iNeuron

LOW LEVEL DOCUMENT

CREDIT CARD DEFAULT PREDICTION

October, 2022

Written BY Namdeo Patil

Contents

DOCUMENT VERSION CONTROL

Change Record

Date Date	Version Version	Comments Comments	Author Author
18/10/2022	0.1 0.1	Introduction and architecture defined	Namdeo Patil
21/10/2022	0.2 0.2	Architecture updated Architecture updated and unit test case and unit test case defined defined	Namdeo Patil

Review

Date	Version	Reviewer	Comments

Approval Status

Version	Reviewer	Approved By	Comments

ABSTRACT

Financial threats are displaying a trend in the credit risk of commercial banks as the incredible improvement in the financial industry has arisen. In this way, one of the biggest threats faces by commercial banks is the risk prediction of credit clients. The goal is to predict the probability of credit default based on the credit card owner's characteristics and payment history.

With the help of Data Science and Machine learning technology, I developed an application, which allows a banker to determine the probability Of Default in just a few seconds.

1 Introduction

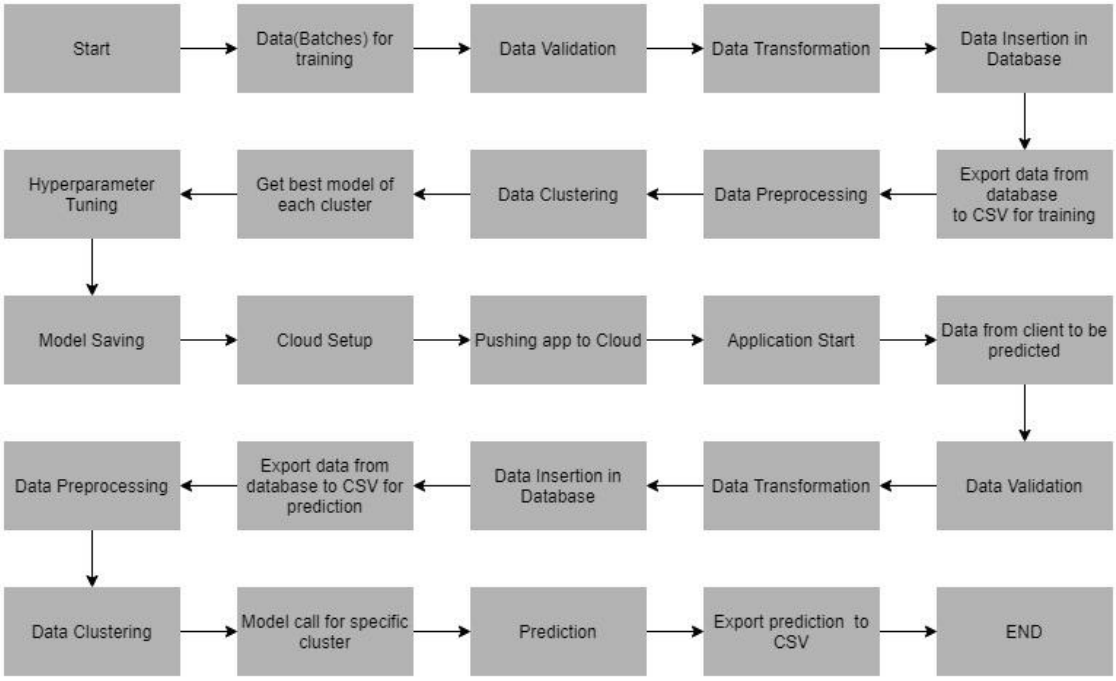
1.1 What is Low-Level design document?

The goal of LLD or a Low-level design document is to give an internal logical design of the actual program code for the Credit Card Default Probability Prediction. LLD describes the class diagrams with the methods and relations between classes and
the program specs. It describes the modules so that the
programmer can directly code the program from the document.

1.2 Scope

Low-level design (LLD) is a component level design process that follows a step-by-step refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then defined during data design work.

