Backorder Prediction Project

High-Level Design (HLD)

Document Version:1.0

Control Version Date: 22-Feb-24

Author : Akshay kumar BM

Comments: Introduction & Architecture defined

Contents

Abstract

This High-Level Design (HLD) outlines the architectural blueprint and strategic approach for developing a machine learning-based credit card default prediction system. In the contemporary financial landscape, the ability to accurately forecast credit card defaults is indispensable for risk management and maintaining a healthy credit portfolio. The HLD encapsulates the various stages of the project lifecycle, commencing with the acquisition of diverse datasets encompassing historical credit card transactions, customer demographics, economic indicators, and potentially supplementary sources of data. Rigorous preprocessing techniques are applied to cleanse, engineer features, and standardize the datasets, priming them for subsequent modeling endeavors. Employing a supervised learning paradigm, the project explores an array of algorithms, including logistic regression, decision trees, random forests, gradient boosting, and neural networks, to construct predictive models. Algorithm selection is contingent upon factors such as performance metrics, interpretability, and computational efficiency. Validation and evaluation protocols are paramount to ascertain the efficacy and generalizability of the models. Techniques such as cross-validation, hyperparameter tuning, and comprehensive performance metrics evaluation are deployed to assess model performance rigorously. Following model development, extensive testing procedures are undertaken to validate the robustness and reliability of the predictive models across diverse real-world scenarios. This phase encompasses simulating various conditions and evaluating the model's performance under different circumstances. The deployment phase entails seamless integration of the trained model into the existing credit risk management infrastructure. The model may be deployed either as a standalone application or as an integral component within a broader decision-making framework. Continuous monitoring and iterative updates are pivotal to adapt to evolving market dynamics and sustain model efficacy over time

High-Level Design Document for Backorder Prediction Project

I. Introduction

The Backorder Prediction Project is a machine learning-based solution designed to predict whether a product will go on backorder. This document provides a high-level overview of the project's design and architecture.

II. General Description

2.1. Product Perspective

The product is a machine learning model that predicts product backorders based on various features such as national inventory, lead time, in-transit quantity, forecast sales for the next three months, sales quantity in the prior month, minimum recommended stock level, and six-month average performance.

2.2. Problem Statement

Predicting backorders is crucial for businesses to manage their inventory efficiently and meet customer demand. The challenge is to build a model that can accurately predict whether a product will go on backorder.

2.3. Proposed Solution

The proposed solution is a machine learning pipeline that includes data ingestion, data transformation, model training, and prediction. The pipeline is implemented in Python, leveraging various libraries for data processing and machine learning.

2.4. Technical Requirements

The project requires Python 3.x and the following Python libraries: pandas, numpy, sklearn, os, sys, pickle, and logging.

2.5. Data Requirements

The model is trained on a dataset with features such as national inventory, lead time, in-transit quantity, forecast sales for the next three months, sales quantity in the prior month, minimum recommended stock level, and six-month average performance. The target variable is whether the product went on backorder.

2.6. Tools Used

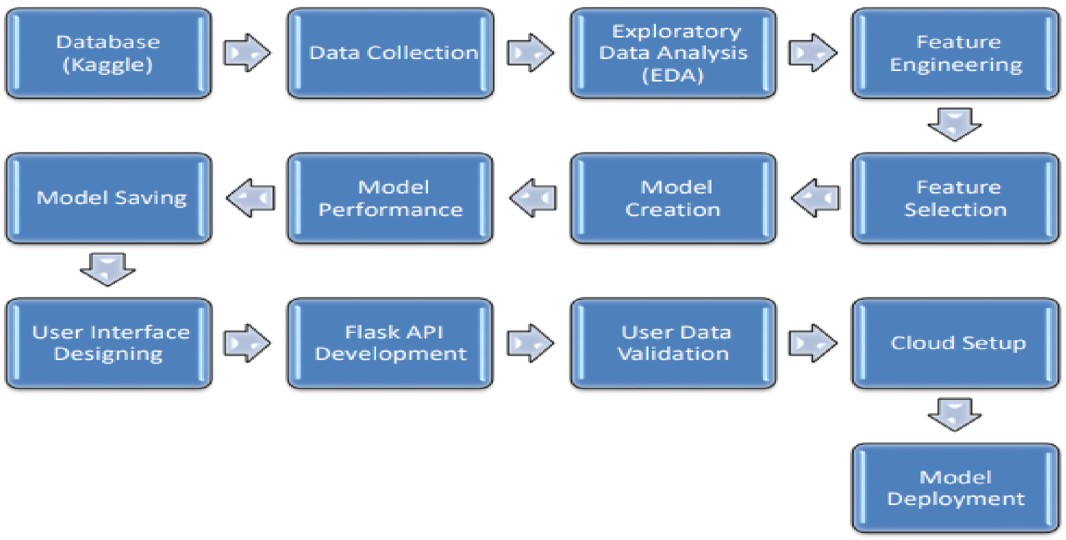
The project uses Python as the primary programming language, with libraries such as pandas for data manipulation, sklearn for machine learning, and logging for generating logs.

2.7. Constraints

The accuracy of predictions is dependent on the quality and representativeness of the training data. Additionally, the model may need to be retrained periodically to adapt to changing patterns in the data.

