Walmart Retail Weekly Sales

A prediction algorithm for weekly sales at Walmart retail locations

Capstone 2 Presentation
Damilola T. Olaiya

Proposal

- Hypothesis → How can Walmart use its reported sales data to
  - predict and take advantage of future sales/demand
  - potentially improve inventory allocation/scheduling?

- Criteria for success
  - Creating a model that can accurately predict the sales with regards to single and multiple features

Data Wrangling

- The dataset contains sales information from 45 walmart stores. Source:
  - https://www.kaggle.com/datasets/yasserh/walmart-dataset

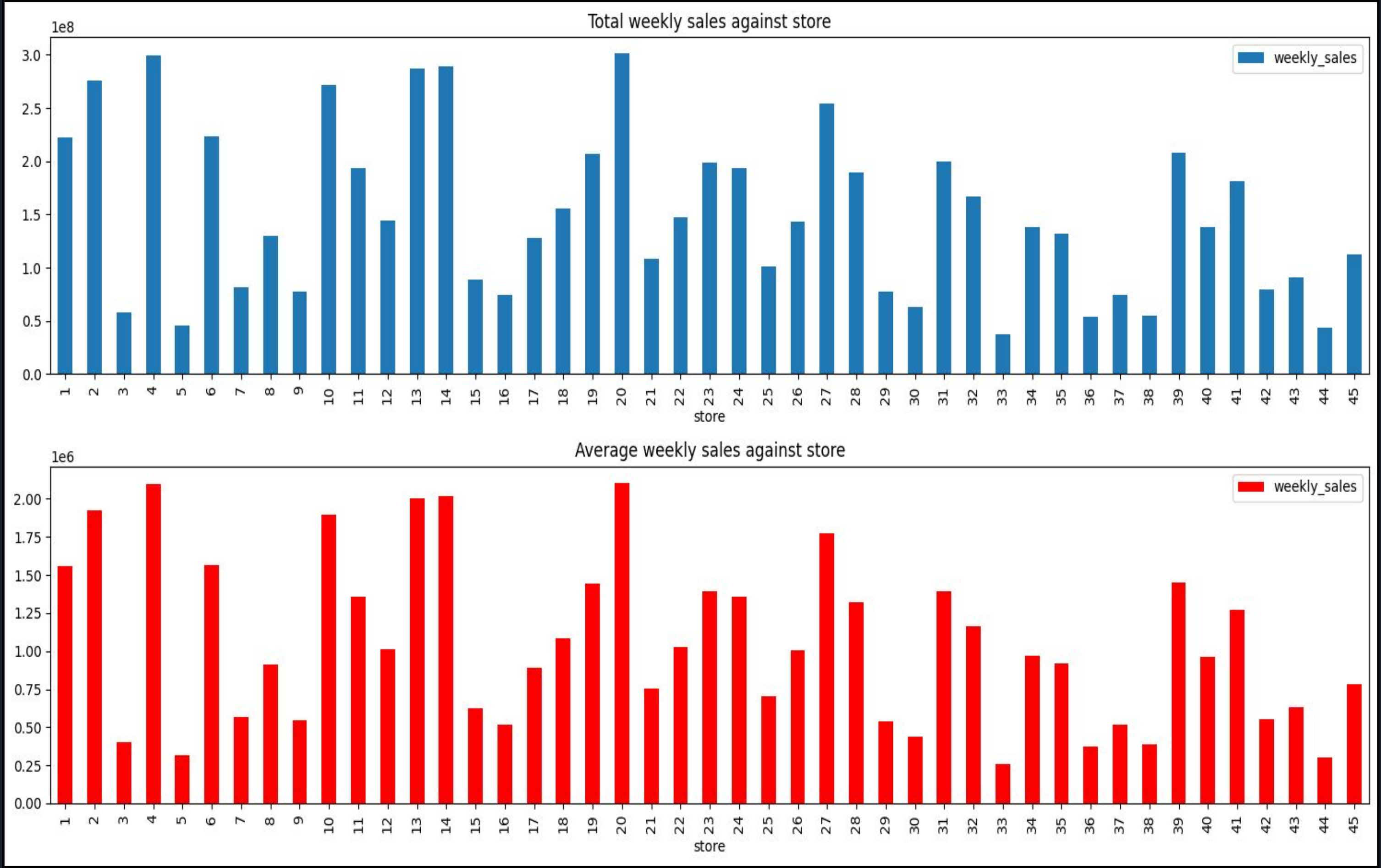
- There were 7 features/columns:
  - store
  - weekly_sales
  - holiday_flag, temperature
  - fuel_price
  - cpi
  - unemployment.

