Credit Card Default Prediction Model

High Level Design (HLD)

Ramchandra Tukaram
Padwal

Revision Number 3.0
Revised Date 08/06/2023

Document

Date Issued	Version	Description
20/04/2023	1.0	Initial HLD
22/05/2023	2.0	Flowchart
08/06/2023	3.0	Final

Contents

Abstract

In present day majority of people are using credit card for online shopping, EMI payments. Moreover, banks are encouraging customers to use it by approving them freely without any initial charges. Because credit card interest income is the major share of profit for all commercial banks. Even if customers are not willing to buy, bank marketing team reaches out through phone call to sell it. At first the interest rate is normal to grope the customer into the practice of using it, later they will levy charges on everything to loot money from the customers. Due to this reason many customers are facing difficulty in repaying the credit amount on time. If they missed the deadline the interest rate will get multiplied leading to defaulting the account if left unpaid even for a short span of time. The model we are going to build will consider all inputs from user to predict whether the account will be defaulted or not based on various parameters.

Introduction

Why High-Level Design?

The main purpose of HLD is to add the necessary details 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. Document includes the following aspects:

✓ Architecture Design and Definition

✓ User Interface Implementation

✓ Hardware/Software Interface

✓ Performance Requirements

✓ Non – functional attribute definitions

✓ Security, Reliability, Maintainability, Portability, Reusability, Application Compatibility, Resource Utilization, Serviceability

Scope

Documentation provides the system structure, such as DB architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non– technical terms which should be understandable to the administrator.

Definitions

Term	Description
CCDD	Credit Card Default Detection Model
CIBIL	Credit Information Bureau Limited
IDE	Integrated Development Editor
MLA	Machine Learning Algorithm
EDA	Exploratory Data Analysis
ETL	Extract, Transform, Load

General Description

Problem Statement

To create machine learning model to detect anomalies in credit repayment transaction and predict whether the account will be defaulted or not.

Proposed Solution

The proposed model will be capable of studying customers based on various parameters like gender, education qualification, marital status, previous repayment history and current outstanding balance. Based on that it can predict the account default status in advance and notify the respective bank official for further action.

Further Improvements

Further we can enhance our model with additional information like customer expenditure pattern, CIBIL score, loan EMI or availability of any fixed deposit mapped to the account. These information's will improve the prediction accuracy and can suggest alternative methods to retrieve the pending dues in-case of future default.

Requirements

Model performance depends on various factors like dataset we use to train it, prediction algorithm, hyper tuning parameters

- ✓ Input Dataset – Data should be clean without any outliers, missing values, garbage values
  - ✓ Outliers – can greatly impact the model performance
  - ✓ Missing/Garbage values – Data collected often have missing and garbage values which must cleaned before training the model
- ✓ Choice of algorithm – Depending on the problem statement appropriate algorithm is used to train

Tools

Python programming language is our primary tool to build the model with few open-source libraries for various subtasks

Pandas

Python package providing fast, flexible and expressive data structure designed to make working with relational or labelled data both easy and intuitive

NumPy

Python library used for working with arrays with domain specific functionalities in Linear algebra, Fourier transform and Matrices.

Scikit-Learn

High level framework designed for supervised and unsupervised machines learning algorithms. It provides efficient tools for statistical and scientific models for illustrations.

Plot Visualization

Seaborn and Matplotlib libraries used to create 2D graphs and plots to visualize the data

HTML/CSS

HTML is used to create static web pages and web application whereas CSS is a style sheet responsible for presentation of documents written in HTML

