Enhancing Walmart Sales Forecasting: A Comprehensive Analysis and Model Optimization

Hamid Razavi, John Fahim and Rajiv Kumar

Date: 12/06/2023

1. Introduction:

- Project Overview:
  - The purpose of the **Walmart Sales Forecasting** project is to create a predictive model to forecast weekly sales in Walmart stores using a dataset that contains sales from 2010 to 2012.
  - Key aspects of the project: data preprocessing, such as feature engineering and outlier detection using the Interquartile Range (IQR) method, Multiple Linear Regression (MLR), exploration of the influence of holidays on sales trends.
    Our contributions focused on refining these methods with data imputation for missing values, model selection via cross-validation, and hyperparameter optimization.
  - We also introduced more complex models, such as, gradient boosting machines and random forests, as well as time series analysis using the Holt-Winters model.
- The **dataset consists** of:

Sr. #	Features	Description
1.	Weekly Sales	The primary target variable, which represents the total sales per store on a weekly basis. Average sales around $1,046,965, ranging from $209,986 to $3,818,686.
2.	Store	Identifies each store, numbered from 1 to 45.
3.	Holiday Flag	Binary indicator (0 or 1) for holidays in a given week.
4.	Temperature	Average around 60.66°F, varying from -2.06°F to 100.14°F.
5.	Fuel Price	Mean value of approximately 3.36.
6.	Consumer Price Index (CPI):	Average of about 171.58.
7.	Unemployment:	Mean rate around 7.99%.

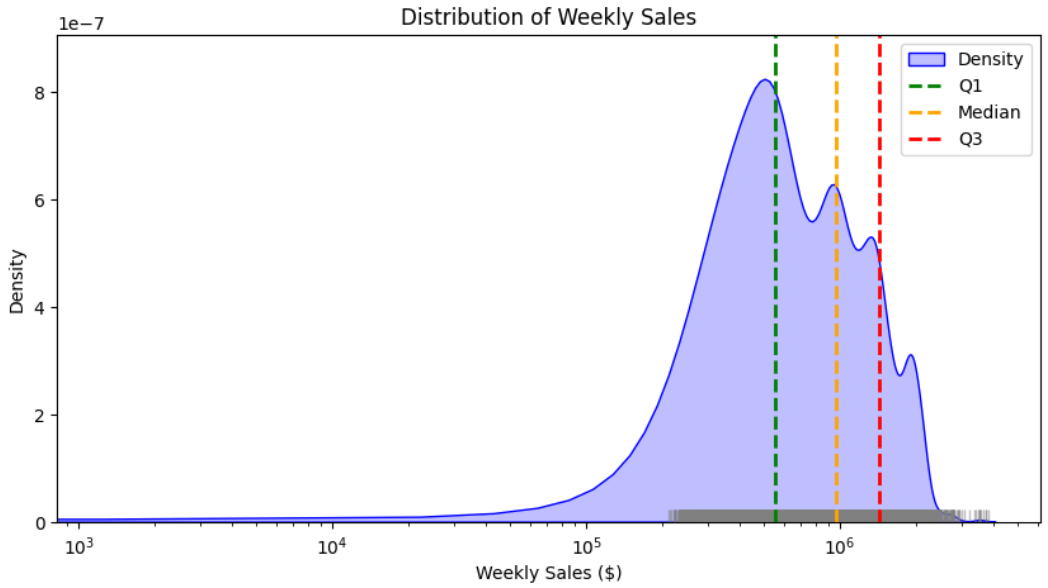
2. Work Reviewed and Analyzed:

- Yasser's Approach:
  - Reference: Walmart Sales Prediction - (Best ML Algorithms) by M Yasser
  - link : Walmart Sales Prediction - (Best ML Algorithms) | Kaggle
  - Yasser's approach involved feature engineering, scaling, and effective outlier removal using the Interquartile Range (IQR) method.
  - His unique contribution was the implementation of IQR for outlier detection and the application of Principal Component Analysis (PCA) for feature reduction.
- Ahmedov's Approach:
  - Walmart Sales Forecasting by Aslan Ahmedev
  - Link : https://www.kaggle.com/code/aslanahmedov/walmart-sales-forecasting
  - Ahmedov utilized the Exponential Smoothing model and evaluated its performance using the Weighted Mean Absolute Error (WMAE).
  - The unique approach was to focus on the impact of holidays on sales, which provided valuable insights into seasonal sales trends.

