High Level Design (HLD)

Backorder Prediction

Document Version Control

Date Issued	Version	Description	Author
26 July 2023	1	Initial HLD – V1.0	Shubham Vedi
			Akshay Nikam
			Nitin Udmale
			Nikhil Nigudkar

Table of Content

Table of Contents

Abstract

A Backorder is an order which can’t be fulfilled at the given time due to lack of supply or the product is currently out of stock or not in inventory but can guarantee delivery of the goods or service requested by a certain date in the future because the production of goods or replenishment of inventory is underway. Unlike in the situation of Out-of-stock where the delivery date of the goods can’t be promised, in the Backorder scenario the customers are allowed to shop for the products and order. Simply put Backorder can be thought of as an order with a delayed delivery date. The primary goal of all the companies is to increase the demand for the products they offer. Having a poor sales forecast system could one of the reasons for failing to predict the demand. Despite having a good sales forecasting system sometimes these situations are inevitable because of the factors which can’t be controlled or unpredictable events. If many items are going on Backorders consistently it is a sign that companies’ operations are not properly planned and also there is a very high chance of missing out business on the products.

1. Introduction

This document will be used for documenting High-level designs of project.

1.1 Purpose of the Document

The purpose of this plan is to

- Identify different design approaches.
- Identify core modules/sub-systems of the system and sub-system boundary.
- Identify the best suitable technology for various sub-systems.
- Identify areas that need R&D.
- Identify third party components required in the system.
- Identify components, state, life cycle, and communication mechanisms between different sub-systems, and also identify the external interface.
- Identify various usage scenarios.

1.2 Objective of HLD

1. To provide an overview of the entire system.
2. To provide a module-wise breakup of the entire system.
3. To provide introduction and high level working of every module involved.

1.3 Scope of HLD

This HLD covers all areas of system.

2 System Overview

2.1 Product Prospective

The Backorder prediction problem using classification-based Machine Learning algorithms.

2.2 Problem statement

Backorders are unavoidable, but by anticipating which things will be backordered, planning can be streamlined at several levels, preventing unexpected strain on production, logistics, and transportation. ERP systems generate a lot of data (mainly structured) and contain a lot of historical data; if this data can be properly utilized, a predictive model to forecast backorders and plan accordingly can be constructed. Based on past data from inventories, supply chain, and sales, classify the products as going into backorder (Yes or No).

2.3 Proposed Solution

The solution here is a Classification based Machine Learning model. It can be implemented by different classification algorithms (like Logistic Regression, Random forest, Decision tree , XGBoost and so on.Here First we are performing Data preprocessing step, in which Data Profiling ,feature engineering, feature selection, feature scaling ,PCA steps are performed and then we are going to build model.

2.4 Technical Requirements

In this Project the requirements to check to predict the backorder sales for a particular product according to the provided dataset. For that, in this project we are going to use different technologies. Here is some requirements for this project.

- Model should be exposed through API or User Interface, so that anyone can test model.
- Model should be deployed on cloud (Azure, AWS, GCP).
- Cassandra database should be integrated in this project for any kind of user input.

2.5 Data Requirements

Data Requirement completely depend on our problem.

- For training and testing the model, we are using Backorder prediction dataset that is provided by Ineuron Company.
- From user we are taking following input:

→ national_inv
lead_time
in_transit_qty
forecast_3_month
forecast_6_month
forecast_9_month
sales_1_month
sales_3_month
sales_6_month
sales_9_month
min_bank
potential_issue
pieces_past_due
perf_6_month_avg
perf_12_month_avg
local_bo_qty
deck_risk

