Machine Learning Analysis Report

Data Selection

The data was taken from Kaggle. The link to source is:
https://www.kaggle.com/aditya6196/retail-analysis-with-walmart-data

Data Cleaning

The data was cleaned using an ETL function which was described in the Walmart_Wkly_Sales_ETL.ipynb file of the ETL folder of master branch. The cleaned data was then stored in the postgres as 'Weekly_Sales', 'Features' and 'Holidays' tables. The data was then stored in the RDS database of the Amazon Web Services (AWS), so that it can be easily imported to some other remote file.

Importing the Data in the jupyter notebook

The data was imported from the RDS database of the AWS for the machine learning.

Link to the machine learning code (Initial Analysis):
https://github.com/Franceskling/final_project/blob/machine_learning/machine_learning/machine_learning_Models.ipynb

Link to the machine learning code (Final Analysis):
https://github.com/Franceskling/final_project/blob/machine_learning/machine_learning/ALLstores_sales_forecast_ML.ipynb

Data Preprocessing

The imported data has been copied into another dataframe and sorted by "Store" and "Date". A new column named 'sales_diff' has been created which consists of the difference between the weekly sales. Then some of the unwanted rows are dropped which could cause the error in the final sales predictions. After this, a dataframe is generated which consists of 12 lag columns. The data has been saved at various stages to create visualizations, compare predicted values and performance metrics in the dashboard.

Feature Selection

The preprocessed data was then divided into the input(X) and the target/output(y) features. Also, the non-relevant columns were dropped from the data. All the columns to be used in the model must contain a numerical data type.

Input features (X):
“Store","Holiday_Flag","Temperature","Fuel_Price","CPI","Unemployment","Month","Year"and "Week"

Target/Output feature(y): "Weekly_Sales"

Splitting the data into Training and Testing datasets

The data needs to be split into the training and testing datasets in the ratio of 75-25% before fitting in the Standard Scaler instance. This prevents testing data from influencing the standardization function.

Scaling the Data

Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. It is performed during the data pre-processing to handle highly varying magnitudes or values or units. If feature scaling is not done, then a machine learning algorithm tends to weigh greater values, higher and consider smaller values as the lower values, regardless of the unit of the values. All the input training and testing datasets are scaled before fitting in the machine learning models.

Performance Metrics

The following metrics have been calculated in this project to access the performance of the machine learning models.

- **Mean Squared Error:**

  In statistics, the mean squared error (MSE) or mean squared deviation (MSD) of an estimator (of a procedure for estimating an unobserved quantity) measures the average of the squares of the errors- that is, the average squared difference between the estimated values and the actual value.

- **Root Mean Squared Error:**

  Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells how concentrated the data is around the line of best fit.

- **Mean Absolute Error:**

  In statistics, mean absolute error (MAE) is a measure of errors between paired observations expressing the same phenomenon.

- **R-squared:**

  R-squared (R2) is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model. R-squared is always between 0 and 100%:
  - 0% indicates that the model explains none of the variability of the response data around its mean.
  - 100% indicates that the model explains all the variability of the response data around its mean.
  - Usually, the larger the R2, the better the regression model fits the observations.

Machine Learning Algorithms

In this project, the following machine learning algorithms are used.

- **Linear Regression Model**

  Linear regression is a statistical model that is used to predict a continuous dependent variable based on one or more independent variables fitted to the equation of a line. Multiple linear regression builds a linear regression model with two or more independent variables. In this case, the dependent variable (target variable i.e. y) is dependent upon several independent variables(X). A regression model involving multiple variables can be represented as:

  $y = b0 + m1b1 + m2b2 + m3b3 + \dots \dots mnbn$

This is the equation of a hyperplane.

- **Results (All stores):**
  Root Mean Squared Error = 529802.065
  Mean Absolute Error = 440285.203
  R-squared = 0.145
- Since R-squared is only 14.5%, it means that this Linear Regression Model is not good in prediction and needs some improvement.
- Source link:
  https://github.com/Franceskling/final_project/blob/machine_learning/machine_learning/machine_learning_Models.ipynb

- **Results (Store-1):**
  Root Mean Squared Error = 160321.168
  Mean Absolute Error = 115030.621
  R-squared = 0.159
- Since R-squared is only 15.9%, it means that this Linear Regression Model is not good in prediction and needs some improvement.
- Source link:
  https://github.com/Franceskling/final_project/blob/machine_learning/machine_learning/Store1_salesForecast_ML.ipynb

- **Linear Regression Model using "lag"**

The Linear Regression Model can be improved by using the "lag". A "lag" is a fixed amount of passing time; One set of observations in a time series is plotted (lagged) against a second, later set of data. The kth lag is the time period that happened “k” time points before time i.

