Bank Credit Risk Classification

Low Level Design

Gaurav Singh
12th July, 2023

Contents

Document Version Control

Date Issued	Version	Description	Author
12th July 2023	1.0	Initial LLD	Gaurav Singh

Abstract

Credit risk plays a major role in the banking industry business. Banks' main activities involve granting loan, credit card, investment, mortgage, and others. Credit card has been one of the most booming 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 credit risks. This project discusses the implementation of model which classifies a given profile as a good risk or a bad risk.

1. Introduction

1.1. Why this Low-Level Design Document?

The purpose of this document is to present a detailed description of the Bank Credit Risk Classification application. 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 etc. This document is intended for both the stakeholders and the developers of the system and will be proposed to the higher management for its approval.

1.2. Scope

This software system will be a Web application. This system is designed to classify a customer profile into good or bad risk category, from customers' information such as demographics, credit details etc.

2. Technical Specifications

2.1. Dataset

File Name	Finalized	Source
SouthGermanCredit.asc	Yes	https://archive.ics.uci.edu/ml/datasets/South+German+Credit

2.1.1. Dataset Overview

The data obtained from the repository is in the form of. asc file, from which the data is extracted and stored in the Cassandra database. It contains the personal and behavioral data of about 1000 customers, out of which 700 are classified as Good Risk and 300 as Bad Risk.

status	duration	credit_history	purpose	amount	savings	employment_duration	installment_rate	personal_status_sex	other_debtors	present_residence	property	age	other_installment_plans	housing	number_credits	job	people_liable	telephone	foreign_worker	credit_risk
1	18	4	2	1049	1	2	4	2	1	4	2	21	3	1	1	3	2	1	2	1
1	9	4	0	2799	1	3	2	3	1	2	1	36	3	1	2	3	1	1	2	1
2	12	2	9	841	2	4	2	2	1	4	1	23	3	1	1	2	2	1	2	1
1	12	4	0	2122	1	3	3	3	1	2	1	39	3	1	2	2	1	1	1	1
1	12	4	0	2171	1	3	4	3	1	4	2	38	1	2	2	2	2	1	1	1
1	10	4	0	2241	1	2	1	3	1	3	1	48	3	1	2	2	1	1	1	1
1	8	4	0	3398	1	4	1	3	1	4	1	39	3	2	2	2	2	1	1	1
1	6	4	0	1361	1	2	2	3	1	4	1	40	3	2	1	2	1	1	1	1
4	18	4	3	1098	1	1	4	2	1	4	3	65	3	2	2	1	2	1	2	1
2	24	2	3	3758	3	1	1	2	1	4	4	23	3	1	1	1	2	1	2	1
1	11	4	0	3905	1	3	2	3	1	2	1	36	3	1	2	3	1	1	2	1
1	30	4	1	6187	2	4	1	4	1	4	3	24	3	1	2	3	2	1	2	1
1	6	4	3	1957	1	4	1	2	1	4	3	31	3	2	1	3	2	1	2	1
2	48	3	10	7582	2	1	2	3	1	4	4	31	3	2	1	4	2	2	2	1
1	18	2	3	1936	5	4	2	4	1	4	3	23	3	1	2	2	2	1	2	1
1	6	2	3	2647	3	3	2	3	1	3	1	44	3	1	1	3	1	1	2	1
1	11	4	0	3939	1	3	1	3	1	2	1	40	3	2	2	2	1	1	2	1
2	18	2	3	3213	3	2	1	4	1	3	1	25	3	1	1	3	2	1	2	1
2	36	4	3	2337	1	5	4	3	1	4	1	36	3	2	1	3	2	1	2	1
4	11	4	0	7228	1	3	1	3	1	4	2	39	3	2	2	2	2	1	2	1
1	6	4	0	3676	1	3	1	3	1	3	1	37	3	1	3	3	1	1	2	1
2	12	4	0	3124	1	2	1	3	1	3	1	49	1	2	2	2	1	1	2	1

2.1.2. Input Schema

Feature Name	Datatype	Null/Required
status	Integer	Required
duration	Integer	Required
credit_history	Integer	Required
purpose	Integer	Required
amount	Integer	Required
savings	Integer	Required
employment_duration	Integer	Required
installment_rate	Integer	Required
personal_status_sex	Integer	Required
other_debtors	Integer	Required
present_residence	Integer	Required
property	Integer	Required
age	Integer	Required
other_installment_plans	Integer	Required

housing	Integer	Required
number_credits	Integer	Required
job	Integer	Required
people_liable	Integer	Required
telephone	Integer	Required
foreign_worker	Integer	Required
credit_risk	Integer	Required

2.2. Predicting Credit Risk class

- The system presents the set of inputs to the user.
- The user gives required information.
- The system should be able to predict whether the customer is likely to pose a risk to the bank or not.

