Walmart Store Sales Forecasting

Ahmad Al-Dhalaan

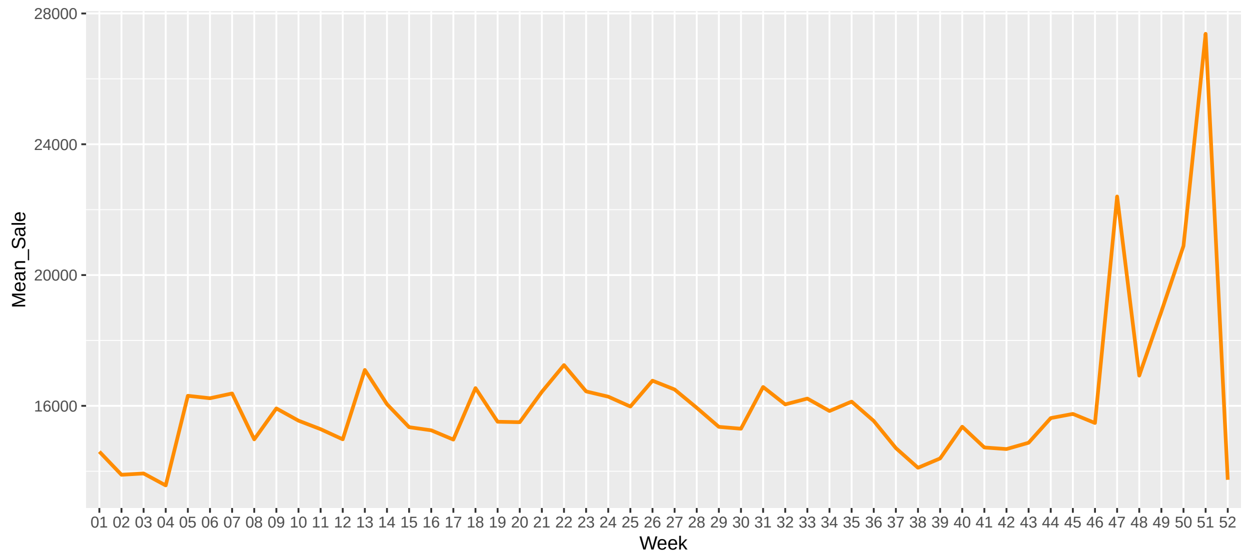
11/9/2020

Pre-processing

In this project, we use the sales data for 45 Walmart stores to predict the future weekly sales for each department in each store. The data is split into a training set, comprising the sales data from 2010-02 to 2011-02 and ten folds of test data from 2011-03 to 2012-10, with each fold consisting of 2 months of sales data.

The data is fairly clean and we do not pre-process any data per se. We do, however, note that many store-department combinations do not have weekly sales values in either the train or test datasets (or both). In those cases, we remove the store-department combination in order to avoid null values and speed up the processing.

Visualizing the training time-series by plotting the mean weekly sales for all stores over the given training time period. We note a non-seasonal time series with a trend component from January to October and an irregular component in November and especially December. However, this is expected as both have the two major American holidays, Thanksgiving and Christmas, which are major shopping seasons with high retail sales.



Implementation

Given the time-series nature of the data, we predict weekly sales for each store-department combination separately. Therefore, we loop over existing store-department combinations and fit a linear regression model with the date (Year and Week) being the predicting variable.

We use the weighted mean absolute error (WMAE) to evaluate our predictions:

WMAE = $\frac{1}{\sum w_i} \sum_{i=1}^{n} w_i |y_i - \hat{y}_i|$

where

n is the number of rows
$\hat{y}_i$ is the predicted sales
y_i is the actual sales
w_i are weights. w = 5 if the week is a holiday week, 1 otherwise

Given that the weights penalize errors during a holiday week, we give special consideration to test fold 5 as it covers the time period from 2011-11-01 to 2012-01-01, which has two holidays (with the most sales). A possible source of error is that the weeks might not overlap exactly between the train and test data. Indeed, the Christmas holiday week in the training set is from 2010-12-24, whereas it is from 2011-12-23 in the test set. This one-day difference will be influential in our predictions and must be taken into consideration. Therefore, we "shift" our predictions for this week (Week 52) and the preceding three weeks in order to somewhat align our test and training time intervals. We accomplish this by splitting the prediction in two, with one prediction staying in its current week (for the six "correct" days) and one prediction shifting to the previous week (for the one day beloning to the previous training week):

$WeeklySale_{52} = [(6/7) * WeeklySale_{52}] + [(1/7) * WeeklySale_{51}]$
$WeeklySale_{51} = [(6/7) * WeeklySale_{51}] + [(1/7) * WeeklySale_{50}]$
$WeeklySale_{50} = [(6/7) * WeeklySale_{50}] + [(1/7) * WeeklySale_{49}]$
$WeeklySale_{49} = [(6/7) * WeeklySale_{49}] + [(1/7) * WeeklySale_{48}]$

Results

We use a 64-bit Windows Operating System - Intel Core i5-3317U CPU with 4GB of RAM to run the ten folds. The running time for all ten folds was 1155.89 seconds (~20 minutes). The mean WAME was 1629.389, with the ten folds shown as below. We were able to decrease the WMAE for Fold 5 from 2324.496 (no shifting) to 2021.303, by using the shifting method as discussed earlier.

Table 1: WMAE for train/test splits

Split	WMAE
1	2045.243
2	1466.912
3	1449.852
4	1593.998
5	2021.303
6	1677.483
7	1722.274
8	1428.212
9	1443.960
10	1444.656