**High Level Design (HLD)**

**Credit Card Defaulters Prediction System**

Revision Number: 1.0

Last Date of revision: 06-Mar-2022

Document Control

Change Record:

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| 1 | 06-Mar-2022 | Subhajit Roy |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Reviews:

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Reviewer** | **Comments** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Approval Status:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Review Date** | **Reviewed By** | **Approved By** | **Comments** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Contents

[Abstract 1](#_Toc97469352)

[1. Introduction 1](#_Toc97469353)

[3.1 Why this High-Level Design Document? 1](#_Toc97469354)

[3.2 Scope 1](#_Toc97469355)

[3.3 Definitions 1](#_Toc97469356)

[2. General Description 2](#_Toc97469357)

[2.1 Product Perspective 2](#_Toc97469358)

[2.2 Problem Statement 2](#_Toc97469359)

[2.3 Proposed Solution 2](#_Toc97469360)

[2.4 Further Improvements 2](#_Toc97469361)

[2.5 Technical Requirements 2](#_Toc97469362)

[2.6 Data Requirements 2](#_Toc97469363)

[2.7 Tools Used 3](#_Toc97469364)

[2.8 Constraints 3](#_Toc97469365)

[2.9 Assumptions 3](#_Toc97469366)

[3. Design Details 4](#_Toc97469367)

[3.1 Process Flow 4](#_Toc97469368)

[Process Methodology 4](#_Toc97469369)

[3.1.1. Model Training and Evaluation 4](#_Toc97469370)

[3.1.2. Deployment Process 5](#_Toc97469371)

[3.2 Event Log 5](#_Toc97469372)

[3.3 Error Handling 5](#_Toc97469373)

[4. Performance 6](#_Toc97469374)

[4.1 Reusability 6](#_Toc97469375)

[4.2 Application Compatibility 6](#_Toc97469376)

[4.3 Resource Utilization 6](#_Toc97469377)

[4.4 Deployment 6](#_Toc97469378)

[5. Conclusion 6](#_Toc97469379)

# Abstract

With the tremendous advancement in the financial sector, credit risk now plays a significant part in the banking industry's operations. Banks' primary businesses include loan, credit card, investment, and mortgage lending, among others. Credit cards have been one of the most successful financial services offered by banks in recent years. However, as the number of credit card users grows, banks face an increasing credit card default rate. This work aims to identify the potential defaulters before allotting any credit card to the customer. This study uses the dataset containing information on default payments, demographic factors, credit data, history of payment, and bill statements of credit card clients in Taiwan from April 2005 to September 2005.

# Introduction

## Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) Document is to add the necessary detail 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.

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 |
| Database | Collection of all information monitored by this system |
| IDE | Integrated Development Environment |
| AWS | Amazon Web Services |
| EC2 | Elastic Compute Cloud 2 |
| S3 | Simple Storage Service |

# General Description

## Product Perspective

The credit card default prediction application is a machine learning model which will help to identify a potential defaulter before issuing a credit card to the customer.

## Problem Statement

Creation of an application to detect the Credit Card Defaulters and to achieve the following objectives:

* Understanding and exploratory analysis of the dataset
* Implementations of various machine learning algorithms
* Deployment of the models in the cloud

## Proposed Solution

The solution proposed here to build the application is to create separate models for different sets of data as a result of which the result will be aggregated and will provide a more accurate value. To separate the data into different sets, the data will be clustered and in each cluster various machine learning algorithms will be applied. The best performing model will be selected for that particular cluster. Similarly, during the prediction, the data will be clustered and in each cluster the already saved model will be applied to get the prediction.

## Further Improvements

The application can be further improved to propose a safe credit limit for the defaulters so that they can also be converted to potential customers which will be a gain for the bank.

## Technical Requirements

* To build the model minimum 6GB of RAM is required with minimum 2GB of HDD space.
* AWS EC2 instance of Ubuntu is required for the deployment of the model.
* Amazon S3 Bucket is required to store the log and output files.
* Cassandra DB is required to store the dataset.
* Visual studio code is required as the IDE.
* Python3 needs to be installed in the system
* The various important libraries like Pandas, Sklearn, Numpy, Boto3 etc. needs to be installed in the system

## Data Requirements

The dataset contains information on default payments, demographic factors, credit data, history of payment, and bill statements of credit card clients in Taiwan from April 2005 to September 2005.

