Walmart Store Sales Forecasting

Krishna Kinnal, kkinnal@eng.ucsd.edu, (A11359054)
Ashutosh Giri, a2giri@eng.ucsd.edu, (A53243535)
Asha Tulsyan, atulsyan@eng.ucsd.edu, (A53234785)

I. Introduction

Walmart is an American Multinational retail corporation that operates as a chain of hypermarkets, discount department stores, and grocery stores. As part of this assignment, we are provided with historical sales data for 45 Walmart stores located in different regions, with the objective being to predict forecast sales for all given stores.

Sales forecasting is the process of estimating future sales based on previous sales and other trends. Accurate sales forecasts enable companies to make informed business decisions and predict short-term and long-term performance. Companies can base their forecasts on past sales data, industry-wide comparisons, and economic trends. It is easier for established companies to predict future sales based on past business data of many years. Sales forecasting gives insight into how a company should manage its workforce, cash flow, and resources. In addition to helping a company allocate its internal resources effectively, predictive sales data is important for businesses when looking to acquire investment capital.

The predictive models we have implemented for this dataset try to best capture the most important aspects of the data and make the best possible sales forecast, evaluated by a metric known as the weighted mean absolute error (WMAE).

II. The Dataset

We chose our dataset as the historical sales data for 45 Walmart stores located across different regions. Each store may contains many departments (there are 98 departments in total in our dataset). We obtained our dataset from Kaggle, and only used the dataset information for our purpose. The information about the 45 stores, including the type and size of the store has been anonymized and abstracted for our use.

Historical data from 2010, 2011 and 2012 (i.e., from 02/05/2010 to 11/01/2012) with the following fields from the `train.csv` file are:-

- Store - The Store ID
- Dept - The Department ID
- Date - Given for each week, starting from 02/05/2010
- Sales - Sales for this Store-Department-Week triad
- Is_Holiday - Whether this week is a special holiday week or not

Additional data related to the store, department, and other date-specific information are given in the `features.csv` file, containing the following fields:

- Store - The Store ID
- Date - Given for each week
- Temperature - Average temperature in the region of this store for this week
- FuelPrice - Fuel Price in the region of this store for this week
- MarkDown - Anonymized data related to promotional markdowns that Walmart ran on this particular week. MarkDown data is only available after Nov 2011, and is not available for all stores all the time. Any missing value is replaced with the average of the markdowns.
- CPI - The consumer price index on this week
- Unemployment - The unemployment rate on this week
- Is_Holiday - Whether this week is a special holiday week or not

The four holidays fall within the following weeks in the dataset (not all holidays are in the data):

1) SuperBowl: 12-Feb-10, 11-Feb-11, 10-Feb-12
2) LaborDay: 10-Sep-10, 9-Sep-11, 7-Sep-12
3) Thanksgiving: 26-Nov-10, 25-Nov-11, 23-Nov-12
4) Christmas: 31-Dec-10, 30-Dec-11, 28-Dec-12

We also have the following features about the stores given in the `stores.csv` file, containing the following fields:-

- Store - The Store ID
- Type - The store's type, takes values 'A', 'B', and 'C'
- Size - The size of this store in square feet.

In total, we have 4,21,570 entries for store, department and week triplet. Which means, training is roughly 2,81,046 data samples and validation and testing set being almost equal to 70,262 each.

A. Data Summary

We have three years data of sales data with us. We are using the data from first two years for creating the Training set and the third year data we have divided into Testing and Validation sets respectively.

B. Data Analysis

We explored different data features from the training data and below are the observations and deductions in both literature and graphical form.

We can see from Fig. 1 that the store types, which are A,B and C, are almost a linear function of the average type size, indicating Type A stores are large, Type B ones are mid-sized and Type C are small ones. So, type feature does not add any extra information, hence it is not really needed to be included in our model if we are already including the size as feature. Now, we found the days with the largest sale in 2010 and 2011 (for the training data), they are as follows:-

- 12/24/2010 - 27378.69 (Christmas 2010)

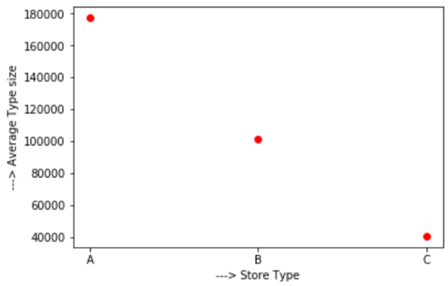


Fig. 1: Average Type Size v/s Store Type

- 12/23/2011 - 25437.14 (Christmas 2011)
- 11/26/2010 - 22403.33 (Thanksgiving 2010)
- 11/25/2011 - 22043.56 (Thanksgiving 2011)
- 12/17/2010 - 20892.46

Interestingly they are on Christmas and Thanksgiving of 2010 and 2011 with another week preceding Christmas 2010 completing the top five.

