trained model will be able to take in data from a particular week and predict what the sales should be for the week.

In addition, we would like to analyze how different hyperparameters affect the learning time and accuracy of the two proposed models. The code for the initial solution is viewable in the "Class Project Networks.ipynb" file.

7 Linear Regression

In the accompanying notebooks, 'linearreg_walmart' and 'lienarreg_rossman'. Two Linear Regression models were implemented and tested, along with two Ridge Regression models.

Both datasets (Walmart and Rossmann) were partitioned with a split of '0.3', resulting in 70% of the instances being used as training data and 30% as testing data. Comparisons are made between both standardized and non-standardized data, but there was no observed difference in the performance of the linear regression model. In future iterations, different normalization techniques will be tested.

Several observations were made amongst features, and some were adjusted before model fitting. Details on any adjustments made to the data can be observed in both respective notebooks.

Linear Regression was able to train on the Walmart data in about 0.2 seconds on average. This is with about 300 thousand of the initial 421 thousand instances. This model provided a R2 score of 0.0851 for training, and 0.0860 for testing, as well as a mean-squared-error of 470713421 for training, and 474215398 for testing. In future iterations, normalization techniques will be applied to the data for any observable changes in performance. Ridge Regression was able to cut down on the execution time to about 0.05 seconds for the same data. The metrics of Ridge Regression are comparable to that of Linear Regression, with an R2 score of 0.085 for training, and 0.086 for testing, as well as a mean-squared-error of 470713465 for training, and 474214913 for testing.

Linear Regression was able to train on the Rossman data in about 0.15 seconds on average, slightly faster

than the Walmart data. This is with almost 600 thousand instances. This model provided an R2 score of 0.765 for training, and 0.767 for testing, as well as a mean-squared-error of 2264671 for training, and 2262029 for testing. Ridge Regression was not able to cut down on the execution time in this case and was often comparable to that of Linear Regression. Ridge provided an R2 score of 0.764 for training, and 0.766 for testing, which is also comparable to the performance of Linear Regression.

8 Random Forest Regressor

Two Random Forest Regressor models were made with the following parameters: n_estimators = 1000, criterion = 'squared_error', random_state = 1, and n_jobs = 10. One model was trained with the walmart dataset, and the second was trained with the rossmann dataset. Both datasets were partitioned to be 70% training and 30% testing data. The model was trained and measured using both standardized data values and the original data values.

Using the original rossmann data, the model had a mean squared error of: 1879525.318 for the training and 2053968.324 for the testing. When using the standardized rossmann data, the model had a mean squared error of: 0.127 for the training and 0.861 for the testing. The R^2 value for training is: 0.873 and the testing R^2 value is 0.861. The recorded fitting time for the original data was: 335.184430 seconds and the fitting time for the standardized data was: 327.803883 seconds.

Using the original Walmart data, the model had a mean squared error of: 50512141.196 for the training and 49053600.029 for the testing. When using the standardized walmart data, the model had a mean squared error of: 0.098 for the training and 0.095 for the testing. The R^2 value for training is: 0.902 and the testing R^2 value is 0.905. The recorded fitting time for the original data was: 153.596803 seconds and the fitting time for the standardized data was: 170.678420 seconds.

Although the mean squared error for both the original rossmann and walmart datasets are uncomfortably large, both models received good performing R^2 values. With rossmann getting a 0.861 on testing, only 0.012 less than its training R^2. And the walmart got a 0.905 on testing, which is 0.003 higher than its training R^2. Code for the Random Forest Regressor are viewable in the "RFR_rossmann.py" and "RFR_walmart.py" files.

9 Feed-Forward Neural Network

The model used to train the network consisted of an input layer, two hidden layers, and an output. The input layer consisted of twenty, one for each row in the data. The two hidden layers contained 1024 perceptrons for each. We trained the neural network with Adam's stochastic gradient descent method, with a learning rate of 1e-7. The learning rate is critical as it determines the step size taken toward the global/local minimum. A small learning rate takes smaller step sizes but increases the risk of falling into a local minimum. A more significant learning rate takes larger step sizes but increases the risk of overshooting the global minimum. The neural network also utilizes the tanh activation function for each perceptron, which outputs values between negative one to one. We choose the tanh activation function as weekly sales can be a negative (loss) or a positive value (profit). Since we wish to minimize the difference between the actual and predicted values, we train the network to minimize the mean squared logarithmic error cost function. Another important hyperparameter is kernel regularization which allows the model to converge optimally by applying penalties on the layer's bais, kernel, and output parameters.

