Kaggle Competition: Walmart Weekly Sales Forecasting

William West

April 30, 2014

1 Introduction

Kaggle is a website that hosts data mining competitions. Typically, companies will organize a competition on Kaggle by offering a sample dataset, a well-defined problem, and some incentive for competing, whether it be some monetary prize, internship, or job. Data Scientists can then compete with one another for the best performance, sometimes collaborating with one another and learning new things in the process.

The Walmart Weekly Sales Forecasting competition asks competitors to predict the weekly sales for a set of store/department pairs in 2013, given the weekly sales from 2010-2012. Competitors are not permitted to use any outside data source; only data provided through the competition may be used.

1.1 Dataset Description

The dataset consists of four data files: *features.csv*, *train.csv*, *test.csv*, and *stores.csv*.

- The *train.csv* and *test.csv* files contain the weekly sales for each (store, department, date) triple. Note that the test file contains null entries for the weekly sales column, as expected
- The *features.csv* file contains the temperature, fuel price, markdowns, CPI, unemployment rate, and a holiday indicator for each (store, department, date) triple
- The *stores.csv* file contains the size and type of each store

We combine all files into two distinct datasets–the training set and test set. Each set contains all features and the weekly sales for each (store, department, date) triple. We will assume that these files are combined for the remainder of this report.

The training data contains weekly sales from February 2010 to October 2012, and the test data contains weekly sales from November 2012 to July 2013.

1.2 Task

The task for this competition is to predict the weekly sales in 2013 for several stores/departments. While the evaluation method is not explicitly released, we are told that greater importance is given to weeks in which holidays occur.

2 Background: Forecasting and Time Series Analysis

A time series is simply a group of measurements which are taken in a non-random order. For example, the set of data captured by a temperature sensor is most likely taken at regular intervals, whether these be every hour, minute, or second. We can assume that each measurement is related to all other measurements by some linear or (most often) non-linear function. Given a measurement of 77 °F, for example, the two surrounding measurements (taken before and after, respectively) have a high probability of being close to 77 °F. This is due to a characteristic of time series data called *seasonality*, which we discuss more below.

2.1 Seasonality

Seasonality is used to describe the way in which a time series goes through cycles as time goes on. For example, consider measurements of monthly water consumption in New Jersey over ten years. While in some states, water consumption may stay relatively constant year over year, New Jersey does not follow the same pattern. Since New Jersey experiences hot summer months, water consumption will be higher during those months, resulting in a time series that exhibits regular peaks during those months every year. Thus, a graph of the time series would show ten regularly-spaced peaks. This is considered the *seasonality* of the time series.

2.2 Trend

Using the same example of water consumption in New Jersey, consider how water consumption may increase or decrease *over time*. That is, over the entire 10 years, what is the *trend* of water consumption? Perhaps there are strict laws put into place 5 years into the time series that gradually limit the amount of water companies are permitted to use in a given year. In that case, we would see a gradual decline in water consumption over the last 5 years of the time series. This is considered the *trend* of the time series.

2.3 Forecasting: Pattern Recognition for Time Series

When dealing with time series data, we often want to use past time series data to predict what will happen in the future. This can be framed as a type of pattern recognition in which we look to recognize the patterns found within the trend and seasonality of a time series. By doing such, we can make sensible predictions of the future.

It is important to be realistic when presenting forecasting results. Forecasting is different from other types of pattern recognition in that often the *state of the world* changes drastically. Consider the classic pattern recognition example of building a recognition system for classifying types of fish. We can be reasonably certain that the types of fish we encounter will not drastically change. While anything is possible, we can say with high probability that our recognition system will remain accurate for the near future. However, forecasting problems often don't carry this same assumption.