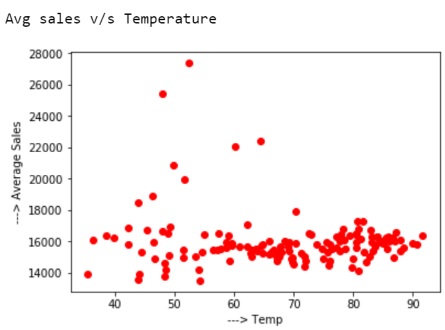


Fig. 2: Average Sales vs Temperature

Plotting a graph of sales v/s temperature as shown in Fig. 2, and analyzing the points for the highest sale, it might suggest that sales are higher on colder days. However the fact is that they are higher because of the festive season which happen to fall in winter and if we just analyze the data on non-holiday weeks, there is not much variation in sale with the temperature. So, temperature as a feature seems to be redundant here. Nevertheless, we have included in the initial multi-linear regression model we have used, as explained in a later section, and later show that not including this feature will improve the predictor. We plotted similar graphs for Fuel, CPI and Unemployment rates, as shown in Fig. 3, 4, and 5.
From the plots, if we ignore the highest sales due to the festive season, the input of these feature does not add any extra or related information to determine the sales value. In fact, since sales value does not change much and is not even close to following a linear pattern with the changing value of these features (although it makes sense at first intuitively to apply linear regression model to solve this task) including these features and using linear regression would not give a good result as these features might just act as contributing to noise rather than providing any useful information to get our prediction anywhere close to the actual value of the sales for that particular sales-department-week triad. Again, this is proved in later sections when we use one model that accommodates these features, and another that doesn't, and compare their predictions.

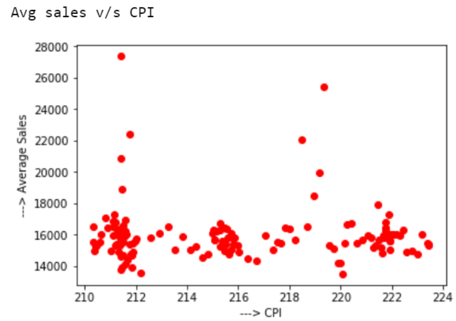


Fig. 3: Average Sales v/s CPI

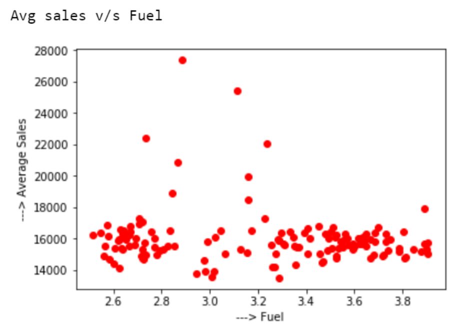


Fig. 4: Average Sales v/s Fuel

We also plotted the standard deviation in sales vs stores and department (in Fig. 6) and found that for stores, data is scattered all over and in general larger stores have higher deviation (and hence, variance) and vice- versa which is

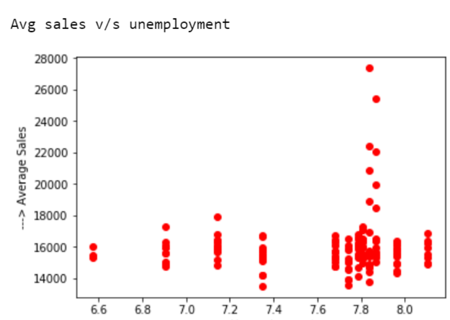


Fig. 5: Average Sales v/s UnEmployment

Fig. 6: Std. Deviation in Sales v/s Stores

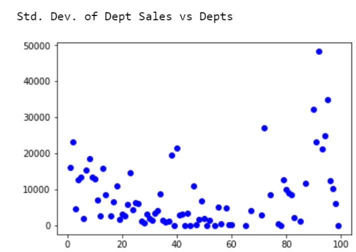


Fig. 7: Std. Deviation in Sales v/s Dept

expected. Also, average variance is higher in this case. The data points in deviation vs department (in Fig. 7) shows that some departments show more variance than others, while on average the variance is lesser. This will prove useful later when we implement an averaging model iterating over the departments, when we make the department as the base parameter and calculate averages over all stores and all weeks.

