# DPR

**Credit Card Fraud Detection**

***PARAS DAHIYA***

**ABSTRACT**

In the dynamic landscape of financial risk assessment, the challenge of predicting credit defaults for commercial banks shares conceptual parallels with classification tasks in diverse domains. This project employs classical machine learning techniques encompassing Data Exploration, Cleaning, Feature Engineering, Model Building, and Testing to craft a robust solution. The goal is to predict the probability of credit default based on the distinctive characteristics and payment histories of credit card owners.

**CONTENTS**

Title Page i

Abstract ii

Content iii

|  |  |  |
| --- | --- | --- |
|  | Introduction | 1 |
|  | Why this DPR Documentation? | 1 |
|  | Key points: | 1 |
| 1 | Description | 1 |
| 1.1 | Problem Perspective | 1 |
| 1.2 | Problem Statement | 1 |
| 1.3 | Proposed Solution | 1 |
| 2 | Technical Requirements | 2 |
| 2.1 | Tools Used | 2 |
| 3 | Data Requirements | 3 |
| 3.1 | Data Gathering from Main Source | 3 |
| 3.2 | Data Description | 3 |
| 3.3 | Import Data into Database | 3 |
| 3.4 | Export Data from Database | 4 |
| 4 | Data Pre-Processing | 4 |
| 5 | Design Flow | 4 |
| 5.1 | Modelling | 4 |
| 5.2 | UI Integration | 4 |
| 5.3 | Modelling Process | 5 |
| 5.4 | Deployment Process | 5 |
| 6 | Data Validation | 6 |
| 7 | Model Training | 7 |
| 8 | Prediction | 8 |
| 9 | Deployment | 8 |
| 10 | Conclusion | 8 |
|  |  |  |

# INTRODUCTION

## Why this DPR Documentation?

The main purpose of this DPR documentation is to add the necessary details of the project and provide 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

## Problem Perspective

Credit Card Default Prediction is a machine learning model which helps us to predict w a person going to default or not.

## Problem Statement

Financial threats are displaying a trend about 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 credit card owner's

characteristics and payment history.

## Proposed Solution

The solution proposed to take dataset from the client and process all the provided data to meet the requirements of the machine learning model and finally displaying the output csv file.

# 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.

## Tools Used

* + - Python 3.9 is used as the programming language and frame works like numpy, pandas, sklearn and other modules for building the model.
    - PyCharm is used as IDE.
    - For visualizations seaborn and parts of matplotlib are being used.
    - For data collection SQl database is being used.
    - Front end development is done using HTML/CSS/JAVASCRIPT.
    - Flask is used for both data and backend deployment.
    - GitHub is used for version control.
    - Heroku is used for deployment.

# 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 excel sheet(.xlsx). As the main theme of the project is to get the experience of real time problems, we are again importing the data into the SQL 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/mushroom-classification

## 3.2 Data Description

Data Description: This dataset describes credit card data. They are classified into: Default payment or not.

1. ID: ID of each client
2. LIMIT\_BAL: Amount of given credit in NT dollars (includes individual and family/supplementary credit
3. SEX: Gender (1=male, 2=female)
4. EDUCATION: (1=graduate school, 2=university, 3=high school, 4=others, 5=unknown, 6=unknown)
5. MARRIAGE: Marital status (1=married, 2=single, 3=others)
6. AGE: Age in years
7. 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)
8. PAY\_2: Repayment status in August, 2005 (scale same as above)
9. PAY\_3: Repayment status in July, 2005 (scale same as above)
10. PAY\_4: Repayment status in June, 2005 (scale same as above)
11. PAY\_5: Repayment status in May, 2005 (scale same as above)
12. PAY\_6: Repayment status in April, 2005 (scale same as above)
13. BILL\_AMT1: Amount of bill statement in September, 2005 (NT dollar)
14. BILL\_AMT2: Amount of bill statement in August, 2005 (NT dollar)
15. BILL\_AMT3: Amount of bill statement in July, 2005 (NT dollar)
16. BILL\_AMT4: Amount of bill statement in June, 2005 (NT dollar)
17. BILL\_AMT5: Amount of bill statement in May, 2005 (NT dollar)
18. BILL\_AMT6: Amount of bill statement in April, 2005 (NT dollar)
19. PAY\_AMT1: Amount of previous payment in September, 2005 (NT dollar)
20. PAY\_AMT2: Amount of previous payment in August, 2005 (NT dollar)
21. PAY\_AMT3: Amount of previous payment in July, 2005 (NT dollar)
22. PAY\_AMT4: Amount of previous payment in June, 2005 (NT dollar)
23. PAY\_AMT5: Amount of previous payment in May, 2005 (NT dollar)
24. PAY\_AMT6: Amount of previous payment in April, 2005 (NT dollar)
25. default.payment.next.month: Default payment (1=yes, 0=no)

