Predicting Future Sales Using Random Forest

Caleb Seely

Computer Science

CS 477 -Python for Machine Learning (Xian)

Spring 2021

*Introduction*

This project attempts to predict the number of sales for different products. The goal is to practice machine learning techniques on real data. This means data that may be inconsistent or incomplete. This is a regression problem requiring a regression model. In this case, a random forest algorithm was used.

*Approach*

One advantage to using a Random Forest algorithm is its ability to consider multiple features. In this case, the dataset has several categories that could be important to the model. A Random Forest can take each into consideration and determine different importance levels for each.

*Data*

In this case the data set is relatively small for a machine learning dataset. It starts in 2013 and continues through October of 2015. It contains five thousand one hundred different items, from forty-two different stores. It also contains a record of how many times an item was sold each day from each store. This is the most important data because it is what the model is trying to predict. This dataset is incomplete, there is not a record of every item at every store. There are several reasons the dataset would be incomplete, but it none the less provides an opportunity to work with real world data. Some items are only sold for part of the time and inconsistencies like this will have an effect on any model’s ability to predict future sales.

*Parameter Tuning*

There are a number of parameters that could be adjusted to change the model. Parameters such as the number of trees, the depth of each tree, minimum or maximum depth of each tree, and several others. There is no “proper” way to search for the ideal parameters to use, it is a game of guess and check. Two things were important factors in the process. Time was the first factor, how long did the model take to train with these parameters. Secondly was the score given, determined by R^2 or (1-u/v) where u is the residual sum of squares and v is the total sum of squares.

I found that it was best to use ten trees. This provided a model in less than a minute while still producing a score of about 93%. Anything lower than ten would start to have a significant impact on the score. Increasing the number of trees above ten would start to have an impact on time while not significantly effecting the score. Playing with other parameters did not have any significant impact on the score or time.

*Results*

The dataset has a lot of different data across a range of categories. So, although the model does attempt to predict the total sales for each product at each store for a whole month, it is easier to look at total sales across the board. By looking at the sum of all sales over time we can better see trends and what the model is doing. The following graphs will compare the testing data and trends to what the model has predicted.

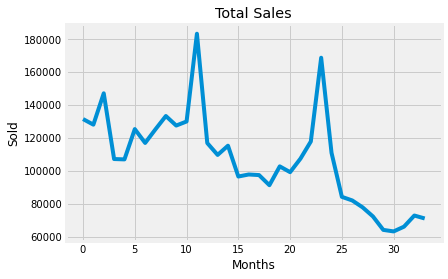


Figure 1: The total number of sales made each month

Here is the total number of sales of every product over the thirty-four-month time. The number of sales does trend downward, but it should be noted that the image distorts this because the bottom is not zero, but sixty thousand items sold. There is also a clear seasonal trend. Since the data starts tracking in January, every twelve months is a December, a time where holidays are known to have a positive affect on product sales. This is evident in the two large spikes going into the twelfth and twenty fourth months, respectively.

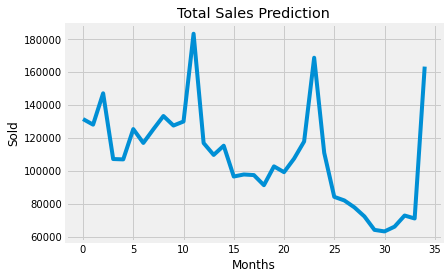
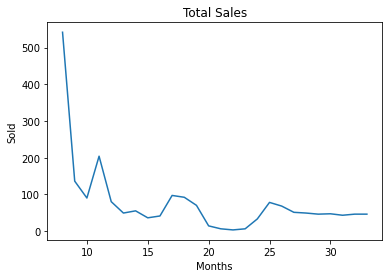


Figure 2: The total number of sales each month including the model's prediction

This graph is largely the same as the previous but with an additional month of data. This last month is using the data predicted by the model. Here there is a clear spike that follows the previous twelve-month patter seen earlier. This final spike is slightly lower than the other two as there is a general negative trend to the data.

This does provide support that the model is working, from a general perspective. Taking a closer look at individual items will provide insight as to how this spike was formed.



