CSE422 lab project

TOWFIQ KARIM

MD.Alvi Rahman

27th April 2024

1 Introduction

The BigMart Sales Prediction project serves as a valuable tool for forecasting the sales of various item products in various outlet stores. This will enable companies to make informed decisions on their marketing strategies and business plans by analyzing the patterns between the sales of the item product and other feature variables affecting it.

This project provides insights of the overall sales predictions that will help companies to identify the top-performing products in the outlets. For instance, if the analysis reveals a particular product with the highest sales, it becomes crucial to understand the factors that are contributing to its popularity so that its volumes can be increased. On the other hand, it is also important to identify the factors that are causing insufficient sales of certain products in order to improve them.

Thus, the business owners will be immensely benefited as they will be able to earn higher profits. The companies will also be able to predict how much revenue they are going to earn after a certain period of time so that they can be more confident in their future expenditures and savings. This will be a great advantage to many startups so that they do not incur significant losses at the beginning of their career by having a good understanding about the conditions that affect the sales of items in an outlet.

2 Dataset Description

2.1 Source

Link: Bigmart Sales data

Reference: BigMart Sales data. (2018, January 16). Kaggle.
https://www.kaggle.com/datasets/brijbhushannanda1979/bigmart-sales-data

2.2 Dataset Description

The dataset for the BigMart Sales has 12 features which are described below:

- Item_Identifier:Unique product ID
- Item_Weight: Weight of the product
- Item_Fat_content: Whether the product is low fat or not
- Item_ Visibility:The percentage of total display area of all products in a store allocated to the particular product.
- Item_Type: The category to which the product belongs
- Item_MRP:Maximum Retail Price (list price) of the product
- Outlet_Identifier: Unique store ID
- Outlet_Establishment_Year: The year in which store was established
- Outlet_size:The size of the store in terms of ground area covered
- Outlet_Location_Type:The type of city in which the store is located
- Outlet_Type:Whether the outlet is just a grocery store or some sort of supermarket
- Item_Outlet_Sales:Sales of the product in the particular store (This is the outcome variable to be predicted)

The problem we are focusing on is a regression problem as our goal is to predict the continuous numeric value of ‘Item_Outlet_Sales’ which will have a range of values rather than a specific category. It is not a classification problem as we are not going to assign a category to a given input based on its features.

In this dataset we have 8523 rows and in 12 columns.We have obtained a shape of (8523, 12) which represents 8523 samples, and each sample has 12 variables associated with it.

Our dataset has both categorical and quantitative features.’Item_Identifier’,’Item_Fat_Content’,’Outlet_Ident
and ’Outlet_Type’ are categorical features which represents categories with limited values. On the other hand,’Item_Weight’,’Item_Visibility’,’Item_MRP’,’Outlet_Establishment_Year’ and ’Item_Outlet_sales’ are quantitative features which represents numeric values.