2 Architecture



3 Architecture Description

3.1 Data Collection

For training and testing the model I used a publicly available dataset on Kaggle. This 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.

https://www.kaggle.com/datasets/uciml/default-of-credit card-clients-dataset

3.2 Data Dictionary

- **ID: ID of each client droped because it is unnecessary**
- **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, 4=others, 5=unknown, 6=unknown)**
- **MARRIAGE: Marital status (1=married, 2=single, 3=others)**
- **AGE: Age in years**
- **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: 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)**

iNeuron

Summary Statistics

There are 25 columns .Out of which 20 are integer columns and 4 are decimal columns.Total there are 30,000 transactions.

3.3 Variable Information

This is a classification problem. The target variable is default.payment.next.month. The aim of the project is to predict the probability of default given various attributes of the customer given below.

4. Data Ingestion

- data in data ingestion folder
- Split data in train and test data
- Saving data in data ingestion folder

4.1 Data Validation

- data in data validation folder
- data drift and EDA
- Report in evidently to show and analysis Data Distribution
- Saving report in data validation Folder

4.2 Data Transformation

- data in data Transformation folder
- Using Pipeline and columns transformer to processing Data
- Splitting data in array by Standard Scalar and different technique
- Putting train array data and test array data in transformation folder and creating prepossessing Object file.
- Saving Processing pkl in data transformation folder for future Transformation

5. Model Trainer

- We have built various models like Logistic Regression, Random Forest, Gradient Boosting, etc. • Each of the above models was built taking their default parameters.
- We have use Gridsearch CV to have best model in comparison all models
- We save best model file in Model Trainer folder

5.1Data Evaluation

- we used Metric for Classification : Recall Score, F1 Score and Model accuracy In train and test model to evaluate our best model
- we save the evaluation model in evaluation Folder
- Random Forest Classifier was the best Model
- Reason For Choosing This Model :
- Apart From a good training and test score, the reasons for choosing this model are as follows :
- • Can handle missing values.
- • Can work well even on imbalanced data.

