MINISTRY OF EDUCATION AND TRAINING

EASTERN INTERNATIONAL UNIVERSITY

MIS 451
MACHINE LEARNING FOR BUSINESS

Group Final Project

Classification: Default of Credit Card Clients

Lecturer: Mr. Dang Thai Doan and Ms. Huynh Gia Linh

Prepared by: **4 Pieces team**

Full Name	**IRN**
Nguyễn Thanh Giang	2132300593
Bùi Gia Tuệ	2132300511
Bùi Thị Thanh Thảo	2232300157
Trần Tiến Thảo Hiếu Ngân	2032300513

Quarter 3/2024-2025

TABLE OF CONTENTS

EXECUTIVE SUMMARY

This project aims to predict credit card defaults using the UCI "Default of Credit Card Clients" dataset. By applying both traditional machine learning and deep learning models, we seek to identify key factors influencing default risk.

The dataset comprises 30,000 records with 23 features, including demographic information, payment history, and billing details. Our analysis will assist financial institutions in making informed decisions regarding credit risk assessment.

1. INTRODUCTION

Credit card default poses significant challenges to financial institutions, impacting profitability and risk management. Predicting potential defaulters enables proactive measures to mitigate losses.

In this study, we utilize the UCI "Default of Credit Card Clients" dataset to develop predictive models that classify clients based on their likelihood of defaulting. By analyzing various attributes such as age, education, and payment history, we aim to uncover patterns that distinguish defaulters from non-defaulters.

2. DATASET OVERVIEW

The dataset, sourced from the UCI Machine Learning Repository, contains information on 30,000 credit card clients in Taiwan from April to September 2005. It includes 23 features:

- **Demographic Factors**: Gender, education level, marital status, and age.
- **Financial Data**: Credit limit, bill statements, and previous payments over six months.
- **Payment History**: Repayment status for each month from April to September 2005.

The target variable is a binary indicator of default payment in the next month (1 = default, 0 = no default). This dataset provides a comprehensive view of client behaviors and financial standings, making it suitable for classification tasks in credit risk modeling.

3. EXPLORATORY DATA ANALYSIS (EDA)

In this section, we explored the dataset to understand its structure, detect any missing or duplicate values, and examine the distribution of key features. These initial steps are essential to guide our data cleaning and modeling process.

To summarize the structure and properties of the dataset, we provide the following table:

Table 1: Structure and properties of the dataset

Aspect	Description
Number of records	30
Number of attributes	23 features + 1 target
Target variable	default payment next month (binary: 0 = no default, 1 = default)
Data types	Mostly numerical (integer and float); includes categorical numerical fields
Missing values	None detected
Duplicate entries	None detected
Class distribution	~77% non-default (0), ~23% default (1)

We also used visualizations to analyze the target variable.

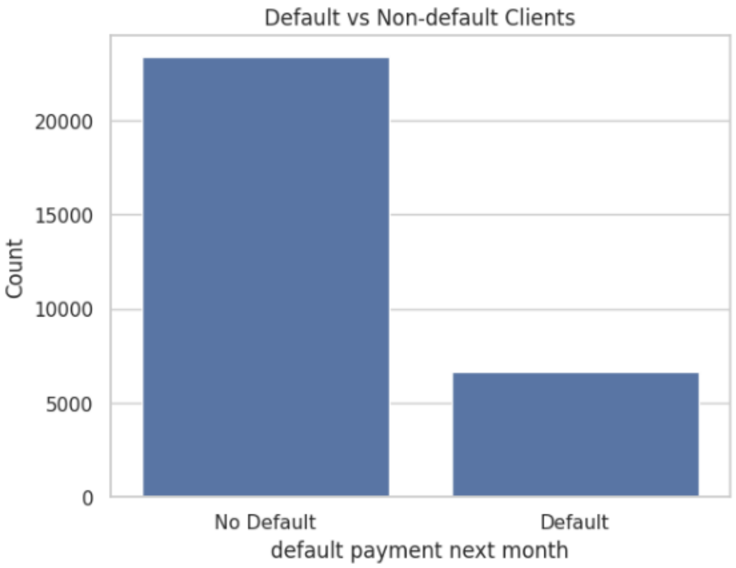


Figure 1: Count of default vs non-default clients

The distribution showed a significant class imbalance, which will need to be addressed in the preprocessing phase to avoid biased model learning.