Data Wrangling

- The target feature is weekly_sales.

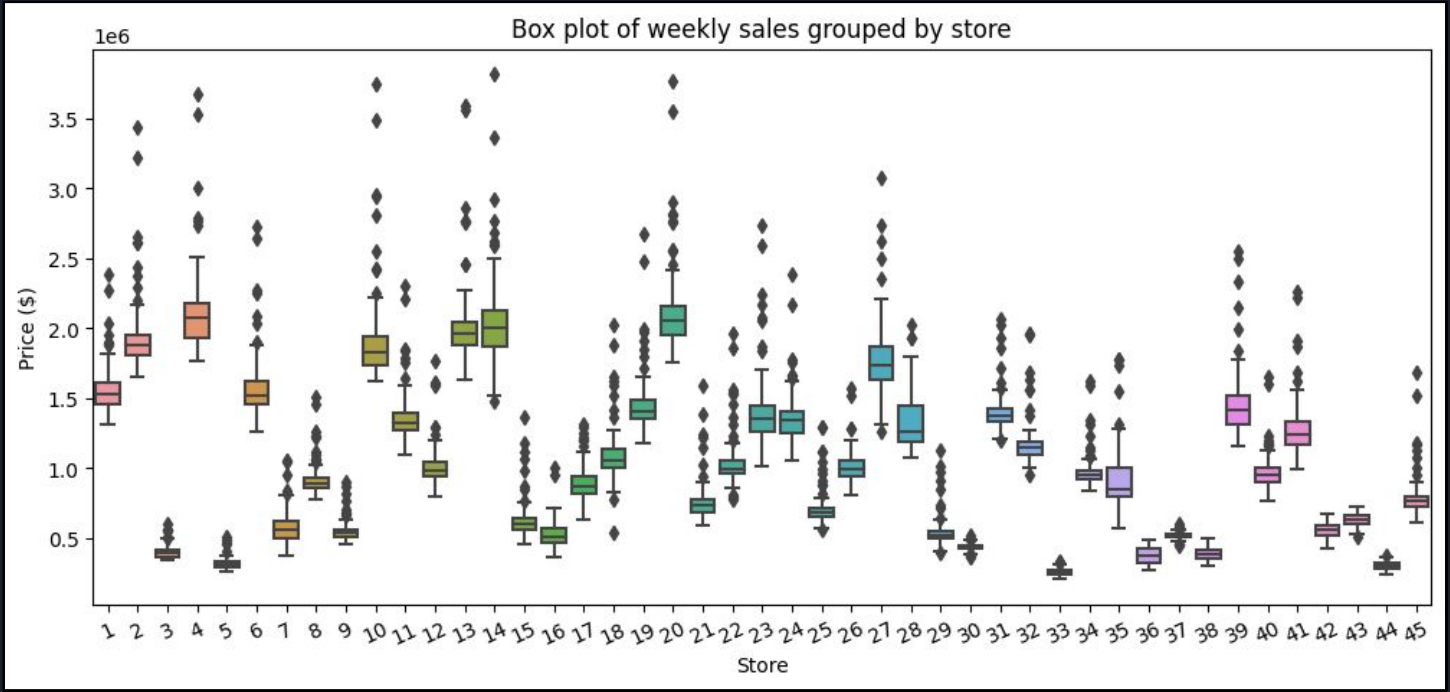
- There were 6435 entries/samples in the data. This corresponded to 143 entries each for 45 stores.