Label-> went_on_backorder

2.6 Tools Used

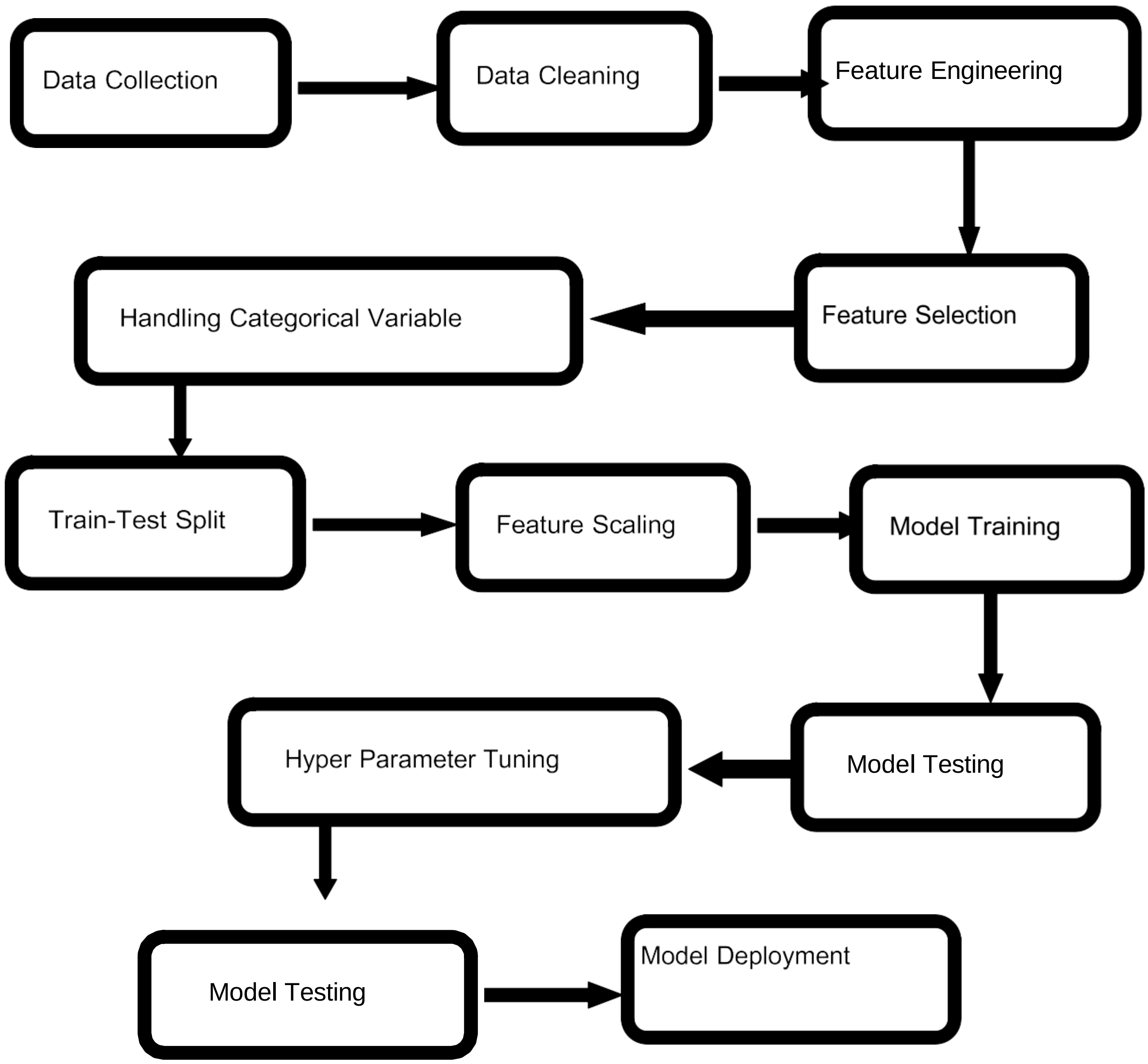
- Visual Studio Code is used as IDE.
- For visualization of the plots, Matplotlib, Seaborn are used.
- Aws is used for deployment of the model..
- Front end development is done using HTML/CSS, Bootstrap.
- Flask is used for Application development
- GitHub is used as version control system.

