HIGH LEVEL DOCUMENT

Backorder Prediction

Abstract: Material backorder is a common supply chain problem under unpredictable demand and pertinent supply risk. We aim to use Machine Learning predictive models in the area of the business decision process. By predicting products backorder predictive models providing flexibility to the decision authority, better clarity of the process, and maintaining higher accuracy. The tree-based machine learning is chosen to predict material backorder the model. The backorders of products are predicted in this project using Random Forest (RF), sampling techniques and ensemble learning are employed in this particular task.

1.Why this HIGH LEVEL DOCUMENT Document? The purpose of this HIGH LEVEL DOCUMENT (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding, and can be used as a reference manual for how the modules interact at a high level.

1.1. Scope The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system.

2. Project Overview When a customer orders a product, which is not readily available due to lack of unavailability of the product in the store or not in the inventory but can guarantee the delivery of the ordered product at a certain date in the future and the customer waits for the same. This scenario is called backorder of that specific product. Backordering has good and bad faces in the business. So there has to be some balance between the demand and supply. ML Predictive analysis can schedule the production, supply chain and inventory helps to forecast product delivery delay which in return increase the customer satisfaction If the back orders are not handled promptly it reflects high impacts on the company’s revenue, share market price, and customer’s trust that leads to losing the customer as well as the order. On the other hand, to satisfy backorders leads to enormous pressure on different stages of the supply chain which may break (exhaust) or can cause extra costs on production, shipment …etc. Nowadays, most of the company’s use

Machine learning predictive analysis to predict products back orders to overcome the tangible and intangible costs of backorders. We have performed some hypothesis tests considering backorder scenarios. The outcomes of the hypothesis's tests are helpful to choose the appropriate machine learning model for prediction.

3. Backorders vs. Out of Stock An item is out of stock when the seller doesn't have the item in inventory and has no sure date to restock, or the item is seasonal or a limited run. Backordered items are expected to be available in a reasonable timeframe.

4. Problem Statement: Classify the products whether they would go into Backorder (Yes / No) based on the available data from inventory, supply chain and sales. The task is to classify whether a product would go to Backorder based on the given input data. The target variable to predict consists of two values, if it is "Yes" - the predicted product to be treated as Backorder. If the output is "No"- the predicted product to be not going to Backorder So we can have considered this is a Binary Classification problem.

5. Exploratory analysis:

5.1Data Overview

The dataset of this project work has been published in Kaggle. The dataset is divided into the training and testing datasets.

⇥ The dataset is highly imbalanced which should be addressed for accurate predictions by the model.

⇥ Each dataset contains 23 attributes with 1,687,862 and 242,077 observations for the training and testing sets, respectively.

⇥ We have chosen inventory, lead time, sales, and forecasted sale as our predicting variables and 'went on backorder' as our response variable.

⇥ Our target variable is labelled with two classes. Hence, this scenario falls under the binary classification problem.

⇥ The inventory feature indicates an available stock of products, although it contains high numbers of negative records. The negative inventory may arise due to the machine or human error. It may also occur when a shipment is recorded as complete before it arrives.

⇥ The 'lead time' feature indicates the elapsed time between the placement of products' orders and delivery of those products to the customers.

⇥ The lead time in our dataset ranges from 0 to 52 weeks.

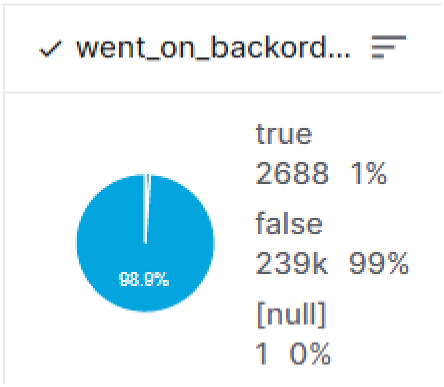
⇥ The sales features are divided into four parts as one-month sale, three months' sale, six months' sale, and nine months' sale.

⇥ The forecasted sale is divided into three columns showing the forecast of three months, six months, and nine months.

⇥ It is observable that the data points of different features have many outliers with different ranges. In both training and testing datasets, a large number of missing values across the predicting variables are observed. Moreover, our response variable is highly imbalanced with 0.669% data from 'Yes' class, and 99.33% data from 'No' class.

⇥ The data samples are distributed among two classes, where 0 indicates 'No' class or no backorder items and 1 indicates 'Yes' class or backorder items.

⇥ Highly Class imbalance: Most of the samples did not went to back order (99.83%) of the time, while actual back orders occurs (0.17%) of the time. The data points which are responsible for extreme positive skewness may not actually be outliers.



✓ went_on_backord...

true
2688 1%

false
239k 99%

[null]
1 0%

5.2. Observations:

⇥ From the 23 features given in the dataset 15 are numerical and 8 (including the target variable) are categorical features.

