EXPERIMENT REPORT - Predictive Model

Student Name	Ronik Karki
Project Name	Assignment 2 Predictive Model
Date	10/10/2023
Deliverables	Notebook Name: karki_ronik-24886412-predictive_decision_tree_regressor.ipynb Github Repo: https://github.com/ronik999/Machine_learning_as_a_service/tree/master API URL: https://mla-at2-e0654ce6deb8.herokuapp.com/

1. EXPERIMENT BACKGROUND	
1.a. Business Objective	The main goal of this project is to predict the revenue of the specific item and store of the US Retail Store. This project helps the business to predict the prices that the items of a particular store will be making in the future. This helps the businesses to understand how much profit can be generated and if the price doesn't come as expected, then it can be analyzed to a specific store to look at the issues. As the problem is to predict the revenue, this problem requires a regression model for prediction. Therefore the good metrics for this problem would be having low Root Mean Squared Error(RMSE) and Mean Absolute Error(MAE). Furthermore, R-squared metrics would also be good to analyze how well the model would be able to explain the variance. If there is a high RMSE and MAE score then the model won't be able to predict the revenue properly. It defines the range(either higher or lower) than the real value which gives the estimation of errors the model made at the time of prediction. Therefore, a good model should be able to minimize both errors in creating a reliable model.
1.b. Hypothesis	The dataset consists of ~47 Million rows and the hypothesis of this project is that the model is most likely to overfit with high data points and tuning the hyperparameters might reduce the overfitting of the model. Since the prediction is more hierarchical, the tree-based model might outperform linear or other models.

1.c. Experiment Objective	The expectation from this experiment is that a tree-based model like Decision Tree Regressor will perform better than the Linear Regression model or other models. Furthermore, it might perform better than the naive model.

---

2. EXPERIMENT DETAILS	
2.a. Data Preparation	For the data preparation part, there were 4 different CSV files (sales_train.csv, calendar.csv, calendar_events.csv, and items_weekly_sell_prices.csv) which stored important information about the data set. All four CSV files were merged to create one pandas data frame for prediction. The following steps were performed for the dataset preparations: 1. Initially, the sales_train file had information about each item ID, store ID, category ID, department ID, and state ID with respect to the quantities that were sold on a specific date from the year 2011 to 2015. 2. The sales_train file was in wide format which was melted using pandas df_melt function to convert to the long format. 3. This was further merged with the calendar to get the date information about the items of a particular store being sold. 4. Furthermore, the merged data were again merged with the items_weekly_sell_price file to get the information about the price for that particular product. 5. In addition, each unit was multiplied by its sell price to get the revenue of each product. 6. Finally, columns like id, d, department id, wm_yr_wk, sales, and sell_price were dropped as they had information captured by other columns and these columns were redundant which would just increase the complexity of the large dataset. Then this table was used for feature engineering and further model processing.
2.b. Feature Engineering	For this experiment, three types of feature engineering techniques were used: 1. Data imputation: a. There were Nan in the sell_price column which led to the revenue having Nan values, it might be because at that period the product wasn't sold and had 0 sales. Therefore, these Nan values were imputed with 0 values which means no sales were observed on that particular date. 2. Ordinal encoding: a. All of the categorical features were converted to integers by applying Ordinal encoding techniques and this encoder was saved for applying in the future unseen data as well. 3. Feature transformation: a. The Date feature was converted to get specific information for the month and day which were also encoded using the ordinal encoder. b. The date was also converted to numerical using the DateTime to ordinal function to put the information of the date as required for prediction.

2.c. Modelling

Train and Test splits were performed on the final table with a split size of 80% for the training model and 20% for evaluating the performance on the test set. The data set was shuffled prior to the split in order to remove any bias the dataset has for the model-building purpose. All of the features were used as the predictor variables and revenue was set as the dependent variable which was used for the prediction. 4 models were used for training and testing to predict the revenue of each item and store. The four models were the XGBRegressor model, Linear Regression model, Decision Tree Regressor model, and Tuned Decision Tree Regressor model. All of the models were evaluated and compared with the naive model where the naive model is just the average revenue of the products.