By nature, the domain spaces where forecasting is used are unpredictable–questions about weather patters, sensor data, economics, etc. can vary drastically with small or large changes in environment. For example, perhaps most years, the week before Christmas brings drastically high profits for grocery stores. As such, a forecasting system built to predict weekly grocery store sales predicts very high sales for the week before Christmas next year. What happens if there is a massive blizzard that week? Our prediction will be drastically incorrect. It is vital to keep in mind the dramatic variation that can occur between predicted values and actual values in forecasting problems.

3 Data Exploration

First, to get an idea of what the time series looks like, we average the sales across all stores and departments for each week. The graph is shown in Figure 1. We see that there are significant peaks around the end of each year, one during November (Black Friday) and one during December (Christmas). The seasonality of the time series is evident, and it is also clear that there is a general downward trend between the peaks in 2011 and 2012. Shoppers spent less during the weeks before Christmas in 2012 than in 2011.

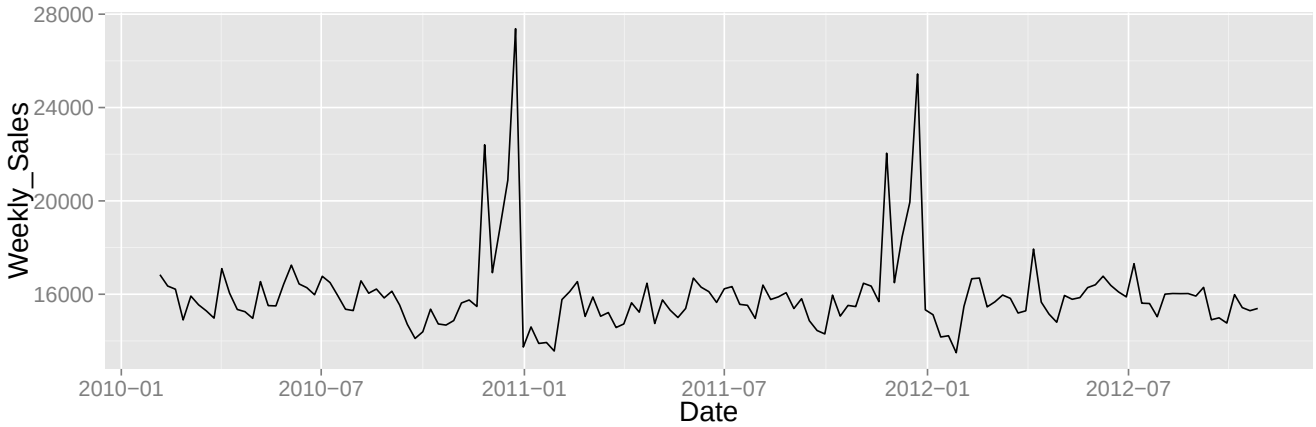

Figure 1: Line graph of time series

3.1 Features

To get an idea of how each of the predictor variables (features) interacts with one another, we look at a scatterplot matrix capturing the correlation between variables, and the distribution of each variable.

