Final Project: Forecasting Walmart Store Sales

Vidhan Bhatt (vpb24) and Steven Jaroslawski (sj393)

December 5, 2016

Problem Description

Walmart, an American multinational retail corporation, hired us as consultants to forecast future sales by department for its stores. Walmart is interested in knowing weekly sales from November 2, 2012 to July 26, 2013 for 45 of its stores spread across multiple regions in the United States. Walmart executives want to know how much inventory to carry by department within a given store, at a given time. Store managers want to determine optimal employee shift schedules in order to satisfy demand. This is a valuable problem to solve: accurate predictions will help Walmart allocate resources efficiently and improve its bottom line.

There are a few challenges associated with this project. Predicting retail sales is tricky because there are many potential predictors: external variables like inflation and unemployment can heavily affect sales. Not only is a department's sales history relevant, but it is also important to consider which departments and stores are similar to one another. This interplay, coupled with the chaos inherent in hundreds of thousands of data points, makes this a particularly hairy problem with no obvious solution.

Description of Data

We used data from Kaggle, which consisted of four files: stores.csv (stores data) features.csv (features data), train.csv (training data), and test.csv (test data). The stores data contains one row for each of the 45 stores, and two columns for each store: the type of store, classified into three buckets (which are presumably Walmart Supercenters, Walmart Discount Stores, and Walmart Neighborhood Markets, the company's three most popular store types), and the square footage of each store.

The features data contains macroeconomic indicators for the region in which each store is located, for each week from February 5, 2010 to November 01, 2012. These indicators are the average temperature, average fuel price, average consumer price index, which is a measure of inflation, and average unemployment rate. Additionally, this file contains markdown data for five promotions that Walmart ran. This data is anonymized, only available after November 2011, and only in select stores each week. The features data also has a boolean column indicating whether a week is a Walmart-designated holiday week or not. Walmart designates four weeks each year as holiday weeks: the week of the Super Bowl, the week of Labor Day, the week of Thanksgiving, and the week of Christmas.

The training data contains weekly sales data, in dollars, for each department (numbered 1-99) within each store (numbered 1-45), and a boolean column indicating whether each week is a Walmart-designated holiday week not. Lastly, the test data contains all the same information as the training data except for weekly sales (which we will predict) for every week from November 2, 2012 to July 26, 2013. We performed a simple SQL join to add the columns for the four aforementioned

macroeconomics variables from the features data to both the training and test data, matching by date in each row.

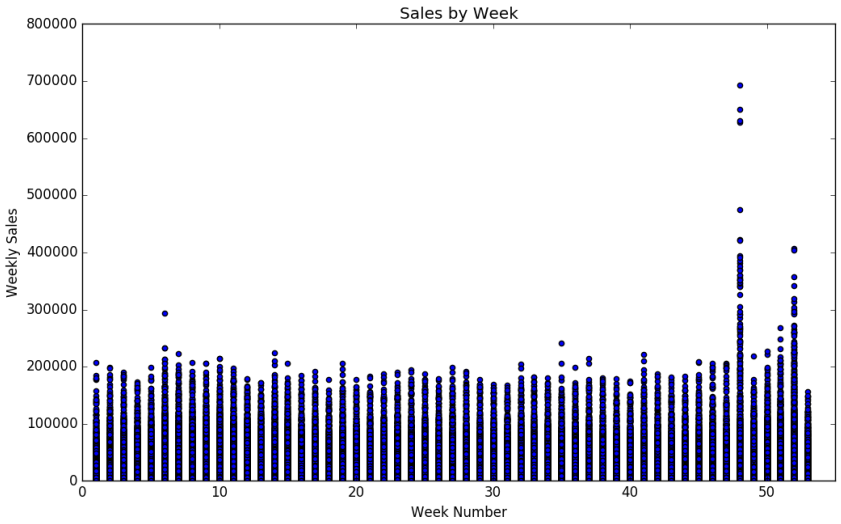
Because we did do not have access to the weekly sales in the test set, we created a validation set from the training set in order to check multiple hypotheses. We partitioned the last 20% of the provided training set as our validation set (starting from April 20, 2012), and the remainder (first 80%, up to April 13, 2012) as our training set.[1] Our training set had 333738 rows and 10 columns.