2.7 Constraints

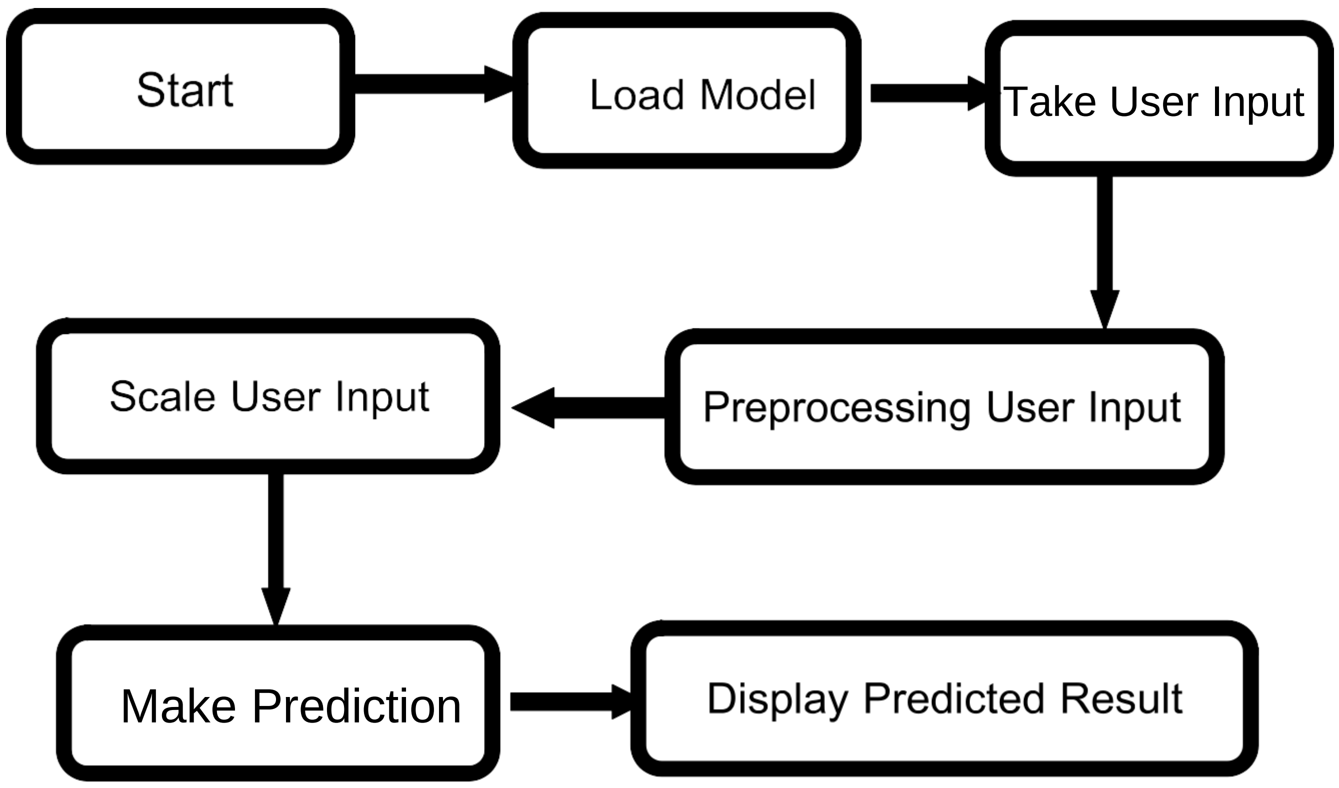
The Backorder prediction system must be user friendly, errors free and users should not be required to know any of the back end working.

3 Design Flow

3.1 Process Flow



3.2 Prediction Process



3.3 Event Log

In this Project we are logging every process so that the user will know what process is running internally.