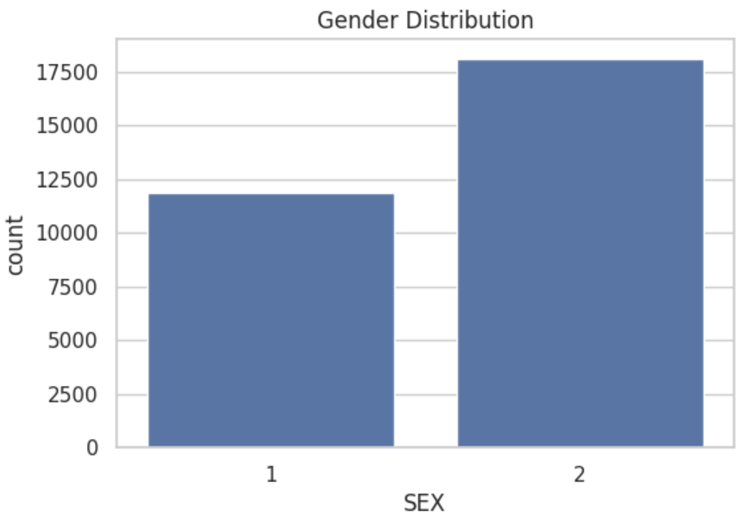


Figure 2: Gender Distribution

The dataset contains more female clients than male clients. Male (1) accounts for about 60% and female (2) is 40%.

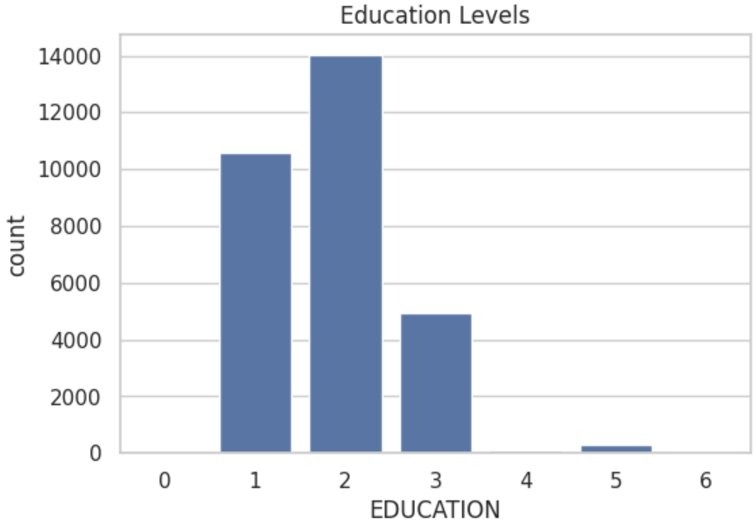


Figure 3: Education Levels

The most common level is university (label 2), followed by graduate school (1) and high school (3). The existence of codes like 0, 5, and 6 indicates input errors or undefined categories, which justifies our recoding in the part data transformation.

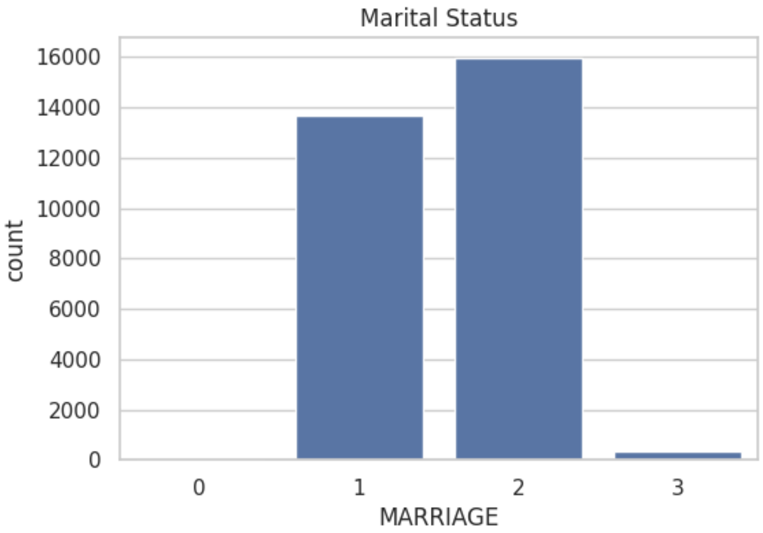


Figure 4: Marital Status

The majority of clients are married (1), followed by single (2). The presence of code 3 again signals the need for recoding into "others" (in data transformation part).