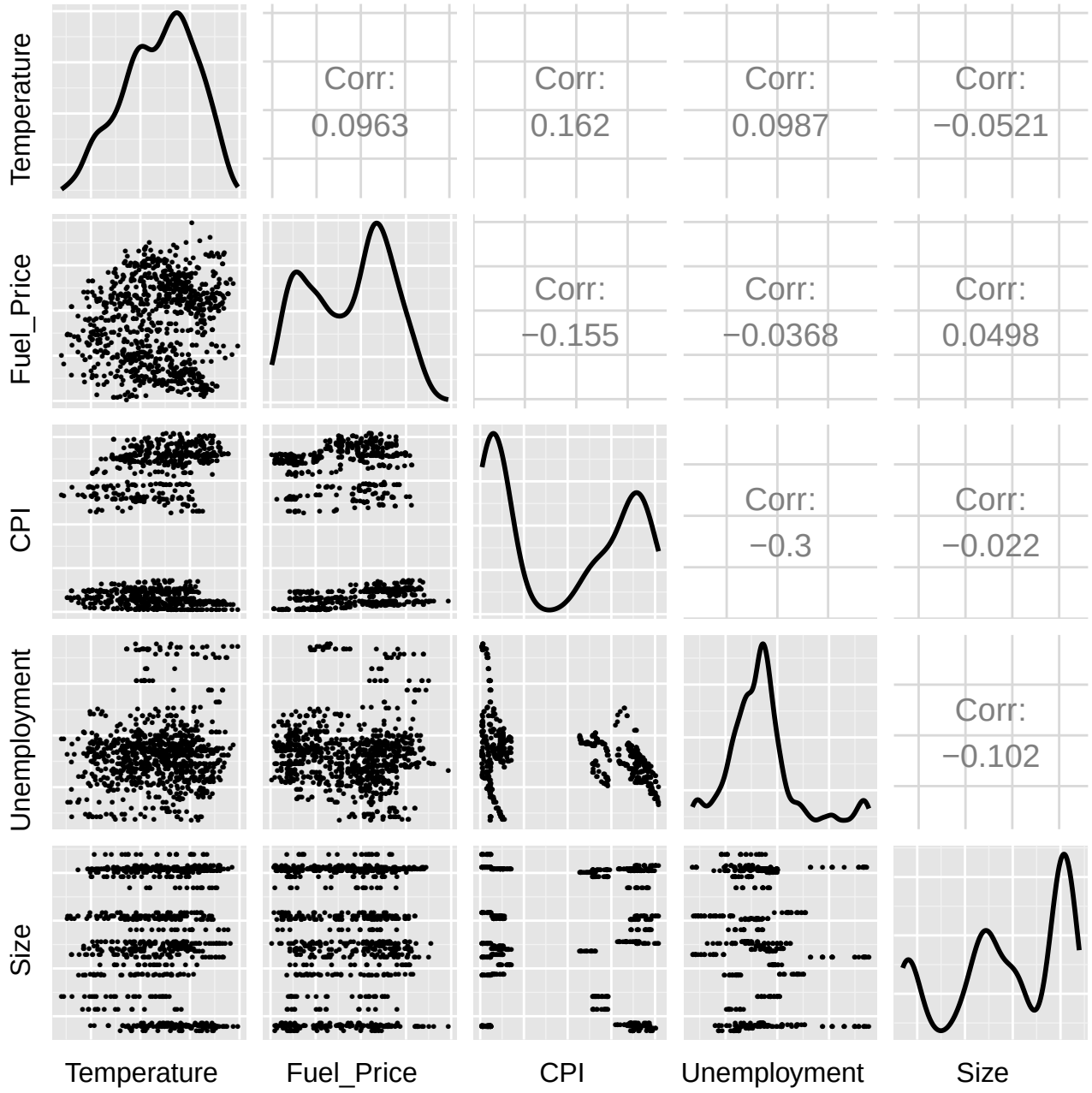

Figure 2: Scatterplot matrix: random sample of feature values. Upper boxes show correlation values between features, diagonal boxes show the distributions, and lower boxes are scatter plots.

We can see that there is a very slight correlation between fuel price and temperature, and a slightly negative correlation between unemployment and CPI.

3.2 Holidays

In order to better visualize how holidays affect sales figures, we choose a random store and look at the time series, highlighting the weeks in which there is a holiday. This is shown in Figure 3. We can see that often peaks can be found in the sales of the surrounding weeks, but not necessarily in the holiday week itself. This intuitively makes sense: while the week before Christmas may bring large sales numbers, the week *of* Christmas is more likely to bring lower sales numbers, as people are traveling and spending time with family.

