CREDIT CARD DEFAULT PREDICTION

Low Level Design (LLD)

AAKASH BHUTE

**1.INTRODUCTION**

Sometimes, even debts that seem manageable, like credit card debt, can spiral out of control due to unexpected life events such as job loss, medical emergencies, or business failure. Credit card debt is especially susceptible to this due to high finance charges and penalties. Many people can relate to missing a credit card payment or two due to forgetfulness or cash flow issues, but what happens when this becomes a consistent problem? To mitigate the risk of default, a model has been developed to predict customer default based on demographic data such as gender, age, and marital status, as well as behavioral data like past payments and transactions.

**2.PROBLEM STATEMENT**

The financial industry has made incredible strides, but commercial banks still face the challenge of predicting credit risk. One of the biggest threats they face is predicting the likelihood of credit default among their clients. The objective is to develop a model that can accurately predict the probability of credit default based on the characteristics and payment history of credit card owners.

**3.DATASET INFORMATION**

**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, 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, (-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, (scale same as above)

**PAY\_3:** Repayment status in July, (scale same as above)

**PAY\_4:** Repayment status in June, (scale same as above)

**PAY\_5:** Repayment status in May, (scale same as above)

**PAY\_6:** Repayment status in April, (scale same as above)

**BILL\_AMT1:** Amount of bill statement in September, (NT dollar)

**BILL\_AMT2:** Amount of bill statement in August, (NT dollar)

**BILL\_AMT3:** Amount of bill statement in July, (NT dollar)

**BILL\_AMT4:** Amount of bill statement in June, (NT dollar)

**BILL\_AMT5:** Amount of bill statement in May, (NT dollar)

**BILL\_AMT6:** Amount of bill statement in April, (NT dollar)

**PAY\_AMT1:** Amount of previous payment in September, (NT dollar) **PAY\_AMT2:** Amount of previous payment in August, (NT dollar)

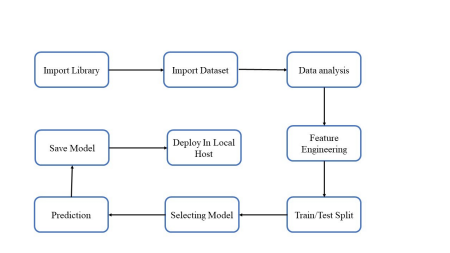
**PAY\_AMT3:** Amount of previous payment in July, (NT dollar)

**PAY\_AMT4:** Amount of previous payment in June, (NT dollar)

**PAY\_AMT5:** Amount of previous payment in May, (NT dollar)

**PAY\_AMT6:** Amount of previous payment in April, (NT dollar)

**default.payment.next.month:** Default payment (1=yes, 0=no)



**4.Architecture Description.**

**4.1. Data Description:** The dataset used in this project was obtained from Kaggle and it contains information on credit card clients in Taiwan from April 2005 to September 2005, including data on default payments, demographic factors, credit data, history of payment, and bill statements.

**4.2. Data Pre-processing:** The pre-processing step involved importing important libraries such as seaborn, matplotlib, pandas, and importing the dataset from Kaggle.

**4.3. Data Analysis**: In this step, we handled null values, changed column names, and created multiple visualizations using libraries such as seaborn and matplotlib to understand the data and the distribution of information. We did not find any null values in the data, so we proceeded with the analysis.

**4.4. Feature Engineering:** We merged two or more columns to gain more in-depth knowledge and information regarding the data.

**4.5. Train/Test Split:** We used the Sklearn library to divide the final dataset into a 80-20% ratio, where 80% of the data was used to train the model and the remaining 20% was used to predict the same.

**4.6. Selecting Model:** We tried and tested multiple models such as XGBoost, RandomForest, Decision Tree, and ADABoost, and found that the Random Forest Classifier performed the best.

**4.7. Prediction:** The accuracy of the Random Forest model was found to be 81.7% and the F1 score was 47.3%.

**4.8. Save Model:** We saved the model using the pickle library, which saves the file in a binary mode.

**4.9. Deploy in Local Host**: We created an HTML template and deployed the model through Flask.