Step-By-Step Description:

- In this Project we defined logging for every function, class.
- By logging we can monitor every insertion, every flow of data in database.
- By logging we are monitor every step which may create problem or every step which is important in file system.
- We have designed logging in such a way that system should not hang even after so many logging's, so that we can easily debug issues which may arises during process flow.

3.4 Error Handling / Exception Handling

We have designed this project in such a way that, at any step if error occur then our application should not terminate rather it should catch that error and display that error with proper explanation as to what went wrong during process flow.

4 Performance

Solution of Backorder prediction is used to predict the product went on backorder or not in advance, so it should be as accurate as possible so that it should give as much as possible accurate prediction.
That's why before building this model we followed complete process of Machine Learning. Here are summary of complete process:

1. First, we cleaned our dataset properly by removing all null value and duplicate value present in dataset.
2. Then we performed Data profiling where we check no of categorical features and numerical features. We have done statistical analysis,

For most of the features the mean value is greater than the 75th percentile value which depicts that for each feature distribution is extremely right skewed.
Also, the difference between the 75th percentile value and max value is 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 has max. Value as 1 and min. value as -99. It seems that the missing values are replaced with -99.

3. After that we performed feature engineering step in which the sku column has a unique value for each row, so we can use it as the index column or drop it and use the existing index. We have handled categorical value, Check the outlier, and replace the Nan value also.
4. We convert categorical variable by performing python map function.
5. Then I split the whole data set train-test split. After that I performed scaling on X_Train and X_test.
6. After performing above step I was ready for model training. In this step, I trained my dataset on different classification algorithm (Logistic, Random-Forest, Decision Tree, Support Vector Classifier,). After training the dataset on different algorithms I got highest accuracy of 92.27% on Decision Tree

1. After that I applied hyper-parameter tuning on all model which I have described above. Here also I got highest accuracy of 93.89% on test dataset by same Random Forest (Classification).
2. After that I saved my model in pickle file format for model deployment.
3. After that my model was ready to deploy. I deployed this model on various cloud storage(AWS) and GCP and also dockerize this model.

	Train F1 Score	Total F1 Score	Train Score	Test Score	Model Accuracy
Logistic Regression	0.5319	0.5872	0.572	0.558	0.558
Random Forest	0.9607	0.9181	0.9609	0.9389	0.9389
Decision Tree	0.9077	0.8963	0.9511	0.9227	0.9227
Support Vector Machine(SVC)	0.5442	0.5881	0.6396	0.5653	0.5653

4.1 Re-usability

We have done programming of this project in such a way that it should be reusable. So that anyone can add and contribute without facing any problems.

4.2 Application Compatibility

The different module of this project is using Python as an interface between them. Each modules have its own job to perform and it is the job of the Python to ensure the proper transfer of information.