## 3.3 Import Data into Database

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

* + - Connection is made with the database.
    - Created a database with name Prediction.
    - SQL command is written for creating the data table with required parameters.
    - And finally, a SQL command is written for uploading the dataset into the data table by bulk insertion.

## 3.4 Export Data from Database

In the above created api, the download url is also being created, which downloads the data into a csv file format.

# 4. Data Pre-Processing

Steps performed in pre-processing are:

First the data types are being checked

There was no missing vlaues in the data. But when we go through the data, we find that missing values in one column is replaced with '?'.

Replaced such values with numpy "nan" so that we can handle the missing values.

Performed categorical imputing for the required columns.

Added quote on the the whole 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, HTML and JAVASCRIPT 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 Validation

In this step, we perform different sets of validation on the given set of training files.

1. Name Validation- We validate the name of the files based on the given name in the schema file. We have created a regex pattern as per the name given in the schema file to use for validation. After validating the pattern in the name, we check for the length of date in the file name as well as the length of time in the file name. If all the values are as per requirement, we move such files to "Good\_Data\_Folder" else we move such files to "Bad\_Data\_Folder."
2. Number of Columns - We validate the number of columns present in the files, and if it doesn't match with the value given in the schema file, then the file is moved to "Bad\_Data\_Folder."
3. Name of Columns - The name of the columns is validated and should be the same as given in the schema file. If not, then the file is moved to "Bad\_Data\_Folder".
4. The datatype of columns - The datatype of columns is given in the schema file. This is validated when we insert the files into Database. If the datatype is wrong, then the file is moved to "Bad\_Data\_Folder".
5. Null values in columns - If any of the columns in a file have all the values as NULL or missing, we discard such a file and move it to "Bad\_Data\_Folder".

# 7. Model Training

1) Data Export from Db - The data in a stored database is exported as a CSV file to be used for model training.

2) Data Preprocessing

a) Check for null values in the columns. If present, impute the null values using the KNN imputer.

b) Check if any column has zero standard deviation, remove such columns as they don't give any information during model training.

3) Clustering - KMeans algorithm is used to create clusters in the preprocessed data. The optimum number of clusters is selected by plotting the elbow plot, and for the dynamic selection of the number of clusters, we are using "KneeLocator" function. The idea behind clustering is to implement different algorithms

To train data in different clusters. The Kmeans model is trained over preprocessed data and the model is saved for further use in prediction.

4) Model Selection - After clusters are created, we find the best model for each cluster. We are using two algorithms, "XGBOOST" and "Naïve bayes". For each cluster, both the algorithms are passed with the best parameters derived from GridSearch. We calculate the AUC scores for both models and select the model with the best score. Similarly, the model is selected for each cluster. All the models for every cluster are saved for use in prediction.

# 8. Prediction

1) Data Export from Db - The data in the stored database is exported as a CSV file to be used for prediction.

2) Data Preprocessing

a) Check for null values in the columns. If present, impute the null values using the KNN imputer.

b) Check if any column has zero standard deviation, remove such columns as we did in training.

3) Clustering - KMeans model created during training is loaded, and clusters for the preprocessed prediction data is predicted.

4) Prediction - Based on the cluster number, the respective model is loaded and is used to predict the data for that cluster.

5) Once the prediction is made for all the clusters, the predictions along with the Wafer names are saved in a CSV file at a given location and the location is returned to the client.

# 9. Deployment

The tested model is then deployed to AWS. So, users can access the project from any internet devices.

# 10. Conclusion

The Credit Card Default web app can predict whether the person is going to do default or not based on the trained data set in the algorithm.