3. Proposed and Implementation:

- Data Preprocessing Enhancements including imputation:

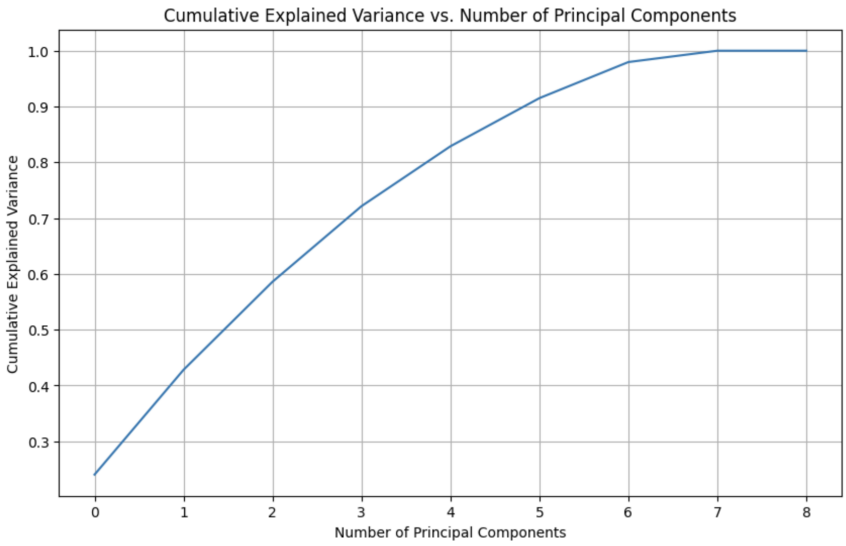
  We visualized 'Weekly Sales' using Seaborn and Matplotlib, finding a right-skewed distribution with many low sales and some high outliers, best shown on a logarithmic scale."



We then used visualizations and correlation matrices to efficiently explore how specific features are related.

We initially had 6435 samples in our dataset. After applying IQR for outlier removal, the count reduced to 5951 samples.

For PCA feature reduction, we retained 7 components to capture 95% variance. Through Recursive Feature Elimination (RFE), the selected features were Store, Holiday_Flag, Fuel_Price, month, and year.



Imputation Techniques: We implemented imputation methods to address missing values in the dataset. The imputation process revealed no missing values in the dataset, maintaining its original state with 5,951 samples. All features showed a 0% missing value percentage, and the principal components' statistics remained consistent before and after imputation.

- **Advanced Model Selection:**
  - Cross-Validation: We applied cross-validation to our model to mitigate overfitting and improve generalization.
    - Pre-cross-validation, RMSE was 328,509.75 and R2 was 0.692.
    - However, post-cross-validation, the mean RMSE rose to approximately 593,609 with a standard deviation of about 334,497, indicating that cross-validation exposed overfitting, underscoring the need for further model refinement.
  - Hyperparameter Optimization:
    - Grid and Random Search: We **applied both grid search and random search techniques for the fine-tuning of model hyperparameters** to optimize the performance of the regression models.
  - Model Exploration:
    - We implemented additional models such as gradient boosting machines and random forests. They were used for their ability to **capture the non-linear relationships** in the data more effectively.
  - Time Series Analysis:
    - Holt-Winters Model: We implemented the Holt-Winters model to capture the temporal dynamics and trends in Walmart's sales data.

4. Machine Learning Modeling

Before performing modeling, the PCA data was split into training, validation and test (70%, 15%, 15%) This enabled us to reserve 15% of the data as unseen until after all models had been tuned and fit. The cross-validation was performed on the training set and the validation set was used to compare results for all five machine learning methods. **The test set was used only at the end to ensure no overfitting to the validation set** and to verify our final results are reproducible.