4.3 Resource Utilization

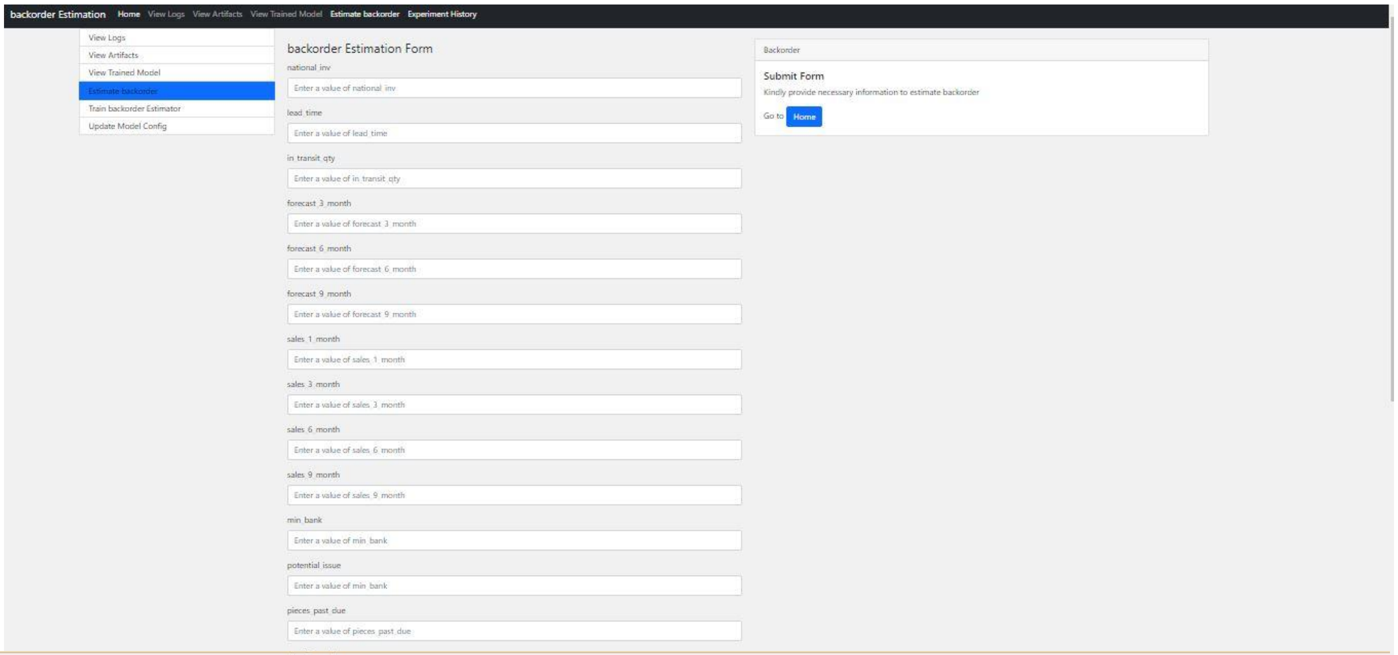
In this project, when any task is performed, it will likely that the task will use all the processing power available in that particular system until it's job finished. By keeping this in mind, In this project we have used the concept of multithreading.

4.4 Deployment

we have deployed this on cloud using AWS.

4.5 User Interface

We have created an UI for user by using HTML and CSS.