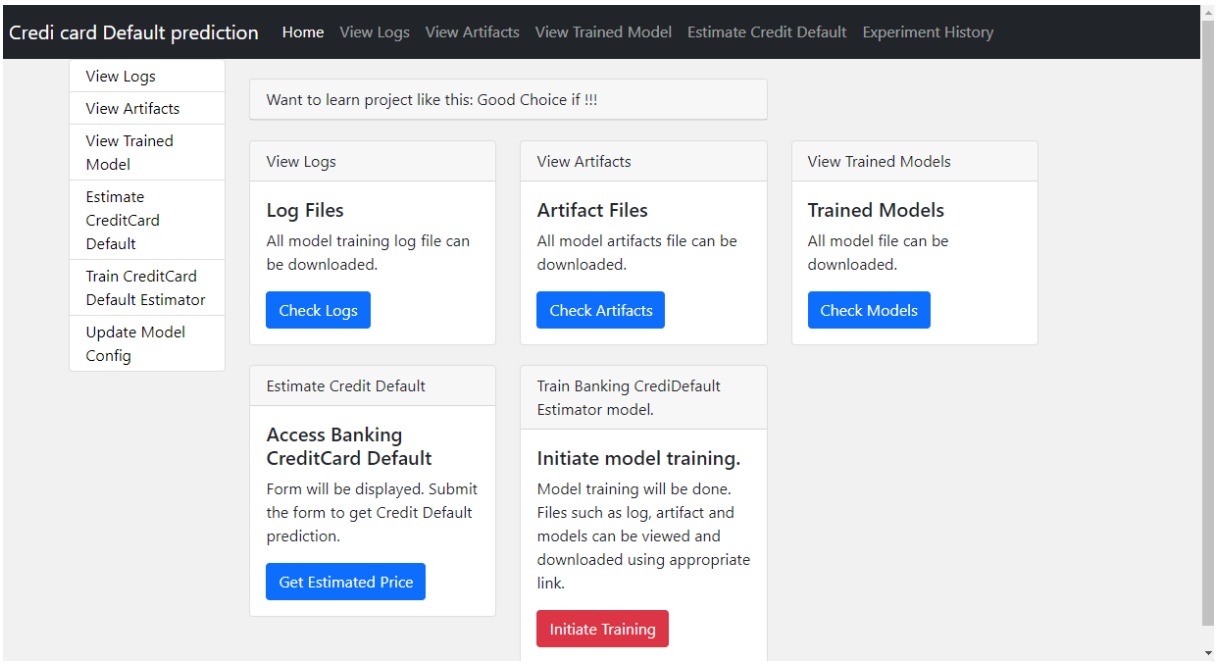
5.2 Model Pusher

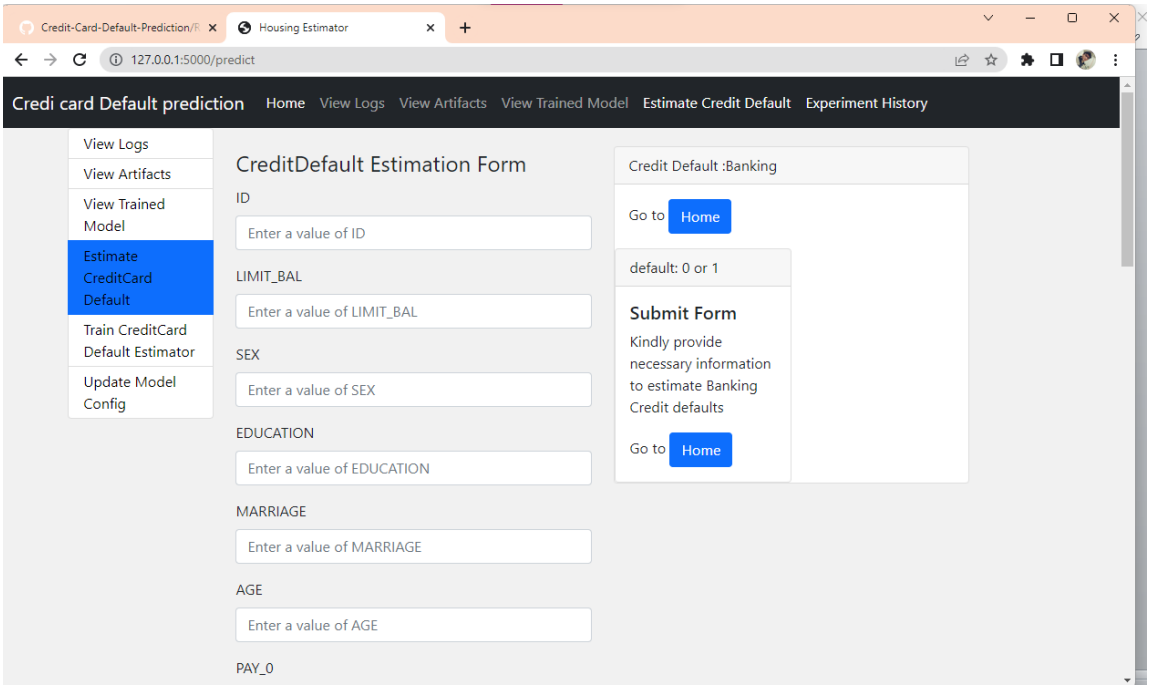
- **The best model was Random Forest.**
- **The model has been saved in 'Model.pkl'**
- **If we have an updated better model by new data , we can save it in model pusher save models folder**

5.3 Heroku Deplyement

- **I deployed the application on the web using Heroku**
- **The deployment part of the code runs in the "app.py" file, connecting with the web page designed using HTML with CSS styles.The html front end template can be found in templates folder.**

6 User Interface





iNeuron

7 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 can edit all the input fields	1. Application URL is Accessible. 2. Application loads completely for the user. 3. All the input fields Loaded.	User should be able to edit all the input fields
Verify whether user gets “Predict” button to make predictions on the given inputs	1. Application URL is Accessible. 2. Application loads completely for the user. 3. All the input fields Loaded.	User should get a “Predict” button to make predictions on the given inputs.
Verify whether user is Presented with recommended results on clicking the “Predict” button	1. Application URL is Accessible. 2. Application loads completely for the user. 3. All the input fields Loaded.	Users should be presented with recommended results on clicking the “Predict” button.