[Dataset Link](https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset)

## Tools Used

Mainly the application has been built using python programming language and various other tools have been integrated with it.

* VS Code is used as an IDE.
* For the front end, HTML, CSS, JS is used with the Bootstrap framework.
* For visualizations of the plots Matplotlib and Seaborn are used.
* Cassandra database is used to store the dataset.
* Python programming language is used to develop the ML models.
* Flask is used to build the web apps.
* AWS is used for the deployment of the model.
* S3 bucket is used for storing the files.
* Boto3 and S3FS are used for file operations in S3 bucket.
* GitHub is used as version control system.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | https://logos-world.net/wp-content/uploads/2021/10/Python-Symbol.png |  |  |  |  |
|  |  |  |  |  |  |
|  |  | |  |  |  |

## Constraints

The credit card default prediction application must be simple, user friendly and as automated as possible with the least amount of inputs from the users and users should not be required to know any of the workings.

## Assumptions

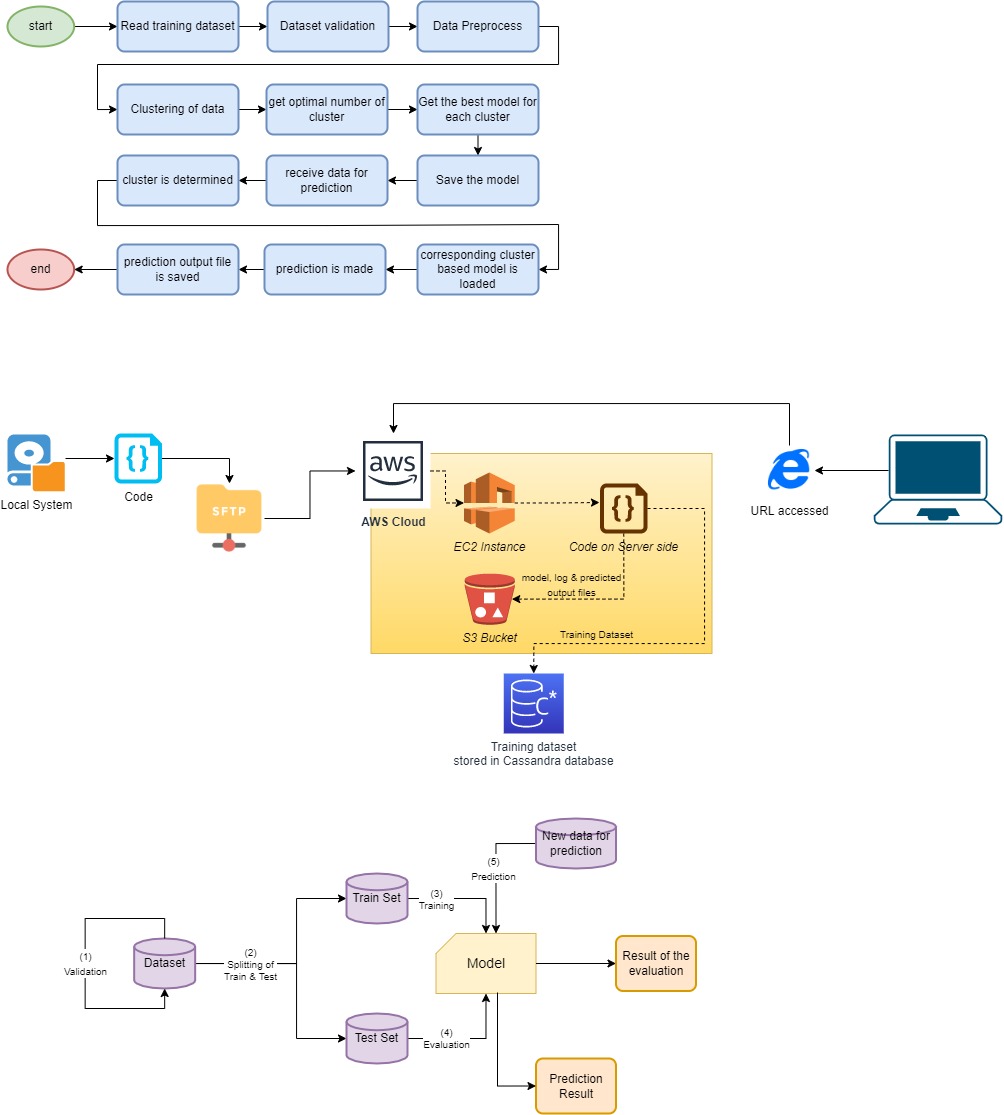
The model is working based on the dataset that was collected. Only the features of the datasets are considered as the important feature to determine a defaulter. It is also assumed that all aspect of this project have the ability to work together in the way the designer is expecting.