Data Exploration and Transformation

Given the overwhelming size of our training data, our first step was to perform some transformations to organize our data into a more intuitive and manageable format. It was clear to us that the historical sales in a certain department within a certain store would be the most predictive of future sales for that department in that store. However, we did not want to restrict our scope to just one department in one store for each prediction as this would result in a loss of a lot of valuable data. On the other hand, regressing on store number and department number does not make much sense either because the department and store numbers are all nominal values. Store 3 is not expected to have thrice the sales as store 1 simply because its store number is thrice that of store 1. Department 40 is not necessarily more similar to department 41 than it is to department 1. We needed to impose some sense of similarity between stores and departments, and using store/department number had little affect on similarity.

As for the time, we decided to consider time only by the week number in the year. We disregarded the year in which a sale took place and only focused on week of the year because we are only working with 2 years worth of data, so trends from year to year cannot be accurately determined. Further, we noticed that in 2010 and 2011, the first week was week 2 and holidays fell on weeks 7, 37, 48 and 53. However, in 2012 and 2013, the first week was week 1 and holidays fell on weeks 6, 36, 47 and 52. Therefore, we accounted for this by adding 1 to Excel's weeknum() function in 2012 and 2013 so that the weeks in every year was aligned. Our initial thought was to treat week number as a time series, and build an auto-regressive model. However, AR models are ineffective when predicting more than a few time periods in advance, and we are tasked with predicting sales for weeks 9 months in the future. And while we do expect sales from week 1 to be more related to sales in week 2 than compared to sales in week 30, there could be a deeper, more non-obvious relationship among sales across various weeks. Therefore, we felt it was necessary to consider each week of the year and its impact on sales across all stores and departments.

[1] For clarity, all references to training and validation data from this point onward correspond to *our modified* training and validation data sets following the split from the given training set



And the visualization of sales by week shows that it was a noisy data set that would be difficult to conventionally regress on. From the graph above, we could not develop much intuition as to why a linear or polynomial model would fit the data. One trend we did notice is that sales jumped during some holiday weeks, which we decided to model in the Macroeconomic Modeling section.

As a result, we created three matrices from our training set:

1. store_dept/time("s_dt"): The rows of this matrix are the store numbers. The columns of this matrix are deptartment/time pairs, meaning there is a column for each department each week. The values in each cell are the sales for the store corresponding to that row and the department/time corresponding to that column. This matrix has dimensions 45 by 8910.

2. dept_store/time("d_st"): The rows of this matrix are the department numbers. The columns of this matrix are store/time pairs. There is a column for each store at each time period. The values in each cell are the sales for a department, for each store at each time. This matrix has dimensions 81 by 5175.

3. time_store/dept("t_sd"): The rows of this matrix are week numbers, ranging from 2-53, in ascending order (the weeks range from 2-53, not 1-52, because that is what Excel's weeknum() function returned given the dates in the training data). The columns of this matrix are store/deptartment pairs. There is a column for each department within each store. The values in each cell are the sales for a week of the year, for each department within each store. This matrix has dimensions 52 by 3313.

These three matrices were then reduced in dimension to induce a sense of similarity, which will be explained in greater detail in the Store, Department, Time-Specific Modeling section.

We quickly noticed that many values in these matrices were missing. We conducted some basic analysis to determine if there was any discernible pattern to the distribution of NAs. By comparing the number of NAs in d_st within each department for all stores, we learned that there were 18 departments that had missing values for every store at every time between February 5, 2010 and April 13, 2012. Consequently, we removed all data corresponding to these departments because it was clear that they did not operate as functioning departments within any store.

The rest of the NA values were more difficult to deal with in one fell swoop. Some were likely data entry errors, where sales occurred but were entered incorrectly. Others could be attributed to situations where certain departments do not exist at certain times. For example, a Christmas-related department may only be active for several weeks in November in December, and disappear for the other months. Some departments also did not exist in certain stores. For example, a hunting department may not exist in stores located near urban centers where hunting is not predominant. So these missing values could in fact contain some valuable information about sales in certain stores at certain times.

There were also a few negative weekly sales values. We were not convinced that all negative values were bad data, or even that any negative values were erroneous. For example, it is conceivable that a low-volume department had negative net sales if it registered more returns than sales in a week, or if Walmart recognizes purchases via gift-card as a negative. Further, negative values accounted for less than 0.1% of total data, so we decided to include these values because ultimately they would have a negligible impact.

In order to model the effect of macroeconomic variables, we created another matrix, store total, which sums sales across all departments for each store at each time. We did this because the values of all macro indicators do not vary across department, but only vary by geography (hence, stores, which are in different regions) and time. Again, we made use of SQL join and group by functions to deftly consolidate sales of all departments within each store.

