CS598 PSL Project II Report

Team Members: xxx (netID), xxx (netID), xxx (netID)

Introduction

This project uses data from the Walmart Store Sales Forecasting Kaggle challenge, with the goal of predicting the future weekly sales by department in each store based on historical weekly sales data. The sales data are collected from 45 Walmart stores located in different regions with multiple departments within each store. In addition, selected holiday sales events are included in the dataset which makes the task of sales prediction more challenging. We trained linear models to predict future weekly sales, and evaluated its accuracy using a 10-fold split of the test set. Since the data are time series, when predicting a new fold, all previous folds are treated as training data. We used linear interpolation and truncated-SVD for preprocessing the training data, and post forecast adjustment to further improve prediction accuracy.

The evaluation metric used in this project is the Weighted Mean Absolute Error (WMAE), which can be expressed as $\texttt{WMAE}=\frac{1}{\Sigma w_i}\sum_{i=1}^{n} w_i\left|y_i-\hat{y}_i\right|$, where n is the number of rows, $\hat{y}_i$ is the predicted sales, y_i is the actual sales, w_i are weights ($w_i = 5$ if the week is a holiday week, 1 otherwise). Thanks to t-SVD denoise and a post forecast adjustment that accommodates for the discrepancy of the Christmas day between training and test set, our model is able to achieve a mean WMAE of 1578.89 over the 10 test folds, which is well below the target mean WMAE of 1610. In the end, a discussion of important learnings and interesting findings is presented, along with individual contributions of each team member.

Data Pre-processing and Feature Engineering

We can decompose the weekly sales prediction problem into multiple sub-problems, one for each department in each store. We do this by extracting unique pairs of (Store, Department) combinations that exist in both training and test sets. We also encode independent variables (Dates) as dummy variables to allow for easy interpretation and increase coefficient stability. The package `lubridate` provides a convenient way to convert date to week (`Wk`) and year (`Yr`). Representing individual weeks as different levels of a categorical variable `Wk` helps to capture seasonality and holiday events of the time series data. We keep `Yr` as a numerical variable instead of categorical to accommodate for unseen values (`2012`) in the test data and to avoid treating one of the `years` as the reference level.

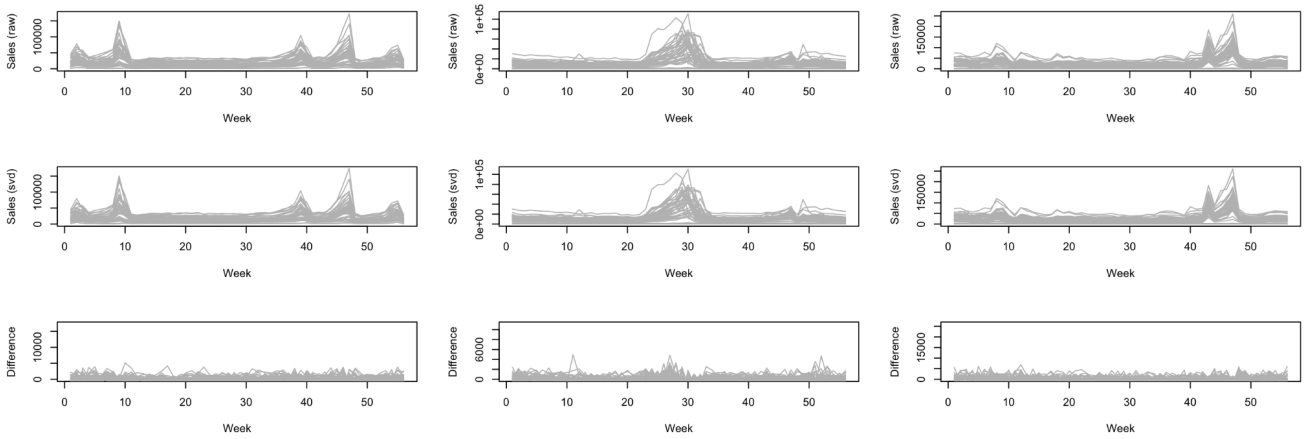

Figure 1. Original time series (left), t-SVD (middle) and difference (right) for different departments (1, 3, 5).

There are two major challenges when using the raw time series data to construct the design matrix for linear regression: missing data points in the time series and noisy time series data. Missing data will cause NA coefficients in the model, while noise in the data could lead to overfitting. The next section provides a detailed description of how we handle such challenges in the model building step, by the methods of linear interpolation and truncated Singular Value Decomposition (t-SVD). T-SVD is motivated by the observation that the sales patterns for the same department are similar across different stores. This effect is illustrated in Figure 1 above. The three columns correspond to department 1, 3 and 5, and the three rows correspond to the original data, t-SVD result with 8 components and their differences plotted with 1/10 scale of the original data. We can see that t-SVD removes mostly random noise.