- **Results (All stores):**
- The results in this case are better than the Linear Regression without using lag. The scores can be seen in the code’s “Store_scoring_01” dataframe which are promising.
- This means that Linear Regression Model using "lag" is accurate and can make reasonably good predictions than the Random Forest Regressor Model using "lag".
- Source link:
  https://github.com/Franceskling/final_project/blob/machine_learning/machine_learning/ALLstores_sales_forecast_ML.ipynb

- **Results (Store-1):**
  Mean Absolute Error = 29271.619
  Mean Squared Error = 1494186688.734
  Root Mean Squared Error = 38654.711
  R-squared = 0.958
- The value of root mean squared error is 38654.71, which is ~ 2.4% of the weekly sales mean value which is 1.561364e+06.
- This means that this algorithm is accurate and can make reasonably good predictions than the Linear Regression without using lag.

- Source link: https://github.com/Franceskling/final_project/blob/machine_learning/machine_learning/Store1_salesForecast_ML.ipynb

- **Random Forest Regressor Model**

A random forest is an ensemble model that consists of many decision trees. Predictions are made by averaging the predictions of each decision tree.

- **Results (All stores): without "lag"**
  Root Mean Squared Error = 118809.742
  Mean Absolute Error = 65539.967
  R-squared = 0.957
- Since R-squared is 95.7%, it means that this Random Forest Regression Model is good in prediction as compared to the Linear Regression Model (without using "lag"). This prediction is very good but it can be due to the overfitting of the model.
- Source link: https://github.com/Franceskling/final_project/blob/machine_learning/machine_learning/machine_learning_Models.ipynb

- **Results (All stores): with "lag"**
- The scores can be seen in the code's "Store_scoring_02" dataframe which are good and have reasonable values. By using the "lag" for all the stores, this model is less prone to overfitting and hence the results vary from store to store.
- Source link: https://github.com/Franceskling/final_project/blob/machine_learning/machine_learning/ALLstores_sales_forecast_ML.ipynb

- **Random Forest Regressor Model (Naïve Forecasting Technique for Future Forecast)**

In this project, Naive forecast technique has been used for forecasting a standard week by using the entire prior week as the forecast for the week ahead. It is based on the idea that next week will be very similar to this week.

- **Results (All stores):** Scenario-1 (Using all input features)
  Average R2 = 0.313
  Average RMSE = 138121.880
  Average Naive FCST RMSE = 173170.190

- **Results (All stores):** Scenario-2 (Using only Previous Week Sales and Weekly Sales)
  Average R2 = 0.456
  Average RMSE = 113317.595
  Average Naive FCST RMSE = 173170.190

- **Results (All stores):** Scenario-3 (Using Previous Week Sales and original features i.e. unemployment, CPI, temperature, fuel price, holiday)
  Average R2 = 0.480
  Average RMSE = 114397.396
  Average Naive FCST RMSE = 173170.190

- Source link: https://github.com/Franceskling/final_project/blob/machine_learning/machine_learning/ALLstores_sales_future_forecast_ML.ipynb

Conclusion

In this project, we hoped to predict the weekly sales by using the above regression models. Our Random Forest Regression model is able to predict the week-50th sale by using the Naïve forecasting technique with an average R2 equal to 48%. The R2 value can be further improved by increasing the number of estimators.

Further Improvements

In the next phase of project, further improvements can be done by using LSTM or ARIMA and by adding more data that includes location and other features with higher correlation to the sales in attempt to get better accuracy.

Data Transformation

The predicted data is exported or saved into a simpler format for storage and future use, such as a CSV, spreadsheet, or database file. These output results can be used in further analysis and for dashboard creation of the project.

Link to the Tableau Presentation:
https://public.tableau.com/profile/frances.klingenberger#!/vizhome/Final_Presentation_00/SalesForcasting?publish=yes