Team

- Amratya Roy Chaudhary (amratya4@illinois.edu)
- Aman Arora (amana4@illinois.edu)
- Kumar Gaurav Dubey (kgdubey2@illinois.edu)

Introduction

For this project we were provided with historical sales data for 45 Walmart stores located in different regions 2010-02 (February 2010) to 2012-10 (October 2012). This data contains 5 different variables.

Store : numeric, Dept : numeric, Date : date format, Weekly_Sales : price which is again a number and the Isholiday which is marked as it's a holiday or not. Total we have 421570 records. The Isholiday column is an indicator variable that shows if each of the given week is a special holiday week or not.

The objective for our project is to forecast the future weekly sales for each department in each store based on the historical data given to us

Data Pre Processing

The train.csv file that was provided to us was split into the following files: 1. Train_ini.csv – This file contains the same weekly sales data as the train.csv file but from the time frame 2010-02 to 2011-02. 2. Test.csv – This file contains 4 variables Store, Dept, Date and IsHoliday for the test time frame 2011-03 to 2012-10. The Weekly_Sales column was removed from this dataset. 3. Fold1.csv... Fold10.csv – The train.csv file provided for the test time frame, 2011-03 to 2012-10, was further split into 10 folds each with each fold containing 2 months' worth of weekly sales data. For each of the 10 folds the following initial data processing steps were performed: 1. Since the dates are not same across each fold, we computed the unique test dates for each fold that needed prediction. 2. Similarly, we computed the unique stores and unique department within each store that need prediction and stored them all together in a dataset. Same steps were performed for the train dataset as well. 3. We then create a time series view for each store's department for the train dataset. 4. For the test dataset, we created a place holder dataset with a time series view.

Approach

We check the following approach:

- Consider seasonal naive with Singular value decomposition.
  The singular value decomposition (SVD) provides another way to factorize a matrix, into singular vectors and singular values. We set each forecast to be equal to the last observed value from the same season of the year.

- Consider seasonal naive without Singular value decomposition.
  The disadvantage is that it does not consider any possible causal relationships that underly the forecasted variable. This model adds the latest observed absolute period -to-period change to the most recent observed level of the variable.

- Consider time series linear model.
  We use tslm from forecast package and applied over each column of the training matrix. This is the same as fitting a linear regression model on the time series of a given store, and then the sales of this store for the next two months are forecasted using the fitted model. This is done for every store of a given department.

Data Post Processing

We saw Fold 5 has a high WMAE because it contains two holiday weeks and therefore receives higher weights in WMAE. The high WMAE for fold 5 is due to a slight shift of the Christmas shopping season from 2010 to 2011. TO overcome this post forecast adjustment we made with a shifting method for fold 5 (weeks 46 to 52). If the average sales for weeks 49, 50 and 51 of a given department were at least 10% higher than for weeks 48 and 52, then circularly shift a fraction of the sales from weeks 48 through 52 into the next week (and from 52 back to 48)

Result

Seasonal Naïve with svd

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10
2214.901	1742.840	1740.698	1662.677	2093.030	1625.173	2019.327	1673.862	1649.592	1620.931

```
>
> print(wae)
 [1] 2214.901 1742.840 1740.698 1662.677 2093.030 1626.173 2019.327 1673.862 1649.595 1620.931
> mean(wae)
[1] 1804.403
```

Seasonal Naïve without svd

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10
2262.422	1787.081	1779.052	1716.117	2112.743	1696.900	2086.967	1750.283	1719.887	1680.956

```
> print(wae)
 [1] 2262.422 1787.081 1779.052 1716.117 2112.743 1696.900 2086.967 1750.283 1719.887 1680.956
> mean(wae)
[1] 1859.241
```

Time series linear model

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10
2042.401	1440.083	1434.716	1596.988	2029.388	1674.185	1718.577	1420.817	1430.801	1447.034

```
> print(wae)
 [1] 2042.401 1440.083 1434.716 1596.988 2029.388 1674.185 1718.577 1420.817 1430.801 1447.034
> mean(wae)
[1] 1623.499
>
```

Conclusion

We consider taking the time series liner model (tslm) and implement that for the focasting resulting the below result. TSLM has advantages over seasonal naive models because it is able to learn both the trend and seasonality of the time series data. Although there is not much trend in the Walmart time series data since there are only up to two years of training data when forecasting for sales price in 2012, it might still be useful to include. Trend is modelled by including a degrees of freedom associated with time in the linear model

Best result is from tslm: 1623.499 mean(wae)

Technical specification

OS Name	Microsoft Windows 10 Home
Processor	Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz, 1992 Mhz, 4 Core(s), 8 Logical Processor(s)
Installed Physical Memory (RAM)	8.00 GB
Total Physical Memory	7.88 GB
Available Physical Memory	1.52 GB
Total Virtual Memory	17.1 GB
Available Virtual Memory	3.10 GB

OS Name	Microsoft Windows 10 Home
Processor	Intel(R) Core(TM) i7-1061 CPU @ 1.80GHz, 2.3 Ghz, 4 Core(s), 8 Logical Processor(s)
Installed Physical Memory (RAM)	16.00 GB

Acknowledgement

- Professor and TAs: Amazing help were provided in the office hours to understand concepts of different elements of the projects.
- All the OH helped a lot to understand the issues and the approach.
- Piazza posts.
- Josh code : https://liangfgithub.github.io/Example_Code_Project2_Josh.html
- https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting/data