III. The Predictive Task

Our task is - given the sales of all the stores and departments over the first two years, forecast the sales of these stores and departments for the third year. Here, the data in the third year is split in equal proportion into the validation and the test set. Since, we already have the actual value of the sales for the third year, we can calculate the error with respect to the predicted value.

The predicted value for the data in the test set is evaluated based on the weighted mean absolute error (WMAE) value which is given by:

$$WMAE = \frac{1}{\sum w_i} \sum w_i |y_i - y_i'| \tag{1}$$

where w_i is the weight that is associated with a particular week, i.e, weight=5 if it a holiday,=1 if it isn't.

Essentially, this means that the penalty for inaccuracy in the sales for holidays is higher. Hence, we should develop a model that prioritizes a more accurate prediction for holidays. However, since the number of holidays is quite less compared to the number of non-holidays, the contribution of the error from non-holidays would still be substantially higher than the error from the sales predicted from the holiday-weeks. So, this means that while it is rewarding to predict more accurately on holidays, we can not really afford to have higher inaccuracy on non-holiday weeks either.
From the data described earlier, it is fairly simple to parse the various fields and parameters and store them in appropriate data structures (w have used Python lists and dictionaries). The models we used were linear regression with multiple variations of different features, support vector regression, gradient boosting, random forest, simple model of using the sales value from previous year averaged over certain features such as store, week. The intuition behind using these models and features, which of them worked well, and which of them did not, and the probable reasons for the same have been covered in the later sections.

IV. The Different Models Used

Throughout this section, we will talk about models, with each subsequent model being an increment of the previous model, unless mentioned otherwise.

A. Simple Multiple Linear Regression Model

In the simple regression model, we build a feature vector that contains the average sales of the store, average sales of the department, average sales of the week, the temperature, fuel price, CPI, and unemployment values. Also, we append a value of 5.0 if that particular week is a holiday or not (by holiday, here we mean whether that particular week is a Super bowl, Labor Day, Thanksgiving, or Christmas week). Therefore the feature vector looks like this.

$$[Avg\_Sales\_Store, Avg\_Sales\_Dept, Avg\_Sale\_Week$$
$$Temperature, FuelPrice, CPI, Unemployment, Weight] \tag{2}$$

where Weight=5 if the week is a holiday; otherwise Weight=1. We perform simple least squares multiple linear regression on this feature vector with the sales for that particular store-department-week triad.

$$y = \theta_0 + x_1\theta_1 + x_2\theta_2 + ... + x_9\theta_9 \tag{3}$$

We tested this on the validation set and got a very bad error rate of close to 15000, almost as bad as a trivial predictor that predicts the average of all sales in the training set, which has an error rate of 15331.46. We knew we could do better by optimizing the regular model with the regularization parameter :

$$argmin_\theta = \frac{1}{N}||y - X\theta||_2^2 + \lambda||\theta||_2^2 \tag{4}$$

We choose a particular value of λ by finding out which λ gives the least WMAE value.

This value was found to be λ=10. that had an error of 9293.78. Using this lambda on the test set, we found WMAE=9382.86.
To further optimize prediction, we tried using the following regression techniques.

1) Support Vector Regression: Using the Support Vector Regresion model, given by the `sklearn.svm.LinearSVM` function in Python, and using the same feature vector as explained previously, we trained this model on the training set, and by trying out different values of C (which is the penalty parameter 'C' of the error term), we try to find the least WMAE for the validation set. This was found to be C=0.001 with an error of 8932.76. Testing this model on the test set with the penalty parameter set to 0.001 gave WMAE = 8899.97, which turns out to be slightly better than the gradient descent model we used earlier. However, we decided we could do better with the help of ensemble methods to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability/robustness over a single estimator.