⇥ The first column 'sku' corresponds to the product identifier which is unique for each data point in the dataset. So this feature can be dropped as it adds no value in output prediction.

⇥ The feature distribution is extremely right skewed because the features mean value is greater than 75th percentile value which depicts the same. Also the difference between the 75th percentile value and max value for each feature is very high which might be due to the presence of outliers.

⇥ The columns perf_6_month_avg and perf_12_month_avg have max. value as 1 and min. value as -99. It seems that the missing values are replaced with -99.

⇥ Out of 1929937, only 13981 i.e., 0.5% of data is Out of Stock (Flag as Yes... as it went to Backorder), and more than 99.5% is in Stock (Flag as No). So its result is biassed

6. Data Cleaning

⇥ Clean missing values.

⇥ Dropped the first column ‘sku’ as it as an identifier and is unique for all the rows in dataset so dropped ‘sku’ column as it is random product id.

⇥ Replace all categorical columns with 1 for yes and 0 for no.

6.1. Handling Missing values:

⇥ Used model based imputation to impute the missing values

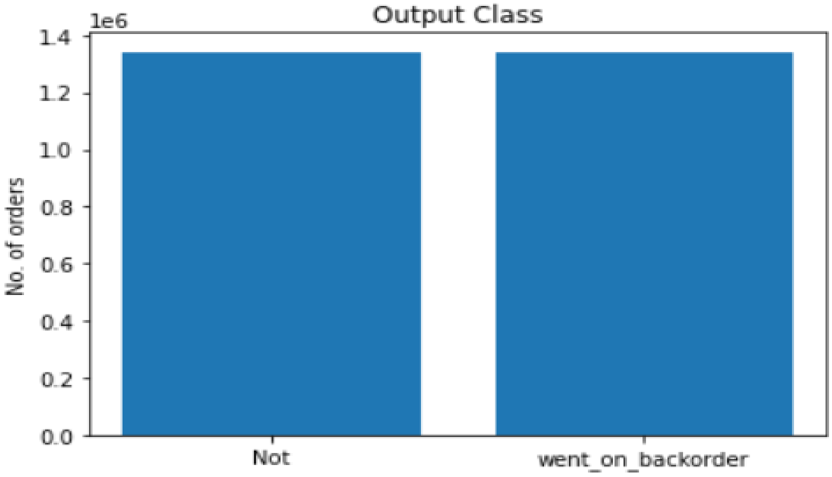
⇥ Few null values observed in the column lead-time.

⇥ Nan values observed in the last row of the training dataset should be dropped. Replaced missing values from the categorical features

6.2. Handling class imbalance:

From EDA it can be concluded that the dataset is extremely right skewed and also is highly imbalanced. Class imbalance can be handled in two ways. Balancing the dataset using different Oversampling or Under sampling techniques or while fitting the model itself different ensembles can fit their base learners on balanced sub-samples.

For handling the imbalance data we have used an Over Sampling technique. After doing this data is completely balanced.



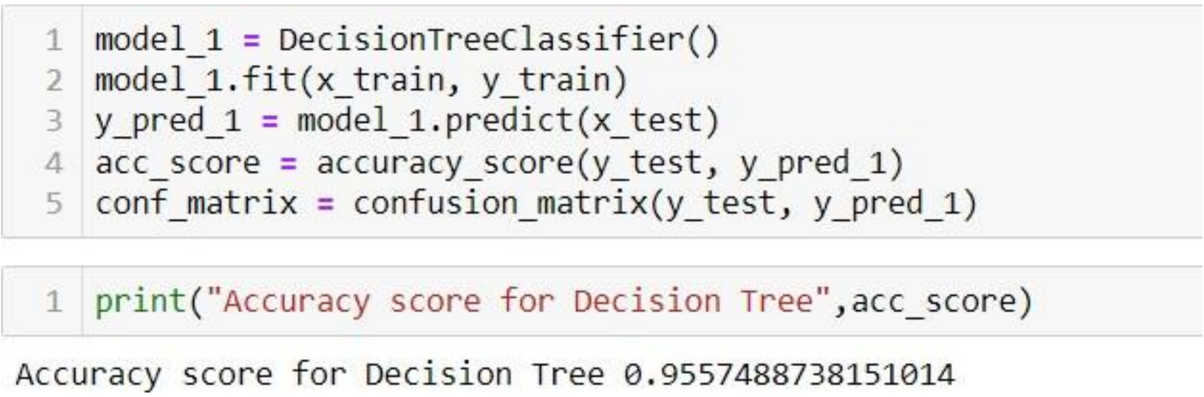
7. Splitting the Dataset:

Split whole data into train and test (80%–20%) using the ‘train_test_split’ method.