backorder Estimation Form

national inv
Enter a value of national inv

lead time
Enter a value of lead time

in transit qty
Enter a value of in transit qty

forecast 3 month
Enter a value of forecast 3 month

forecast 6 month
Enter a value of forecast 6 month

forecast 9 month
Enter a value of forecast 9 month

sales 1 month
Enter a value of sales 1 month

sales 3 month
Enter a value of sales 3 month

sales 6 month
Enter a value of sales 6 month

sales 9 month
Enter a value of sales 9 month

min bank
Enter a value of min bank

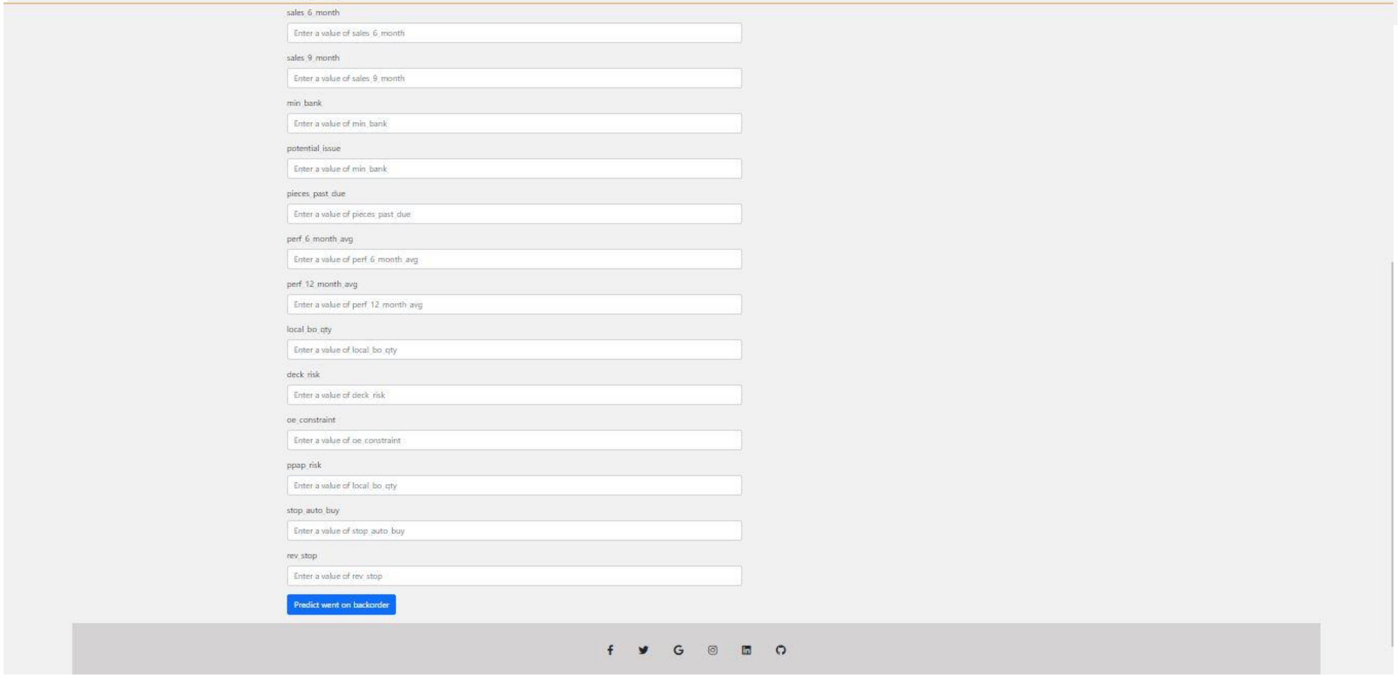
potential issue
Enter a value of min bank

