Walmart Sales Predictions

PSL Fall 2020 - Project 2

Project Members

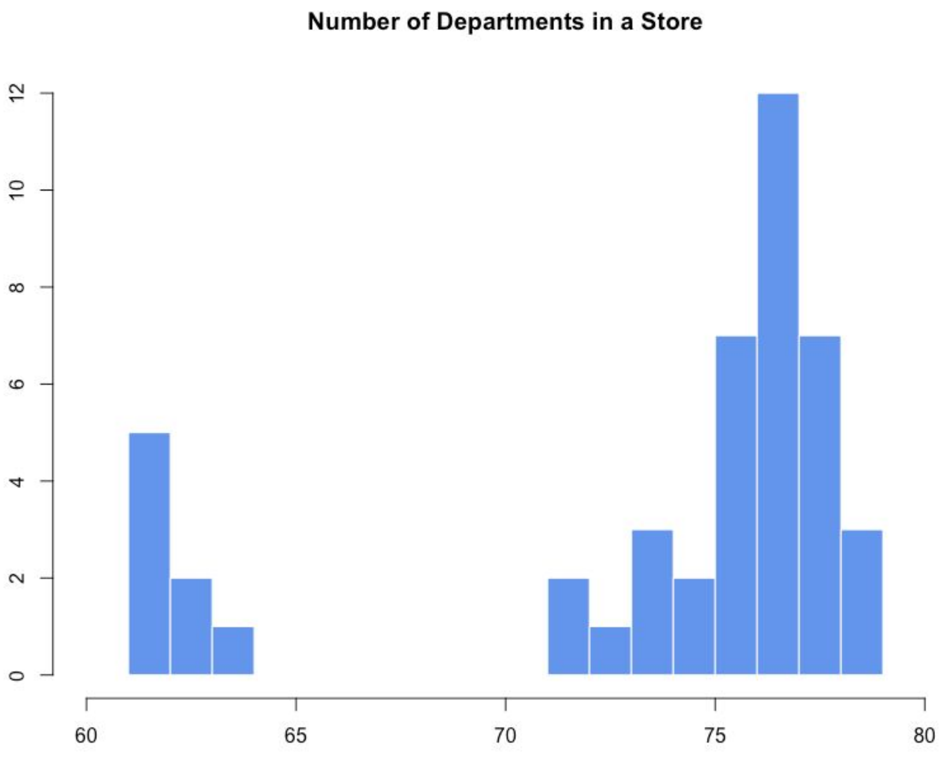
- Derek Chapman (derek4)
- Zeed Jarrah (zjarrah2)

Overview

Our task for this project was to predict future weekly sales of Walmart stores/departments based on previous sales data. We split the training data up into a variety of files that are then read in sequential order based on date. After reading in the training data we predict the first two months of sales data for each store/dept that reported information during that time period. We then read in the true sales for those two months for each store/dept and add that to our training data for the following two month block of data. This is repeated over the course of 10 sections ending with a prediction of the last two months using all of the previous months added onto the original training data.

Dataset / Interesting Observations

Project 2 (Walmart Sales) in some ways was the complete opposite of our first project (Ames Housing). The Walmart sales data has almost no predictors since we would consider the first few columns of Store and Dept to be organizational information. We have in effect only the weekly sales and whether that week was a holiday week or not. There are a total of 45 stores and 81 possible departments. However that is where the firm answers end. Within a given Store/Dept combination we often have a sparsity of data. Not all stores have every department, but in



addition NO store has all of the departments (see figure above). The most number of departments that any store has is 79 while some stores have as few as 61. We also see a huge disparity in the number of records for each store/dept combination. About 85% of the stores/depts have what we would consider 'full records' in the training data. That is they reported weekly sales for every week (56) that is included in the 13 months of starting training data. For many of the stores/depts they have only one record for the *entire* 13 months of data. And to make matters more confusing there are a surprisingly large number of negative weekly sales! (457 in the training data)

Our Process

We developed our solution based upon the provided piazza code ("What we have tried III" and "Project 2: The forecast package"). Our main process closely follows the provided code. We loop through each department and gather all of the data for all stores that have that department. We then create a joined list of only stores that are in both the training data and in the current testing data. If a store is not in both testing and training data then we are unable to make a sane prediction for them, or we don't need to make a prediction for it. We then create a training and testing data frame that is in the form of *dates x store* so that a single column will be all available information for a single store. The testing data frame is really our 'prediction' data frame since it will hold our predictions for each week for each store. After this we create a timeseries object from the data for a single store/dept and use `tslm` to create a model, then use `forecast` to make our predictions with a set 'horizon' or number of weeks out that we would like to predict on. Since department sales may follow both long and short term trends we include both `trend` and `season` in our `tslm` model.

After this we 'shift' the data by moving part of the sales volume to the week after the current week. This has two effects: it accounts for the shift in the holiday week between 2010 and 2011, it also seems to smooth out the sales volumes. We chose to shift 1/7th of the sales data as this seemed to get the best results for us. After all of this we return our predictions data frame into a format that is the same as our original data where each row is in the format of Dept, Date, Store, and Weekly_Pred.

We made a few modifications along the way. In the current prediction period if there is only one store that has the given department we skip making predictions. Not only do the data types get scrambled but these departments in a single store tend to be non-consequential. The majority of them have either very low sales volume (a few hundred dollars) or negative sales during the training data. We also call `shift` on every fold instead of just the 5th fold that has the Christmas shopping season. This improved several of the folds while only very minimally decreasing a few of the folds. Our guess is that the 'smoothing' of the data helped prevent inappropriately wild guesses based on limited data that may have had some isolated and independent hiccups that are not indicative of future sales.

We also tried the 'naive' approach, the 'seasonal naive' approach, as well as combining predictions in an average. Although they each had improved results none of them approached the time series lm method. We also tried 'brute forcing' a better score by looking at which departments varied the most from our general predictions and inflating/deflating the predictions for all stores with that department. This worked quite well at first since adjusting the largest departments to better match reality had large initial gains. Some of the departments were consistently off in a single direction which meant making a slight adjustment improved the predictions for every one of those departments. Combined with the fact that some of the individual departments had upwards of a million dollars in weekly sales meant a large difference. But ultimately this trick fell victim to the law of diminishing returns, and the amount we could lower our score with each adjustment quickly fell.

Scoring

For this project we are again scored based primarily on how close our predictions are to the actual sales data. However, one major difference is that holidays are weighted differently. Holiday week predictions are weighted five times more than a regular week. Meaning that a sizable difference between the actual and predicted results during a holiday week will be a huge hit to the final score. We are given a weighted average score from each of the sections and then also look at the overall prediction quality based on the mean of all 10.

Our scores:

Fold	1	2	3	4	5	6	7	8	9	10
WMAE	2039.46	1440.08	1432.85	1596.2	2029.24	1674.19	1717.77	1421.41	1424.63	1447.03

Our overall score is:
1622.286

The processing time for our code is 286.596 seconds or about 4:46 on a Macbook Pro with 8gbs of ram and a 2.7 GHz Intel i5 processor. This time includes predicting on all 10 sections but does not include loading/separating/reading files which is relatively quick.

This is a significant improvement in both score and running time over our original solution. That solution had a top overall WMAE of around 1640 and took over 12 minutes to run (740 seconds).