2) Random Forest Regression: The `sklearn.ensemble` module includes two averaging algorithms based on randomized decision trees: the RandomForest algorithm and the Extra-Trees method. Both algorithms are perturb-and-combine techniques specifically designed for trees. This means a diverse set of classifiers is created by introducing randomness in the classifier construction. The prediction of the ensemble is given as the averaged prediction of the individual classifiers. A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting. This function makes use of the n_estimators parameter (essentially the number of the trees in the forest) performed best on the validation set with a value of 30, giving an error of 8667.99. Initially we tried setting this parameter to a higher value of 50, 100, etc., but this resulted in memory issues on our system. Perhaps we could've implemented this on a better system, but with what we had, this is the best we could do. Using this value of n_estimators for testing on the test set, we get a WMAE=8437.19. Obviously, this is better than the techniques we tried previously, but not good enough.

3) Gradient Boosting Regression: Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function. The module `sklearn.ensemble` provides methods for both classification and regression via gradient boosted regression trees. By tweaking the n_estimators parameter (which is the number of boosting stages to perform), we found that for n_estimators=100, with the error being 8502.59. Initially, we tried tweaking the parameter to a higher value, as usually a higher value will result in a better model. However this resulted in memory errors. Also, when we tried increasing the learning rate, the prediction was way off (giving an error of close to the trivial predictor), and hence we used the default learning rate of 1. Using this value and testing the model on the test set, we get a prediction error of 8335.50. Obviously, this performed the best among all techniques so far, but we can still do better by changing the way we store the feature vectors to represent the training data.

B. Multiple linear regression for each store

As an incremental change to the first model, we decided to store the feature vectors of the data by store (we used a Python dictionary to implement this). Essentially, for each store of the 45 Walmart stores, we have a θ vector that has the coefficient values for each of the features in the feature vector (which is the same as mentioned before). Hence, what we are doing is a store-wise prediction, i.e., for each store-department-week triad in the test set, we do a prediction on the sales for that store based on its particular θ values. This level of specificity could greatly improve predictions, according to our intuition. We followed the same procedure as mentioned in simple multiple linear regression model, except this time we stored the X and y vectors as dictionaries with the storeID as the key value. Hence the feature vector looks like this.

$$Feature\_Vector_{store} = [Avg\_Sales\_Dept, Avg\_Sale\_Week \\ Temperature, FuelPrice, CPI, Unemployment, Weight]$$

where Weight=5 if the week is a holiday; otherwise Weight=1. Implementing multiple linear regression with regularization done for each store with gradient descent on the validation set, we found that the WMAE was the least when λ=1000, with an error of 5875.88. Implementing this model on the test set gave an error of 5811.96. This is a significant improvement from Model 1. We followed the same steps as before, trying to optimize our predictions with the following regression techniques.

1) Support Vector Regression: Using the same function as mentioned above, we tested the model on the validation set and found that the WMAE is the least when the penalty parameter C=0.001, giving WMAE=5770.08. Testing this model on the test set gave an error of 5696.61; slightly better than the previous technique, but not by too much.

2) Random Forest Regression: Using the same function as mentioned above, we test the model on the validation set and find that the WMAE is the least when the number of trees in the forest n_estimators=20, giving an error of 8655.78. Testing this model on the test set gave WMAE=8642.66. This performed worse than the SVR model.

3) Gradient Boosting Regression: Using the same function as mentioned above, we test the model on the validation set and find that the WMAE is the least when the number of boosting stages n_estimators=2000, giving the error measure equal to 8996.08. Testing this model on the test set gave an error of 8975.89; again a lot worse than the SVR model.

C. Multiple linear regression with an improved feature vector

As explained in the data analysis section earlier, we found that the variance of the features like CPI, Temperature, Fuel Price, and Unemployment were not too much, and were unlikely to improve the prediction of the sales. It was in fact found (after running this model) that the aforementioned features in fact contributed to noise, and didn't do anything to make the prediction better, i.e, they were superfluous features. Also, we tried adding four new binary features to this model one each for Super Bowl, Labor Day, Thanksgiving, and Christmas. The new feature vector now looks like this.

$$[Avg\_Sales\_Dept, Avg\_Sales\_Week, IsSuperBowl \\ IsLaborDay, IsThanksgiving, IsChristmas, Weight] \tag{5}$$

where Weight = 5 if any one of IsSuperBowl, IsLaborDay, IsThanksgiving, or IsChristmas is true; otherwise Weight = 1.

