

UTM

UNIVERSITI TEKNOLOGI MALAYSIA

FACULTY OF COMPUTING
SEMESTER II SESSION
2025/2026

SECP2753-02 PERLOMBONGAN DATA
(DATA MINING)

PROJECT 1

NAME	MATRIC NUMBER
KAVIVARTHAN A/L MANNIVANAN	A24CS0093
RICHARD YONATHAN JULIO CLAY HENG	X25EC3019
NANDINI VENGAT	X25EC3021
NIVEETHITA A/P PANDIA RAJAN	A24CS0148
KALAITHARAN A/L PALANYVELU	A24CS0091

Table of Contents

1.0 Introduction **4**
2.0 Business and Data Understanding **5**
2.1 Problem Formulation 5
2.2 Dataset Overview 5
2.3 Importance of the Dataset 7
3.0 Data Preparation and Processing **7**
3.1 Overview of Data Preparation and its Importance 7
3.2 Data Preparation Steps 8
3.3 Application of Suitable Processing Methods 11
3.4 Use of Suitable Tools (Python) 11
3.5 Generation of Clean Dataset 16
3.6 Data Preparation Results 17
4.0 Model Development **18**
4.1 Overview 18
4.2 Data Splitting Strategy 18
4.3 Feature Engineering and Preprocessing 18
4.3.1 Feature Engineering 18
4.3.2 Categorical Encoding 18
4.3.3 Missing Value Imputation 19
4.3.4 Feature Scaling 19
4.4 Classification Algorithms 19
4.4.1 Logistic Regression 19
4.4.2 Decision Tree Classifier 19
4.4.3 Random Forest Classifier 19
4.4.4 Gradient Boosting Classifier 19
4.4.5 Support Vector Machine (SVM) 19
4.4.6 K-Nearest Neighbours (KNN) 20
4.5 Model Validation Strategy 20
4.6 Results Summary 22
5.0 Model Performance & Evaluation **23**
5.1 Application of Evaluation Metrics 23
5.2 Analysis of Model Performance 24
5.3 Reporting of Model Performance Results 25
6.0 References **26**

1.0 Introduction

In modern banking, loan approval decisions directly affect financial risk, business growth, and customer trust. Manual assessment of diverse loan applications can be slow and inconsistent, as it relies heavily on individual judgment and is prone to both approving high-risk applicants and rejecting qualified ones. To address this, lenders increasingly need data-driven tools that can support more consistent, transparent, and efficient credit evaluation, without removing human oversight from the decision-making process.

With this prepared data, several supervised learning models are implemented in Python, including Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, Support Vector Machine, and K-Nearest Neighbours. Using these algorithms, the program learns patterns that link applicant characteristics to loan approval outcomes, and evaluates the models using metrics such as accuracy, precision, recall, F1-Score, and ROC-AUC. From a business perspective, these programmed models help banks estimate the risk of approving each application, reduce manual workload, and improve the consistency of loan decisions, while still allowing credit officers to keep final control. The result is a practical, code-driven decision support tool that turns raw loan data into actionable insights for managing credit risk and identifying promising customers.

2.0 Business and Data Understanding

2.1 Problem Formulation

Financial institutions receive a large number of loan applications from individuals with different financial backgrounds, employment statuses, and credit histories. Evaluating each application manually can be a time-consuming process and may result in inconsistencies due to human judgment. Furthermore, incorrect loan approval decisions can lead to significant financial losses when borrowers fail to repay their loans. On the other hand, rejecting qualified applicants may reduce business opportunities and negatively affect customer satisfaction. Therefore, banks and lending institutions require a more efficient and reliable approach to assess loan applications.

The problem addressed in this project is the challenge of predicting loan eligibility based on applicant information. By analysing factors such as income level, marital status, education, employment status, credit history, and property area, it is possible to identify patterns associated with successful loan approvals. Data mining techniques can be utilised to automate this decision-making process and support financial institutions in making accurate and consistent lending decisions.

This project aims to explore how applicant characteristics can be used to predict loan approval outcomes. The insights obtained from the analysis can assist financial organizations in reducing risk, improving operational efficiency, and enhancing the overall loan evaluation process. Since loan approval prediction involves determining a specific outcome based on historical data, it is well-suited for predictive analytics and supervised learning approaches within the field of data mining.

2.2 Dataset Overview

The Loan Prediction Dataset contains historical records of loan applicants and their loan approval status. The dataset includes several attributes that describe applicants' demographic, financial, and credit-related characteristics.