- There were no missing values. The data was clean.


Total weekly sales against store
1e8
weekly_sales
store
Average weekly sales against store
1e6
weekly_sales
store

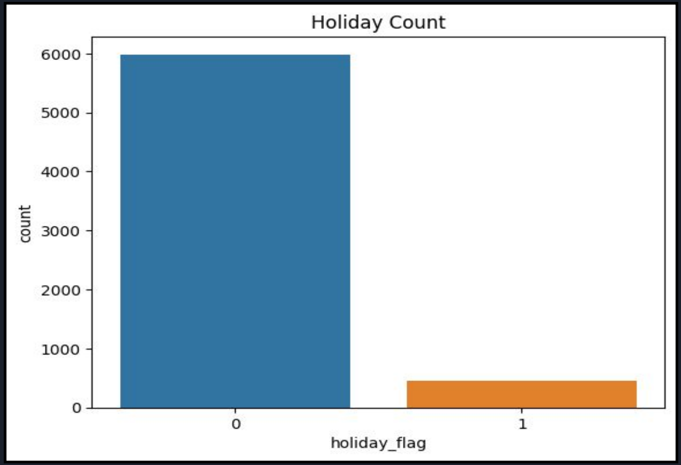
Data Wrangling

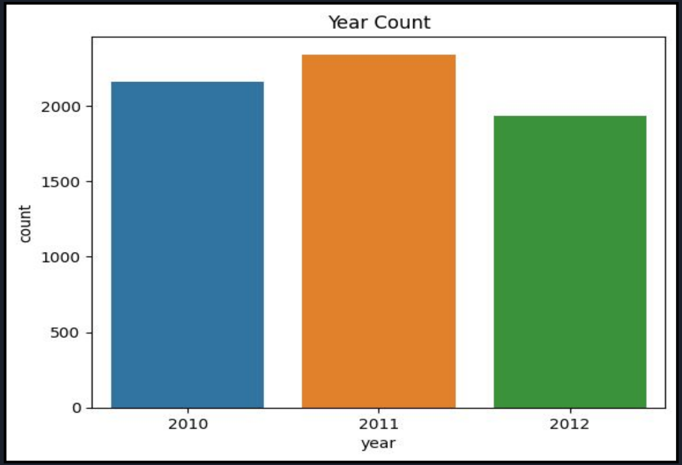
- Weekly sales grouped by store



Exploratory Data Analysis (EDA)

