Low Level Design

BACK ORDER PREDICTION

WRITTEN BY M.SHABARISH

Table of Contents

1. Introduction

1.1 What is Low-Level design document

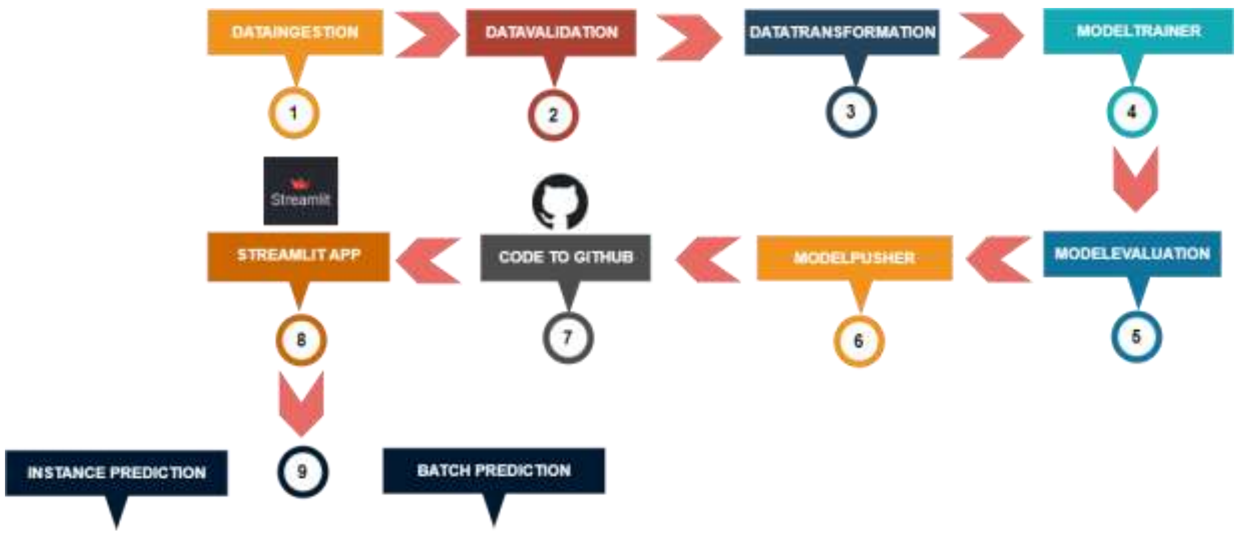
A low level design document (LLD) is a detailed technical document that describes how a software system or application will be implemented at a low level. The purpose of an LLD is to provide a blueprint for developers and stakeholders to understand the technical details of the system, including the architecture, components, modules, interfaces, algorithms, data structures, and other implementation details.

1.2 Scope

Low-level design (LLD) is a component-level design process that follows a step-by-step refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work.

2. Architecture

2.1 Project Architecture:



3. Architecture Description

3.1 Data Gathering:

The data for the current project is taken from the Github raw file

https://github.com/rodrigosantis1/backorder_prediction/blob/master/dataset.rar

3.2 Data Description:

There are about 16lakh+ records in backorder dataset. The information includes inventory level of the products, transit time, forecast sales for prior months, performance level of the products for prior months, sales quantity for the products for prior months and some risk factor labels of the products.

3.3 Tool Used:

• Python 3.8 is used while creating the environment.
• Amazon S3 buckets are used to store the data.
• VS Code is used as IDE.
• GitHub is used as code repository.
• Streamlit application is used to host the application.

3.4 Insert data to Amazon S3

The data has huge records for efficient storing and retrieving the data is stored in parquet form and inserted in Amazon S3 buckets.

3.5 Data Transformation:

• Removing the duplicate records from the data.
• Handling unusual data in the columns by filling the missing values.
• Encoding the categorical variables using OneHotEncoder.
• Transforming the Numerical features.

3.5 Model Building:

Pre-processed data has been passed to various machine learning classification algorithms. All the models are checked with f1-scores.The RandomForestClassifier perform among the given classification algorithms.

3.6 Model Evaluation:

Currently trained model is evaluated with the previously saved trained model, if available. Ifcurrently trained model is better than the previously trained model then the current model and transformer objects are pushed and saved for the future use.

3.7 Model Pusher:

Pushing the model to saved models if the current trained model is better than the previous model.

3.8 Instance and Batch Prediction:

Both the instance and batch prediction can be performed using the code.

- In instance prediction we need to give the inputs through the streamlit application and can get prediction as output on the screen.
- In Batch prediction you need to enter the github raw file URL to perform batch prediction.
- And it displays the predictions as well as predictions file will be downloadable.

Github url For batch prediction:
https://github.com/medashabari/BackorderPrediction/raw/main/Kaggle_Test_Dataset_v2.csv

3.9 Deployment:

Streamlit framework is used to deploy the application.