Attribute	Description
Loan_ID	Unique applicant identification
Gender	Applicant gender
Married	Marital status
Dependents	Number of dependents
Education	Education level
Self_Employed	Self-employment status
ApplicantIncome	Applicant monthly income

Attribute	Description
CoapplicantIncome	Co-applicant monthly income
LoanAmount	Requested loan amount
Loan_Amount_Term	Loan repayment period
Credit_History	Credit history record
Property_Area	Property location category

The loan prediction dataset contains demographic, financial, and credit-related information of loan applicants. The dataset consists of 367 records and 12 attributes, where each attribute represents a specific characteristic that may influence the loan approval decision. The variables can be categorized into categorical attributes such as Gender, Married, Education, Self_Employed, and Property_Area, as well as numerical attributes such as ApplicantIncome, CoapplicantIncome, LoanAmount, and Loan_Amount_Term. These attributes collectively provide a comprehensive representation of an applicant's financial background and repayment capability.

Among all the variables, Credit_History is expected to have a significant influence on loan approval decisions because it reflects an applicant's previous repayment behavior. Similarly, ApplicantIncome and CoapplicantIncome indicate the financial capacity of applicants to repay the loan, while LoanAmount represents the amount requested from the financial institution. Demographic attributes such as Gender, Marital Status, Education Level, and Number of Dependents provide additional information that may contribute to the prediction process.

The dataset contains both numerical and categorical data types, making it suitable for applying various data preprocessing techniques such as missing value treatment, data transformation, and categorical encoding.

Datatype:

Categorical Variables	Numerical Variables
• Gender • Married • Dependents • Education • Self_Employed • Property_Area	• ApplicantIncome • CoapplicantIncome • LoanAmount • Loan_Amount_Term • Credit_History

2.3 Importance of the Dataset

The loan prediction dataset is important because it contains relevant information for analysing factors that affect loan approval decisions. The dataset includes demographic attributes such as Gender, Married, Education, and Dependents, as well as financial attributes such as ApplicantIncome, CoapplicantIncome, LoanAmount, and Credit_History. These variables provide valuable insights into an applicant's financial capability and repayment potential.

Furthermore, the dataset reflects a real-world banking scenario in which financial institutions must determine whether applicants are eligible for loans. By analyzing these attributes, data mining techniques can identify patterns and relationships that support more accurate and efficient decision-making. Therefore, this dataset serves as a suitable foundation for developing predictive models that can assist banks in reducing risk and improving the loan approval process.

3.0 Data Preparation and Processing

3.1 Overview of Data Preparation and its Importance

Data Preparation is an essential step in the data analysis and machine learning process. Often, the initial data set will have missing information, anomalies, and other problems such as inconsistencies, noise, and formatting that can distort analysis or lead to inaccurate predictions. Data preparation involves cleaning the data to prepare it for further analysis.

The reason for performing data preparation for the Loan Prediction dataset was that there were missing values in some attributes. This problem needs to be resolved to avoid producing erroneous predictions or even preventing algorithms from functioning properly due to dirty data.

3.2 Data Preparation Steps

The data preparation process was carried out systematically to ensure that the Loan Prediction Dataset was accurate, complete, and suitable for further analysis and machine learning modelling. The following steps were performed:

Step 1: Data Acquisition

The first stage involved obtaining the original dataset from the provided archive file. The archive contained two datasets:

- **Training Dataset** (train_u6lujuX_CVtuZ9i.csv) containing 614 records and 13 attributes.
- **Testing Dataset** (test_Y3wMUE5_7gLdaTN.csv) containing 367 records and 12 attributes.

This dataset consisted of information on applicants like gender, marital status, income level, loan amount, credit score, and loan status. These datasets were then imported in Python through the Pandas library for easy manipulation.

The aim of this step was to load the data successfully into the program for further analysis.

Step 2: Data Inspection

After loading the datasets, an initial inspection was conducted to understand the structure and characteristics of the data.

Several checks were performed:

- Determining the number of records and attributes.
- Identifying data types for each attribute.
- Examining sample records using the head() function.

- Generating summary statistics for numerical variables.
- Detecting missing values and potential data quality issues.

This step provided a comprehensive understanding of the dataset and helped identify attributes that required cleaning before analysis.

Example findings from the inspection process included:

Dataset	Records	Attributes
Training Dataset	614	13
Testing Dataset	347	12

The inspection stage revealed the presence of missing values in several attributes, indicating the need for preprocessing.

Step 3: Missing Value Analysis

A missing value analysis was performed to detect records with missing values that might hinder the performance of the machine learning algorithm.

Attributes having missing values were:

Attribute	Missing Value
Credit_History	50
Self_Employed	32
LoanAmount	22
Dependents	15
Loan_Amount_Term	14
Gender	13
Married	3

Step 4: Data Cleaning

Data cleaning was performed to handle missing values and improve overall dataset quality.

