HIGH LEVEL DESIGN

Back order Prediction

Revision Number: 1.0

Last date of revision: 02/08/2021

Document Version Control

Date Issued	Version	Description	Author
02/06/2021	1	Initial HLD	Naveen Vinayak S

Contents

Document Version Control……2

Abstract……4

1 Introduction……5

1.1 Why this High-Level Design Document?……5
1.2 Scope……5
1.3 Definitions……5

2 General Description……6

2.1 Product Perspective……6
2.2 Problem Statement……6
2.3 Proposed Solution……6
2.4 Further Improvements……6
2.5 Technical Requirements……6
2.6 Data Requirements……7
2.7 Tools Used……8
2.8 Constraints……9
2.9 Assumptions……9

3 Design Details……10

3.1 Process Flow……10

3.1.1 Model Training and Evaluation……10
3.1.2 Deployment Process……11

3.2 Event Log……11
3.3 Error Handling……11
3.4 Performance……12
3.5 Reusability……12
3.6 Application Compatibility……12
3.7 Resource Utilization……12
3.8 Deployment……12

4 Dashboards……13

4.1 KPIs (Key Performance Indicators) ……13

5 Conclusion……14

Abstract

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 also 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).

1 Introduction

1.1 Why this High-Level Design Documentation?

The purpose of this High-Level Design documentation is to add necessary details to the current project description to represent 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

The HLD will:

- Present all the design aspects and define them in detail
- Describe the user interface being implemented
- Describe the hardware and software interfaces
- Describe the performance requirements
- Include design features and architecture of the project
- List and describe the non-functional attributes like
  - Security
  - Reliability
  - Maintainability
  - Portability
  - Reusability
  - Application Compatibility
  - Resource utilization
  - Serviceability

1.2 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.

1.3 Definitions

TERM	DESCRIPTION
DATABASE	Collection of all the information monitored by this system
IDE	Integrated Development Environment
AWS	Amazon Web Services

2 General Description

2.1 Product Perspective

The Backorder prediction solution system is a Machine Learning-based predictive model which help us to predict whether the product will go backorder or not

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 also 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 proposed here is backorder prediction system, the model is built on ERP generated historical data. Based on inventories, supply chain, sales model will classify the product as going into backorder or not, so that the management can take appropriate action.

2.4 Further Improvements

Backorder prediction can be added with more ERP system in Ecommerce sites to classify whether the product will go backorder or not so that the necessary stocks can be added in inventory beforehand itself.

2.5 Technical Requirements

This document addresses the requirements for classifying the backorder for products in Ecommerce using ERP system generated data at early stages itself and recommending the necessary and rapid action to avoid backorder in inventory.

Cassandra Data Base is required to store the data.

2.6 Data Requirements

Data Requirement completely depends on our problem statement.

- We need inventory and sales data which is generated by ERP system (historical data).
- We need dataset to be properly balanced for each class with notations.
- Data set can be a Live streaming dataset or CSV or Excel file with comma or space or point as delimiter.

2.7 Tools Used

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

- PyCharm is used as IDE.
- For visualization of data Plotly, Matplotlib, Seaborn are used.
- AWS is used for deployment of model.
- Tableau/Power BI is used for dashbord creation.
- Cassandra is used to retrieve , insert, delete and update the database.
- Front end development is done using HTML/CSS.
- Python Diango/Flask is used for backend development.
- GitHub is used as version control system.

2.8 Constraints

Backorder based prediction solution system must be user friendly, as automated as possible and user should not be required to know any of the workings.