2.3. Logging

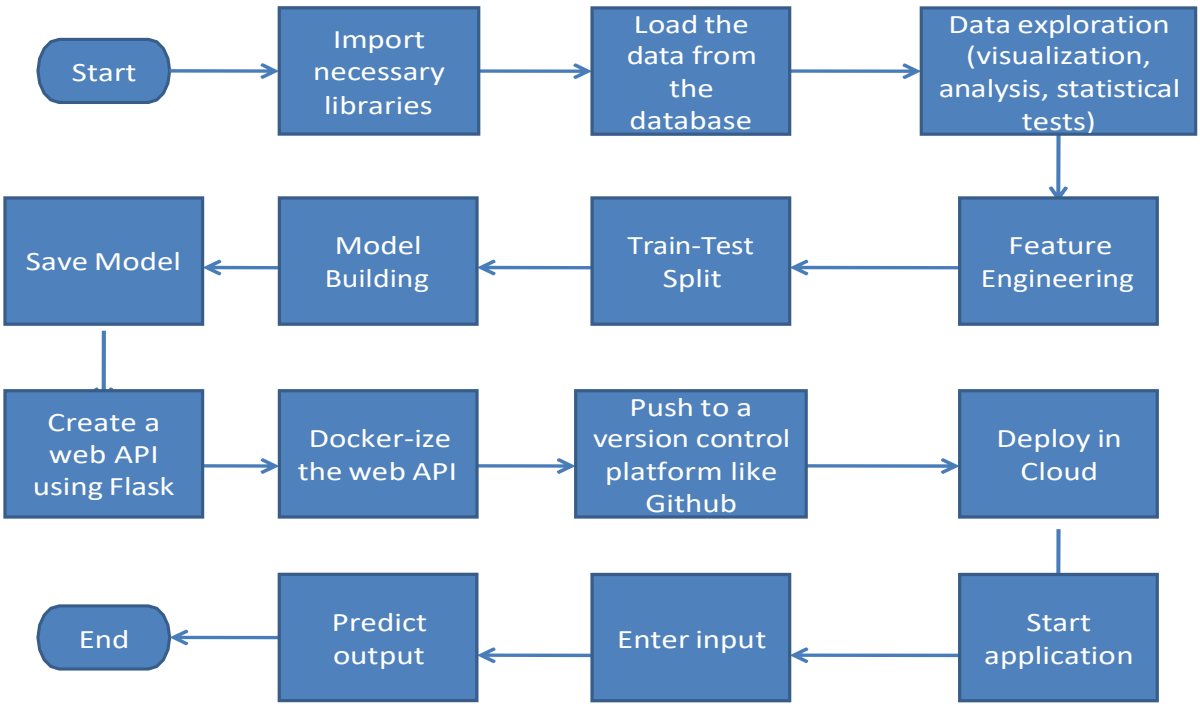
We should be able to log every activity done by the user.

- The System identifies at what step logging required.
- The System should be able to log each and 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 just because we can easily debug issues so logging is mandatory to do.

2.4. Deployment

The application is deployed locally on windows system as Heroku has stopped its free version and cloud providers need credit cards for the same. But model can be easily upload to any cloud providers through dockerization.

3. Architecture



4. Architecture Description

4.1. Data Description

This dataset is taken from the UCI Machine Learning Repository (url: https://archive.ics.uci.edu/ml/datasets/South+German+Credit). It contains information on demographic factors, credit data etc. of customers. There are 21 variables:

Content There are 21 variables:

- **status** : status of the debtor's checking account with the bank (categorical)
  - **1** : no checking account
  - **2** : ... < 0 DM
  - **3** : 0<= ... < 200 DM
  - **4** : ... >= 200 DM / salary for at least 1 year

- **duration** : credit duration in months (quantitative)

- **credit_history** : history of compliance with previous or concurrent credit contracts (categorical)
  - **0** : delay in paying off in the past
  - **1** : critical account/other credits elsewhere
  - **2** : no credits taken/all credits paid back duly
  - **3** : existing credits paid back duly till now
  - **4** : all credits at this bank paid back duly

- **purpose** : purpose for which the credit is needed (categorical)
  - **0** : others
  - **1** : car (new)
  - **2** : car (used)
  - **3** : furniture/equipment
  - **4** : radio/television
  - **5** : domestic appliances
  - **6** : repairs
  - **7** : education
  - **8** : vacation
  - **9** : retraining
  - **10** : business
- **amount** : credit amount in DM (quantitative; result of monotonic transformation; actual data and type of transformation unknown)