Hence we followed the same regression model as used in the previous section (performing regression for every store) except this time, we used the new feature vector as described above. Training this model on the training set and running it on the validation set for different values of λ, we found that for λ=10, the error 5578.77. Testing this model on the test set for the same value of lambda gave WMAE=5539.63.
This gave us a better result compared to the previous model. We have gradually improved our prediction with each iterative model. However, we decided to try completely different methods that did not involve multiple linear regression or its variants, models that were simple in nature, easy to implement, and provided better predictions, as we shall see in the next two models.

D. Simple Average Product model

Here, in the training set, for each unique department-store pair, we find the average of all sales over all the weeks. Then, for each unique department-week pair, we find the average over all stores. These average values for every department-week pair and department-store pair are stored in respective dictionaries that can be accessed by using the pair as the key value. Also, as used in the previous models, we also find the average sales for each department, which is also stored in a dictionary.
Then, reading from the test set, for every store-department-week triad, we find the following product

$$\frac{Avg\_Sales_{dept,week} * Avg\_Sales_{dept,store}}{Avg\_Sales_{dept}} \tag{6}$$

The above product is the sales prediction for that particular store-department-week triad. Since this model doesn't require any validation set for testing a regularization parameter, we directly proceed to test this model on the test set. This model performed the best among all models, giving us an error measure of 2404.50, close to 2.5 times better than the model that employed the improved feature vector.

E. Average of past year sales

Looking at the dataset, another simple model we had in mind was to implement a predictor that trains on the sales values from the first two years of data and predicts for the third year by simply averaging the values of the sales of the past two years for each respective week. In other words, for each store-department-week triad in the test set, we predict the sales to be the average of the sales for (store, department, (week-52)) and (store-department-(week-104)) in the training dataset. For this purpose, a 3-dimensional dictionary is used to reduce look-up time while iterating over the test set, that stores the sales value for a given store-department-week triad in the training set. The aforementioned prediction of sales for a given store-department-week triad may be represented mathematically as follows.

$$\frac{Sales_{store,dept,week-52} + Sales_{store,dept,week-104}}{2} \tag{7}$$

This model was trained on the training set. As with the previously discussed Average Product Model, we find that there's no need to test it on a validation set to further optimize the model, and hence we directly proceed to test it on the test set. Using the WMAE metric, we got an error measure of 3367.86, which performs better than all of the regression models, and almost as good as the average product model. From this, we can conclude that for this dataset in particular, a simple model based on averages can make the best predictions on the test set, better than those made by linear, support vector, random forest, and gradient boosting regression models.

V. Literature Survey

We obtained our dataset from Walmart recruitment Kaggle competition. Other similar types of datasets involving forecasting over some period of time which we found as a part of our research are:

- Daily minimum temperatures in Melbourne, Australia, 1981-1990
- Zurich monthly sunspot numbers 1749-1983
- Daily total female births in California, 1959
- Occupancy Detection Data Set
- Ozone Level Detection Data Set
- Sales of shampoo over a three-year period

After completing our task, we went over the discussion about the models used by others in the forecasting task, our result of using linear regression, modified linear regression, SVR, and finally simple model of using the average of last two years data for the same (store,department, week) triad to predict the sales, and the model of taking average of department over stores and weeks are in line with what others have achieved. Of course, the best models have used ensemble of various models and other complex packages and advanced concepts of Statistical Interference. A lot of the competitors have used the `tslm` (time series linear model) package from R language on such tasks of forecasting which have resulted in pretty good results. It allows to fit a linear regression model with a linear trend and weekly dummy variables with this bit of code. Although, from the look of what the models do, it seemed it could have resulted better result in our task as well. However, we wanted to stick to Python and solve the task using the methods and algorithms we have studied and understood well rather than using a library function of a different language that we dont understood completely, even though it may have yielded a good result.

