Juan Rojas

Jesus Mendoza

Brian Canela

Gian Tolentino

Abraham Vega

Data Science, Fall 2018

Walmart Store Sales Forecasting

Project Details:

The project we chose involves sales forecasting for the Walmart company, where the company provides data sets from 45 different stores located in various regions. The original project description wanted to project the sales for each department within each store, however we as a group proposed to project the weekly sales based on the features from the data.

Details about data:

The project provides three data files: features, stores, train. In total of more than 400k rows of data.

Features.csv

Store number, date, temperature, fuel price, 5 columns of markdown, consumer price index, unemployment rate and a boolean of whether or not that week contained a holiday.

Stores.csv

Information store type and size

Data.csv (422k x 5)

information on the store number, the department, date, weekly sales, and whether it is a holiday or not.

Random Forest Regression: Gian

Considering that our project goal was to predict sales forecasting from a weekly basis, I decided to use RandomForestRegressor. One of the biggest benefits of using random forest is that when you use enough trees for predictions overfitting will not be an issue, however using too many trees can drastically affect the time it takes for the algorithm to run completely.

Another benefit of using random forest was that it provided good results without necessitating too much parameter adjustments. In fact the only parameter I adjusted for this portion was the n_estimators, which is the number of trees the algorithm builds to perform the necessary prediction.

I ran the algorithm with various n_estimators

num_est = [2,4,10,20,30,40,50,60,70,80,90,100]

To which I compared the resulting r2_score and rsme values to see which number of trees would best fit the dataset we were using. After running the algorithm through all 12 of the n_estimators, I found that the "60" gave the highest r2_score and the lowest rsme value, indicating that it was the most suitable value to use for n_estimators for the algorithm.

Linear Regression with Coefficient: Juan

Since our goal was to estimate weekly sales based on features provided to us, Linear Regression sounded like a good fit. Linear regression models the relationship between a scalar variable and a set of independent variables. Using the merged dataset which matched all stores to their specific features, and the yielded predictive model would tell us which features were most important to predicting weekly sales. The predictive model showed that the most important feature in determining weekly sales was the 'IsHoliday' feature. This means that the weeks that contained a holiday were the most successful when it came to weekly sales, while the least important feature was temperature. The dates that were taken into account contained important days such as the Super Bowl, Labor Day, Thanksgiving (most likely including black friday and cyber monday), and Christmas.

Logistic Regression and Linear Regression with Cross Validation: Brian

In the beginning of the project, we didn't have one complete dataset. So, I had to merge the datasets into one complete dataset by using inner join based on the storeIDs. Once the dataset was merged into one we all worked in the same dataset. Also before working on a specific model, I was looking more in depth into the datasets. By querying data such as what is the highest weekly_sales, when did the highest weekly_sales happened, and also visualizations, like scatter plots based on weekly_sales to features. Like that our group can have a better idea of our dataset. Another team member predicted the rmse score of our dataset with linear regression and end up getting root mean square error of 21830. I figured that using Linear Regression with Cross Validation might help reduce our rmse for linear regression. Using Linear Regression with Cross Validation it helped reduce our rmse by a bit by getting an rmse of 21549. Then since looking at the Linear Regression model and having seen that the feature “IsHoliday” is the most important feature, I figured if we can see a correlation using Logistic Regression, but with “IsHoliday” as out target/label. Logistic Regression is a classifier and considering our goal is to predict Walmart stores sales, our label of weekly_sales is continuous values, thus making Logistic Regression algorithm use weekly_sales label is much more difficult. Since our dataset has a classifier of “IsHoliday”, true or false whether if that week was a holiday or not, we used Logistic Regression and our label as the “IsHoliday” feature. Thus, Logistic Regression with Cross Validation we will like to predict if is Holiday, such as Super Bowl, Labor Day, Thanksgiving (most likely including black friday and cyber monday), and Christmas, has a good evaluation of accuracy and if so could be correlated towards having “IsHoliday” the most important feature based our Linear Regression model. Looking at the results for Logistic Regression we get an accuracy to be exact is 0.9316523155506006 = 93%, and after doing 10-fold cross validation we get a list of accuracies of [0.92957446, 0.93139929, 0.9332258, 0.93426952, 0.93310719, 0.93429324,0.92793605, 0.9327751, 0.92964395, 0.92949995]. Doing the mean of the list the accuracy is = 0.9315724551687609, which is also 93%.

Decision Tree Regressor: Abraham

I decided to go with Decision Tree Regressor after getting an error when using Decision Tree Classifier. That error was that the data was continuous. Decision Tree Regressor was the solution found. It works similar to Decision Tree Classifier, not exactly the same but close enough, however, one of the key differences is that Regressor works better with continuous or numeric response variables. Regressor works well for prediction as opposed to classification. In the end the evaluation accuracy came out as 95%, with IsHoliday as the most important feature, followed by Fuel_Price.