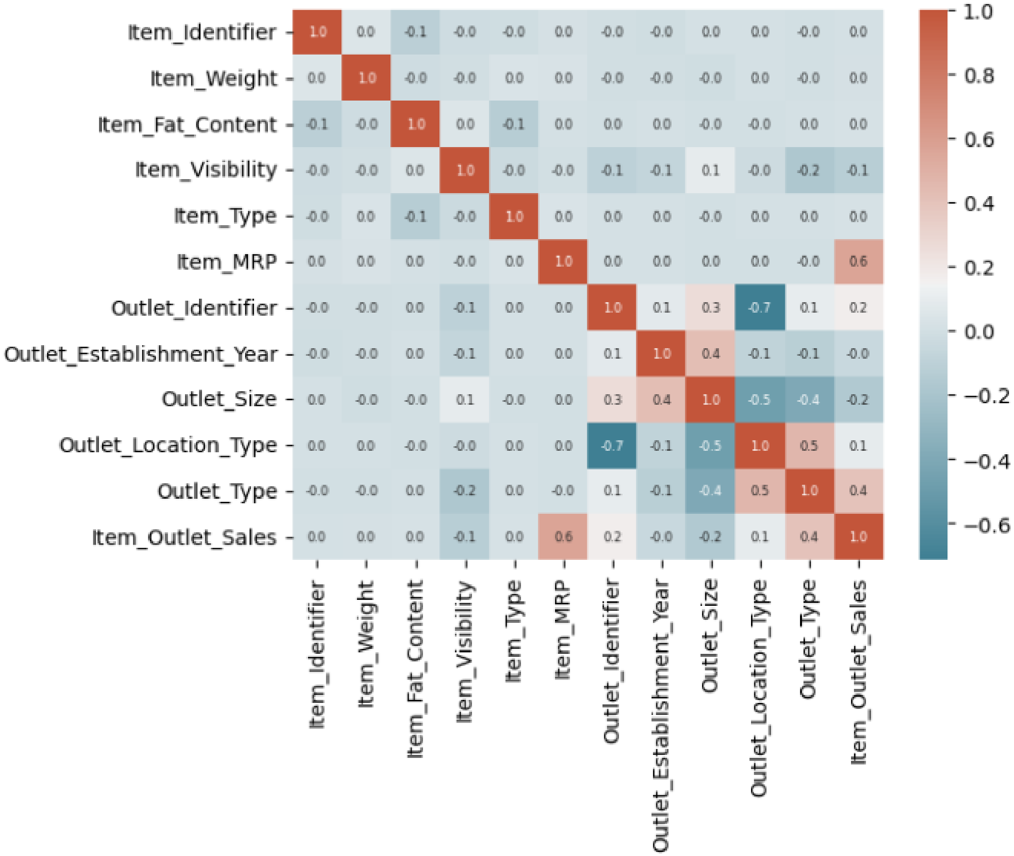

Figure 1: Co-relation between variables.

The Correlation of all features including both input and output features are shown by applying heatmap using the seaborn library:

This matrix shows the relationship by representing the correlation coefficient between each pair of variables in the dataset. Correlation coefficients measure the strength of a linear relationship between two variables which range from -1 to 1, where:

1: Perfect positive correlation
0: No correlation
-1: Perfect negative correlation

With the help of correlation coefficient, we can get the idea of the most important features of the dataset. It can be observed that the output variable 'Item_Outlet_Sales' shows positive correlations with 'Item_MRP', 'Outlet_Identifier', 'Outlet_Location_Type' and 'Outlet_Type'. The output also shows negative correlations with 'Item_Visibility' and 'Outlet_Size'. The rest of the other input variables show no correlation with the output which means that they have no linear relationship but may have non linear or complex relationships or dependencies between them. That is why we did not omit those features.

3 Dataset Pre-processing

At first, we investigated the duplicate values of the rows to reduce redundancy and found out there were no duplicates present in the dataset using 'bigmart_data.isnull().sum() method.
Then we looked for the null values using the 'bigmart_data.isnull().sum() method. We discovered that the features 'Items_Weight' and 'Outlet_Size' contained 1463(approx 17%) and 2410 (approximately 28%) null values respectively. Such a significant proportion of missing values can indeed lead to inaccurate or biased results if not appropriately handled. Therefore, instead of deleting the rows or columns with missing values, we impute valid values to handle the missing data so that important data values from the same rows and columns are not lost. Since the feature 'Item_Weight' contains numerical values, we chose the mean value for imputation because the variance for the mean was slightly lower than the median. This means that the data values were closer to the mean. In a regression problem, the narrower spread of data tends to result in better model performance so we used the '.fillna()' method to replace the null values with the mean value. This approach ensures that the missing values are replaced with reasonable estimates, allowing us to retain the valuable information contained within the feature.
As for the categorical feature 'Outlet_Size', we used the mode for imputation. To accomplish this, we created a pivot table using the Pandas library. This table created the column 'Outlet_Type' having 'Outlet_Size' as the values and by applying lambda function we were able to extract the first of the most frequently occurring values of 'Outlet_Size' for each 'Outlet_Type'. This allowed us to determine the mode of 'Outlet_Size' corresponding to the 'Outlet_Type'