Flask

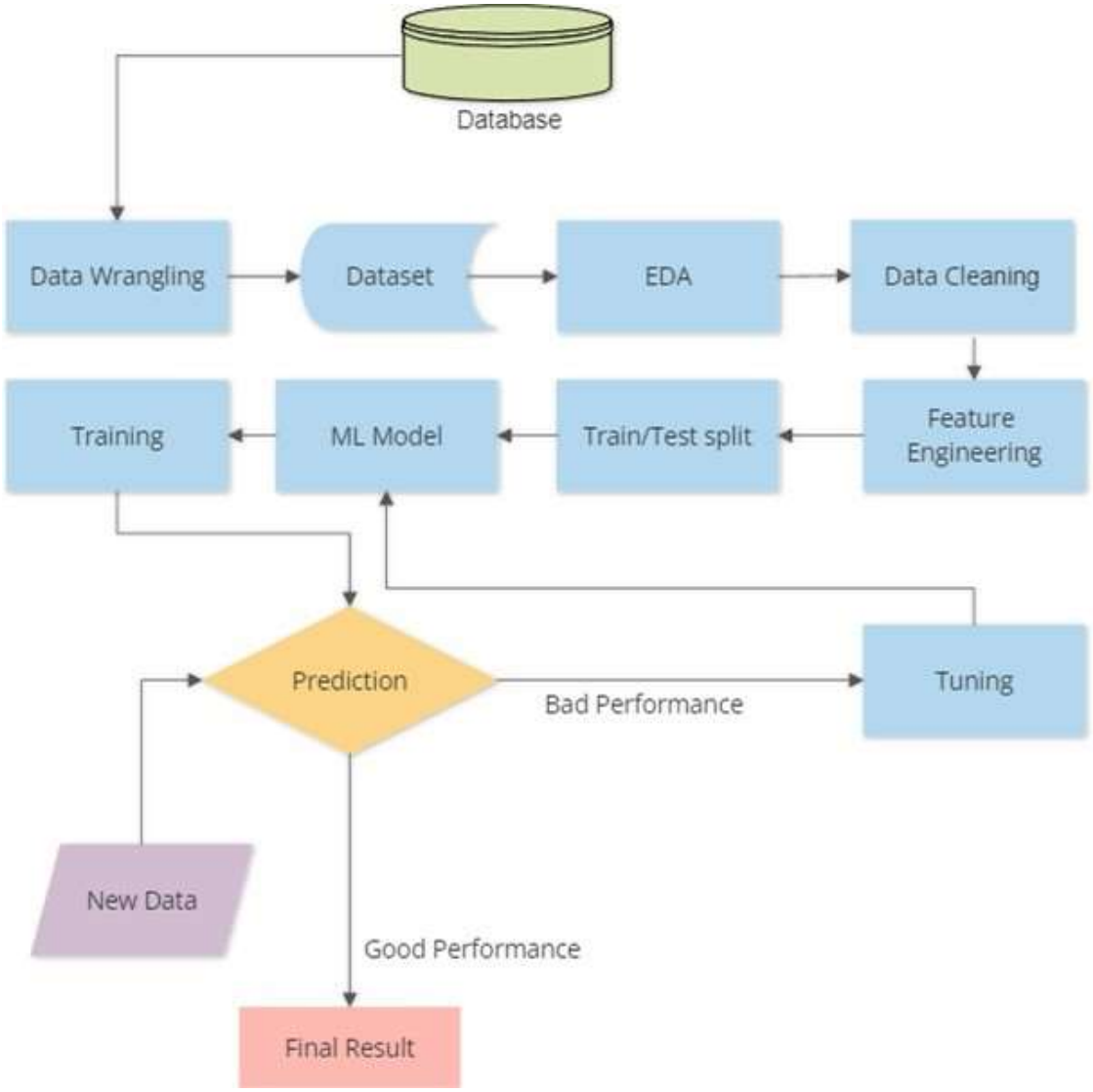
Small lightweight python web framework that provides useful tools and features to make API based web applications.

GitHub

Cloud based online software development platform used for storing, tracking, and collaborating on software projects. It enables developers to upload their own code files and to collaborate with fellow developers on open-source projects.

Design Architecture

Model Training and Evaluation



Data Wrangling

Process of transforming and mapping the raw data into desirable format with the intent of making it more valuable dataset