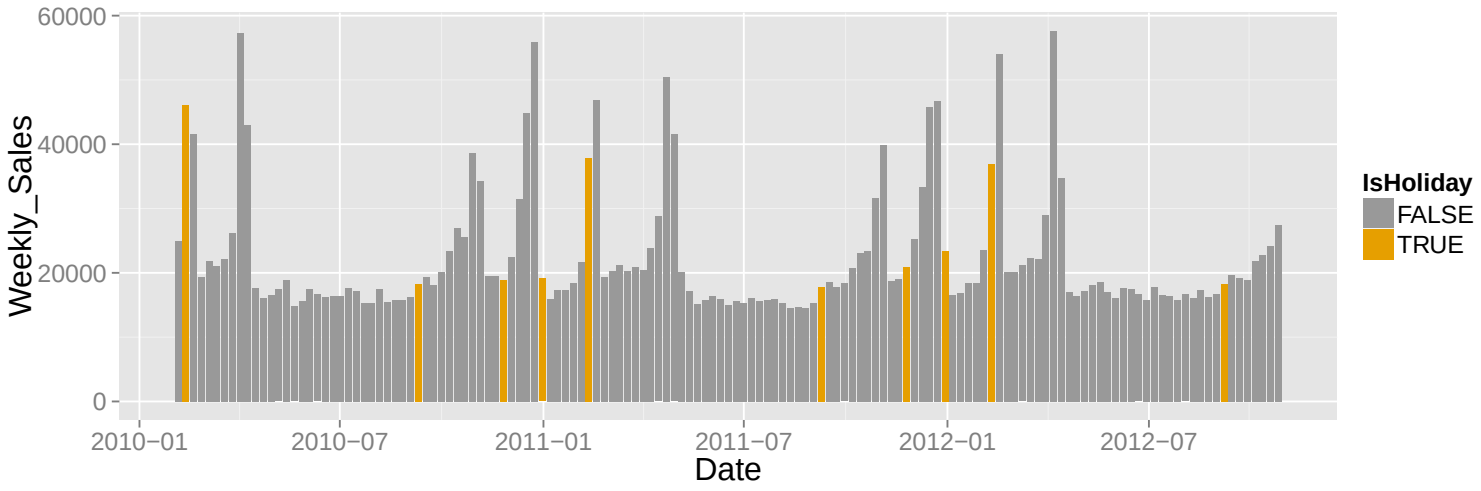

Figure 3: Weekly Sales of random store, highlighting weeks marked as holidays

Though Figure 3 gives us a good idea of how weekly sales are distributed for a random store, it would be better to see how holidays impact sales over the entire dataset. Figure 4 shows the average weekly sales across all stores and departments, again highlighting the weeks marked as holidays. The absence of the peaks we saw for the random store in Figure 3 is an indication that these peaks may only occur for specific departments or stores. Thus, we must take the store number and department number into account when forecasting, or our predictions will be too general.