which we used to impute the missing values in the 'Outlet_Size' column. By using the mode for imputation, we can fill in the missing categorical values with the most common value, providing a reasonable estimate for the missing data.
Moreover, after visualizing the feature variables we observed that the feature 'Item_Fat_Content' had different categorical values with the same exact meaning due to differences in spellings. Therefore, we combined the values of 'low fat', 'Low Fat' and 'LF' into one value and merged 'reg' and 'Regular' with another single value.
Finally, we handled all the categorical features in the dataset by using the Label Encoder. This assigns specific integers to unique values in the categories as the machine learning models are designed to work better with numerical input. We applied the method 'encoder.fit_transform(bigmart_data[col])' to fit the encoder on the data and simultaneously transform the data into numerical values. Label encoding is advantageous compared to one-hot encoding as it reduces the dimensionality of the data.

4 Feature Scalling

Feature scaling is an essential step in preparing our dataset for machine learning models. It involves normalizing the range of features so that they are interpreted on the same scale by the models. We applied StandardScaler to the features 'Item_Weight', 'Item_Visibility' and 'Item_MRP' for standardization by removing the mean and scaling each variable to unit variance. This ensures that the features are transformed to a consistent range making them comparable and easier for the machine learning models to interpret so that the learner does not give more importance to features with high variance.

5 Dataset Splitting

We divided the column features into x and y variables where y contains the dependent variable 'Item_Outlet_Sales' which will be predicted and x contains the remaining other features. As there is no issue of imbalanced data, we used a random split on the dataset using the 'train_test_split(x, y, test_size=0.3, random_state=2)' method. Setting the 'random_state' to 2 ensures the same exact split in the future with the same 'random_state' value. Setting the 'test_size' to 0.3 determines that 30% of the data will be used for testing and the remaining 70% will be used for training. Moreover, we used the variables 'x_train' and 'y_train' to train the model while the variables 'x_test ' and 'y_test 'are used to test the model.

6 Model Training and Testing

We have used the following four regression models to train and test the dataset:

- RandomForestRegressor()

  This model builds a collection of decision trees and combines each of their predictions to improve accuracy and generalization. At each split of a decision tree, a random subset of features are considered for diversity among the trees which prevents the overfitting of the training data.

- LinearRegression()

  This model creates the relationship between the dependent variable and the independent variables by using a linear equation. It solves the regression problem by learning coefficients and intercept of a linear equation that best fits the training data by minimizing the summation of squared differences between the actual and predicted values, and then uses this equation to make predictions on new data.

- Lasso()

  This model is a linear regression technique that introduces a penalty term to the objective function. This penalty term is based on the absolute values of the regression coefficients which makes the coefficient values of less important features to become exactly zero resulting in a much simpler and more interpretable model.

- XGBRegressor () This model uses an implementation of the gradient boosting algorithm to solve regression problems. It combines the predictions of multiple weak models, often decision trees which are constructed in a sequential manner, and each tree will correct the errors of the previous ones.

7 Model Selection/Comparation Analysis

After fitting both 'x_train' and 'y_train' into our selected models, we predicted the values of y by using the 'x_test'. We then compared the results with the 'y_test' by calculating R-Squared and Root Mean Squared Error values. R-squared represents the proportion of the variance in the dependent variable that is explained by the independent variables in a regression model while the Root Mean Square Error represents the square root of the average of the squared differences between the predicted values and the actual values

The R-squared values for the selected models are as follows:

- RandomForestRegressor = 0.5738
- LinearRegression = 0.5098
- Lasso = 0.5097
- XGBRegressor = 0.5241