Analysis

Overview

We wanted our model to utilize all the columns of our training to maximize the accuracy of our prediction. It is entirely possible that some of these features may not predict sales at all, but we wanted to build a model that works in case they do. Our thought process was as follows: we were tasked with predicting sales for each store at each department at a given time, so naturally, our model should at least be functions of of store, department, and time. The transformations above create data in this format, but all three matrices have thousands of columns. An ordinary regression would yield thousands of coefficients. To reduce dimensionality and combat the sparsity of the s_dt, d_st, and t_sd matrices, we utilized the *LowRankModels* package in Julia, created by Professor Udell. We conducted principal components analysis (PCA), sparse PCA, and matrix completion methods using this package.

We also wanted to account for the impact of the given macro variables (temperature, CPI, fuel price, unemployment rate, and holiday week). Unlike the s_dt, d_st, and t_sd matrices, the macroeconomic data was not missing for any of the weeks in our training, validation, and tests sets.[2] And because there the predictors only span 5 columns, we did not have to look to specialized methods for high dimensioned or sparse data outside of polynomial regression.

Our final model combined the results from both approaches. We created a matrix that combined the results of the dimension reduction and the columns of macroeconomic data, and regressed on all columns to determine optimal coefficients. We had six different types of models, explained in greater detail in the Store, Department, Time-Specific Modeling section, which we ran on our validation set to determine which produced the smallest error. We then applied the best model on our test set to

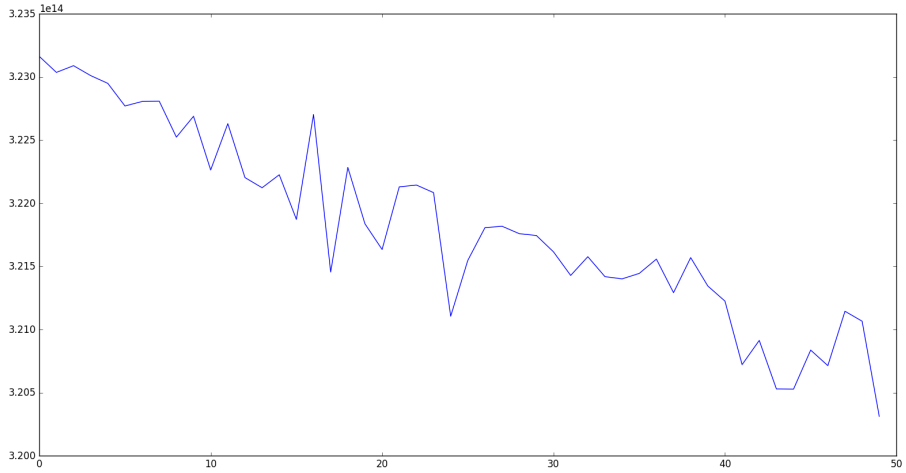
[2]This is not an unexpected phenomenon. Because this data is collected by government agencies and is of national importance, it is expected to reliable and dense, at least to a greater degree than Walmart sales data.

predict store sales.

Store, Department, Time-Specific Modeling

Once again, we knew that the store number, department number, and week of the year mattered, but the value was simply a label. To achieve some sense of similarity across stores, and across departments, and across times, we first created the s_dt, d_st, and t_sd matrices. Taking s_dt for example, if two stores often have similar sales in the same departments, those stores are likely to be similar to each other. If we think of a row in s_dt as a 8910-dimensional vector representing a store, then similar stores will be "close" to each other in this 8910-dimensional space. Similarly for d_st, if two different departments have similar sales numbers for many stores in many weeks then the departments probably sell similar items. The same goes for the week of the year. However, 8910 dimensions (5175 dimensions for d_st and 3313 dimensions for t_sd) is much too high for feasible computational speed. We needed to reduce the dimension of these three matrices to be able to regress on their similarity.

To prevent under-fitting, we wanted to have sufficient hypotheses to try on the validation set. We therefore employed multiple methods of reducing dimensionality learned from class so that we had many options to validate. PCA was the first technique that we learned, and is the oldest and most notable unsupervised learning technique. Because it is such a fundamental and widespread technique, we figured it would be a useful method to employ. Further, given that our matrices are relatively sparse, we decided that sparse PCA and matrix completion would work well for our data. Taking s_dt for example, the X matrices generated from these techniques would consist of 45 rows. Each row would represent each store in a lower dimension, which would make the resulting X matrices easier to work with than the bulky s_dt matrix. When deciding which k to choose for the various methods, we plotted how the objective value of the glrm() function within the *LowRankModels* package changed with various values for k. One of those plots, for PCA on d_st, is shown below with the x axis being the value for k and the y axis being the objective value:



As expected, there is a clear downward trend as k increases. This is because as k increases, the number of principal components retained from PCA (or matrix completion or sparse PCA) increases, so accuracy will improve. This trend was true for all methods on all three of the data sets. Because we wanted the highest accuracy we could achieve, and because 81 dimensions (45 dimensions for

s_dt and 52 dimensions for t_sd) is not all that high, we decided to choose the highest k possible.