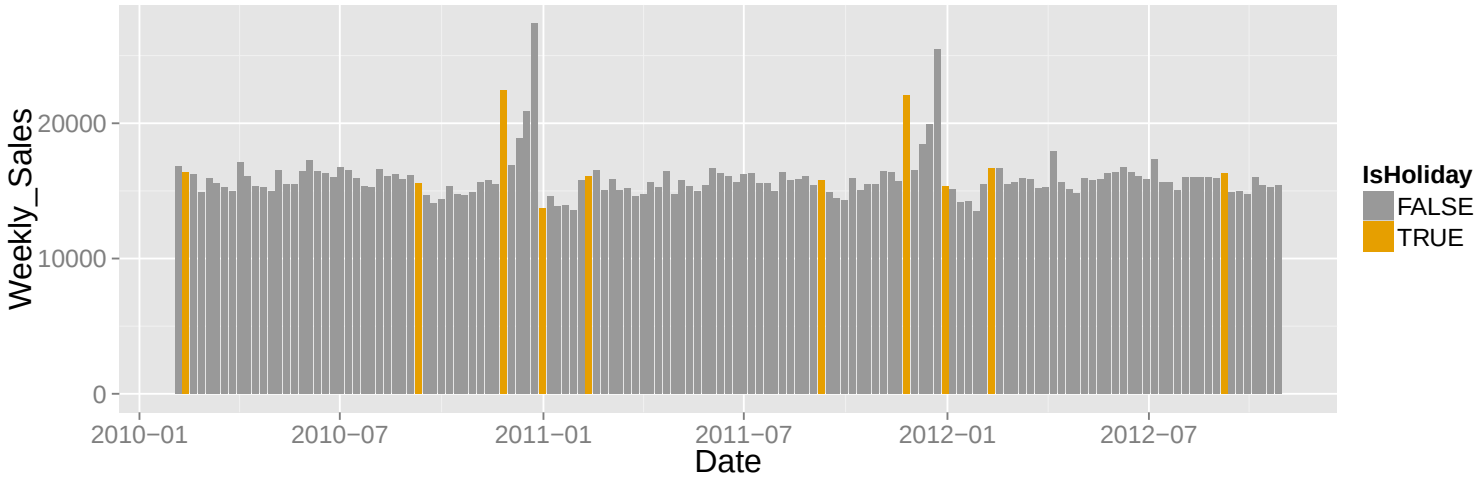

Figure 4: Average weekly sales across all stores/departments, highlighting weeks marked as holidays

4 Feature Extraction

4.1 Previous Year's Sales

Intuitively, it makes sense that a good predictor for weekly sales in 2013 are the weekly sales that we saw in 2012. Thus, we create a submission that, for each testing instance, simply predicts that the weekly sales will be equal to the weekly sales from the year before. To do this, we assign a month_id for each week in 2011/2012, and 2012/2013 from 1 to 52, and then simply perform a merge between the month_ids in the training set vs the test set.

4.2 Smoothed Sales

While the previous year's sales proved to be a good predictor, they don't take into account that the weeks between two years occur at different intervals. That is, June 1st may occur on a Tuesday one year, and a Friday another year. Since each measurement consists of a week from Saturday - Friday, the week containing June 1st in one year would contain many

different days than the week containing June 1st in another year. In order to mitigate this problem of shifting intervals, we choose to generate smoothed sales predictions that are an average of the weeks *surrounding* the previous year's week.

For each week in the test set, we extract the date of Friday in that week. We then subtract a year, obtaining the same date but a year before. Then, we calculate which day of the week that date is on, and find the date of the previous Friday and the next Friday. The measurements for those two weeks are then averaged and used as the previous year's sales predictor. We found this performance obtained a better score than the simple previous year's sales model.

4.3 Other Predictors

The other features provided by the dataset include: temperature, consumer price index (CPI), unemployment rate, fuel price, markdowns, store size, and store type. For each numerical feature, we scale the feature to be between 0 and 1 using libSVM's scaling library [1]. For each categorical variable, we use dummy coding to represent each n-category feature by $n-1$ indicator features.

5 Modeling

For modeling, we use all features given in the provided dataset. We test using a factorization machine model, which can be used for regression or classification [2]. This model decomposes the feature space into a set of latent factors, which capture the interactions between each of the feature variables, including categorical variables. That is, it will be able to capture the interaction between the various features and the store/department type. The equation is given below:

$$\hat{y}(\mathbf{x}) := w_0 + \sum_{j=1}^{p} w_j x_j + \sum_{j=1}^{p} \sum_{j'=j+1}^{p} x_j x_{j'} \sum_{f=1}^{k} v_{j,f}, v_{j'}, f \tag{1}$$

6 Evaluation

Surprisingly, the best predictor was the smoothed last year's sales. When using the factorization machine, we found that our performance gradually decreased as we added additional features. It is likely that more statistical modeling is necessary to extract meaning from these features, such as manually correcting for department-specific trends. While this is trivial to calculate, obtaining good performance through manual modeling is outside of the scope of this project, though we hope to attempt to add this before the final contest deadline.

References

[1] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[2] Steffen Rendle. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 995–1000. IEEE, 2010.