Project 2: Walmart Store Sales Forecasting

xxxx (netID)

November 8, 2021

1. Introduction

This supervised learning study is based on a time series dataset consisting of weekly sales data from 45 Walmart stores from February, 2010 through October, 2012. Each of the more than 400,000 observations used as test and/or train data consists of a date (week), store ID number, department ID number, sale volume (presumed to be in U.S. dollars), and a boolean value indicating whether or not the particular week is a holiday. The goal of our study is to build a model that uses past sales data to accurately predict future sales data for future $(store,\ department,\ week)$ triplets.

The initial training set consists of data spanning 12 months: February, 2010 – February, 2011. Predictions are made for two month time blocks (beginning with March – April of 2011). After a set of predictions is made and compared against a two-month test dataset, the test data is then incorporated into the train data which is then used to make predictions for the subsequent two months. This process is repeated ten times (for a total of 20 months worth of predictions). The accuracy of each prediction is measured using the weighted mean absolute error (WMAE) formula:

$$WMAE = \frac{1}{\Sigma w_i}\sum_{i=1}^{n} w_i\left|y_i - \hat{y}_i\right|$$

is the number of $(store,\ department,\ week)$ triplets, $\hat{y}_i$ and y_i are the respective predicted and actual sales, and w_i is a weight factor equal to 5 for holiday weeks, and 1 for non-holiday weeks. This WMAE formula is the same one that was used for a 2013 Kaggle competition based on a more elaborate version of the dataset. For the current assignment, an overall mean WMAE below 1610 is needed for maximum credit.

The model described in subsequent sections uses singular value decomposition (SVD) filtering with a dynamic method for determining the number of SVD components to use for the projection at each prediction fold. The SVD filtered data are then used to fit a linear regression model for each unique $department$ (using data across all stores for each department type). These linear regression models are used to calculate the predictions and achieve an overall mean WMAE below 1602. This approach draws heavily on the code for the general SVD plus linear regression provided in CampusWire Post #654. However, by dynamically tuning the SVD cutoff dimensionality, our approach achieves a lower WMAE than any values achieved by using the same number of SVD components for all folds (as was done in CampusWire #654).

2. Data Processing and Analysis

2.1 Software Tools and Packages

All pre-processing, modeling and predictions were performed using **R** programming language version 4.1.1 and the RStudio IDE. Prior to data processing and analysis, packages `lubridate` and `tidyverse` were explicitly loaded into the **`R_GlobalEnvironment`**. Due to package dependencies, the commands to load the above two packages caused RStudio to load additional packages such as `dplyr` and `tidyr`.

2.2 Raw Data Files

The set of raw data used for this study consisted of 12 .csv files. `train_ini.csv` contained the initial 12 months of training data, `test.csv` containing predictor values for 20 months of "future" test observations, and 10 `fold_xx.csv` files - each containing 2 months of predictor and response data for the test observations. Train_ini.csv was loaded into R as the initial train data. After each cycle of model building, predicting and

comparing against data from a particular `fold_xx.csv` file, the data from that `fold_xx.csv` file were added to the train dataframe for use in subsequent cycles.

2.3 Data Structures

As described in the assignment specifications, the test and train data frames were stored in the Global environment. When building and optimizing the model methods and parameters, a separate Project Environment object was created inside the Global Environment. A total of 10 Prediction Environments (one for each fold) were created, and references to these environments were stored as elements of a list in the Project Environment. Since the test code for the final performance evaluation iteratively makes predictions for one split at a time, the version of code submitted for the assignment uses a single Prediction Environment for each iteration and does not use a Project Environment.

2.4 Pre-processing

2.4.1 Singular Value Decomposition

Before calculating predictions for a particular fold, the training data were split by department ID, and each of these splits was reshaped ("pivoted wide") into a matrix with one row per store and one column per week of available data. Each of these department-specific matrices was centered by subtracting the mean of each row and then (if the matrix dimensionality was greater than some chosen cutoff) projected onto a parameter space with the cutoff dimensionality using SVD. This SVD-reduced matrix was then un-centered (by adding the row means back in) and then before being used to fit a regression model.