Different imputation methods were applied according to the attribute type:

Mode Imputation for Categorical Variables

The following categorical attributes were completed using the most frequently occurring value (mode):

- Gender

- Married
- Dependents
- Self_Employed
- Credit_History

Mode imputation was selected because it preserves the existing category distribution while minimizing distortion of the data.

Median Imputation for Numerical Variables

The following numerical attributes were completed using the median value:

- LoanAmount
- Loan_Amount_Term

Median imputation was chosen because it is less sensitive to outliers than the mean and is therefore more suitable for financial datasets where extreme values may exist.

After applying these techniques, all missing values were successfully replaced.

Step 5: Validation

Once the cleansing steps had been completed, the validation step was implemented to check whether the cleansing efforts had been accomplished successfully.

Steps carried out in the validation process include:

- Counting number of missing values again.
- Checking consistency of data in all fields.
- Looking for any duplicate entries.
- Checking that there was no data corruption during the cleansing process.
- Checking the statistical measures to ensure that appropriate figures still existed.

The validation outcomes indicated that:

- There was no missing value left in both data sets.
- No duplicate entries had to be removed.
- Integrity of the datasets had been preserved.

Step 6: Dataset Export

The last step was the export of the processed datasets to use in the next stages.

Processed datasets include:

- train_cleaned.csv
- test_cleaned.csv

Both processed datasets are saved in CSV format due to its compatibility with data analytics and machine learning platforms.

Main characteristics of the exported datasets are as follows:

- No missing values.
- Same attributes.
- Ready for EDA.
- Predictive modeling.

Successful completion of this step resulted in a well-processed dataset, which will be successfully used in further stages of the project.

3.3 Application of Suitable Processing Methods

For categorical data (Gender, Married, Dependents, Self_Employed, and Credit_History), Mode Imputation technique was used since they are categorical features and thus the mode value retains the distribution.

In case of numerical data (LoanAmount and Loan_Amount_Term), the Median Imputation technique was used because the median is robust and does not depend on extreme data present in any dataset.

Finally, the data was further validated to confirm that all missing values had been taken care of during pre-processing.

3.4 Use of Suitable Tools (Python)

Python was selected as the primary preprocessing tool.

Tools and Libraries Used:
• Pandas – Data loading, cleaning, and manipulation.
• NumPy – Numerical processing support.
• Matplotlib – Data visualization.
• Jupyter Notebook – Interactive documentation and workflow execution.

Python was chosen because it provides extensive support for data science tasks, reproducible workflows, and integration with machine learning frameworks.

Here are some important screenshots from the notebook created by using Python:

Load Dataset

[11]:

```
train = pd.read_csv('train_cleaned.csv')
test = pd.read_csv('test_cleaned.csv')

print('Train Shape:', train.shape)
print('Test Shape:', test.shape)

train.head()
```

```
Train Shape: (614, 13)
Test Shape: (367, 12)
```

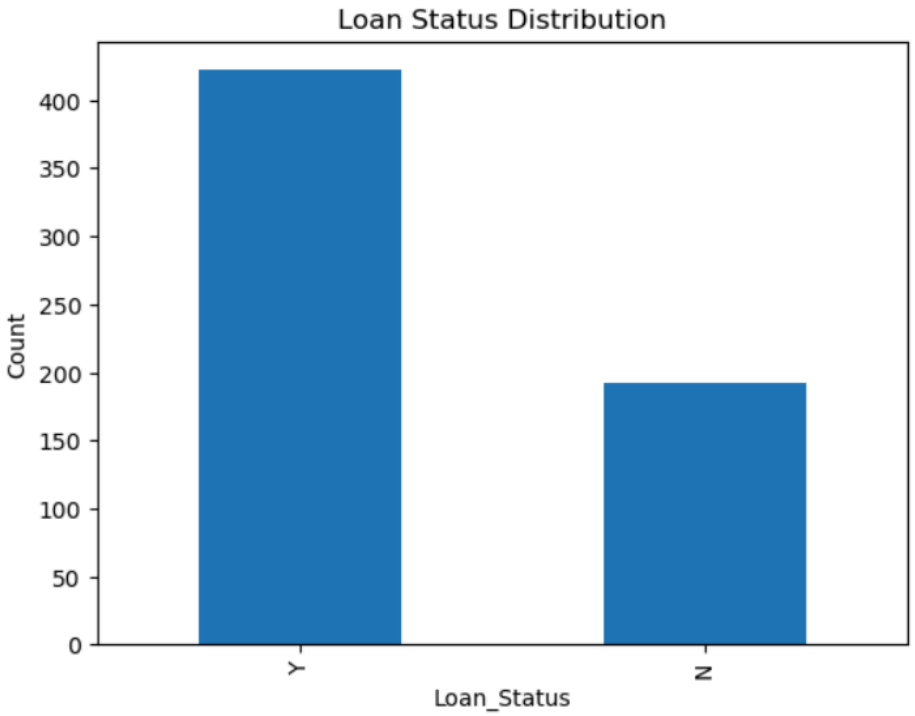
[11]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	Lo
0	LP001002	Male	No	0	Graduate	No	5849	0.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	