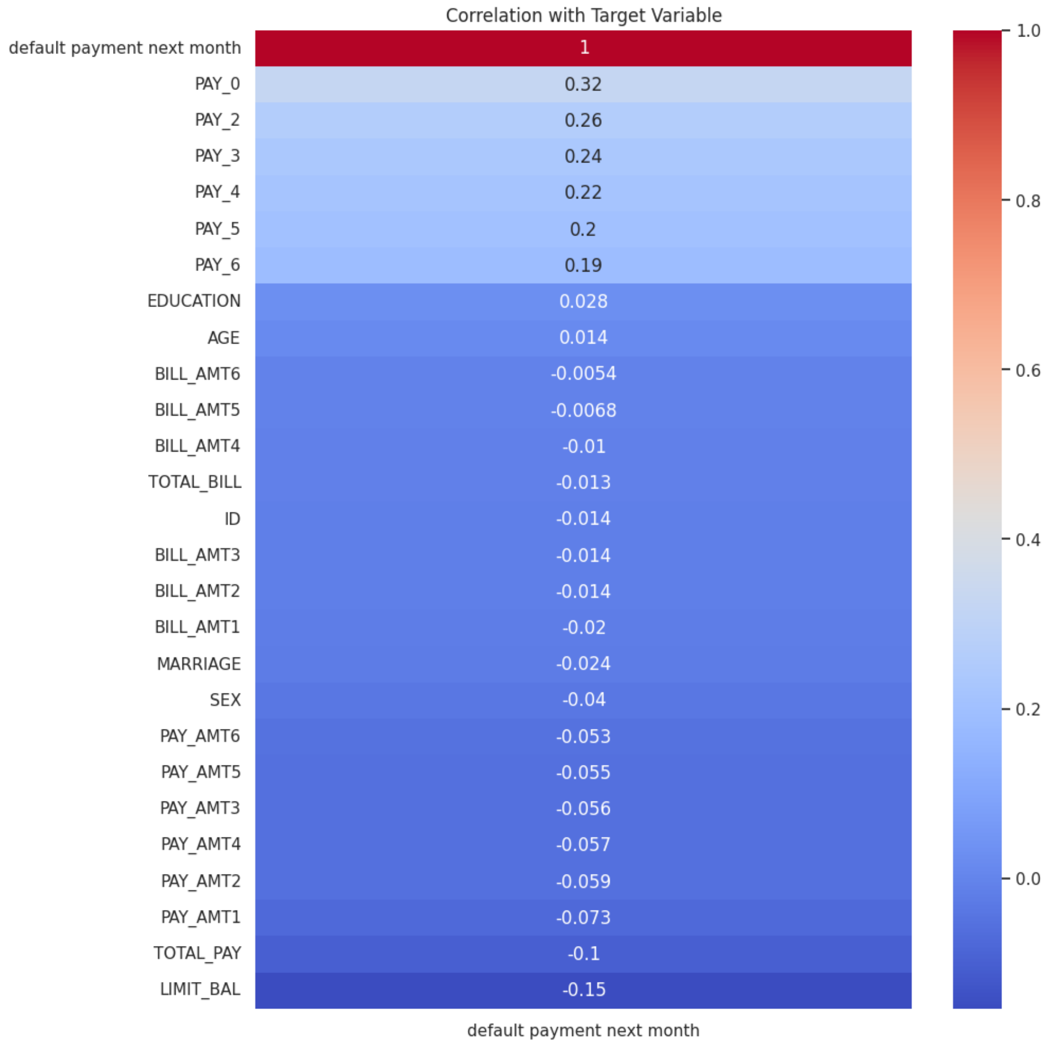


Figure 5: Heatmap correlation with target variables

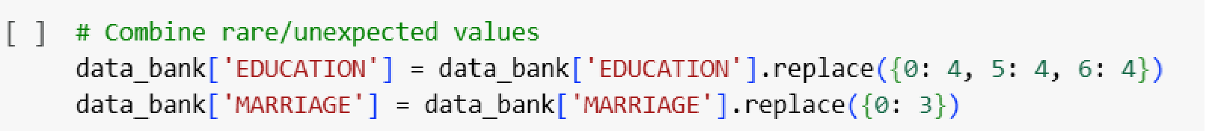
To better understand which features are most related to default, we calculated the correlation between each variable and the target. The result shows that recent repayment status variables (especially **PAY_0** (0.32), **PAY_2** (0.26), and **PAY_3** (0.24)) have the strongest positive correlation with default, suggesting that recent delays in payments are strong signals of risk.