- **Linear Regression:** The performance of the Linear Regression was very poor, however minimal overfitting was observed. Clearly, Linear Regression failed to capture some of the relationships in the data. Validation r^2 = 0.1654.
- **Lasso Regression:** The performance of the Lasso Regression was identical to the Linear Regression, showing that all 7 principle components were necessary. Validation r^2 = 0.1654.
- **Random Forest:** It was extremely overfit and extensive hyperparameter tuning was performed to reduce overfitting without significantly impacting validation set performance. However, it was not possible to reduce overfitting significantly without impacting performance. However, results were a significant improvement over the linear models, indicating the existence of non-linear relationships in our variables. Final validation r^2 = 0.6646.
- **Gradient Boost:** it is a boosting algorithm that continuously trains on the errors of the previous learner as such it is difficult to overfitting, and even a very small number of estimators still showcases overfitting. Using a GridSearch methodology, a high number of estimators was selected thus overfitting is present in this model. However, the performance on the validation set was slightly better than that in the Random Forest model. Validation r^2 = 0.6757
- **SVM**: Performed GridSearch hoping that for some combination of Kernel, C, epsilonSVM would perform well. However, very extreme values for C and epsilon were chosen and the model barely outperformed linear regression. Validation r^2 = 0.2048.

```
# GradientBoos hyperparameter tuning
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import GradientBoostingRegressor
param_grid = {"n_estimators":[300, 500],
              "max_depth":[8,12, 16],
              "min_samples_split":[3, 5],
               "subsample" :[0.8,0.9],
              "learning_rate":[0.05, 0.1],
               }
gb_model = GradientBoostingRegressor()
gb_cv = GridSearchCV(gb_model, param_grid = param_grid, verbose = 3)
gb_cv.fit(X_train, y_train)
print(gb_cv.best_params_)
```

```
# n_estimator = 1000 min_sample_split = 4
from sklearn.ensemble import GradientBoostingRegressor
gb_model = GradientBoostingRegressor(n_estimators = 500, max_depth = 16, subsample = 0.8, learning_rate = 0.05, min_samples_split
gb_model.fit(X_train,y_train)
print(f"Training r^2: {gb_model.score(X_train,y_train):.4f}")
print(f"Valid r^2: {gb_model.score(X_valid,y_valid):.4f}")
```

```
Training r^2: 1.0000
Valid r^2: 0.6757
```

- On the test dataset reserved for this purpose, evaluation of the GradientBoosting model was performed with the parameters selected by the hyperparameter tuning.
    - Mean Absolute Error: 218633
    - Root Mean Square Error: 329996.62
    - r^2: 0.6671

5. Time Series Modeling

We also worked with time series data, which is data collected over time. We used techniques like **resampling and differencing to analyze and predict time-based patterns.** Time Series analysis is fundamentally different from other machine learning algorithms in that it is primarily used to forecast future events from past data. As such, we did not randomly split the data into training and test sets, but rather used the first 80% of the data as the training set and used it to forecast the final 20% from oldest to newest.

We then **performed a first-differenced model** to predict change in sales from week to week, instead of raw sales.