3. EXPERIMENT RESULTS

3.a. Technical Performance

There were many results and testing done throughout this experiment. The observed results for these models are shown below:

Model	Metrics	Training Scores	Testing Scores
Naive Model	MAE	-	4.38058
	RMSE	-	9.17113
	R2	-	0.0
Linear Regression Model	MAE	4.98589	4.92254
	RMSE	9.25654	9.23586
	R2	0.008	0.007
XGBRegressor Model	MAE	3.94789	4.21470
	RMSE	8.63260	9.34613
	R2	0.07238	0.05405
Decision Tree Regressor	MAE	0.0	2.91908
	RMSE	0.0	7.50662
	R2	1.0	0.33005
Decision Tree Regressor Tuned	MAE	2.16505	2.41497
	RMSE	5.16482	5.70069
	R2	0.68279	0.61362

Starting with Linear Regression model, even though there were so many data points, the model couldn’t learn anything from the data. Similarly, XGBRegressor also didn’t perform well in predicting the revenue with these features(As shown in Fig 1 Below). It may be because the data are hierarchically structured and these models found it difficult to get any interpretation from them.

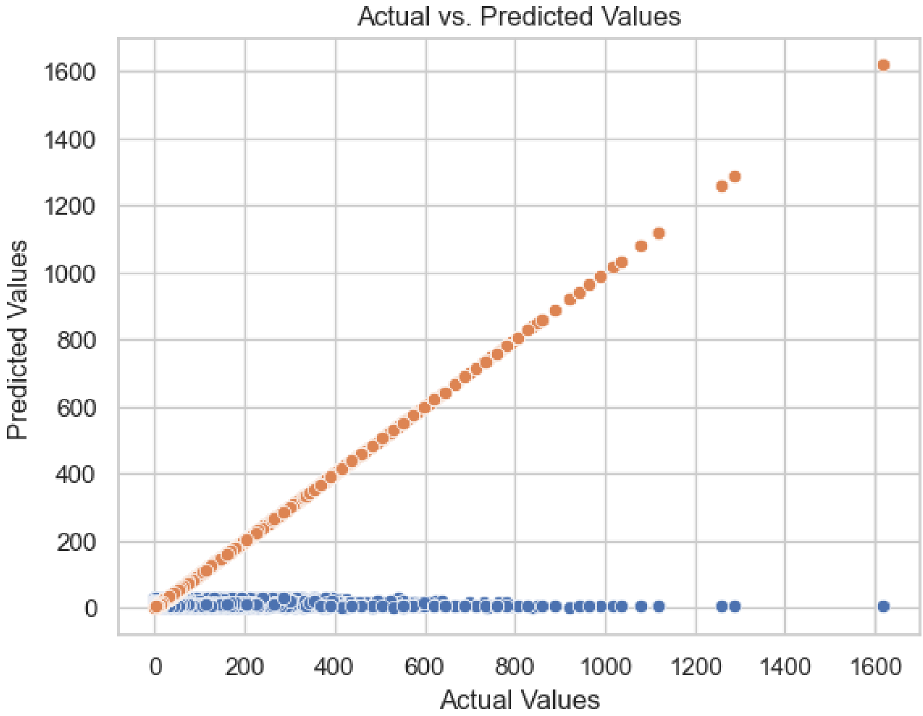

Fig 1: Actual vs Predicted graph for XGB Regressor

The decision Tree Regressor performed as expected for these data sets. Without any tuning, the model overfits the data. We can see from the model performance that MAE and RMSE were absolutely 0 and the model was perfectly fit providing 1 as the R2 score. But, the testing scores prove that the model wasn’t fitted well and it learned too many specific details and noise of the data.

Observing the way that the model could learn something, the Decision Tree Regressor was selected for further hyperparameter tuning. Each of the hyperparameters was manually tuned for each step. However, for each step, GridSearchCV was used to find the best parameters for 5-fold cross-validations and it was optimized to find the best parameters that would give the least RMSE score. The tuned Decision Tree Regressor model had the following hyperparameters:

1. max_depth = 60
2. min_samples_split=39
3. min_samples_leaf=18
4. max_features=None

This model reduced overfitting and performed the best among the other models.