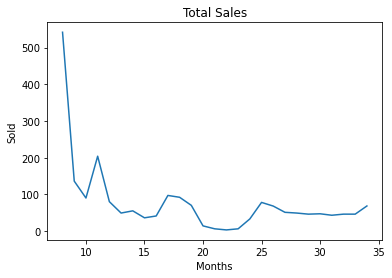
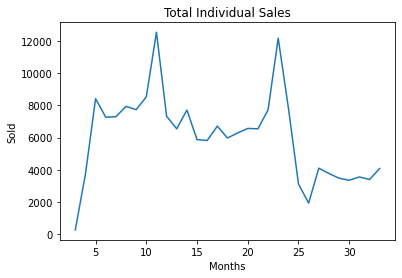


Figure 3: ID 4233

These two graphs look at the total number of sales for a single item over time. On the left is before the prediction and on the right is after the model’s prediction. The important part is the final months, where the model shows a slight increase in sales.



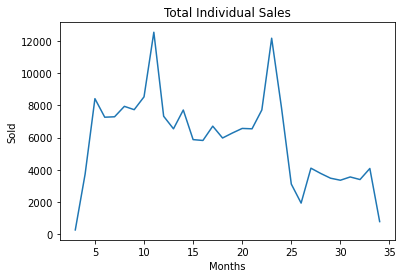
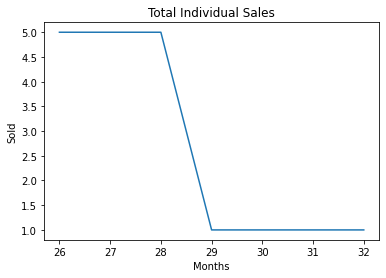


Figure 4: ID 20949

Here is another example of individual sales of an item over time. This graph looks drastically different from the other and the model compensates for this behavior by making a different prediction.

There are thousands of these graphs, each representing a unique item. As discussed earlier, the dataset is incomplete and those whole can become obvious and problamatic for some items.



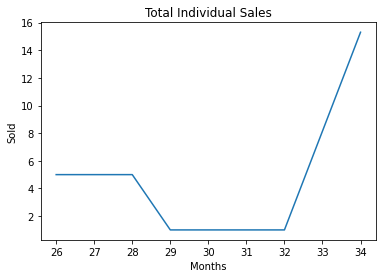


Figure 5: ID 6371

Here is just one example where the model breaks down. Notice the months on the bottom. There is significantly less data to work with compared to the other examples.

It is clear this model has problems scattered about. Those inconsistencies are likely why the Total Sales Prediction graph looks slightly off. To see what the Total Sales Prediction graph would possibly look like with a complete dataset, we can use a tool called Prophet. This will forecast the total number of sales by combining all the individual sales data over time. The difference between Prophet and the random forest model is Prophet is not predicting individual item sales. It uses every sale across every item to forecasts just the total number of sales for the following month.

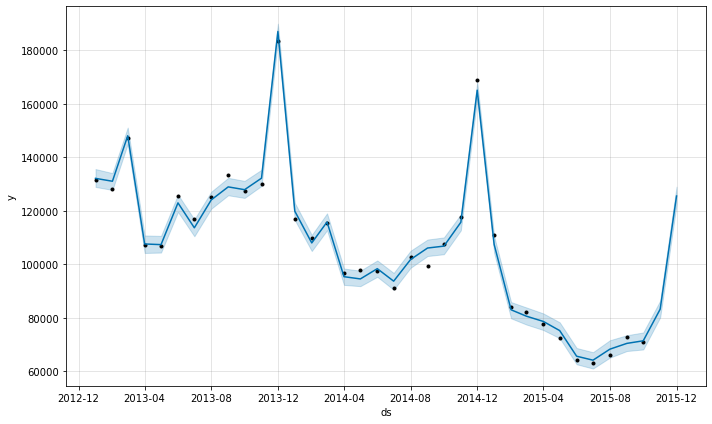


Figure 6: Prophet forecasting total number of sales for one month

This graph is similar to what the random forest model predicted. The spike seen here is significantly lower than what the model came up with. This suggest that with more data points the random forest model would have an even lower spike, something similar to what is seen in figure 6.

*Sources*

* <https://www.kaggle.com/c/competitive-data-science-predict-future-sales/data>
* <https://www.kaggle.com/jagangupta/time-series-basics-exploring-traditional-ts>
* <https://www.kaggle.com/sarvajna/random-forest-with-variable-importance/comments>
* <https://pythondata.com/forecasting-time-series-data-with-prophet-part-1/>
* <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>
* <https://facebook.github.io/prophet/docs/quick_start.html#python-api>