DPR
Credit Card Default Prediction Model

Revision Number – 1.2

Last Date of Revision – 08/06/2023

Ramchandra Tukaram Padwal

Document Version Control

Date	Version	Description	Author
24-04-2023	1.0	Abstract, Introduction	Ramchandra
25-05-2023	1.1	Deployment and Process	Ramchandra
08-06-2023	1.2	Q and A	Ramchandra

Contents

Abstract 4
INTRODUCTION **5**
Why this DPR Documentation? **5**
Key points: 5
1 Description **5**
1.1 Problem Perspective **5**
1.2 Problem Statement 5
1.3 Proposed Solution 5
1.4 Solution Improvements 5
2 Technical Requirements **6**
2.1 Tools Used 6
3 Data Requirements 7
3.1 Data Gathering from Main Source 7
3.2 Data Description 7
3.3 Import Data into Database 7
3.4 Export Data from Database 7
4 Data Pre-Processing 8
5 Design Flow **9**
5.1 Modelling **9**
5.2 UI Integration **9**
5.3 Modelling Process **9**
5.4 Deployment Process **9**
6 Data from User **10**
7 Data Validation **10**
8 Rendering the Results **10**
9 Deployment **10**
Conclusion **10**
Q & A: **11**

Abstract

In present day majority of people are using credit card for online shopping, EMI payments. Moreover, banks are encouraging customers to use it by approving them freely without any initial charges. Because credit card interest income is the major share of profit for all commercial banks. Even if customers are notwilling to buy, bank marketing team reaches out through phone call to sell it. At first the interest rate is normal to grope the customer into the practice of using it, later they will levy charges on everything to loot money from the customers. Due to this reason many customers are facing difficulty in repaying the credit amount on time. If they missed the deadline the interest rate will get multiplied leading to defaulting the account if left unpaid even for a short span of time. The model we are going to build will consider all inputs from user to predict whether the account will be defaulted or not based on various parameters.

INTRODUCTION

Why this DPR Documentation?

The main purpose of this DPR documentation is to add the necessary details of the project andprovide the description of the machine learning model and the written code. This also provides the detailed description on how the entire project has been designed end-to-end.

Key points:

- Describes the design flow
- Implementations
- Software requirements
- Architecture of the project
- Non-functional attributes like:
    - Reusability
    - Portability
    - Resource utilization

1 Description

1.1 Problem Perspective

The credit card default prediction is a machine learning model which helps us to predict whether the account will be defaulted or not.

1.2 Problem Statement

To create machine learning model to detect anomalies in credit repayment transaction and predict whether the account will be defaulted or not.

1.3 Proposed Solution

The proposed model will be capable of studying customers based on various parameters like gender, education qualification, marital status, previous repayment history and current outstanding balance. Based on that it can predict the account default status in advance and notify the respective bank official for further action.

1.4 Solution Improvements

Further we can enhance our model with additional information like customer expenditure pattern, CIBIL score, loan EMI or availability of any fixed deposit mapped to the account. These information's will improve the prediction accuracy and can suggest alternative methods to retrieve thepending dues in-case of future default.

2 Technical Requirements

There are no hardware requirements required for using this application, the user must have an interactive device which has access to the internet and must have the basic understanding of providing the input. And for the backend part the server must run all the software that is required for the processing the provided data and to display the results.

2.1 Tools Used

- Python 3.8 is used as the programming language and frame works like NumPy,pandas, sklearn and other modules for building the model.
- VSCode is used as IDE.
- For visualizations seaborn and parts of matplotlib are being used.
- For data collection Cassandra database is being used.
- Front end development is done using HTML/CSS.
- Flask is used for both data and backend deployment.
- GitHub is used for version control.

