Credit Card Default Prediction

Low-Level Design

Puneet Soni
26 October, 2023

Contents

Document Version Control

Date Issued	Version	Description	Author
26 October 2023	1.0	Initial LLD	Puneet Soni

Abstract

Credit risk plays a major role in the banking industry business. Banks' main activities involve granting loans, credit cards, investments, mortgages, and others. The credit card has been one of the mostbooming financial services by banks over the past years. However, with the growing number of credit card users, banks have been facing an escalating credit card default rate. As such data analytics can provide solutions to tackle the current phenomenon and management of credit risks.
This project discusses the implementation of a model which predicts if a given credit card holder has a probability of defaulting in the following month, using their demographic data and behavioural data from the past 6 months.

1. Introduction

1.1. Why this Low-Level Design Document?

The purpose of this document is to present a detailed description of the Deep EHR System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system and will be proposed to the higher management for approval.

1.2. Scope

This software system will be a Web application. This system will be designed to predict the customers' probability of defaulting credit payments at the earliest for better disease management and improved interventions using previous EHR records available. This system is designed to predict the credit card default from customers' information such as demographics, credit payment history etc.

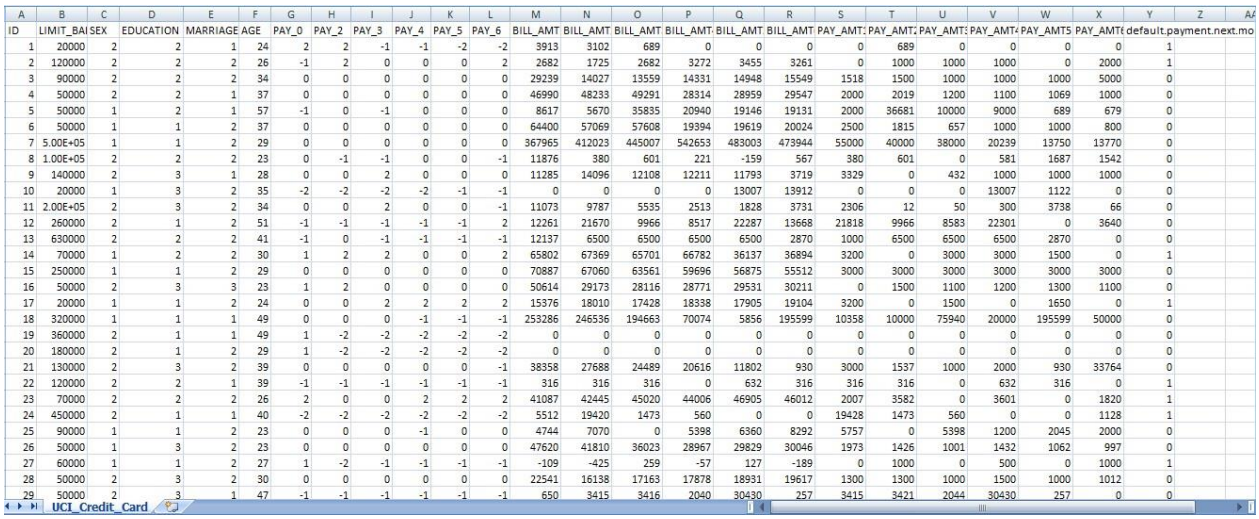
2. Technical Specifications

2.1. Dataset

File Name	Finalized	Source
UCI_Credit_Card.csv	Yes	https://www.kaggle.com/uciml/defaultof-credit-card-clients-dataset

2.1.1. Dataset Overview

The data file consists of one table, UCI_Credit_Card, containing the personal information and historical data about the payments made in the previous 6 months (April to September,in this context), of about 30,000 customers.



ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5	PAY_6	BILL_AMT1	BILL_AMT2	BILL_AMT3	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT2	PAY_AMT3	PAY_AMT4	PAY_AMT5	PAY_AMT6	default.payment.next.mo
1	20000	2	2	1	24	2	2	-1	-1	-2	-2	3913	3102	689	0	0	0	0	689	0	0	0	0	1
2	120000	2	2	2	26	-1	2	0	0	0	2	2682	1725	2682	3272	3455	3261	0	1000	1000	1000	0	2000	1
3	90000	2	2	2	34	0	0	0	0	0	0	29239	14027	13559	14331	14948	15549	1518	1500	1000	1000	1000	5000	0
4	50000	2	2	1	37	0	0	0	0	0	0	46990	48233	49291	28314	28959	29547	2000	2019	1200	1100	1069	1000	0
5	50000	1	2	1	57	-1	0	-1	0	0	0	8617	5670	35835	20940	19146	19131	2000	36681	10000	9000	689	679	0
6	50000	1	1	2	37	0	0	0	0	0	0	64400	57069	57608	19394	19619	20024	2500	1815	657	1000	1000	800	0
7	5.00E+05	1	1	2	29	0	0	0	0	0	0	367965	412023	445007	542653	483003	473944	55000	40000	38000	20239	13750	13770	0
8	1.00E+05	2	2	2	23	0	-1	-1	0	0	-1	11876	380	601	221	-159	567	380	601	0	581	1687	1542	0
9	140000	2	3	1	28	0	0	2	0	0	0	11285	14096	12108	12211	11793	3719	3329	0	432	1000	1000	1000	0
10	20000	1	3	2	35	-2	-2	-2	-2	-1	-1	0	0	0	0	13007	13912	0	0	0	13007	1122	0	0
11	2.00E+05	2	3	2	34	0	0	2	0	0	-1	11073	9787	5535	2513	1828	3731	2306	12	50	300	3738	66	0
12	260000	2	1	2	51	-1	-1	-1	-1	-1	2	12261	21670	9966	8517	22287	13668	21818	9966	8583	22301	0	3640	0
13	630000	2	2	2	41	-1	0	-1	-1	-1	-1	12137	6500	6500	6500	6500	2870	1000	6500	6500	6500	2870	0	0
14	70000	1	2	2	30	1	2	2	0	0	2	65802	67369	65701	66782	36137	36894	3200	0	3000	3000	1500	0	1
15	250000	1	1	2	29	0	0	0	0	0	0	70887	67060	63561	59696	56875	55512	3000	3000	3000	3000	3000	3000	0
16	50000	2	3	3	23	1	2	0	0	0	0	50614	29173	28116	28771	29531	30211	0	1500	1100	1200	1300	1100	0
17	20000	1	1	2	24	0	0	2	2	2	2	15376	18010	17428	18338	17905	19104	3200	0	1500	0	1650	0	1
18	320000	1	1	1	49	0	0	0	-1	-1	-1	253286	246536	194663	70074	5856	195599	10358	10000	75940	20000	195599	50000	0
19	360000	2	1	1	49	1	-2	-2	-2	-2	-2	0	0	0	0	0	0	0	0	0	0	0	0	0
20	180000	2	1	2	29	1	-2	-2	-2	-2	-2	0	0	0	0	0	0	0	0	0	0	0	0	0
21	130000	2	3	2	39	0	0	0	0	0	-1	38358	27688	24489	20616	11802	930	3000	1537	1000	2000	930	33764	0
22	120000	2	2	1	39	-1	-1	-1	-1	-1	-1	316	316	316	0	632	316	316	316	0	632	316	0	1
23	70000	2	2	2	26	2	0	0	2	2	2	41087	42445	45020	44006	46905	46012	2007	3582	0	3601	0	1820	1
24	450000	2	1	1	40	-2	-2	-2	-2	-2	-2	5512	19420	1473	560	0	0	19428	1473	560	0	0	1128	1
25	90000	1	1	2	23	0	0	0	-1	0	0	4744	7070	0	5398	6360	8292	5757	0	5398	1200	2045	2000	0
26	50000	1	3	2	23	0	0	0	0	0	0	47620	41810	36023	28967	29829	30046	1973	1426	1001	1432	1062	997	0
27	60000	1	1	2	27	1	-2	-1	-1	-1	-1	-109	-425	259	-57	127	-189	0	1000	0	500	0	1000	1
28	50000	2	3	2	30	0	0	0	0	0	0	22541	16138	17163	17878	18931	19617	1300	1300	1000	1500	1000	1012	0
29	50000	2	3	1	47	-1	-1	-1	-1	-1	-1	650	3415	3416	2040	30430	257	3415	3421	2044	30430	257	0	0