On the other hand, features like credit limit (**LIMIT_BAL** (-0.15)**)** and total payment (**TOTAL_PAY** (-0.1)**)** show negative correlations, meaning customers with higher limits or payments tend to default less. Demographic variables such as gender, age, and education have very weak correlations and may not significantly affect predictions on their own.

From these analyses, we gained several key insights. First, the target variable is imbalanced, with fewer default cases. Second, some features may contain inconsistent or rare categorical values, which require cleaning. Third, the repayment history features appear to be among the most informative and will be crucial in model training.

4. DATA CLEANING AND TRANSFORMATION

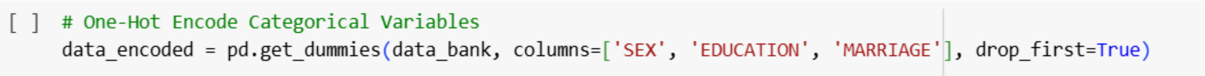
Before training our classification models, we performed several data cleaning and transformation steps to ensure the dataset was properly formatted, balanced, and ready for modeling. These steps included handling unusual values, encoding categorical variables, engineering new features, and balancing the dataset.



```
[ ]  # Combine rare/unexpected values
     data_bank['EDUCATION'] = data_bank['EDUCATION'].replace({0: 4, 5: 4, 6: 4})
     data_bank['MARRIAGE'] = data_bank['MARRIAGE'].replace({0: 3})
```

Figure 6: Handle invalid categorical values

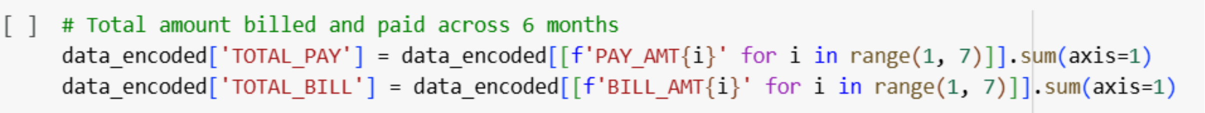
We began by examining the categorical variables for any unexpected or invalid values. For instance, the **EDUCATION** and **MARRIAGE** columns included codes such as 0, 5, and 6, which were not properly defined. We grouped these rare categories into a single new category labeled as “others” to avoid introducing noise into the model.



```
[ ]  # One-Hot Encode Categorical Variables
     data_encoded = pd.get_dummies(data_bank, columns=['SEX', 'EDUCATION', 'MARRIAGE'], drop_first=True)
```

Figure 7: One-hot encode categorical features

Next, we applied **One-Hot Encoding** to convert categorical variables (such as education, gender, and marital status) into numerical format. This step created separate binary columns for each category, allowing the models to process these features effectively.



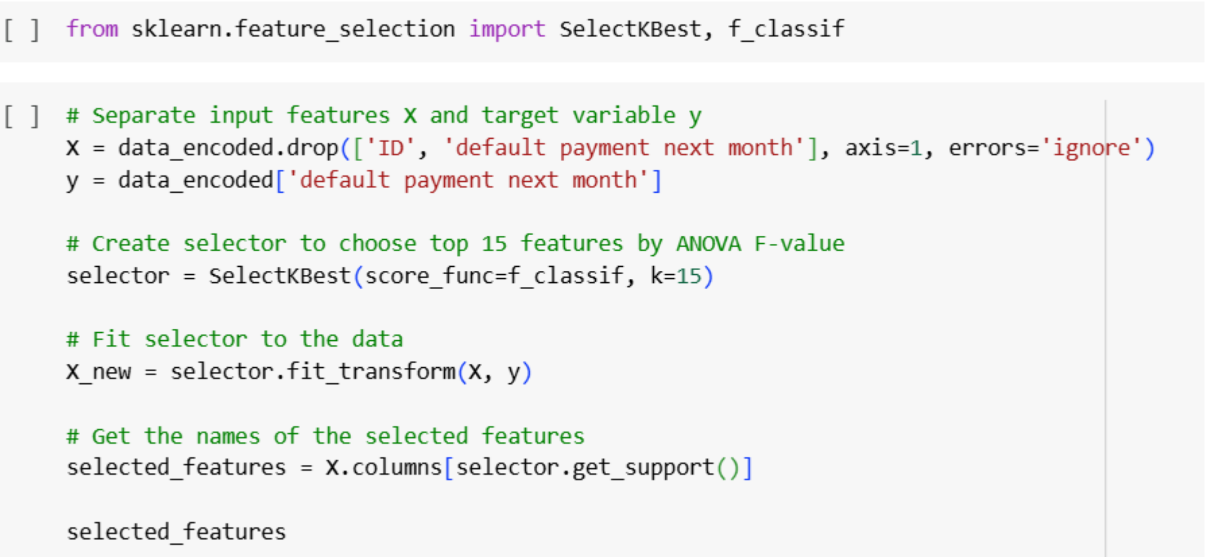
```
[ ]  # Total amount billed and paid across 6 months
     data_encoded['TOTAL_PAY'] = data_encoded[[f'PAY_AMT{i}' for i in range(1, 7)]].sum(axis=1)
     data_encoded['TOTAL_BILL'] = data_encoded[[f'BILL_AMT{i}' for i in range(1, 7)]].sum(axis=1)
```