III. Design Details

3.1. Process Flow

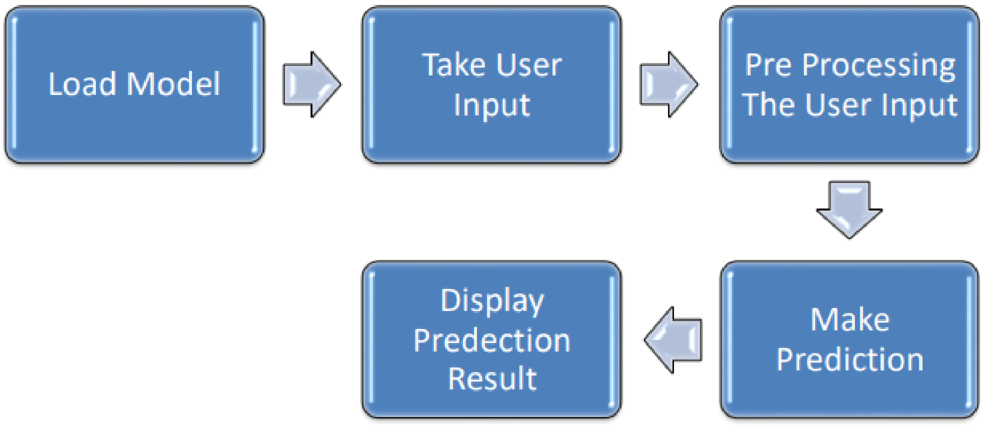


The process flow involves the following steps:

1. **Data Ingestion**: The data ingestion component reads the training and testing data from CSV files.
2. **Data Transformation**: This component preprocesses the data, including handling missing values, scaling numerical features, and encoding the target variable.
3. **Model Training**: Various machine learning models are trained on the preprocessed data, and the best-performing model is selected.
4. **Prediction**: The trained model is used to make predictions on new data.

3.2. Deployment Process

The trained model, along with the preprocessing object, is saved as a pickle file for future use in making predictions.



IV. Performance

4.1. Reusability

The code is modular and components such as data ingestion, data transformation, and model training are implemented as separate classes, enhancing reusability.

4.2. Application Compatibility

The application is compatible with any system that supports Python.

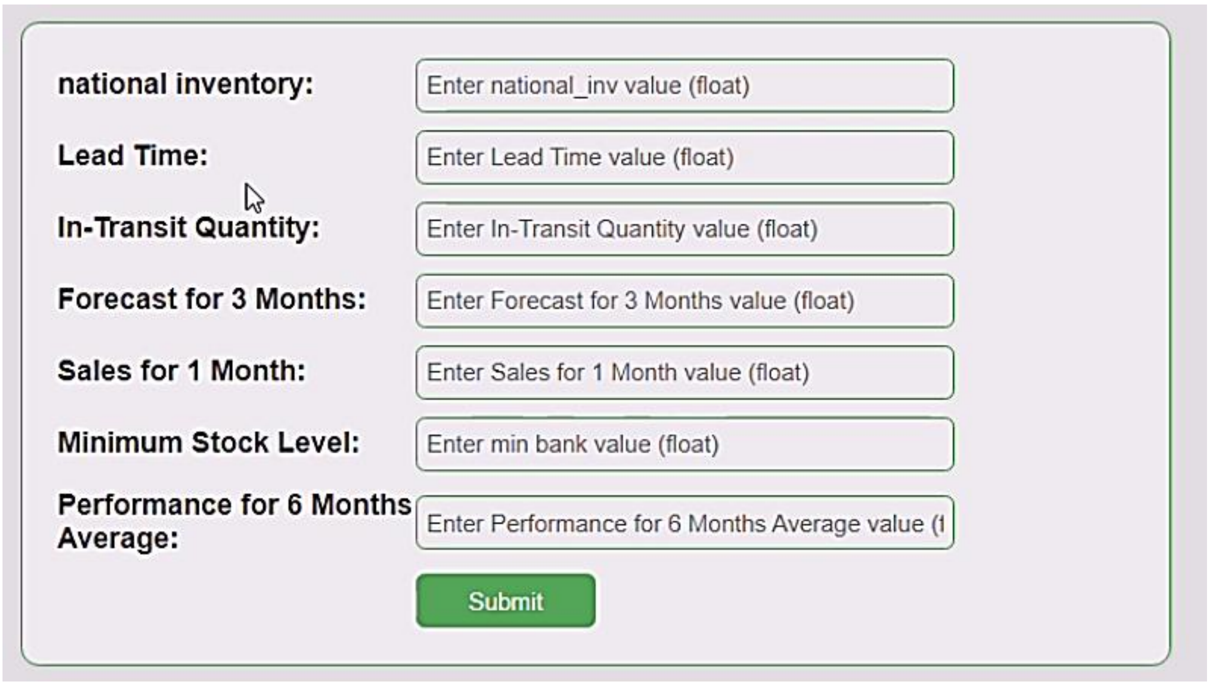
4.3. Resource Utilization

The resource utilization of the application depends on the size of the training data and the complexity of the machine learning models used.

4.4. Deployment

The trained model can be deployed in any environment that supports Python. The model, along with the preprocessing object, is saved and can be loaded to make predictions.

4.5 User interface



national inventory:	Enter national_inv value (float)
Lead Time:	Enter Lead Time value (float)
In-Transit Quantity:	Enter In-Transit Quantity value (float)
Forecast for 3 Months:	Enter Forecast for 3 Months value (float)
Sales for 1 Month:	Enter Sales for 1 Month value (float)
Minimum Stock Level:	Enter min bank value (float)
Performance for 6 Months Average:	Enter Performance for 6 Months Average value (f

V. Conclusion

The Backorder Prediction Project provides a machine learning-based solution to predict product backorders, helping businesses manage their inventory more efficiently. The project demonstrates the use of Python and various libraries to implement a machine learning pipeline, from data ingestion and preprocessing to model training and prediction. The modular design enhances reusability, and the application can be deployed in any Python-supported environment. Future work may include exploring more features and using more advanced machine learning techniques to improve prediction accuracy.