UCI_Credit_Card

2.1.2. Input Schema

Feature Name	Datatype	Null/Required
ID	Integer	Required
LIMIT_BAL	Integer	Required
SEX	Integer	Required
EDUCATION	Integer	Required
MARRIAGE	Integer	Required
AGE	Integer	Required
PAY_0	Integer	Required
PAY_2	Integer	Required
PAY_3	Integer	Required
PAY_4	Integer	Required
PAY_5	Integer	Required
PAY_6	Integer	Required
BILL_AMT1	Integer	Required
BILL_AMT2	Integer	Required
BILL_AMT3	Integer	Required

BILL_AMT4	Integer	Required
BILL_AMT5	Integer	Required
BILL_AMT6	Integer	Required
PAY_AMT1	Integer	Required
PAY_AMT2	Integer	Required
PAY_AMT3	Integer	Required
PAY_AMT4	Integer	Required
PAY_AMT5	Integer	Required
PAY_AMT6	Integer	Required
default.payment.next.month	Integer	Required

2.2. Predicting Credit Fault

- The system presents the set of inputs from the user.
- The user gives the required information.
- The system should be able to predict whether the customer is likely to default in the following month.

2.3. Logging

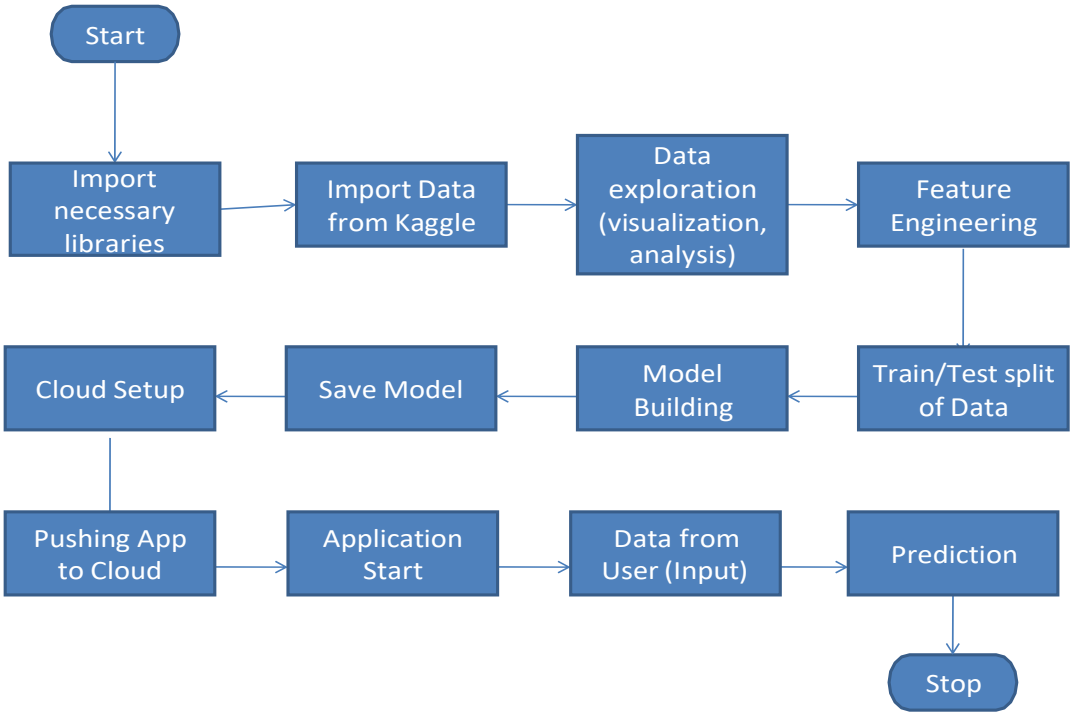
We should be able to log every activity done by the user.

- The System identifies at what step logging is required.
- The System should be able to log every system flow.
- Developers can choose logging methods. You can choose database logging/ File logging as well.
- System should not be hung even after using so many loggings. Logging is just becausewe can easily debug issues so logging is mandatory to do.

2.4. Deployment

Deployed in Heroku.