Figure 8: Aggregate total payments and bills

To improve feature quality, we engineered two new variables:

(1)**TOTAL_BILL_AMT**: the total bill amount across six months (sum of **BILL_AMT1** to **BILL_AMT6**)

(2)**TOTAL_PAY_AMT**: the total payment amount across the same period (sum of **PAY_AMT1** to **PAY_AMT6**)

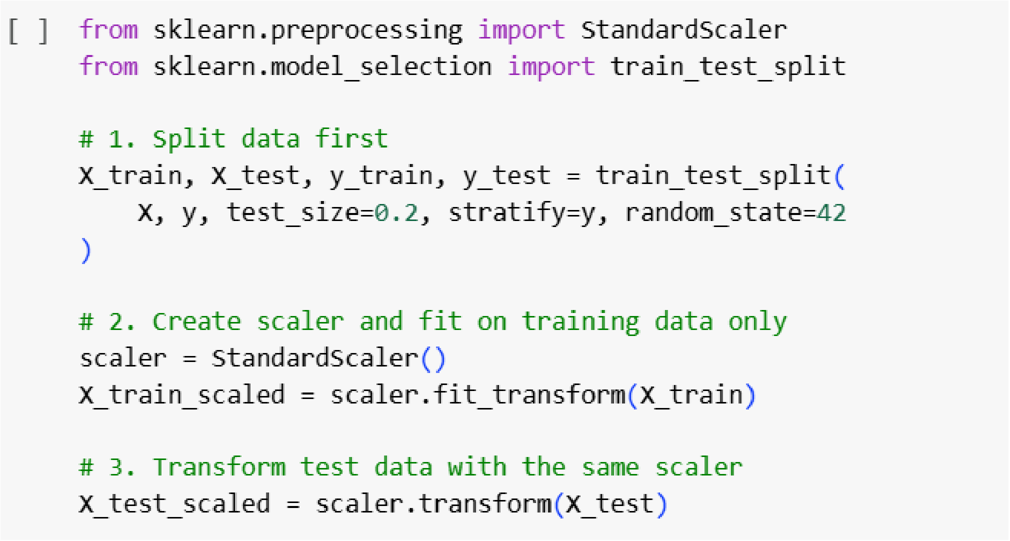


```python
from sklearn.feature_selection import SelectKBest, f_classif
```

```python
# Separate input features X and target variable y
X = data_encoded.drop(['ID', 'default payment next month'], axis=1, errors='ignore')
y = data_encoded['default payment next month']

# Create selector to choose top 15 features by ANOVA F-value
selector = SelectKBest(score_func=f_classif, k=15)

# Fit selector to the data
X_new = selector.fit_transform(X, y)

# Get the names of the selected features
selected_features = X.columns[selector.get_support()]

selected_features
```

Figure 9: Select Features

Afterward, we selected important features using **SelectKBest** with the ANOVA F-test (**f_classif**) method. This step helped reduce dimensionality and retained the most relevant features for predicting default.