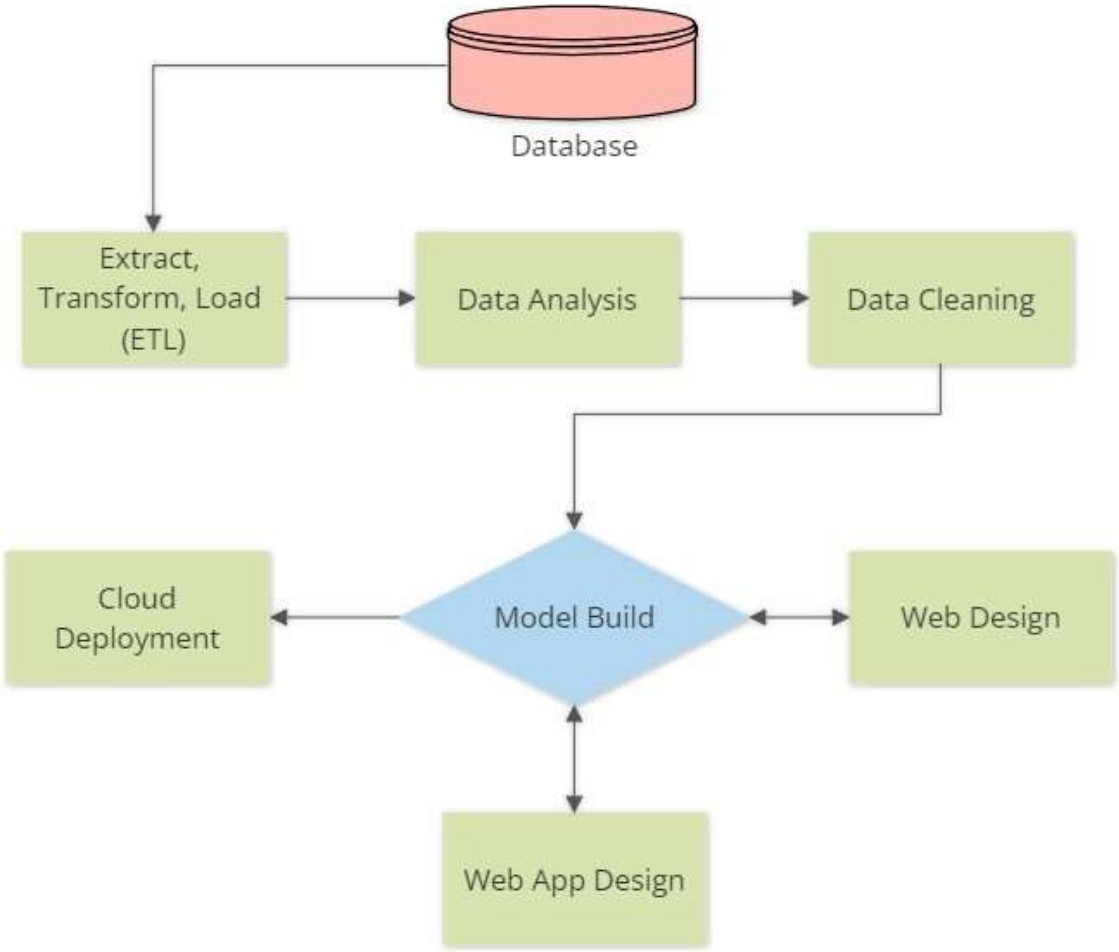
EDA

Analysis of data to summarize their main characteristics often using statistical graphs and visualization methods

Feature Engineering

Addition and construction of additional features to improve model performance and accuracy

Deployment Process



ETL

Three phase process where data is collected from multiple sources, transformed into desired format and loaded into multiple output containers

Web Design

Basic HTML/CSS design for the web application.

Web App

Front end app interface using Flask python library.

Model Management

Event Log

Model should log every event from start to end for debugging purpose. We can spot the error in logs and identify which part of the code need to be addressed and corrected. Logging is essential to see the internal execution flow and can be used to improve the architecture design for faster execution.

Error Handling

Error Handling makes a model robust, reliable ensuring the execution won't stop abruptly. Even in the worst-case scenario the execution will terminate in between intimating the user the reason for termination. We should imagine all possible worst-case scenarios and able to code the handlers to terminate safely or jump a step ahead in the execution after intimating the user.

Performance

As we discussed before model performance entirely depends on training dataset, quality of the data, authenticity of the data, algorithm used and hyper-parameter tuning. But can assure the performance will be good assuming the above criteria are all satisfied.

Reusability

All codes are written in modular fashion so that for future improvements codes can be reused or modified without affecting other modules. Improvements are necessary for better performance, faster execution and updated dataset training for the model.

Compatibility

No model is good if it not compatible. Model should be build in such a way it can perform in all scenarios provided with required input details. For this reason, we need to use widely prevalent version of framework or library supporting maximum systems to run the model.

Portability

Models build in containers have high portability. Portable containers will make sure the model executes and performs well even if the system doesn't have necessary libraries pre-installed.

Conclusion

We have build multiple classification models, trained and tested with same datasets, and finally performed parameter tuning for decently performing model to achieve best prediction and recall accuracy.