3. Architecture



4. Architecture Description

4.1. Data Description

This dataset is taken from Kaggle (url: https://www.kaggle.com/uciml/defaultof-credit-card-clients-dataset). It 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.

Content There are 25 variables:

- **ID**: ID of each client
- **LIMIT_BAL**: Amount of given credit in NT dollars (includes individual and family/supplementary credit
- **SEX**: Gender
  - 1=male,
  - 2=female
- **EDUCATION:**
  - 1=graduate school,
  - 2=university,
  - 3=high school,
  - 0, 4, 5, 6=others)
- **MARRIAGE**: Marital status
  - 1=married,
  - 2=single,
  - 3=divorce,
  - 0=others
- **AGE**: Age in years
- **PAY_0**: Repayment status in September, 2005
  - -2: No consumption;
  - -1: Paid in full;
  - 0: The use of revolving credit;
  - 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**: Repayment status in August, 2005 (scale same as above)
- **PAY_3**: Repayment status in July, 2005 (scale same as above)
- **PAY_4**: Repayment status in June, 2005 (scale same as above)
- **PAY_5**: Repayment status in May, 2005 (scale same as above)
- **PAY_6**: Repayment status in April, 2005 (scale same as above)
- **BILL_AMT1**: Amount of bill statement in September, 2005 (NT dollar)
- **BILL_AMT2**: Amount of bill statement in August, 2005 (NT dollar)
- **BILL_AMT3**: Amount of bill statement in July, 2005 (NT dollar)
- **BILL_AMT4**: Amount of bill statement in June, 2005 (NT dollar)
- **BILL_AMT5**: Amount of bill statement in May, 2005 (NT dollar)
- **BILL_AMT6**: Amount of bill statement in April, 2005 (NT dollar)
- **PAY_AMT1**: Amount of previous payment in September, 2005 (NT dollar)

- **PAY_AMT2**: Amount of previous payment in August, 2005 (NT dollar)
- **PAY_AMT3**: Amount of previous payment in July, 2005 (NT dollar)
- **PAY_AMT4**: Amount of previous payment in June, 2005 (NT dollar)
- **PAY_AMT5**: Amount of previous payment in May, 2005 (NT dollar)
- **PAY_AMT6**: Amount of previous payment in April, 2005 (NT dollar)
- **default.payment.next.month**: Default payment
    - 1=yes,
    - 0=no

4.2. Data Exploration

We divide the data into two types: numerical and categorical. We explore through each type one by one. Within each type, we explore, visualize and analyze each variable one by one and note down our observations. We also made some minor changes in the data like changing column names for convenience in understanding.

4.3. Feature Engineering

Encoded categorical variables.

4.4. Train/Test Split

Split the data into 70% train set and 30% test set.

4.5. Model Building

Built models and trained and tested the data on the models.
Compare the performance of each model and select the best one.

4.6. Save the model

Saved the model by converting it into a pickle file.

4.7. Cloud Setup & Pushing the App to the Cloud

Selected Heroku for deployment. Loaded the application files from GitHub to Heroku.

4.8. Application Start and Input Data by the User

Start the application and enter the inputs.

4.9. Prediction

After the inputs are submitted the application runs the model and makes predictions. The out is displayed as a message indicating whether the customer whose demographic and behavioral data are entered as inputs, is likely to default in the following month or not.

5. Unit Test Cases

Test Case Description	Pre-Requisite	Expected Result
Verify whether the Application URL is accessible to the user	1. Application URL should be defined	Application URL should be accessible to the user
Verify whether the Application loads completely for the user when the URL is accessed	1. Application URL is accessible 2. Application is deployed	The Application should load completely for the user when the URL is accessed
Verify whether the user is ableto see input fields on log ing in	1. Application URL is accessible 2. Application is deployed	The user should be able to seeinput fields on logging in
Verify whether the user is ableto edit all input fields	1. Application URL is accessible 2. Application is deployed	The user should be able to editall input fields
Verify whether a user gets the Submit button to submit theinputs	1. Application URL is accessible 2. Application is deployed	The user should get the Submit button to submit the inputs
Verify whether the user is presented with recommended results on clicking submit	1. Application URL is accessible 2. Application is deployed	The user should be presented with recommended results on clicking submit