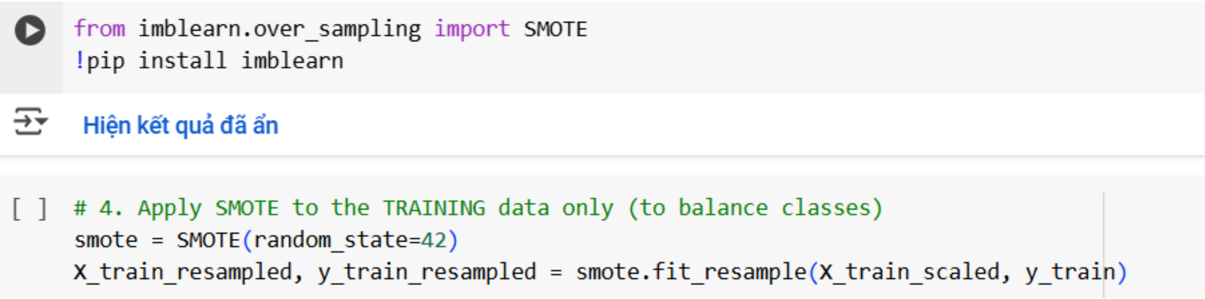


```python
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split

# 1. Split data first
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, stratify=y, random_state=42
)

# 2. Create scaler and fit on training data only
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)

# 3. Transform test data with the same scaler
X_test_scaled = scaler.transform(X_test)
```

Figure 10: Train/Test Split

We then split the dataset into training and testing subsets, ensuring that the distribution of the target variable was preserved in both sets.

To make the features comparable in scale, especially for algorithms like SVM and deep learning, we applied **StandardScaler** to standardize the input features. This transformed the data so that each feature had a mean of 0 and a standard deviation of 1.



```
from imblearn.over_sampling import SMOTE
!pip install imblearn
```

```
# 4. Apply SMOTE to the TRAINING data only (to balance classes)
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train_scaled, y_train)
```

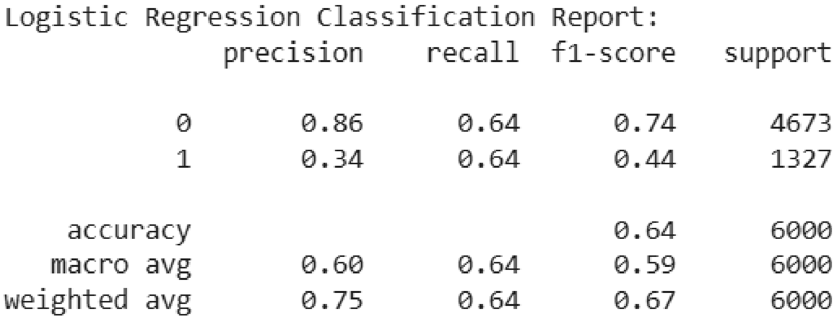
Figure 11: Applying SMOTE for training set

Lastly, we addressed the class imbalance in the training set using **SMOTE (Synthetic Minority Over-sampling Technique)**. This technique created synthetic examples of the minority class (defaults) to balance the class distribution, allowing the models to learn more effectively from both classes.

These preprocessing steps helped improve the data quality, reduce bias, and prepare the dataset for accurate and fair model training.

5. MODEL DEVELOPMENT

a. Logistic Regression



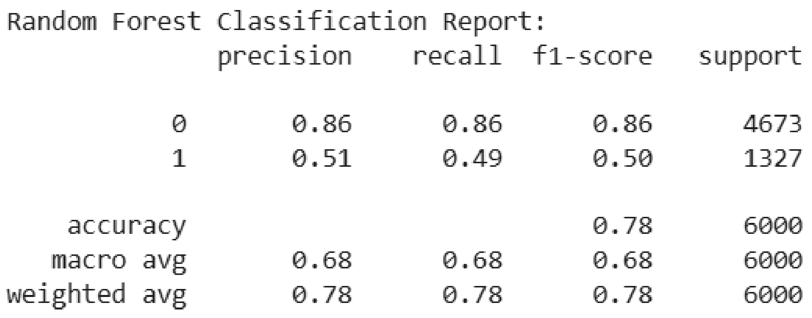
```
Logistic Regression Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.64      0.74      4673
           1       0.34      0.64      0.44      1327

    accuracy                           0.64      6000
   macro avg       0.60      0.64      0.59      6000
weighted avg       0.75      0.64      0.67      6000
```

Figure 12: Logistic Regression Classification Report

We selected logistic regression as a baseline model because it is simple, fast, and interpretable. It is commonly used in binary classification problems like default prediction, especially when we want to understand the influence of each feature.