The results show that the neural network can train a model to predict weekly sales with relatively small errors. Figure 1 shows the loss per epoch for the network training. We can see from the figure that the mean squared logarithmic error does decrease with each epoch, which is a promising sign to allow us to train for much more epochs. Due to the computational load on the local machine from the vast dataset, we could only train the neural network for ten epochs. We are running code on the NMSU Discovery HPC for more epochs since it can handle a much larger computational load.

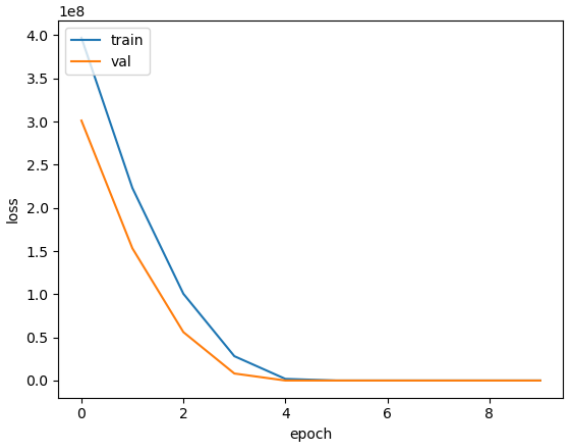

Figure 1: Loss per epoch for the Feed-Forward Neural Network for 10 epochs.

We look to improve the baseline model to decrease the mean square error. The next step is to train the feed-forward neural network for 500 epochs, which will be used as the baseline model. Afterward, we want to change various hyperparameters and perform different dimension-reduction algorithms to compare results to the baseline model.

10 Recurrent Neural Network

The data processing for the RNN is different from other supervised machine learning models. In other supervised machine learning models, a sample is placed into the algorithm, and the output is compared to the actual value. RNN, on the other hand, requires a series of samples to produce an array of historical data. Therefore, data preprocessing requires the programmer to string together a group of previous samples. When the data is placed into the network, each LSTM will have a single sample from the historical data. The results of an LSTM layer will be fed into the next LSTM layer along with the next sample. The process keeps going until the last LSTM.

RNN utilizes a series of LTSMs instead of layers of perceptrons to maintain relevant information and find patterns in the data. The first layer of our designed network is a series of 128 LSTMs. Each LSTM utilizes the tanh activation function to allow for negative (loss) and positive (profit) weekly sales predictions. The output of the LSTM layer is then fed into a dropout layer that will drop 10% of the input units. The dropout layer is a helpful regularization technique used to help reduce overfitting. We then add another set of 128 LSTM with the tanh activation function followed by a dense layer of five perceptrons. The output of the dense layer is then given to the output layer, which predicts the continuous weekly sales variable. We trained the recurrent neural network with Adam's stochastic gradient descent method, with a learning rate of 1e-7. It is also important to note that the network minimizes the loss function with the mean squared logarithmic error function.

Figure 2 shows the loss per epoch for training the above model to predict weekly sales. The results do not look promising as the figure shows relatively high errors throughout the training. However, RNN was designed to handle sequential data. The RNN is much more computationally heavy than the feed-forward neural network as it needs to build and maintain every possible time series in the dataset. We are running code on the NMSU Discovery HPC for more epochs since it can handle a much larger computational load. We will need to continue tuning the model and see if we can improve the results.

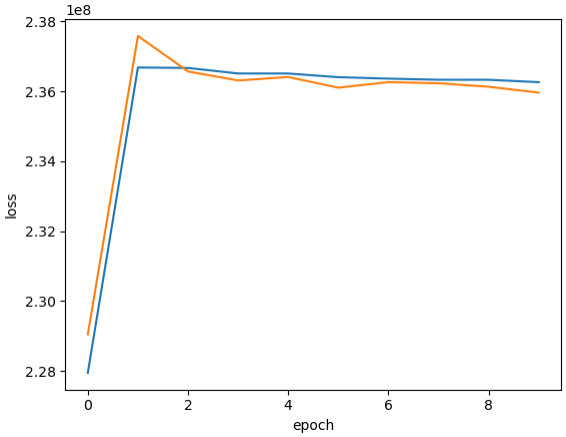

Figure 2: Loss per epoch for the Recurrent Neural Network for 10 epochs.

REFERENCES

[1] Ahemdov, A. (2022, April 29). *Walmart sales forecasting*. Kaggle. Retrieved April 2, 2023, from https://www.kaggle.com/code/aslanahmedov/walmart-sales-forecasting

[2] *Rossmann store sales*. Kaggle. (2016). Retrieved April 2, 2023, from https://www.kaggle.com/competitions/rossmann-store-sales/data

[3] *Walmart recruiting - store sales forecasting*. Kaggle. (2014). Retrieved April 2, 2023, from https://www.kaggle.com/competitions/walmart-recruiting-store-sales-forecasting/data.