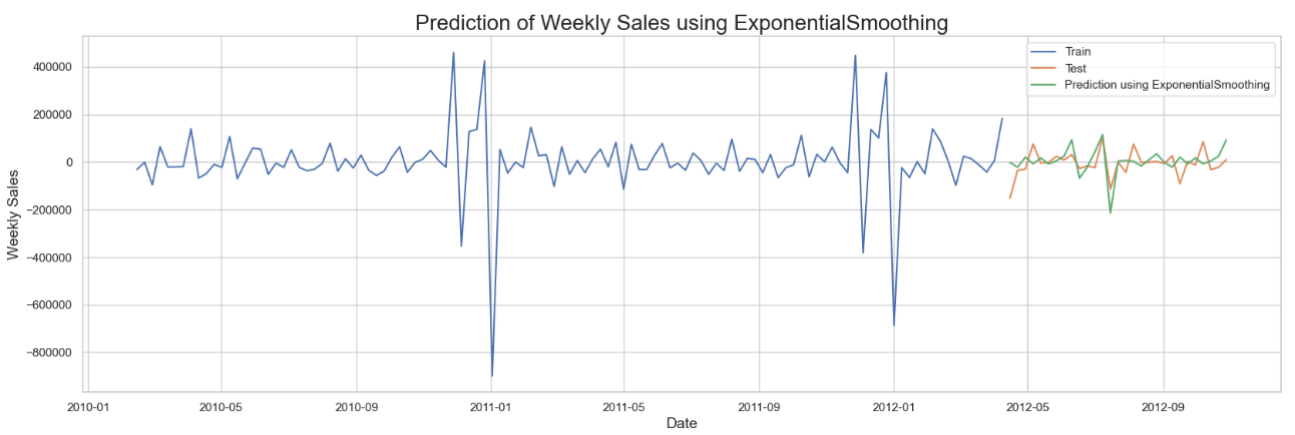
We **chose the Holt-Winters model to capture trend, and seasonality in Walmart sales.** Holtwinters is the model that decomposes the time series variable into trend, seasonality, and other effects. Trend is the part of the data that constantly increases or decreases over time, seasonality are changes that happen on a predicted schedule in this case holiday sales, and other effects are whatever else left over after taking into account the seasonality and trend. **Since the Walmart data shows both trend and seasonality this seemed an appropriate choice of model.**

We are multiplying the error by 5 on holidays and not multiplying by 5 for non holidays. Additive models follow a linear pattern where the effect of the trend and seasonality follow a linear or additive pattern because the data takes place over a short period of time during a stable economy there is no need to look for exponential or multiplicative effects of trend or seasonality.

For time series, python requires the date variable to be a dataframe index and a non random train-test split. Because older information will be used to predict the newer cells. In addition, instead of predicting daily sales, we are predicting change in sales from week to week.

```
model_holt_winters = ExponentialSmoothing(train_data_diff, seasonal_periods=20, seasonal='additive',
                                          trend='additive',damped=True).fit() #Taking additive trend and seasonality.
y_pred = model_holt_winters.forecast(len(test_data_diff))# Predict the test data

#Visualize train, test and predicted data.
plt.figure(figsize=(20,6))
plt.title('Prediction of Weekly Sales using ExponentialSmoothing', fontsize=20)
plt.plot(train_data_diff, label='Train')
plt.plot(test_data_diff, label='Test')
plt.plot(y_pred, label='Prediction using ExponentialSmoothing')
plt.legend(loc='best')
plt.xlabel('Date', fontsize=14)
plt.ylabel('Weekly Sales', fontsize=14)
plt.show()
```



Time Series Results:

- Mean Absolute Error: 44639
- The Mean Absolute Error of the Time Series model is significantly lower than our best Mean Absolute Error result for our machine learning models. This suggests that Walmart's sales data should be viewed as the forecasting question since the assumption of independence of observations appears to be violated as each observation is related to the observation before it. The predictions for the time series model were off on average for 44639 dollars per week. Since this is data from multiple stores, the results are reasonable as shown on the plot above.

6. Conclusion:

- This project has successfully integrated data preprocessing, model selection, and hyperparameter optimization techniques.
  The use of gradient boosting machines and random forests, as well as time series analysis enhanced the predictive accuracy of the Walmart sales forecasts
- **Further analysis into this dataset would include more complex time series models such as ARIMA models.**
  - More complex models would allow us to identify further correlations in sales due to time and deal with additional autocorrelation and partial autocorrelation, as well as account for the non-time variables such as CPI and unemployment that were included in the machine learning models.
  - We did not include those in our Holt-Winters model, but it would be interesting to see how model performance would change if they were included.

Appendices:

- **Link to code in Kaggle:** https://www.kaggle.com/code/krajiv2018/walmart-sales-prediction
- **Link to our video presentation:** https://www.youtube.com/watch?v=sMRCgTcUWBw
- **Link of the dataset:** Walmart Sales Prediction - (Best ML Algorithms) | Kaggle