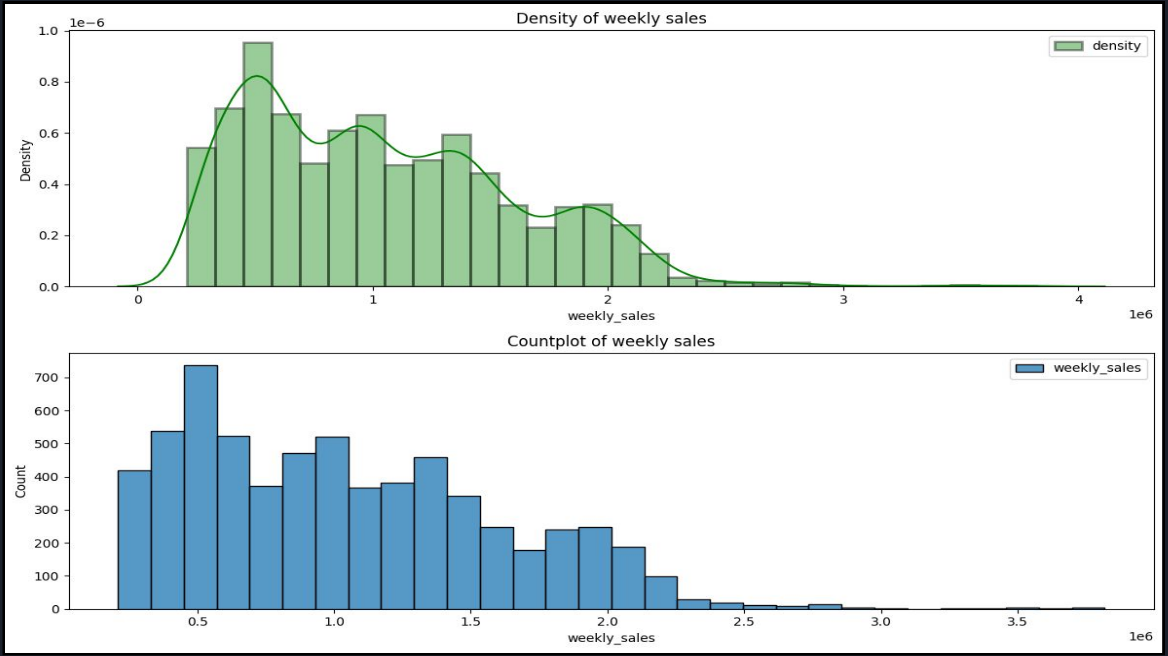
- The date feature was deconstructed. This included extracting weekday, month and year from each sample.
- Categorical and numeric features were parsed and listed out. There were 2 categorical (holiday_flag, store) and 4 numerical features (unemployment, fuel_price, cpi, temperature, weekly_sales).
- As expected, there are far more non-holidays than holidays.





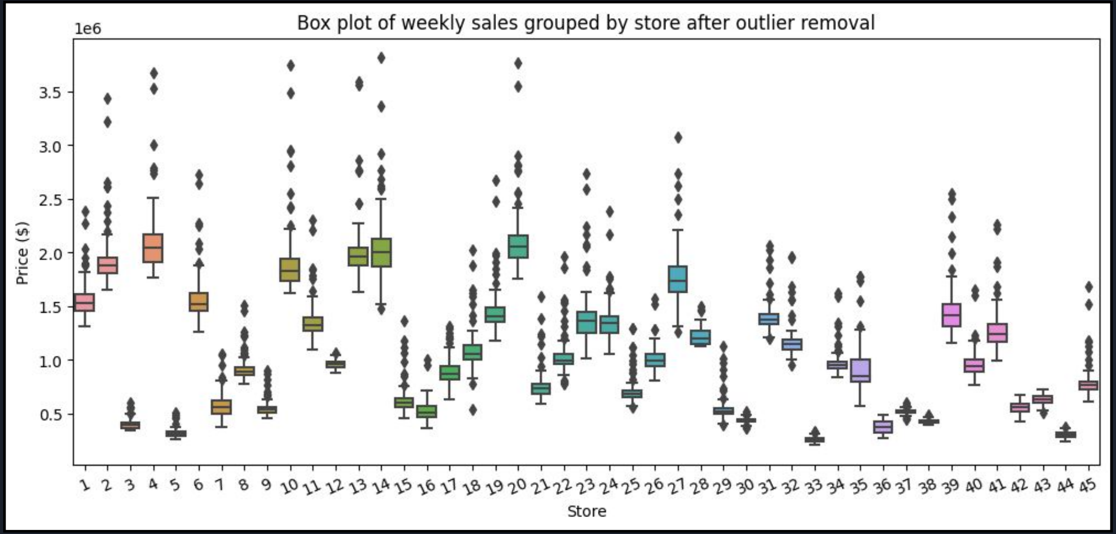
Exploratory Data Analysis (EDA)

- Distribution and Countplot of weekly sales



Pre-processing and Training

- No duplicate rows
- Outliers were removed using Inter Quartile Range (IQR).
  - This dropped total samples by 7.52%.



