

Project 2 Report: Walmart Stores Forecasting

CS 598 Practical Statistical Learning

Graham D Chester, 29-Oct-2018

1 Method

In summary, the overall approach to this project was to firstly perform an Exploratory Data Analysis (EDA) to obtain a good understanding of the available data, secondly to review available materials from this course and Kaggle to short-list potential algorithms, thirdly to apply these algorithms and chose the best performers, and finally to tune the best algorithms to obtain the best possible results.

Since this project was started well before the TA-supplied code template was made available, it takes a different approach by using a hand-coded algorithm as described below, rather than using R forecast library.

The technology used for this project was R-studio with the 'tidyverse' (for data manipulation) and 'lubridate' (for date manipulation) libraries. The modeling was run on an iMac quad-core i7 4.2GHz, 40GB RAM, and the runtime of the R code is 23 seconds. The metric used for this report is Weighted Mean Average Error (WMAE) as required.

1.1 Data Exploration

An Exploratory Data Analysis (EDA) was performed, investigating the size, shape, patterns, and quality of the Walmart Sales data supplied. Whilst many detailed charts were evaluated as part of the EDA, they were too large and detailed to include in this report. However the (somewhat dense) summary charts in Figure 1 reveals a number of key insights that guided the development of the algorithms below.

Firstly we can see that the Sales by Store chart strongly indicates a peak period late in the year, and also indicates that not all stores are subject to the same peaks. The dashed lines show varying magnitude and direction trends by Store, and seasonality by Store can (just barely) be observed, though this was much clearer on the individual charts.

Secondly we can see that the average sales by department varies very significantly with a number of departments having only a tiny effect, and a few departments having a large effect. The dashed trend lines vary by department in a similar manner to that by Store i.e. showing both trend and seasonality, but further EDA showed that a number of departments were largely static, declining or had insignificant sales.



Figure 1

From the EDA, there was no obviously invalid data, and missing data was noted for reference when testing the various algorithms as described below. In the end, the missing data had no appreciable negative impact so no attempt was made to interpolate or otherwise resolve. The final data preparation and preprocessing therefore ended up being minimal and only involved creating new fields from Date (Year and Week) to aid calculations.

1.2 Algorithms

In order to identify potential algorithms, the course materials, the recommended Forecasting Principles and Practice book and Kaggle forums were reviewed to gather ideas. Initially a conventional approach (as suggested in the sample code provided) was taken to loop through stores and departments applying a seasonal naïve algorithm with trends, however the initial results were mediocre and the runtime did not allow for much experimentation and adjustments so this approach was put aside.

A promising approach found on the Kaggle forums (from the 3rd place winner, credited in the Acknowledgements below) was to predict the test week by blending two training weeks from a year ago. These two weeks were aligned (by day) with the current test week by calculating fractions of each week (e.g. taking 6/7 of one week and 1/7 of the other week). However this method does not work for the Thanksgiving and following week, so these weeks were excluded from the blending process but were aligned by shifting the week number instead. Despite this data manipulation, the Thanksgiving fold (5) was still the second-worst performing. The results for this blended seasonal naïve approach were promising, and the runtime was only 20 seconds for all folds.

This approach was enhanced further by applying trends to each store's sales, calculated using linear models (using R 'lm' with a quadratic term to improve accuracy). The results of this are shown as Model One. Then a trend was calculated for each department using a linear model by department (again with a quadratic term to improve accuracy slightly), and this is shown as Model Two. And finally Store's and Department's with a low-trend and high-variances were identified as introducing 'noise' so were skipped in the trending calculation. A further enhancement that improved performance slightly was to assess other holiday weeks, and it was found that the week following Thanksgiving and two weeks around Easter also had a peak in sales that justified exclusion from the week blending process described above. These results are shown as Model Three.

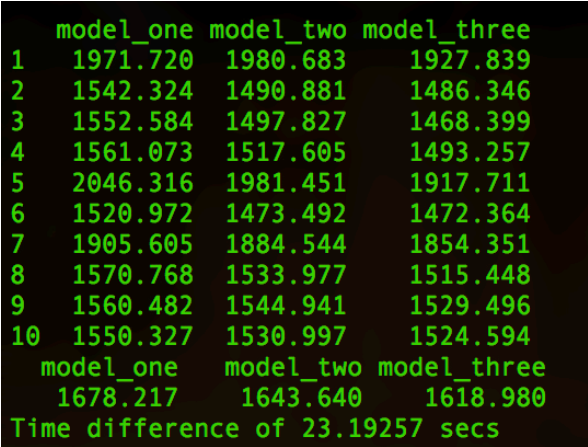
Other potential enhancements such as rounding predictions of less than zero to zero, or establishing a minimum level of sales prediction were investigated, but made very little or no improvement, so were discarded.

Results

The results for each model for each fold and the average, are shown in Figure 2 below.

- Model One: blended weeks seasonal forecast with Store trends, average score 1678
- Model Two: blended weeks seasonal forecast with Store and Dept trends, average score 1644
- Model Three: blended weeks seasonal forecast with tuned Store/Dept trends, average score 1619

Model Performance Summary Table, by fold



	model_one	model_two	model_three
1	1971.720	1980.683	1927.839
2	1542.324	1490.881	1486.346
3	1552.584	1497.827	1468.399
4	1561.073	1517.605	1493.257
5	2046.316	1981.451	1917.711
6	1520.972	1473.492	1472.364
7	1905.605	1884.544	1854.351
8	1570.768	1533.977	1515.448
9	1560.482	1544.941	1529.496
10	1550.327	1530.997	1524.594
	model_one	model_two	model_three
	1678.217	1643.640	1618.980
	Time difference of 23.19257 secs		

Figure 2

Conclusion

Rather than use established libraries with for-loops over Stores and Departments, a hand-coded highly vectorized approach enabled good results to be obtained without lengthy run-times. The run time of only 2.3 seconds per fold (23 seconds total run-time) allowed for significant experimentation/exploration to optimize results which would have been more challenging with a longer-running for-loop/library-based approach.

However whilst the run-time and computational load are very low compared to using the R forecast library, and the results are better than that required for this project, they are not quite as good as those possible using the 'tslm' and 'stlf' algorithms from this library.

Acknowledgements

James King (3rd position in Walmart Sales Kaggle competition) for the idea to blend weeks from the previous year, and for recommendation on the handling of Thanksgiving week (I am not US-based so was not aware of this shopping peak). <https://www.kaggle.com/jfkingiii/competitions>

Appendix

The train.csv data file was sourced from Kaggle as required <https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting>

The code to generate train.csv, test.csv and the fold csv files was sourced from: https://piazza.com/class_profile/get_resource/jky28ddlhmu2r8/jn12q355sd52hn