Model Building and Hyper-parameter Tuning

Given the goal of predicting the future weekly sales given historical sales data, a natural option is to fit a linear model with `Weely_Sales` as response variable and feature variables derived from `Date`. As mentioned in the preprocessing section, to simplify the model, we are training multiple linear models, each for a specific (Store, Department) combination. A straightforward workflow would be to loop over each department and each store that has that department, extract and create the corresponding response (`Weekly_Sales`) and feature variables (`Wk` and `Yr`), perform linear regression and finally perform prediction for the `Weekly_Sales` for the test data. In practice, as pointed out by Peng Xu and the instructor's note, it is more efficient to first discover all unique pairs of (Store, Department) that exist in both the training and test data and use it to filter relevant training and test data. Then one can simply loop over all unique pairs of (Store, Department) and train a linear regression model for each pair.

The linear regression predicts `Weely_Sales` based on a categorical variable `Wk` with 52 levels, and a numerical variable `Yr`. In some cases, not all 52 weeks are available in the training data, which leads to NA model coefficients. In such situations, NA coefficients can be handled in the following three ways.

- **Linear interpolation on model coefficients**: We observe that by doing a linear interpolation of the coefficients, a reasonable improvement in prediction accuracy can be achieved. It can be cumbersome to recursively search for non-NA coefficients. We implement a version of linear interpolation by running nearest-neighbor interpolation (from one side) twice, once forward and once backward, and then taking the average of the two for interpolation.
- **Linear interpolation on design matrix**: The root cause for NA coefficients are missing values in the training data, which corresponds to unobserved weeks of sales for a particular store and department combination. Therefore, one could directly perform a nearest-neighbor or linear interpolation on the design matrix to replace all NA values.
- **Truncated SVD (t-SVD)**: As mentioned in the preprocessing step, a t-SVD workflow will handle denoising and missing data recovery at the same time. Let $X_{m\times n}$ represent the data matrix for each department, where m denotes the number of stores that has the department and n denotes the number of weeks. Before performing t-SVD on the data matrix, we first need to replace NA entries with reasonable values. We experimented with both zero-filling and linear interpolation, and found that linear interpolation only provides negligible improvements but takes longer to compute. The output from t-SVD (multiplied together) is a low-rank approximation to the data matrix, which eliminates the noise components that correspond to smaller singular values. The resulting data matrix is a smoothed version of the original data, with missing values recovered. The number of principal components (or largest singular values) to keep is a hyperparameter. We found that keeping 8 components leads to the best prediction accuracy.

Another major improvement on prediction accuracy is to account for the error introduced by the Christmas holiday and implementing a post forecast adjustment originally introduced by David Thaler. This only needs to be done for fold-5, which contains the entire December of 2011. The source of error is that Christmas occurs in different weeks in 2010 and 2011 (the 2012 data ends in October). For 2011, one needs to circularly shift a fraction (in our case, 1/7) of the weekly sales from weeks 48 through 52 into the next week, i.e. 48 → 49, 49 → 50, …, 52 → 48. To implement the shift trick, we first spread the predicted weekly sales over different dates (weeks), and use linear interpolation to fill in NA values. Next, among weeks 48 through 52, 1/7 of each week's sales are passed onto the next week in a circular fashion. This step alone is able to reduce the WMAE from 1608.36 to 1578.89.

Result

The computer system used for the experiments is a Macbook Pro with 2.3 GHz Quad-Core Intel Core i5 and 8GB Memory. The total run time for all the 10 folds, including both training and prediction time, is 97.4s. Note that this time also includes performing the shift trick for fold 5. The test WMAE for the SVD + LR model with the shift trick is reported below. Without the shift trick, fold 5 will have a WMAE of 2309.82, which results in a mean WMAE of 1608.36. Both cases satisfy the target accuracy.

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	Mean
1941.02	1362.54	1381.93	1526.80	2015.11	1635.72	1682.53	1399.31	1417.83	1426.07	1578.89

Discussion and Conclusion

This project aims to predict future weekly sales with historical sales data. Most challenges of this project are data preprocessing and feature engineering, which involves missing data interpolation, denoising and encoding of categorical variables. For each unique (Store, Department) pair, a linear model is trained for time series prediction. The resulting models satisfy the target prediction accuracy (WMAE). Below we summarize the challenges we met and important learnings we received from this project:

- The prediction task seemed complicated until we learned to decompose it into subtasks.
- For training a large number of linear models, lm.fit() tends to be more efficient than lm().
- For missing data, there are many ways to interpolate based on underlying assumptions. We fill meaningful values based on the individual context.
- T-SVD leverages the similarity among the same department in different stores to denoise the data. When preparing the data matrix for t-SVD, linear interpolation does not provide additional advantages over zero-filling. Our explanation is that although zero-filling introduces additional noise into the data, t-SVD has the effect of denoising that mitigates the error.
- The shift trick helps to further improve prediction accuracy. The lesson here is that to better generalize the model, it's important to identify discrepancies between training and test data.

Finally, we benefited a lot from numerous resources, including posts on Campuswire, office hours and Kaggle discussions. Most of our implementations are inspired by Instructor notes and David Thaler's post (https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting/discussion/8028).

Individual contributions:

- **xxx (netID)**: data pre-processing, model training, code development, report writing and final submission.
- **xxx (netID)**: data pre-processing, model training, code development, report writing.
- **xxx (netID)**: model training, hyper-parameter tuning, code verification, report writing.