3 Data Requirements

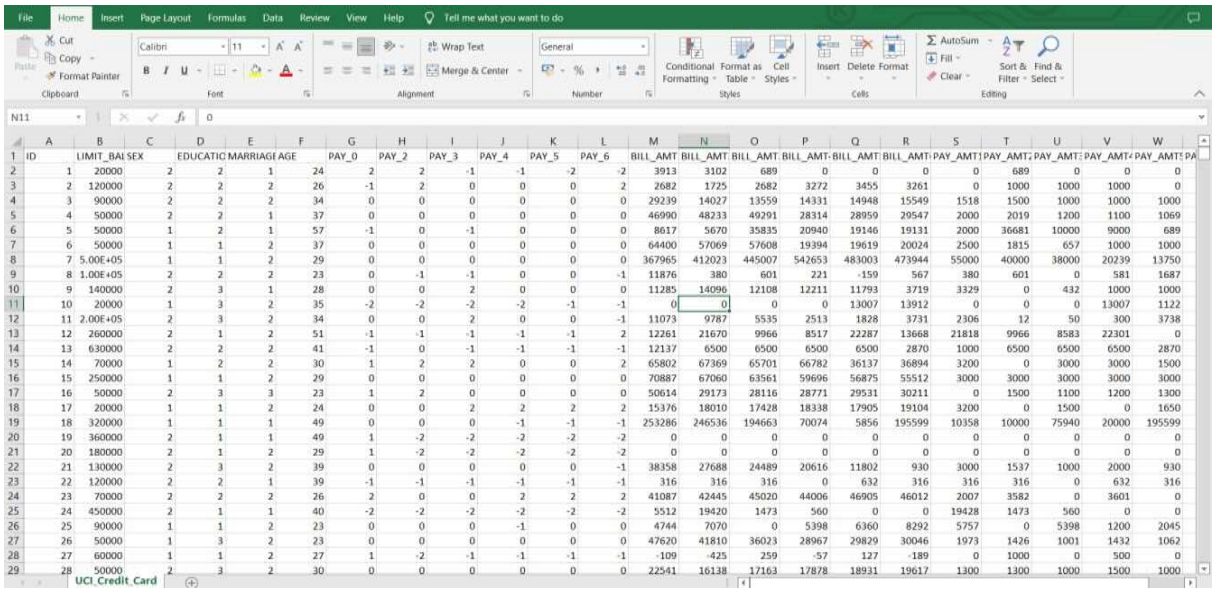
The data requirement is completely based on the problem statement. And the data set is available on the Kaggle in the form of csv(.csv). As the main theme of the project is to get the experience of real time problems, we are again importing the data into the Cassandra data base and exporting it into csv format.

3.1 Data Gathering from Main Source

The data for the current project is being gathered from Kaggle dataset, the link to the data is: https://www.kaggle.com/datasets/uciml/default-of-credit-card-clients-dataset.

3.2 Data Description

There are about 10k+ records of flight information such as airlines, data of journey, source, destination, departure time, arrival time, duration, total stops, additional information, and price. A glance of the dataset is shown below:



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
1	20000	2	2	1	24	2	2	-1	-1	-2	-2	3913	3102	689	0	0	0	0	689	0	0	0	
2	120000	2	2	2	26	-1	2	0	0	0	2	2682	1725	2682	3272	3455	3261	0	1000	1000	1000	0	
3	90000	2	2	2	34	0	0	0	0	0	0	29239	14027	13559	14331	14948	15549	1518	1500	1000	1000	1000	
4	50000	2	2	1	37	0	0	0	0	0	0	46990	48233	49291	28314	28959	29547	2000	2019	1200	1100	1069	
5	50000	1	2	1	57	-1	0	-1	0	0	0	8617	5670	35835	20940	19146	19131	2000	36681	10000	9000	689	
6	50000	1	1	2	37	0	0	0	0	0	0	64400	57069	57608	19394	19619	20024	2500	1815	657	1000	1000	
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	
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	
9	140000	2	3	1	28	0	0	2	0	0	0	11285	14096	12108	12211	11793	3719	3329	0	432	1000	1000	
10	20000	1	3	2	35	-2	-2	-2	-2	-1	-1	0	0	0	0	13007	13912	0	0	0	13007	1122	
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	
12	260000	2	1	2	51	-1	-1	-1	-1	-1	2	12261	21670	9966	8517	22287	13668	21818	9966	8583	22301	0	
13	630000	2	2	2	41	-1	0	-1	-1	-1	-1	12137	6500	6500	6500	6500	2870	1000	6500	6500	6500	2870	
14	70000	1	2	2	30	1	2	2	0	0	2	65802	67369	65701	66782	36137	36894	3200	0	3000	3000	1500	
15	250000	1	1	2	29	0	0	0	0	0	0	70887	67060	63561	59696	56875	55512	3000	3000	3000	3000	3000	
16	50000	2	3	3	23	1	2	0	0	0	0	50614	29173	28116	28771	29531	30211	0	1500	1100	1200	1300	
17	20000	1	1	2	24	0	0	2	2	2	2	15376	18010	17428	18338	17905	19104	3200	0	1500	0	1650	
18	320000	1	1	1	49	0	0	0	-1	-1	-1	253286	246536	194663	70074	5856	195599	10358	10000	75940	20000	195599	
19	360000	2	1	1	49	1	-2	-2	-2	-2	-2	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	
21	130000	2	3	2	39	0	0	0	0	0	-1	38358	27688	24489	20616	11802	930	3000	1537	1000	2000	930	
22	120000	2	2	1	39	-1	-1	-1	-1	-1	-1	316	316	316	0	632	316	316	316	0	632	316	
23	70000	2	2	2	26	2	0	0	2	2	2	41087	42445	45020	44006	46905	46012	2007	3582	0	3601	0	
24	450000	2	1	1	40	-2	-2	-2	-2	-2	-2	5512	19420	1473	560	0	0	19428	1473	560	0	0	
25	90000	1	1	2	23	0	0	0	-1	0	0	4744	7070	0	5398	6360	8292	5757	0	5398	1200	2045	
26	50000	1	3	2	23	0	0	0	0	0	0	47620	41810	36023	28967	29829	30046	1973	1426	1001	1432	1062	
27	60000	1	1	2	27	1	-2	-1	-1	-1	-1	-109	-425	259	-57	127	-189	0	1000	0	500	0	
28	50000	2	3	2	30	0	0	0	0	0	0	22541	16138	17163	17878	18931	19617	1300	1300	1000	1500	1000	

3.3 Import Data into Database

Created an api for the upload of the data into the Cassandra database, steps performed are:

- Connection is made with the database.
- Created a database with name flight fare.
- Cqlsh command is written for creating the data table with required parameters.
- And finally, a cqlsh command is written for uploading the dataset into the data tableby bulk insertion.

3.4 Export Data from Database

In the above created api, the download URL is also being created, which downloads the datainto a csv file format.

4 Data Pre-Processing

Steps performed in pre-processing are:

- First the data types are being checked and found only the price column is of typeinteger.
- Checked for null values if there are few null values, those rows are dropped.
- Converted all the required columns into the date time format.
- Performed one-hot encoding for the required columns.
- Scaling is performed for required data.

And, the data is ready for passing to the machine learning algorithm.