3. Model Description

3.1 Department-specific Regression

This SVD-reduced matrix from each department was used to use a department-specific regression model using the `lm.fit()` method. The coefficients from this model were then used to calculate department train predictions..

3.2 Tuning the number of SVD components

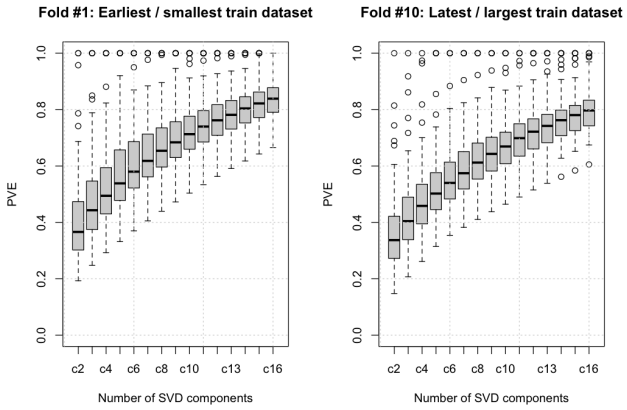
The decision to tune the SVD cutoff parameter (and not use the same number of SVD components for all folds) was based on the observation that the number of SVD components providing the lowest WMAE tended to increase with increasing size of the train dataset (which, for the current study, corresponded to folds at later times). As time progresses, and the data cloud becomes more dense / defined, the amount of "real" information contained in the higher order SVD components tends to increase.

For the current model, we assumed that the increase in the optimal cutoff value (due to gradual accumulation of signal information) would proceed linearly, so we included tuning parameters `filter_relax_rate` (i.e. slope) and `filter_relax_offset` (i.e. intercept) as parameters in our overall model. By guessing (and tuning) this effective rate of increase in the amount of useful information, we were able to obtain an overall mean WMAE of 1601.75. The best possible overall mean WMAE obtained when using the same component cutoff value (ncomp = 7) for all folds was ~1607. If we were to choose the SVD dimensionality cutoff for each fold after the fact (i.e. by calculating WMAE for many different components and then just choosing the best one for each fold), we could have achieved overall WMAE ~ 1595, but this approach would NOT have been predictive. By generalizing and modeling the behavior of how rapidly useful information accumulates, we are using an approach that can allow cross-validation of training data without ever seeing the test data response (but will still help make predictions about the test data).

The final values selected for the tuning parameters were `filter_relax_rate` = 0.014 and `filter_relax_offset` = -0.3. Since we assumed an increase in useful information with time, the units of `filter_relax_rate` are *dimensions per day*. While the computed dimensionality cutoff is computed as a decimal, it is rounded to the nearest integer for use in the SVD computation.

3.3 Proportion of Variance Explained

Attempts were made to use Proportion of Variance Explained in the train data as part of our dynamic SVD component tuning strategy. Although these efforts have been unsuccessful so far, further exploration of this path is recommended. Below are plots of the PVE distributions (of all department-specific models) for the first and final folds with SVD component cutoff values ranging from 2 to 16. For the same number of components, the 10th (larger) dataset exhibits slightly lower PVE than the first dataset.



4. Results

4.1 WMAE Values

The WMAE value for each of the 10 folds, as well as the overall mean are shown in the table below:

Fold	WMAE
1	1892.903
2	1364.558
3	1381.475
4	1527.280
5	2309.253
6	1636.942
7	1682.638
8	1390.588
9	1408.855
10	1423.010
Mean	**1601.750**

4.2 Runtime

The total processing time to model and calculate predictions for all 10 folds was 106 seconds on a MacBook Pro with 2.3 GHz processor cores and 32 GB RAM.

5. Discussion / Conclusions

The assumption that useful information increases linearly with time (used to tune the cutoff values for the number of SVD components) used here was likely an oversimplification, but still resulted in improved predictive power. Further and more sophisticated exploration of the interplay among dimensionality and signal vs. noise tradeoffs are certainly warranted.