2.9 Assumptions

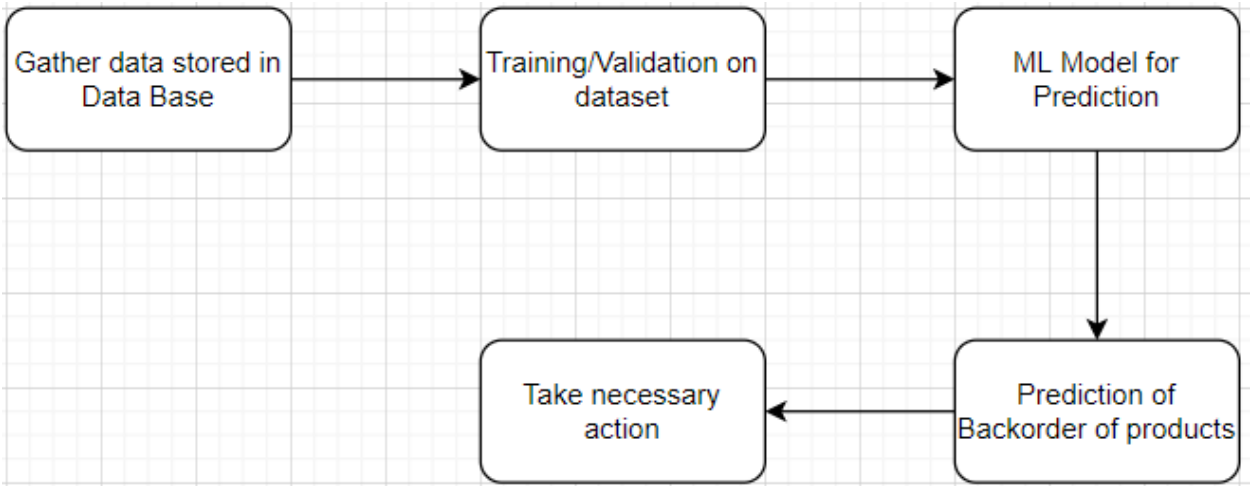
The main objective of the project is the implement the use cases as previously mentioned(2.2 Problem Statement) for new dataset that comes through live schudeler. It is also assumed that all aspects of this project have the ability to work together in the way designer is expecting.

3 Design Details

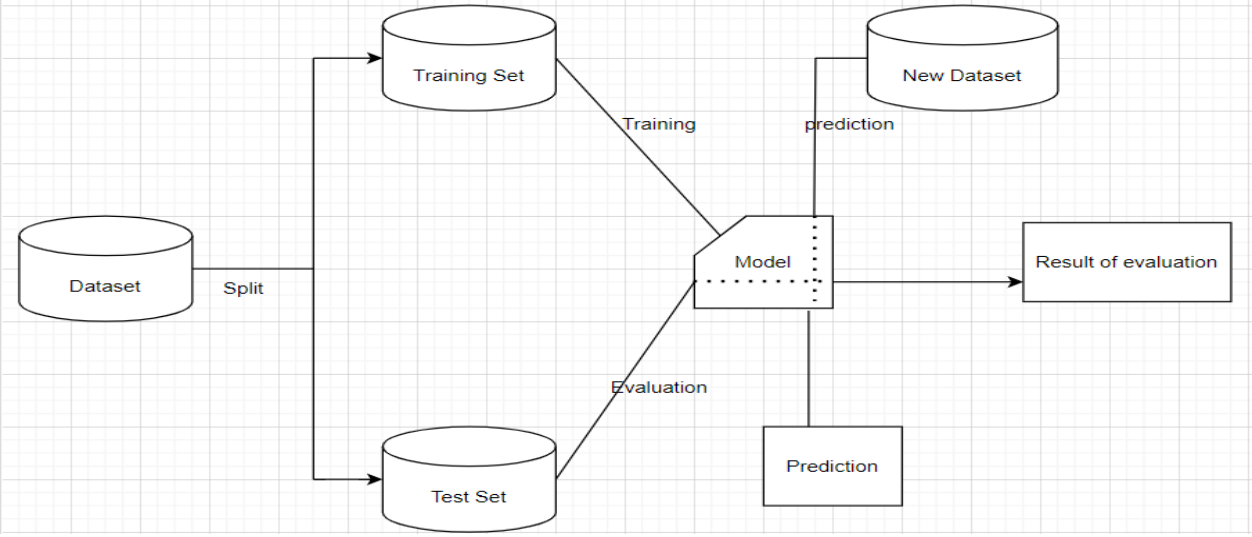
3.1 Process Flow

For identifying the Backorder of product, we will use machine learning model. Below is the process flow diagram.