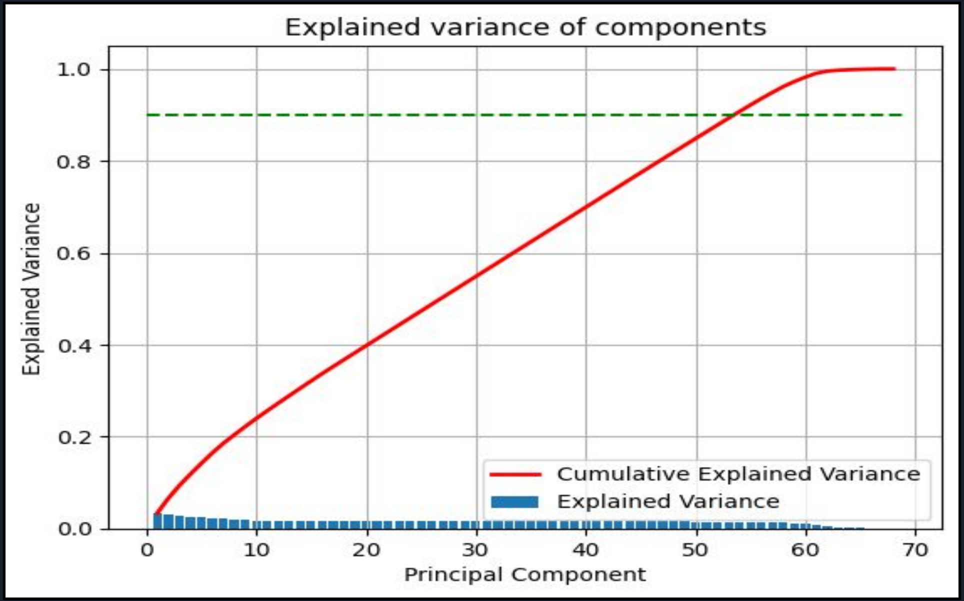
Pre-processing and Training

- Dummy variables were created for holiday_flag, store, weekday, month and year features.
  - The first column was dropped in each case to prevent issues of multicollinearity.
  - The end result was a dataframe with 69 features/columns. This had potential to be cumbersome.

- Data was split into training and test sets
  - Scaling/standardized to have a mean of 0 and a standard deviation of 1.
  - Scaling was fit using X_train only then used to transform X_test.
  - This was done so that the model would be completely unaffected by the testing data.

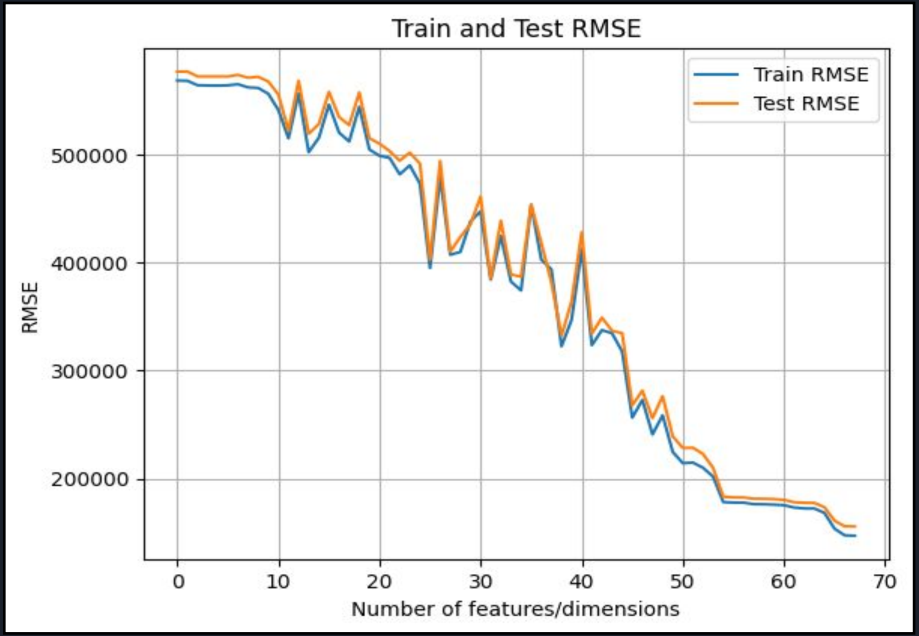
Pre-processing and Training

- Principal component analysis (PCA) was performed on the data.
- Using all features (no reduction), the variance was explained as such:



Pre-processing and Training

- Using PCA and linear regression, RMSE was calculated on the train and test datasets for features ranging in number from 1 to 69. Naturally, the error reduced with more features as more variance was explained.



Modeling

- Using dummy variables for all categorical features makes the data too granular and convoluted (69 features/columns) as evidenced by the PCA decomposition result from pre-processing.

- Going forward, I assumed that all stores (store 1 to store 45) are within the same market segment and ignored store to store differences.

- Additionally, the holiday_flag feature did not need to be standardized as the values were already within scale for analysis.

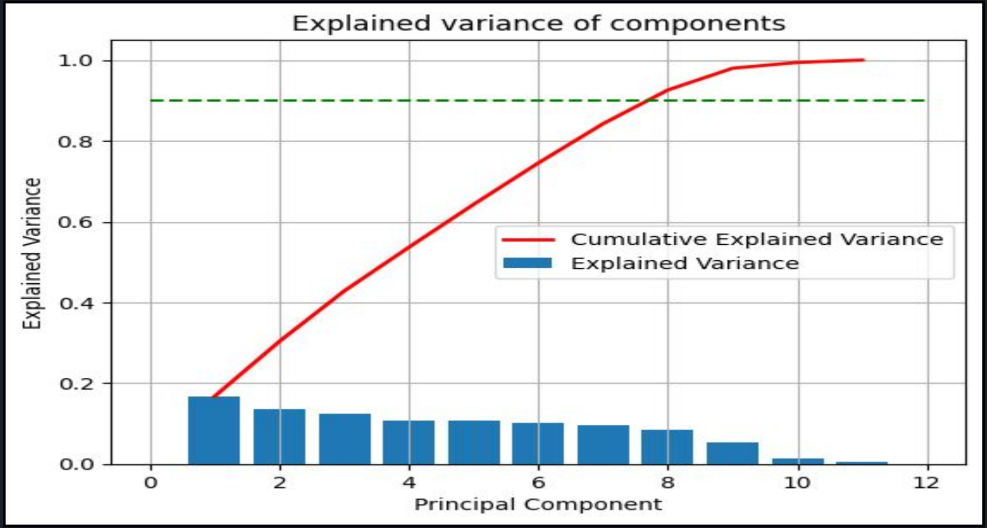
- To that effect, I mapped the holiday_flag feature back to 1s and 0s and eliminated the store, month and year features from the data.

- This left 11 features remaining in the dataset (holiday_flag, temperature, fuel_price, cpi, unemployment, weekday_1, weekday_2, weekday_3, weekday_4, weekday_5, weekday_6).

Modeling

- I was able to infer that feature reduction may be unnecessary as, although 90% of the variance is explained cumulatively by 8/11 principal components, only one of the components had a variance that was significantly lower than the others.

- That, combined with the relatively small number of features, allowed me to ignore feature reduction.



Modeling

- There were four different models used
  - multiple linear regression
  - lasso regression
  - ridge regression
  - random forest regression.

- Cross validation was performed for ridge, lasso and random forest regressions using:
  - alphas of 0.1, 1, 10, 100,1000 and 10000 for lasso and ridge regression
  - parameters {n_estimators: [300,400,500], max_depth:[4,6,8], min_samples_leaf :[0.1,0.2], max_features:['log2','sqrt']} for random forest regression

Modeling

- The best alpha for both lasso and ridge was found to be 100.