# Design Details

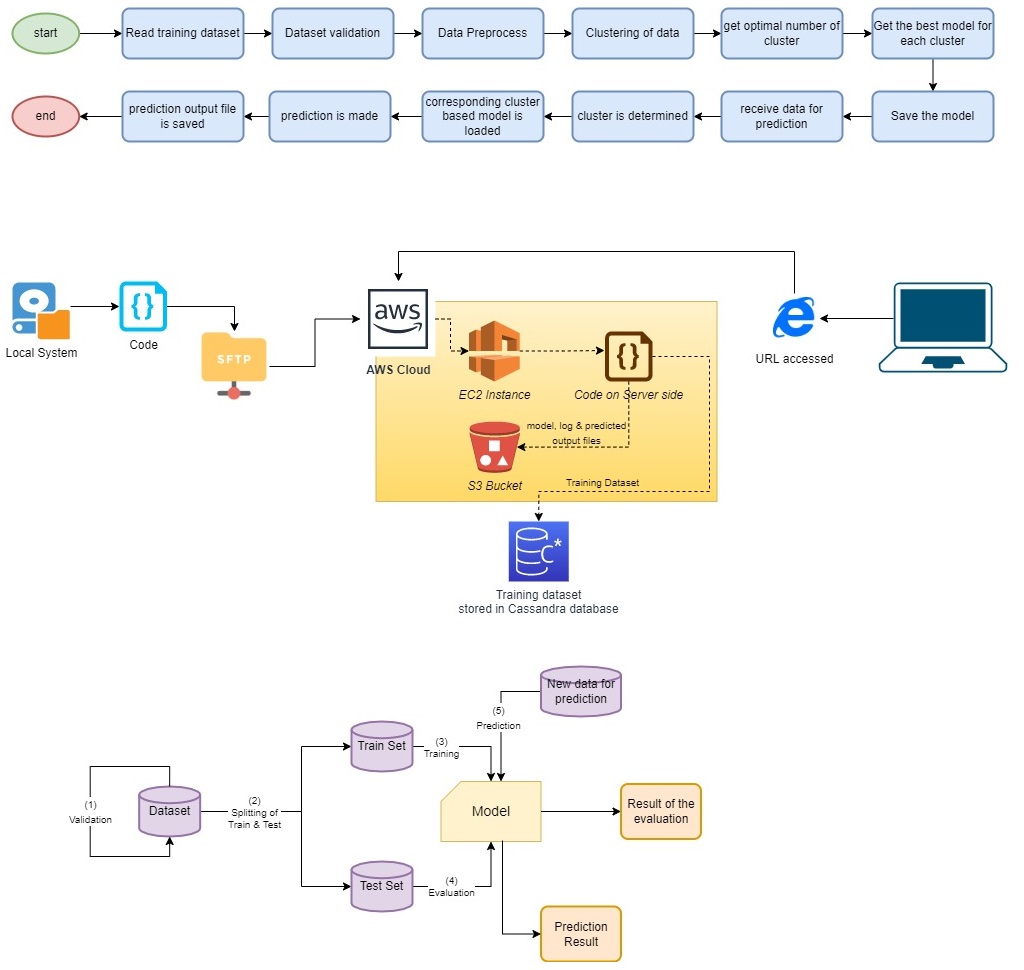
## Process Flow

For identifying the pattern of the defaulters properly, the dataset is first clustered into several groups. Various machines learning model is applied on each clusters and the best model is chosen for that particular clusters. During prediction, it is first checked in which cluster the data resides and the corresponding cluster based model is loaded for the prediction.