8. Modelling:

Decision Tree:

By using Decision Tree Classifier we got an accuracy almost 95%



```
model_1 = DecisionTreeClassifier()
model_1.fit(x_train, y_train)
y_pred_1 = model_1.predict(x_test)
acc_score = accuracy_score(y_test, y_pred_1)
conf_matrix = confusion_matrix(y_test, y_pred_1)
```

```
print("Accuracy score for Decision Tree",acc_score)
```

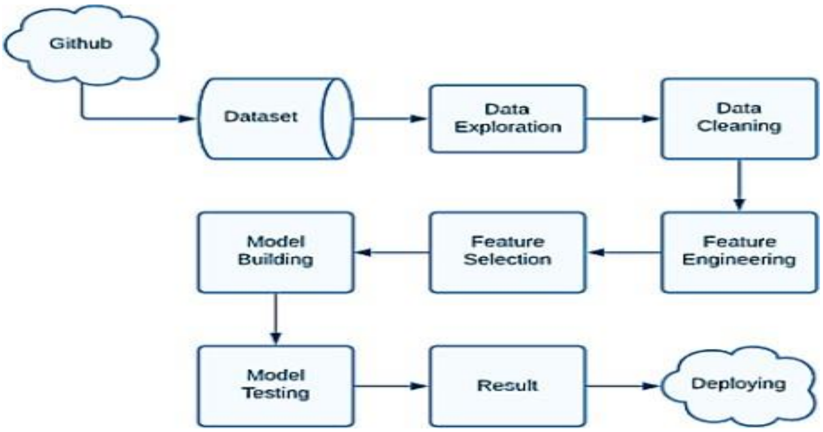
```
Accuracy score for Decision Tree 0.9557488738151014
```

Accuracy of Random Forest and XGBoost is the same as Decision Tree, so we have decided to go with Decision Tree.

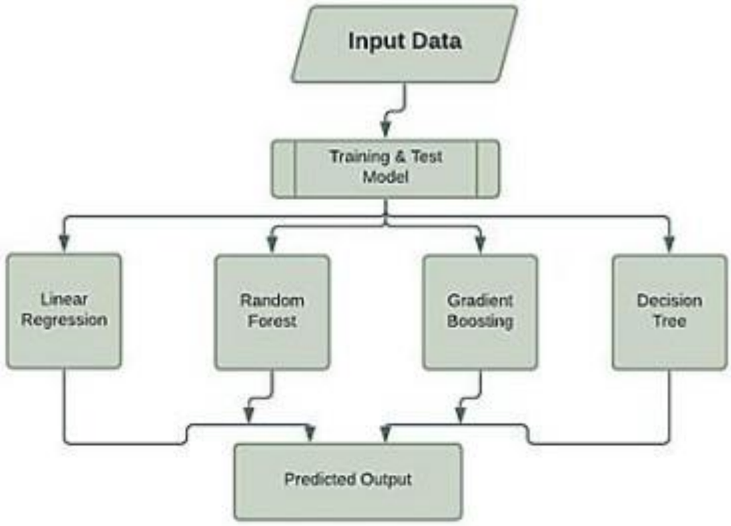
9. Future Work

⇥ Business experts can suggest or with more other effective features.
⇥ Try with more complex ensemble models.
⇥ Deep Learning models tends to works better, we can fine tune the hyper parameters of the existing architecture or can come up with different deep learning architecture.

10. Working procedure of proposed model.



11. Model Training and Evaluation



12. Tools used

Python programming language and frameworks such as NumPy, Pandas, Scikit-learn, Seaborn, Malplotlib are used to build the whole model.

⇥ PyCharm is used as IDE.
⇥ Juypter notebook is used as IDE.
⇥ Matplotlib, Seaborn, Plotly are used for visualisation of plots.
⇥ Flask used as API.
⇥ Cassandra is used to retrieve, insert, delete and update the customer input data.
⇥ Front end development is done using HTML.
⇥ GitHub used as version control.

13. Data Base: - Cassandra DB is used to retrieve, insert, delete and update the customer input database. Flask is a micro web framework written in python. Here we use the flask to create an API that allows sending data, and receives a prediction as a response. Flask API's act as an interface between the user, ML-Model and DB. Users interact with the deployed model on HEROKU and the result will be returned to the user by API. The Flask API will collect all the customer input data stored in Cassandra DB.

14. Reusability:- The written code can be reused and has the possibility to be altered with the help of business experts with more effective features.

15. Conclusion:- This project has been shown to identify those products that will be backordered based on certain features from the known data. The results show that a predictive machine learning classification can control the inventory system, which helps to reduce the pressure of the supply chain. It results in greater flexibility in inventory control and better customer satisfaction at a very low cost. Models like Decision Tree and Random Forest show the highest accuracy score. The XGBoost model also has a good accuracy score.

18.Reference:-

https://apps.dtic.mil/sti/pdfs/AD1141471.pdf
https://journalofbigdata.springeropen.com/articles/10.1186/s40537-020-00345-2