The model achieved an accuracy of **64%**. It performed well on non-default cases (precision: **0.86**, recall: **0.64**), but struggled with default cases (precision: **0.34**, F1-score: **0.44**). This indicates that while the model can detect many defaulters, it also produces many false alarms.

b. Random Forest



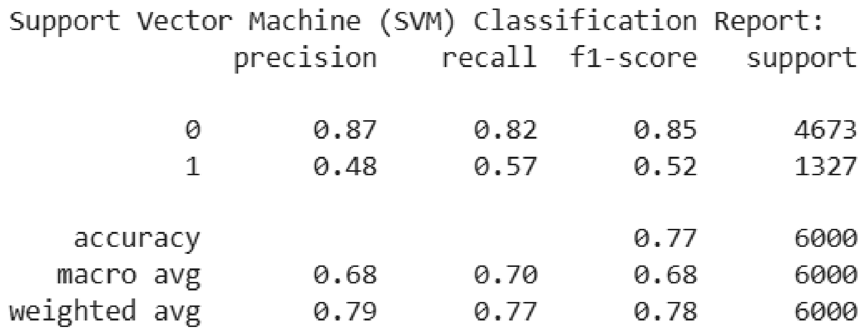
```
Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.86      0.86      4673
           1       0.51      0.49      0.50      1327

    accuracy                           0.78      6000
   macro avg       0.68      0.68      0.68      6000
weighted avg       0.78      0.78      0.78      6000
```

Figure 13: Random Forest Classification Report

We chose random forest because it is robust to noise, handles feature interactions well, and often outperforms simpler models in classification tasks. It also helps reduce overfitting by using an ensemble of decision trees.

The model reached an accuracy of **78%**. It maintained strong performance on non-defaults (F1-score: **0.86**) and improved on defaults compared to logistic regression, with better precision (**0.51**) and a balanced F1-score of **0.50**. This makes it a more reliable model for identifying high-risk clients.

c. Support Vector Machine (SVM)



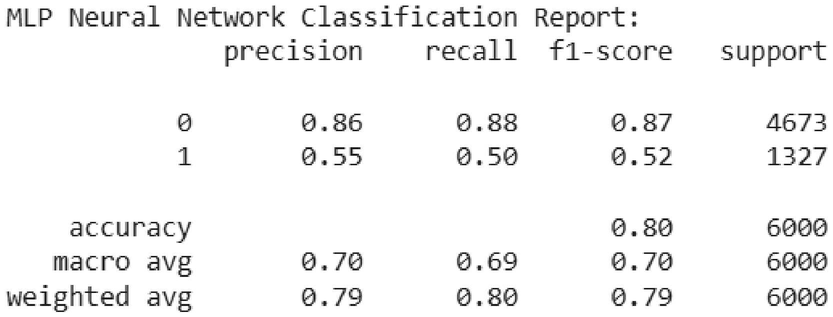
```
Support Vector Machine (SVM) Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.82      0.85      4673
           1       0.48      0.57      0.52      1327

    accuracy                           0.77      6000
   macro avg       0.68      0.70      0.68      6000
weighted avg       0.79      0.77      0.78      6000
```

Figure 14: Support Vector Machine (SVM) Classification Report

We selected SVM with an RBF kernel because it is effective in high-dimensional spaces and can model complex non-linear decision boundaries. It is also less prone to overfitting in cases with many features.

The model reached an accuracy of **77%**. It performed well on non-defaults (F1-score: **0.85**) and better than logistic regression on defaults, with a precision of **0.48** and an F1-score of **0.52**. This shows a balanced ability to detect defaulters without too many false positives.

d. MLP Neural Network

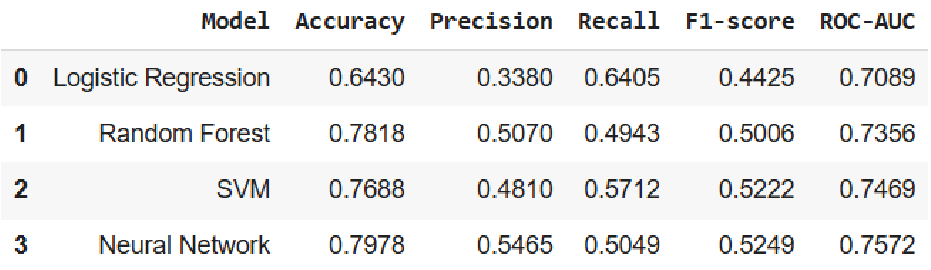


```
MLP Neural Network Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.88      0.87      4673
           1       0.55      0.50      0.52      1327

    accuracy                           0.80      6000
   macro avg       0.70      0.69      0.70      6000
weighted avg       0.79      0.80      0.79      6000
```

Figure 15: MLP Neural Network Classification Report

We applied a Multi-layer Perceptron (MLP) using Keras to test a deep learning approach. MLP is powerful at capturing complex patterns in data, especially after scaling and resampling.