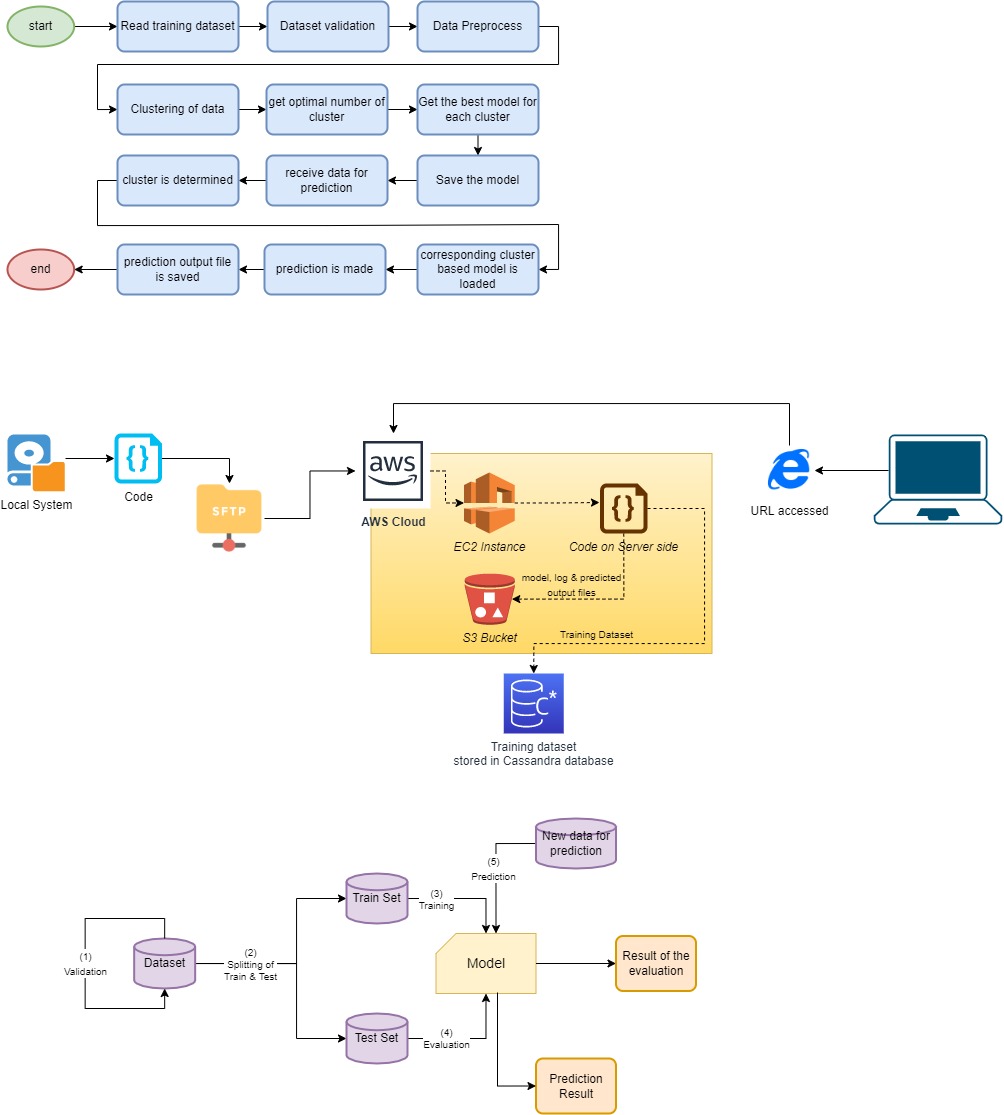
### Process Methodology



### Model Training and Evaluation



### Deployment Process



## 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

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.

# Performance

The default credit card prediction solution is used for the identification of the potential defaulters to reduce the losses that the bank incurs. It uses an existing dataset to learn the pattern of the defaulters using 26 feature variables.

## Reusability

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

## Application Compatibility

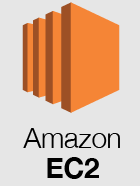
The different components for this project will be using Python as an interface between them. Each component will have its own task to perform, and it is the job of the Python to ensure proper transfer of information.

## Resource Utilization

When any task is performed, it is highly unlikely that the process will use all the processing power available.

## Deployment

The model has been deployed on AWS EC2 instance running on Ubuntu platform. The model also uses S3 bucket to save and store the log and the output files.



# Conclusion

The default credit card prediction solution is used for the identification of the potential defaulters to reduce the losses that the bank incurs.

# Reference

1. Dataset Link: <https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset>
2. Python download: <https://www.python.org/downloads/>
3. VSCode IDE Download: <https://code.visualstudio.com/download>
4. Amazon AWS: <https://aws.amazon.com/>
5. Cassandra Database: <https://www.datastax.com/>