For each of the three methods, each row of the resulting X vector represents the store (taking s_dt for example) in a relatively low dimensional space. Because the goal of PCA was to check the "closeness" of each store, we figured that local smoothing would help improve performance. The local smoothing transformation creates a column for every store, with each row in the X matrix already corresponding to a store. If store i and store j are "closer", local smoothing will give element (i,j) a value closer to 1, and if the stores are distant and dissimilar, (i,j) will have a value closer to 0. Therefore, in the end we had 6 different models to choose from: PCA, sparse PCA, matrix completion, PCA followed by local smoothing, sparse PCA followed by local smoothing, and matrix completion followed by local smoothing. The resulting X matrices all had 178 columns: the store number was translated into a vector of length 45, the department number became a vector of length 81, and the week of the year was turned into a vector of length 52. We then added more columns from the macroeconomic model that is described in the next section.

Macroeconomic Modeling

To determine which degree regression would best fit sales, we emulated the bias variance analysis from Homework 3 for linear, quadratic, and cubic models. Our methodology was to sample 1000 random rows from the store total data set and fit linear, quadratic and cubic models to this sample of data. We repeated the sampling 1000 times and computed bias and variance statistics, which when added together resulted in total error. The total error of the linear model was 4.23×10^{13}, the total error of the quadratic model was 3.17×10^{11}, and the total error of the cubic model was 3.15×10^{11}.

From this, we concluded that the cubic model fits the data best because it has the smallest error. It is worth mentioning that we did not expect the errors to be low — we do not expect this macro-driven approach to accurately model sales *on its own*. Instead, our goal from this process was to determine the model order with the smallest relative error. Furthermore, the reduction in total error from the quadratic to the cubic model is quite small, but we felt as though it provided increased flexibility that may have a bigger impact in our combined model. Our cubic model resulted in a matrix with 14 columns: three for each of the four indicators (CPI, unemployment, fuel price, and temperature) to account for a linear, quadratic, and cubic term for each, one for whether the week is a holiday week or not, and one for the cubic model offset.

Results and Confidence

After running each of the six resulting models on the validation set, it was determined that the model that included PCA followed by local smoothing achieved the lowest error. We used the error metric that Kaggle created for the competition: weighted mean absolute error. Mean absolute error is simply the average of the absolute error, or the absolute value of the difference between the actual value and the predicted value. Because holiday week sales make up a large component of Walmart's revenue, predicting holiday week sales was weighted 5 times as much as predicting non-holiday week sales. PCA followed by local smoothing had a weighted mean absolute error of 8030.59. All of the models that we tested on the validation set had very similar errors, with the PCA local smoothing model having the lowest error by less than 100.

Looking at our w vector, it appears that CPI and unemployment had a very negative correlation with weekly sales. This makes intuitive sense — as unemployment decreases more people have more money to spend. As CPI (inflation) decreases, the dollar is "worth more" so in essence people have more money. Also as expected, the w coefficient for isHoliday is a large positive number, as holidays

mean more shopping and more revenue for Walmart.

In the end, the resulting error was slightly higher than we had hoped. We suspect that the weighting of holiday weeks may have increased our error. The isHoliday column was just one out of about 200 in our X matrix, so it was probably not given as much importance as the Kaggle error metric penalized it for. Another area for improvement could be to devise an alternate means to analyze the time series data that go beyond the scope of tools we learned in the course. While we are confident in our process and understand the procedure, the higher than expected error value precludes us from submitting our recommendation to Walmart with a great deal of conviction. Perhaps our modeling approach was too complex. While we took several steps to prevent over-fitting, including using various regularizers and fitting several models on a validation set, it is possible that a less comprehensive model would have generalized better. Because this is a very difficult problem, our goal was to combine many features to capture as much information from the data as possible. If Walmart is interested in continuing the forecasting project, we would like to continue to explore the problem in order to refine our model without simplifying it.