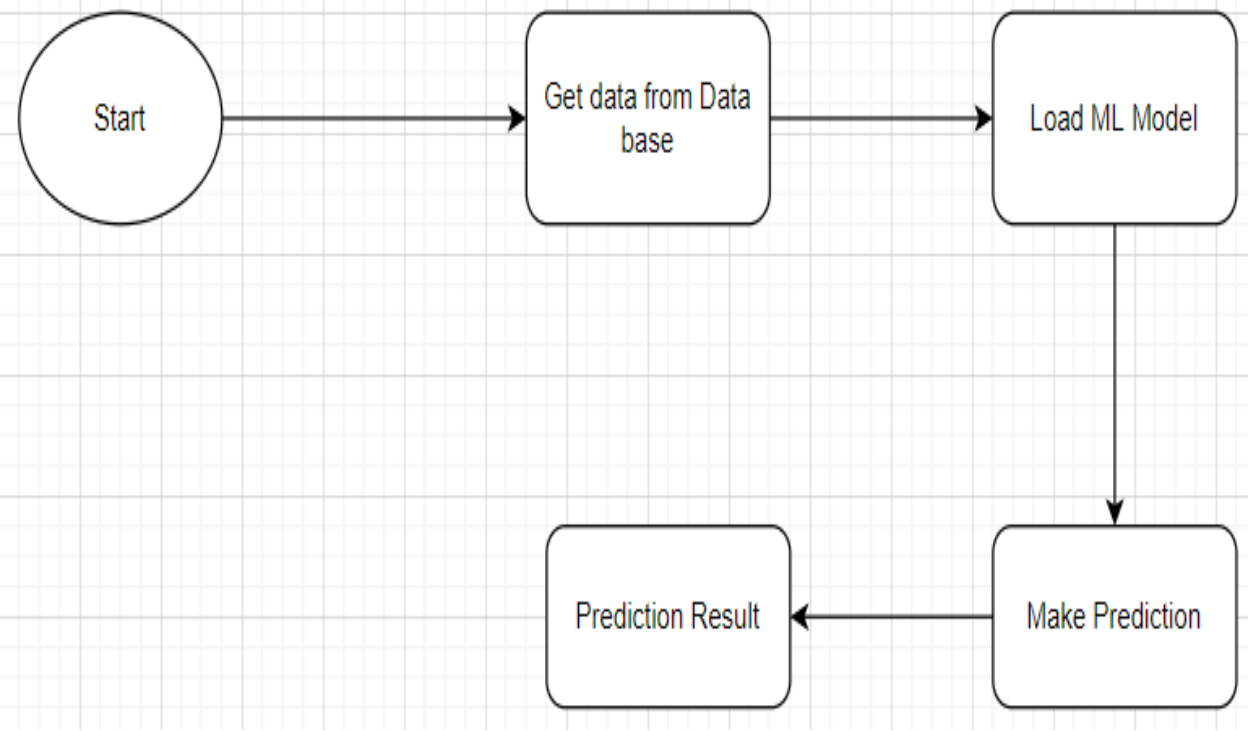
Proposed Methodology



3.1.1 Model Training and Evaluation



3.1.2 Deployment Process



3.2 Event Log

The system should log every event so that the user will know what process is running internally.

Initial Step-By-Step Description:

1. The System identifies at what step logging required
2. The System should be able to log each and every system flow.
3. Developer can choose logging method. You can choose database logging / file logging as well.
4. System should not hang even after using so many loggings. Logging just because we can easily debug issues, so logging is mandatory to do

3.3 Error Handling

Should errors be encountered, an explanation will be displayed as to what went wrong? An error will be defined as anything that falls outside the normal and intended usage.

3.4 Performance

Backorder prediction solution is used for classifying whether the product will go backorder or not, it will also inform to take necessary actions, so it should be accurate as possible, so that will not mislead the concern authorities. Also, model retraining is very important to improve the performance of application.

3.5 Reusability

The code written and the components used should have the ability to be reused with no problems.

3.6 Application Compatibility

The different components for this project will be using Python as an interface between them. Each component will have its own task to perform, and it is the job of the python to ensure proper transfer of information.

3.7 Resource Utilization

When any task is performed, it will likely use all the processing power available until that function is finished.

3.8 Deployment

4 KPI's (Key Performance Indicators)

1. Key indicators displaying a summary of the prediction for the given new dataset.
2. Avoiding the backorder of products

5 Conclusion

The Designed product will predict backorder in the dataset and will be used to train our algorithm, so we can identify the backorder and take necessary action to prevent empty stock of ordered products in inventory.