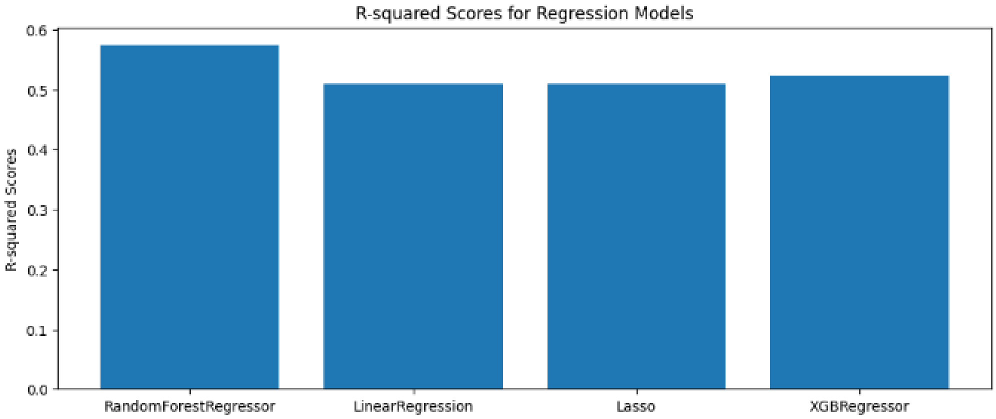

Figure 2: R-squared scores for regression models

The R-squared values will range between 0 and 1 where 1 indicates that the model is perfectly able to predict the value of 'Item_Outlet_Sales' while 0 will indicate that the model is unable to capture the variance. Therefore, we can see from the results that RandomForestRegressor performs slightly better than the other models with almost 57% times predicting the correct sales.

The Root Mean Squared Error values for the selected models are as follows:

- RandomForestRegressor = 1150.6831
- LinearRegression = 1231.1664
- Lasso - R-squared = 1231.2281
- XGBRegressor = 1213.0555

The greater the values of Root Mean Squared Error, the further away the predicted values are from actual ones. Again, it can be observed that RandomForestRegressor has the lowest value out of all the other models which indicates that the prediction is closer to actual value than the other models.

8 Conclusion

In conclusion, this project has equipped us with valuable insights into predictive capabilities of machine learning in forecasting outlet sales of items based on several feature variables. Through data pre-processing, feature scaling, model training and testing, we achieved a good predictive model where Random Forest

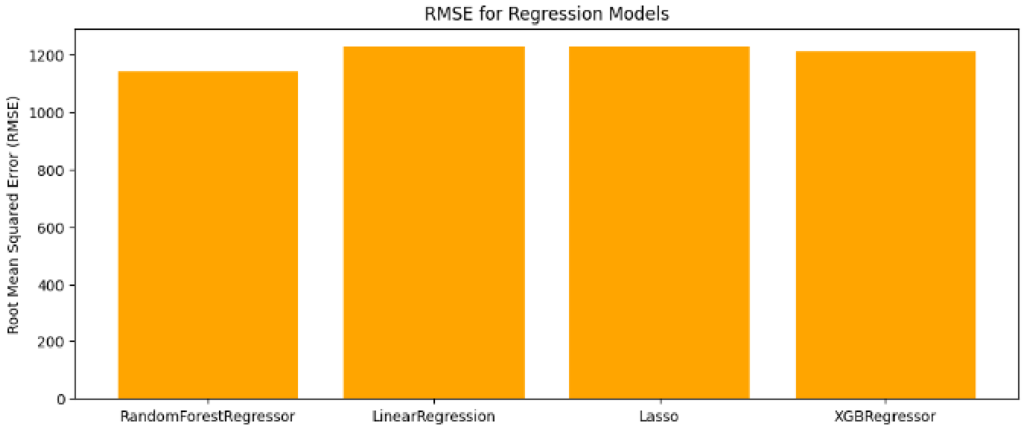

Figure 3: RMSE for Regression Models

Regressor emerged as a slightly better performer than others through our comparative analysis. Thus, our project fulfills the aim of predicting the BigMart sales in an efficient way which would enable businesses to anticipate trends, optimize market strategies and make data driven decisions.