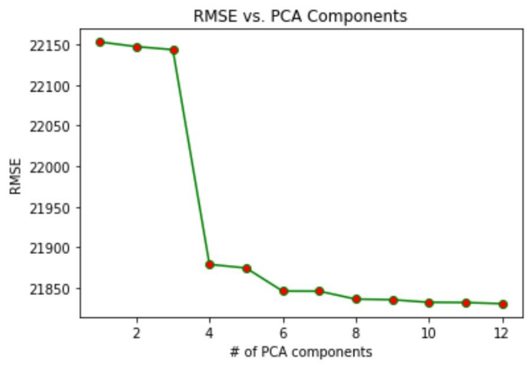
Principal Component Analysis: Jesus

In this part of the project, I was responsible with trying to minimize the number of features by using PCA, while maximizing accuracy and/or minimizing error. We decided to perform PCA on three of the algorithms we worked on: Linear Regression, Decision Tree Regressor, Random Forest.

In order to be able to test each algorithm and determine how many components were optimal for each algorithm, every possibility was tested. From one component in PCA, (meaning that all 12 features would be reduced to 1 feature), to all 12 features. First, I tested every number of PCA components for Linear Regression. However, since on Linear Regression, we already had unexpected results, so there was not much to be able to compare with. For instance, for Linear Regression we got a root mean square error of about 20,000. So when PCA was performed, no improvement was apparent. In this case the 20k range was, and at least, it was also consistent after performing PCA. We could not conclude why our RMSE was so high, especially since the average value for Weekly_Sales was only in the range of 15k. So that RMSE seems high. Yet, still (assuming that our estimates were correct) I concluded that 8 components was a good amount for PCA, since the RMSE did no significantly increase and the number of features was reduced by 33 % as shown here:

Original RMSE without PCA: 21830.539350501567

PCA Components	RMSE	Diff from Original
12	21830.53935050156	+ 0.000
11	21831.95809073203	+ 1.419
10	21832.227514715734	+ 1.688
9	21835.43841747419	+ 3.211
8	21836.167785667436	+ 4.899
7	21845.95820862322	+ 14.69

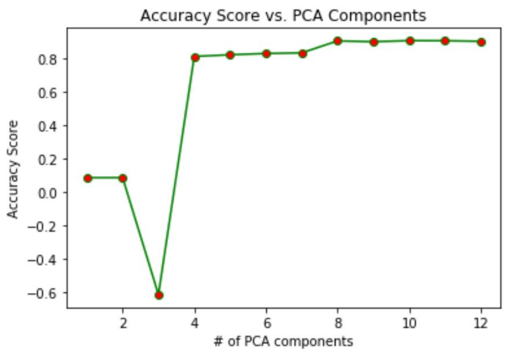


Here you can see that up until after less than 8 components for PCA, then the RMSE increases substantially compared to the larger number of components (as shown in red).

Afterwards, I performed PCA on the Decision Tree Regressor. I followed the exact same procedure as in the previous machine learning algorithm, except using the accuracy score instead of RMSE. In this case, we seemed to get more reasonable results. Also here, the optimal amount of components for PCA was 8 features, since the accuracy did not increase substantially when performing PCA with more features. As shown below:

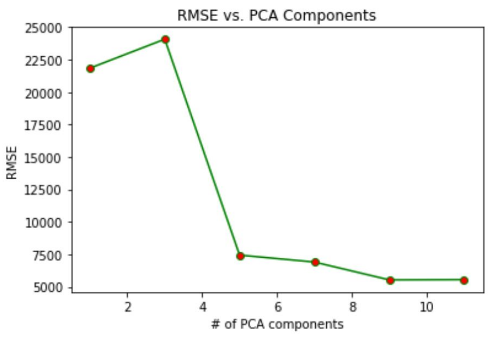
Original Accuracy without PCA : 0.9539125333897658

PCA Components	*Accuracy Score*	*Diff from Original*
12	0.9021163310536108	- 0.0518
11	0.9054183671392539	- 0.0485
10	0.9059802913659352	- 0.0479
9	0.8996297073171443	- 0.0543
8	0.9044101195917806	- 0.0495
7	0.8326874933221743	- 0.1213



In this case, 8 components for PCA is actually the best out of all of the number of components for PCA. Here it is the most optimal since it is only a 4.9 % difference from the prediction that was made without PCA. Therefore, a 5 % sacrifice for a 33 % reduction in features it the best for this set.

Lastly, I performed PCA for the Random Forest algorithm. Considering that the Random Forest classifier is more performance-demanding based on the number of trees in the forest, in order to make it more time efficient and easier to test. Performing PCA on Random Forest for this data set was more interesting. Based on the testing in the Random Forest file, it was determined that 60 was the best number of trees for the Random Forest classifier, so all results are based on a Random Forest classifier consisting of 60 trees. The results are shown below:



Original RMSE without PCA : 3731.685760

PCA Components	*RMSE*	*Diff from Original*
11	5550.357231938524	+ 1,818.67
9	5529.226417204924	+ 1,797.54
7	6902.575618006062	+ 3,170.89

From these results, we can see that PCA did not improve the Random Forest classifier. More specifically that there is no amount of components for PCA that will sacrifice some accuracy / error in order to reduce the number of features. We speculate that it is most likely because there are not that many features to start.

In conclusion, our best machine learning algorithm was the Random Forest. After calculating how many trees would best compose a Random Forest which best predicts our target. In our case, after performing PCA, we decided to run our algorithm without PCA, since it increase RMSE drastically for all number of PCA components tested. Our algorithm was able to predict the Weekly Sales of Walmart within a margin of error 3.7k.

If you would like to view our code, please visit the link below:

https://github.com/jesus-r-mendoza/Walmart-Store-Sales-Forecasting