5 Design Flow

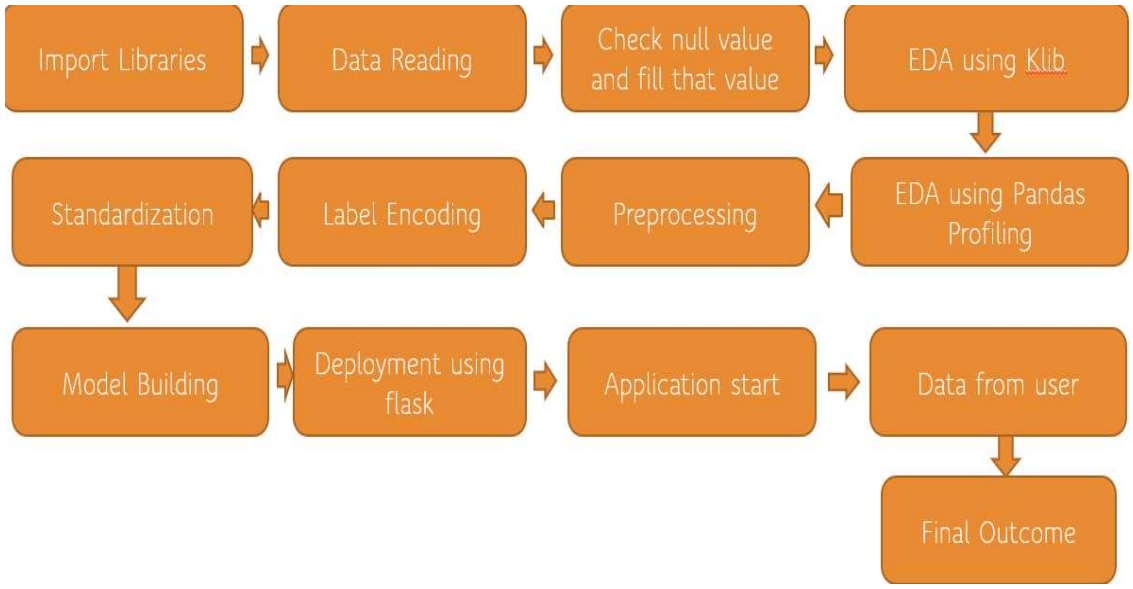
5.1 Modeling

The pre-processed data is then visualized and all the required insights are being drawn. Although from the drawn insights, the data is randomly spread but still modeling is performed with different machine learning algorithms to make sure we cover all the possibilities. And finally, as expected random forest regression performed well and further hyperparameter tuning is done to increase the model's accuracy.

5.2 UI Integration

Both CSS and HTML files are being created and are being integrated with the created machine learning model. All the required files are then integrated to the app.py file and tested locally.

5.3 Modelling Process & 5.4 Deployment Process



6 Data from User

The data from the user is retrieved from the created HTML web page.

7 Data Validation

The data provided by the user is then being processed by app.py file and validated. The validateddata is then sent for the prediction.

8 Rendering the Results

The data sent for the prediction is then rendered to the web page.

9 Deployment

The tested model is then deployed to Heroku. So, users can access the project from any internetdevices.

Conclusion

The flight fare prediction can predict the price based on the trained data set in the algorithm. Hence the user can know the approximate cost for their journey.

Q & A:

Q1) What's the source of data?

The data for training is provided by the client in multiple batches and each batch contain multiple files.

Q 2) What was the type of data?

The data was the combination of numerical and Categorical values.

Q 3) What's the complete flow you followed in this Project?

Refer Page no 6 for better Understanding.

Q 4) After the File validation what you do with incompatible file or files which didn't pass the validation?

Files like these are moved to the Achieve Folder and a list of these files has beenshared with the client and we removed the bad data folder.

Q 5) How logs are managed?

We are using different logs as per the steps that we follow in validation and modeling like File validation log, Data Insertion, Model Training log, prediction logetc.

Q 6) What techniques were you using for data pre-processing?

- Removing unwanted attributes
- Visualizing relation of independent variables with each other and output variables
- Checking and changing Distribution of continuous values
- Removing outliers
- Cleaning data and imputing if null values are present.
- Converting categorical data into numeric values.

Q 7) How training was done or what models were used?

- Before dividing the data in the training and validation set, we performed pre-processing overthe data set and made the final dataset.
- As per the dataset training and validation data were divided.
- Algorithms like Linear regression, SVM, Decision Tree, Random Forest, XGBoost were usedbased on the recall, final model was used on the dataset and we saved that model.

Q 8) How Prediction was done?

The testing files are shared by the client. We Performed the same life cycle on the provided dataset. Then, on the basis of the dataset, the model is loaded and prediction is performed. In the end we getthe accumulated data of predictions.

Q 9) What are the different stages of deployment?

- First, the scripts are stored on GitHub as a storage interface.
- The model is first tested in the local environment.
- After successful testing, it is deployed on Heroku

DPR