- The best parameters for the random forest regression were found to be:
    - max_depth: 6
    - max_features: log2
    - min_samples_leaf: 0.1
    - n_estimators: 400

Model Evaluation Comparison Matrix (MECM)

- The following metric were used to evaluate both training and test datasets
  - R^2 or Coefficient of determination
  - Sum of squared residuals
  - Mean squared error
  - Root mean squared error

	Train-R2	Test-R2	Train-RSS	Test-RSS	Train-MSE	Test-MSE	Train-RMSE	Test-RMSE
Random Forest Regression Model (RF)	0.036973	0.040372	1.490568e+15	3.847395e+14	3.131446e+11	3.230391e+11	559593.245026	568365.259504
Lasso Linear Regression (LLR)	0.019928	0.018191	1.516950e+15	3.936325e+14	3.186869e+11	3.305059e+11	564523.605053	574896.441420
Ridge Linear Regression (RLR)	0.019971	0.017923	1.516884e+15	3.937400e+14	3.186731e+11	3.305961e+11	564511.408328	574974.888760
Multiple Linear Regression (MLR)	0.020051	0.017895	1.516760e+15	3.937511e+14	3.186470e+11	3.306055e+11	564488.278884	574983.013229

Inference

	Train-R2	Test-R2	Train-RSS	Test-RSS	Train-MSE	Test-MSE	Train-RMSE	Test-RMSE
Random Forest Regression Model (RF)	0.036973	0.040372	1.490568e+15	3.847395e+14	3.131446e+11	3.230391e+11	559593.245026	568365.259504
Lasso Linear Regression (LLR)	0.019928	0.018191	1.516950e+15	3.936325e+14	3.186869e+11	3.305059e+11	564523.605053	574896.441420
Ridge Linear Regression (RLR)	0.019971	0.017923	1.516884e+15	3.937400e+14	3.186731e+11	3.305961e+11	564511.408328	574974.888760
Multiple Linear Regression (MLR)	0.020051	0.017895	1.516760e+15	3.937511e+14	3.186470e+11	3.306055e+11	564488.278884	574983.013229

- Lower RMSE implies a better the model. That said, a significant disparity between training and testing scores would suggest overfitting.
- All regression models were fairly similar in terms of training and test R2 and RMSE.
- However, Multiple linear regression performed best in training metrics but worst in test metrics suggesting that it was slightly overfitting.

Inference

	Train-R2	Test-R2	Train-RSS	Test-RSS	Train-MSE	Test-MSE	Train-RMSE	Test-RMSE
Random Forest Regression Model (RF)	0.036973	0.040372	1.490568e+15	3.847395e+14	3.131446e+11	3.230391e+11	559593.245026	568365.259504
Lasso Linear Regression (LLR)	0.019928	0.018191	1.516950e+15	3.936325e+14	3.186869e+11	3.305059e+11	564523.605053	574896.441420
Ridge Linear Regression (RLR)	0.019971	0.017923	1.516884e+15	3.937400e+14	3.186731e+11	3.305961e+11	564511.408328	574974.888760
Multiple Linear Regression (MLR)	0.020051	0.017895	1.516760e+15	3.937511e+14	3.186470e+11	3.306055e+11	564488.278884	574983.013229

- This is in line with what we would expect from lasso and ridge regression which work to combat overfitting.

- Random forest regression performed best for all metrics and gave the best overall results.

Conclusions

1. The dataset was quite small with just 6435 samples initially, which dropped 7.5% after cleaning.
2. Cross validating the Lasso and Ridge regressions allowed us to select the best alpha.
3. We will proceed with the Random forest regression model as it performed best.

Further Steps to Consider

1. Using pca for feature reduction
2. Using more of the generated features in the regression
3. Testing more parameters in the grid search cv at the cost of time
4. Using random forest with bagging, boosting etc
5. Using a polynomial regression model