pieces past due
Enter a value of pieces past due

Submit Form

Kindly provide necessary information to estimate backorder

Go to Home



sales 6 month
Enter a value of sales 6 month

sales 9 month
Enter a value of sales 9 month

min bank
Enter a value of min bank

potential issue
Enter a value of min bank

pieces past due
Enter a value of pieces past due

perf 6 month avg
Enter a value of perf 6 month avg

perf 12 month avg
Enter a value of perf 12 month avg

local bo qty
Enter a value of local bo qty

deck risk
Enter a value of deck risk

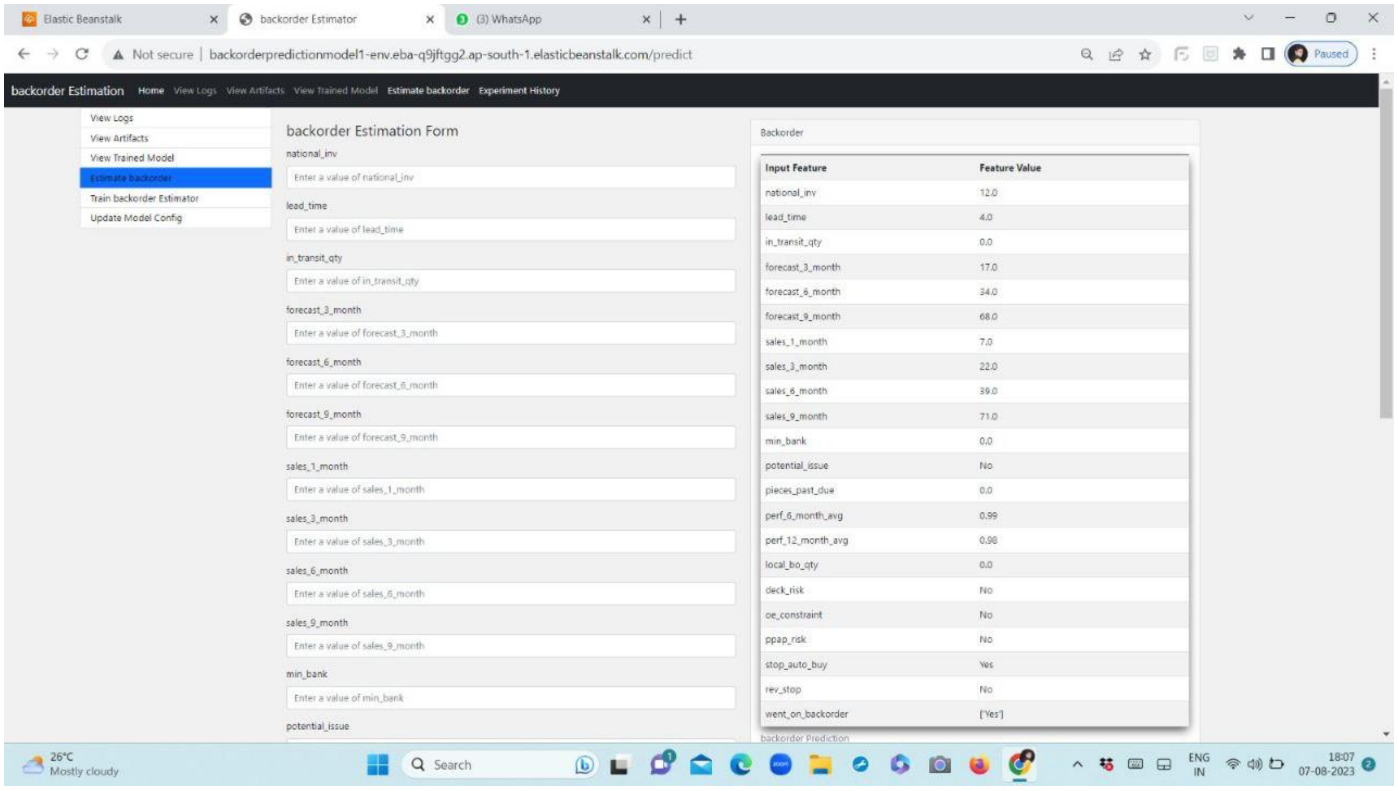
oe constraint
Enter a value of oe constraint

ppap risk
Enter a value of local bo qty

stop auto buy
Enter a value of stop auto buy

rev stop
Enter a value of rev stop

OUTPUT:



backorder Estimation Form

national_inv	Enter a value of national_inv
lead_time	Enter a value of lead_time
in_transit_qty	Enter a value of in_transit_qty
forecast_3_month	Enter a value of forecast_3_month
forecast_6_month	Enter a value of forecast_6_month
forecast_9_month	Enter a value of forecast_9_month
sales_1_month	Enter a value of sales_1_month
sales_3_month	Enter a value of sales_3_month
sales_6_month	Enter a value of sales_6_month
sales_9_month	Enter a value of sales_9_month
min_bank	Enter a value of min_bank
potential_issue	

Backorder

Input Feature	Feature Value
national_inv	12.0
lead_time	4.0
in_transit_qty	0.0
forecast_3_month	17.0
forecast_6_month	34.0
forecast_9_month	68.0
sales_1_month	7.0
sales_3_month	22.0
sales_6_month	39.0
sales_9_month	71.0
min_bank	0.0
potential_issue	No
pieces_past_due	0.0
perf_6_month_avg	0.99
perf_12_month_avg	0.98
local_bo_qty	0.0
deck_risk	No
oe_constraint	No
ppap_risk	No
stop_auto_buy	Yes
rev_stop	No
went_on_backorder	['Yes']

backorder Prediction

5 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.