- **savings** : debtor's savings (categorical)
  - **1** : unknown/no savings account
  - **2** : ... < 100 DM
  - **3** : 100 <= ... < 500 DM
  - **4** : 500 <= ... < 1000 DM
  - **5** : ... >= 1000 DM

- **employment_duration** : duration of debtor's employment with current employer (ordinal; discretized quantitative)
    - **1** : unemployed
    - **2** : < 1 yr
    - **3** : 1 <= ... < 4 yrs
    - **4** : 4 <= ... < 7 yrs
    - **5** : >= 7 yrs

- **installment_rate** : credit installments as a percentage of debtor's disposable income (ordinal; discretized quantitative)
    - **1** : >= 35
    - **2** : 25 <= ... < 35
    - **3** : 20 <= ... < 25
    - **4** : < 20

- **personal_status_sex** : combined information on sex and marital status; categorical; sex cannot be recovered from the variable, because male singles and female non-singles are coded with the same code (2); female widows cannot be easily classified, because the code table does not list them in any of the female categories
    - **1** : male : divorced/separated
    - **2** : female : non-single or male : single
    - **3** : male : married/widowed
    - **4** : female : single

- **other_debtors :** Is there another debtor or a guarantor for the credit? (categorical)
    - **1** : none
    - **2** : co-applicant
    - **3** : guarantor

- **present_residence :** length of time (in years) the debtor lives in the present residence (ordinal; discretized quantitative)
    - **1** : < 1 yr
    - **2** : 1 <= ... < 4 yrs
    - **3** : 4 <= ... < 7 yrs
    - **4** : >= 7 yrs

- **property :** the debtor's most valuable property, i.e. the highest possible code is used. Code 2 is used, if codes 3 or 4 are not applicable and there is a car or any other relevant property that does not fall under variable savings. (ordinal)
    - **1** : unknown / no property
    - **2** : car or other
    - **3** : building soc. savings agr./life insurance
    - **4** : real estate
- **age :**age in years (quantitative)

- **other_installment_plans :**installment plans from providers other than the credit-giving bank (categorical)
    - **1** : bank
    - **2** : stores
    - **3** : none

- **housing :**type of housing the debtor lives in (categorical)
    - **1** : for free
    - **2** : rent
    - **3** : own

- **number_credits :**number of credits including the current one the debtor has (or had) at this bank (ordinal, discretized quantitative); contrary to Fahrmeir and HamerleÃ¢â‚¬â„¢s (1984) statement, the original data values are not available.
    - **1** : 1
    - **2** : 2-3
    - **3** : 4-5
    - **4** : >= 6

- **job :**quality of debtor's job (ordinal)
    - **1** : unemployed/unskilled - non-resident
    - **2** : unskilled - resident
    - **3** : skilled employee/official
    - **4** : manager/self-empl./highly qualif. employee

- **people_liable :**number of persons who financially depend on the debtor (i.e., are entitled to maintenance) (binary, discretized quantitative)
    - **1** : 3 or more
    - **2** : 0 to 2

- **telephone :**Is there a telephone landline registered on the debtor's name? (binary; remember that the data are from the 1970s)
    - **1 :** No
    - **2 :** Yes

- **foreign_worker :**Is the debtor a foreign worker? (binary)
    - **1** : yes
    - **2** : no

- **credit_risk :**Has the credit contract been complied with (good) or not (bad) ? (binary)
    - **0** : bad
    - **1** : good

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. Statistical tests are performed for each independent variable to see if the variable has any significance in determining the output for the target variable.

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.
Compared the performance of each model and selected the best one.
Feature importance and/or hyper-parameter tuning performed to improve the performance of the selected model.

4.6. Save the model
Saved the model by converting into a pickle file.

4.7. Cloud Setup & Pushing the App to the Cloud
Selected Heroku for deployment.
Used the model to develop a flask application which can predict risk class for unseen data.
Dockerised the application and pushed to Github and from there, deployed the application files from to Heroku, all using Github Actions as ci/cd pipeline.

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 is classified as Good Risk or Bad Risk.

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 user is able to see input fields on logging in	1. Application URL is accessible 2. Application is deployed	User should be able to see input fields on logging in
Verify whether user is able to edit all input fields	1. Application URL is accessible 2. Application is deployed	User should be able to edit all input fields
Verify whether user gets submit button to submit the inputs	1. Application URL is accessible 2. Application is deployed	User should get Submit button to submit the inputs
Verify whether user is presented with recommended results on clicking submit	1. Application URL is accessible 2. Application is deployed	User should be presented with recommended results on clicking submit