3.b. Business Impact

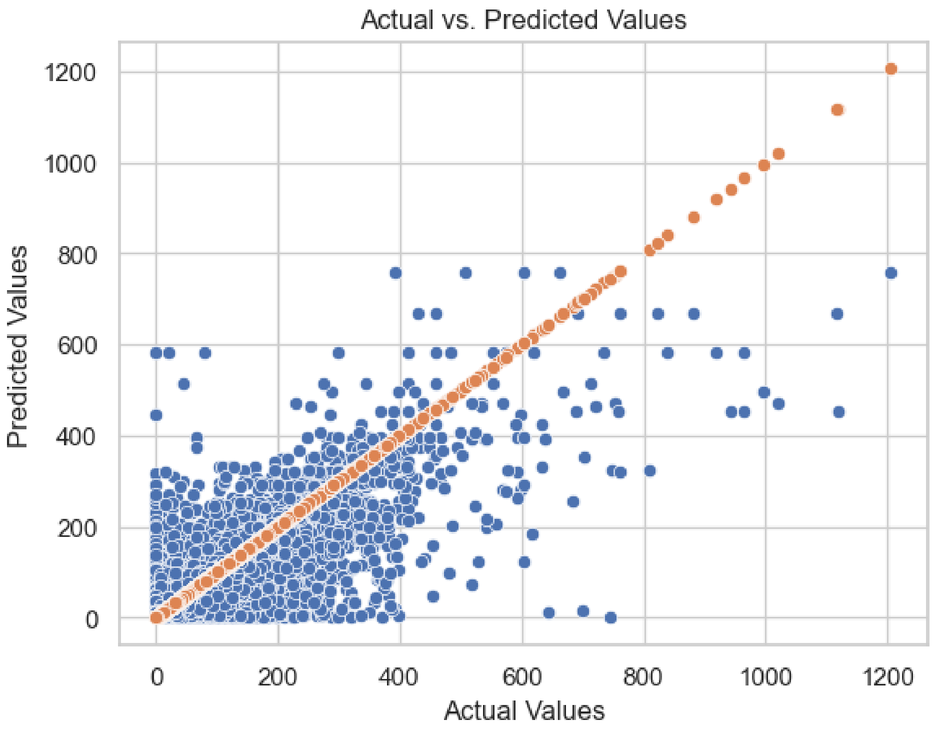

Fig 2: Actual Vs Predicted Values for Tuned Decision Tree Model

MAE score which indicates that the model's predictions are closer to the actual values on average was reduced to 2.41 and the RMSE which indicates that the model's predictions are closer to the actual values, and are sensitive to larger errors was 5.70. R2 score of 0.61 shows that it does have some ability to explain variance. From the figure(i.e. Fig 2) we can see that the model does have some ability to predict and has learned from the dataset.

In terms of business, the model is far better than the naive model, but the impact of prediction errors on inventory management, pricing strategies, and customer satisfaction can vary. According to each item price the MAE and RMSE scores might lead to overstocking or understocking of items which might lead to loss of business profit. This model can be used for estimations but can’t be solely dependent. The regression model might not be a proper fit as it doesn’t observe the trend and seasonality of the data.

3.c. Encountered Issues

To predict or create a robust model for big data like this project, high-powered computer machines or high-power cloud services are required for higher computation and time efficiency. For this project, all of the steps required a minimum of 15 minutes to hours for execution of a single execution and it was difficult to work on advanced techniques due to memory issues.

Due to this, Although cross-validations were used, the model was created on only single test train splits and couldn’t be observed on different data sets which made the model less robust.

4. FUTURE EXPERIMENT	
4.a. Key Learning	Fitting on large data doesn’t always necessarily lead to overfitting of a model. It depends on the type of model and data which overfits. For a hierarchical type of prediction, the tree model performs better than a linear model. A date is required as an input and prediction according to date has more features like the trends and seasonality forecasting models might be better than regression models for solving such problems.
4.b. Suggestions / Recommendations	For this project, if high-powered computation/ resources were available then, more features could be added like event type and names, and also Auto Regressive model combined with tree models might have given more trends and seasonal events information to model which might have increased the model performances. Furthermore, models like Random Forest Regressor could be trained which might have provided more accurate results than just a Decision Tree model. However, it requires higher computational power.