This model gave the highest accuracy so far at **80%**. It performed strongly on non-defaults (F1-score: **0.87**) and achieved an F1-score of **0.52** for defaults, similar to SVM. The overall macro average (F1: **0.70**) suggests balanced performance across both classes.



	Model	Accuracy	Precision	Recall	F1-score	ROC-AUC
0	Logistic Regression	0.6430	0.3380	0.6405	0.4425	0.7089
1	Random Forest	0.7818	0.5070	0.4943	0.5006	0.7356
2	SVM	0.7688	0.4810	0.5712	0.5222	0.7469
3	Neural Network	0.7978	0.5465	0.5049	0.5249	0.7572

Figure 16: Summary metrics

To further evaluate model performance, we considered the ROC-AUC (Receiver Operating Characteristic – Area Under the Curve) score. This metric measures a model's ability to distinguish between the two classes - in our case, default and non-default clients. A ROC-AUC score of 1.0 indicates perfect separation, while a score of 0.5 means the model performs no better than random guessing.

Among the models we tested, the Neural Network achieved the highest ROC-AUC score of **0.7572**, followed closely by the SVM (**0.7469**) and Random Forest (**0.7356**). The Logistic Regression model had the lowest ROC-AUC of **0.7089**, although still above the acceptable threshold of 0.7. These results suggest that while all models demonstrated a decent ability to separate the classes, the Neural Network was the most effective at identifying patterns that differentiate defaulters from non-defaulters.

6. INTERPRETATION AND INSIGHTS

From our model evaluation results, several meaningful insights emerged that could inform real-world business decisions.

First, the most influential features in predicting default were recent repayment behaviors, particularly the **PAY_0**, **PAY_2**, and **PAY_3** variables. These indicate that clients who recently missed or delayed payments are significantly more likely to default in the following month. This supports the idea that **payment patterns over the most recent months are strong early warning signals** of financial risk.

Second, models that captured non-linear relationships and feature interactions—such as **Random Forest, SVM, and MLP Neural Network**—performed better than the simpler **Logistic Regression**. Among them, the **Neural Network achieved the best ROC-AUC score (0.7572)** and the most balanced classification performance, suggesting it is the most reliable model for identifying defaulters.

Third, demographic features like gender, age, education, and marital status showed only weak correlations with default behavior. This implies that while such features may provide additional context, they are **not strong predictors on their own**. Financial behavior and payment history remain the most critical factors.

From a business standpoint, these findings suggest that **monitoring recent repayment status and payment activity can significantly improve early risk detection**. Institutions can use such models to prioritize follow-up with at-risk clients, offer financial counseling, or adjust credit policies proactively.

7. DATA COLLECTION FOR BANKING INSTITUTIONS

To support **credit default prediction models** in this project, Taiwanese banks may consider collecting data from three key sources:

First, internal data sources typically include **readily available information** such as demographic attributes (e.g., age, gender), financial indicators (e.g., credit limits, total bill amounts), and historical repayment behavior. These variables form the foundation for most baseline classification models and are essential for understanding customer profiles.

Second, external public sources can **complement internal data**. In countries like Vietnam, evolving regulatory frameworks now permit financial institutions to access centralized credit information systems. For instance, a customer's registered mobile number may be linked with other financial obligations, offering additional context on cross-institutional borrowing. Such data enhances early warning systems and reduces dependence on interbank sharing.

Third, commercial or third-party sources provide more **granular insights**. These may include credit scores, active loan records from other banks, or consumer transaction trends. Although not always freely available, this data can be obtained through licensed credit bureaus or financial analytics providers, enriching the model’s feature set and supporting a more accurate view of client creditworthiness.

8. REFERENCES

UCI Machine Learning Repository. (n.d.). https://archive.ics.uci.edu/dataset/350/default+of+credit+card+clients