Our best model - the average product model - gave an WMAE value of 2404 which would have put us in the top 10% of the leaderboard. The simple average product model works best on this task, much better than simple linear regression and better than all the other models we tried. Later, we found in our research several others have also found much better result using the simple models as compared to the linear regression, KNN-regression and SVR without making use of any features such as fuel, unemployment etc. Some of the state-of-the-art methods currently employed use various simple and complex models and take the average of them all as the final value.
These are the statistical methods used:

- ARIMA
- UCM

The machine learning methods used:

1) Random Forest
2) Linear Regression
3) K nearest regression
4) Principle Component Regression

Some other models also made use of ensemble of certain complex packages for the forecasting task:

1) svd + stlf/ets
2) svd + stlf/arima
3) standard scaling + stlf/ets + averaging
4) svd + seasonal arima

The abbreviations full-form are below:

- ARIMA - Auto-regressive Integrated Moving Average
- UCM - Unobserved Components Model
- STL - Seasonal and Trend decomposition using Loess, while Loess is a method for estimating nonlinear relationships.
- STLF - STL forecast

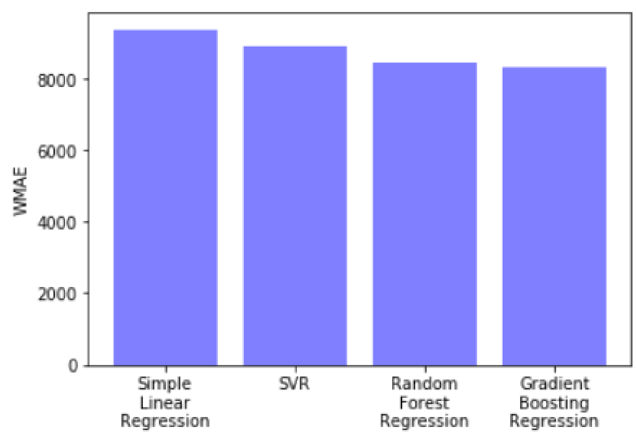
There's also the concept of day-shifting adjustment. There is an excellent analysis by David Thaler on this about how day shift for the same week from one year to the next year affects the forecasting sales value.
In the given data, we had weekly input and weekly output, so we used almost exclusively weekly models, with a 52-week year. For the most part that worked well. The data is short, so the weeks line up pretty well. This is exactly what we have also done in our model which gave the best result.
However, analyzing the same to the holidays week requires deeper analysis. Holidays such as Labor day and Thanksgiving dont occur on a fixed date of the year but on a fixed week, i.e. those events have a fixed relationship to the week boundaries. Christmas is different, it occurs on a fixed date so its day of the week changes, and it has a big sales bulge associated with it, so it matters a lot here. In the first year of the training data, it occurs on a Saturday (with weeks ending on Friday). That causes all of its sales bulge to fall into the week before. In the second year of the training data, it occurs on a Sunday, so there is one pre-Christmas shopping day in week 52. The test set has Christmas for 2012, which is a leap year. That puts Christmas on a Tuesday, with 3 pre-Christmas shopping days in its week.

In the training data, we took a look at the departments that exhibit a bulge in sales around Christmas, and we observed that week 52, the week with Christmas in it, looks pretty normal. Also, week 48, the week after Thanksgiving, does too. So, when we are implementing a model where we are adjusting the shift in days for the same week of one year to the other, we have take care of this too. This was one aspect we did not include in our model and this remains as one of the key tasks for future.
Those who implemented the above were able to achieve WMAE value as low as 2200-2300, which is the best result achieved so far by anyone for this task.

VI. Results and Conclusion

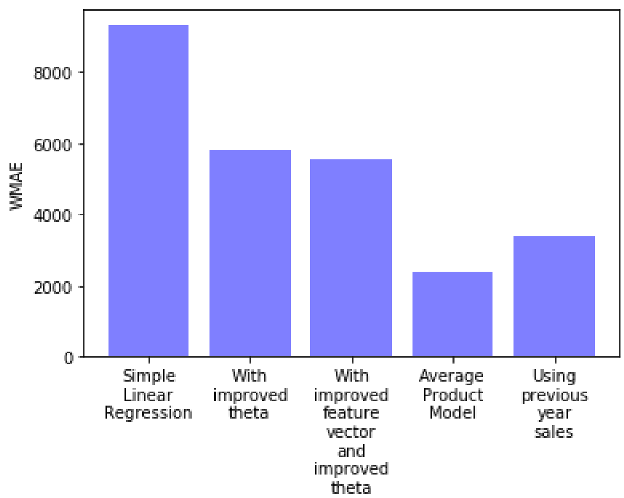
As shown in the graph below, the following are the error values of the regression model and further improvements:-

- Simple Multiple Linear Regression - 9304.87
- Support Vector Regression - 8899.96
- Random Forest Regression - 8437.63
- Gradient Boosting Regression - 8335.61



And as shown in the graph below, following is the summary of the WMAE values for the different types of models we tried, as explained in the previous section

- Simple Multiple Linear Regression - 9304.87
- With Improved Theta - 5811.96
- With improved theta and improved feature vector - 5539.63
- Average Product Model - 2404.50
- Average of previous year's sales - 3367.86



We have discussed in detail while doing the data analysis from the plots on why simple linear regression might not be the best model for this task and same is evident from the results as well where we did not get good result at all.

