High Level Document

Credit Card Default Detection

Written by: Abhishek Nishad

Version:.07.02

Last Revision Date:16July2023

# Document Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Date issued | Version | Description | Author |
| 07/07/2023 | 1 | Base HLD:V1.0 | Abhishek Nishad |
| 16/07/2023 | 2 | Updated HLD:V2.0 | Abhishek Nishad |
|  |  |  |  |

# Contents

Document Version Control

Abstract

1. Introduction
2. Why this High Level Design Document?
3. Scope
4. Definitions.
5. General Description
6. Product Perspective
7. Problem Statement
8. Proposed Solution
9. Further Improvements
10. Technical Requirements
11. Data Requirements
12. Tools Utilised
13. Constraints
14. Assumptions
15. Design Details
16. Process Flow
17. Event Log
18. Error handing
19. Performance
20. Reusability
21. Application Compatibility
22. Resource Utilization
23. Deployment
24. Dashboards
25. KPIs(Key Performance Indicators)
26. Conclusion

# Abstract

The credit card default detection project aims to develop a machine learning solution that can accurately predict the likelihood of credit card default based on historical transaction data. This abstract provides a high-level overview of the project's design, outlining the key components and methodologies involved in achieving the desired outcome.

Credit card default detection plays a crucial role in risk management for financial institutions. By accurately identifying customers who are at a higher risk of defaulting on their credit card payments, banks can take proactive measures to mitigate potential losses. This includes adjusting credit limits, offering assistance programs, or implementing collection strategies to recover outstanding debt.

# Introduction

## Why this High Level Design (HLD) Document?

The HLD document is essential as it facilitates communication, clarifies scope, supports decision making, identifies risks, promotes scalability and maintainability, establishes development guidelines, and aligns technical design with business objectives. It serves as a roadmap for the project, ensuring that all stakeholders have a shared understanding of the system's design and functionality. **They help to identify potential problems early in the development process.** This can help to prevent costly changes and delays later on.

The HLD will:

* Present all of 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 the architecture of the project

List and describe the non-functional attributes like:

* Security
  + Reliability
  + Maintainability
  + Portability
  + Reusability
  + Application compatibility
  + Resource utilization
  + Serviceability

## 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.

## Definitions

|  |  |
| --- | --- |
| Term | Description |
| **LIMIT\_BAL** | Amount of given credit in NT dollars (includes individual and family/supplementary credit |
| **PAY\_0** | Repayment status in September, 2005 (-1=pay duly, 1=payment delay for one month, 2=payment delay for two months, … 8=payment delay for eight months, 9=payment delay for nine months and above) |
| **PAY\_2,** **PAY\_3,** **PAY\_4,** **PAY\_5,** **PAY\_6** | Repayment status in August, July, June, May, April |
| **BILL\_AMT1,** **BILL\_AMT2,** **BILL\_AMT3,** **BILL\_AMT4,** **BILL\_AMT5,** **BILL\_AMT6** | Amount of bill statement in September, August, July, June, May, April |
| **PAY\_AMT1,** **PAY\_AMT2,** **PAY\_AMT3,** **PAY\_AMT4,** **PAY\_AMT5,** **PAY\_AMT6** | Amount of previous payment in September, August, July, June, May, April |

**General Description**

## Product Perspective

Credit card default detection is the process of identifying customers who are at risk of defaulting on their credit card payments. This is important for credit card companies because it allows them to take steps to prevent default, such as increasing interest rates or closing accounts.

## Problem statement

To create a model that will detect a person capability to repay the amount based on historical data.

This will help bank and financial institutions to take steps to prevent default and protect their financial interests

## PROPOSED SOLUTION

The Proposed solution involves implementing a model that takes input from the user with different parameters using app and basis on user data this will predict if person can default on payment or not.

## Technical Requirements

For Building the model Python or R can be used, which have extensive libraries and frameworks for data analysis and machine learning.

System should minimum have 4GB of RAM and Dual Core Processor.

## Data Requirements

The Following are the basic requirements for the data

* The data should have alteast 10000 records.
* The proportionate of Imbalanced data should not vary much.
* The data should be accurate and up-to-date in order to ensure that the model is able to make accurate predictions.
* The data should be anonymized to protect the privacy of the customers
* The data should be stored securely to prevent unauthorized access
* Sparse Data should be avoided

## Tools used

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

* VS Code is used as IDE.
* For visualization of the plots, Matplotlib, Seaborn and Plotly are used
* AWS is used for deployment of the model
* MongoDB is used to retrieve, insert, delete, and update the database
* Front end development is done using HTML/CSS
* Python Flask is used for backend development
* GitHub is used as version control system.

## Constraints

* Precision: The model should minimize the number of false positives, i.e., customers who are predicted to default but do not actually default.
* **Fairness:** The model should not discriminate against any particular group of customers

## Assumptions

The model should not discriminate against any particular group of customers. This is especially important in the context of credit card default detection, as there is a risk that the model could be biased against certain groups of people, such as minorities or low-income individuals.

The model should be able to handle a large volume of data. As the number of credit card holders continues to grow, the model will need to be able to handle more data in order to remain accurate

The model should be secure to protect the sensitive data that it collects. This includes the data that is used to train the model, as well as the predictions that the model makes.

# Design Details

## Process Flow

The Data is entered through app by the user and this is then pre-processed and fed to the model for prediction

## Model Training and Evaluation

Train Dataset

Prediction

Best Model

Test Dataset

Dataset

## Deployment Process

Predicted Result

Make Prediction

Preprocess Data

Get User Input

Load Model

Start

## 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.

## Error Handling

Error Should be Handled properly with try and except block. If Error is occurred then except block executes . The error handling mechanism should be informative

## PerFormance

The performance of the model can be incorporated into the decision-making process in a number of ways. For example, the model can be used to set thresholds for alerts. If the model predicts that a customer is at a high risk of defaulting, then an alert can be sent to the customer. The alert can then be used to take steps to prevent the customer from defaulting. The performance of the model should be evaluated over time. This will help to ensure that the model is not becoming less accurate over time.

## Reusability

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

problems

## 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

## Resource Utilisation

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

## Deployment

* AWS
* Microsoft Azure
* GCP

## Conclusion

By understanding the key concepts of credit card default detection, organizations can improve their ability to prevent credit card defaults and protect their financial interests.