The primary reason is each store behaves a bit differently with respect to season and week and hence using a common theta vector which will be applicable to all the stores will not yield optimum result. To get a better performance, we need to capture the individual variations of each store against week and festivals and feature, so it would be better to run the linear regression individually for each store. i.e. for each store run the model on the training data of that store and get theta vector specific to each store. And, use the theta vector respective to the store for which sales prediction needs to be made. Let's call this model as Linear regression with improved theta. When we implemented this model, the performance improved significantly with WMAE coming equal to 5811.96 compared to the previous value of 9304.87 The secondary reason is the features such as temperature, fuel, CPI, unemployment dont seem to impact the sales values either linearly or in some other pattern so as to be modeled into with certain parameters and the higher sales corresponding the festival weeks. So, in one way the regression model will think that the values of those features which happen to occur on the festival season may have some role in boosting up the sale and try to model the theta values accordingly. This will result in sales being predicted higher even in non-holiday weeks for those feature values. Thus, these features look not only redundant but will also add noise to our prediction model rather than giving any useful information which can help in sales prediction accuracy.
Also, sales increase is higher in the Thanksgiving and Christmas week, but not so much in other festival weeks. So, it's expected that using separate feature for each individual holiday instead of a common one for all the holidays, would give improved result. The above two hypothesis gets confirmed when we implement this (i.e using separate features for each holiday and removing the temperature, CPI, fuel, unemployment from the feature vector) on top of the previous improved theta model and observe that new WMAE improved to 5539.63. Let's call this model as Linear regression with improved theta and improved feature vector.

Intuition for using the dept. averaging model is that we observed that variation of sales in stores from week to week was not much, i.e. if for a certain holiday the sales of a store increases by certain factor, it increases more or less by the same factor for the other stores as well. However, such was not the case was with department v/s week variation. Some of the departments showed huge surge in sales on a particular holiday. Thus, department sales followed seasonal variation across all the stores. We think this also makes sense in the practical world, as on Thanksgiving Day, clothes, and accessories-related department sales will see more surge while on Christmas, gifts, decorations, and toys-related departments will see higher surges. Hence, we thought for each department, we make use of average of the sales for that week over all the stores, sales for that store over all the weeks and average over all the weeks and stores in our prediction, the equation of which is provided in the previous section.

In summary, the following alone results in a quite low WMAE value and little modification and averaging over the parameters further improves the result.

- The best predictor of sales is the sales from the prior year.
- Line up important weeks - for example predict Thanksgiving by Thanksgiving regardless of which week of the year it is.
- Reflect this implied trend by store and department.

VII. Future Work

As mentioned earlier, day-shifting adjustment is one of the key ideas which can improve our result on top of the product averaging model we have, so this remains the very next thing which can be implemented in the future work. Additionally, we are hopeful that the ensemble of multiple simpler and complex machine-learning and statistical models (mentioned earlier in the literature section) could also result in better results. Artificial Neural Network (ANN) is also one of the widely used methodologies being used for the prediction and forecasting tasks of weather, and the same can be extended for sales prediction.

References

[1] https://trackmaven.com/marketing-dictionary/sales-forecasting/
[2] https://en.wikipedia.org/wiki/Walmart
[3] https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting
[4] https://datamarket.com/data/set/2324/daily-minimum-temperatures-in-melbourne-australia-1981-1990
[5] http://archive.ics.uci.edu/ml/datasets/Ozone+Level+Detection
[6] https://datamarket.com/data/set/22r0/sales-of-shampoo-over-a-three-year-period
[7] https://datamarket.com/data/set/235k/daily-total-female-births-in-california-1959

[8] https://www.sciencedirect.com/science/article/pii/S0360544209002539
[9] https://www.researchgate.net/publication/4155484_Short-term_load_forecasting_for_the_holidays_using_fuzzy_linear_regression_method
[10] https://www.sciencedirect.com/science/article/pii/S0960148108003327