Visualization 1: Loan Status Distribution

[16]:

```
train_clean['Loan_Status'].value_counts().plot(kind='bar')
plt.title('Loan Status Distribution')
plt.ylabel('Count')
plt.show()
```



Duplicate Records

[14]:

```
print("Train duplicates:", train_clean.duplicated().sum())
print("Test duplicates:", test_clean.duplicated().sum())
```

```
Train duplicates: 0
Test duplicates: 0
```

```
def clean_df(df):
    df = df.copy()

    cat_cols = ['Gender','Married','Dependents','Self_Employed']

    for col in cat_cols:
        if col in df.columns:
            df[col] = df[col].fillna(df[col].mode()[0])

    if 'LoanAmount' in df.columns:
        df['LoanAmount'] = df['LoanAmount'].fillna(df['LoanAmount'].median())

    if 'Loan_Amount_Term' in df.columns:
        df['Loan_Amount_Term'] = df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].median())

    if 'Credit_History' in df.columns:
        df['Credit_History'] = df['Credit_History'].fillna(df['Credit_History'].mode()[0])

    return df

train_clean = clean_df(train)
test_clean = clean_df(test)

print("Remaining Missing Values (Train):")
print(train_clean.isnull().sum().sum())

print("Remaining Missing Values (Test):")
print(test_clean.isnull().sum().sum())
```

```
Remaining Missing Values (Train):
0
Remaining Missing Values (Test):
0
```

Export Cleaned Data

[19]:

```
train_clean.to_csv('train_cleaned.csv', index=False)
test_clean.to_csv('test_cleaned.csv', index=False)

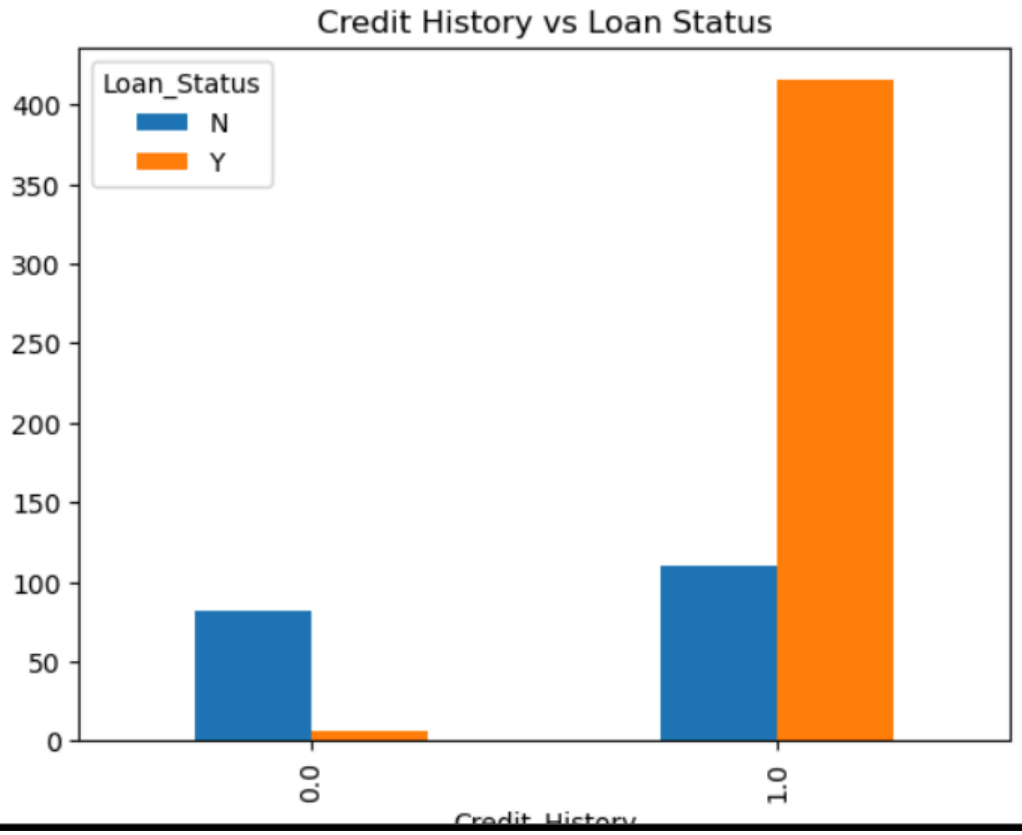
print('Files exported successfully')
```

```
Files exported successfully
```

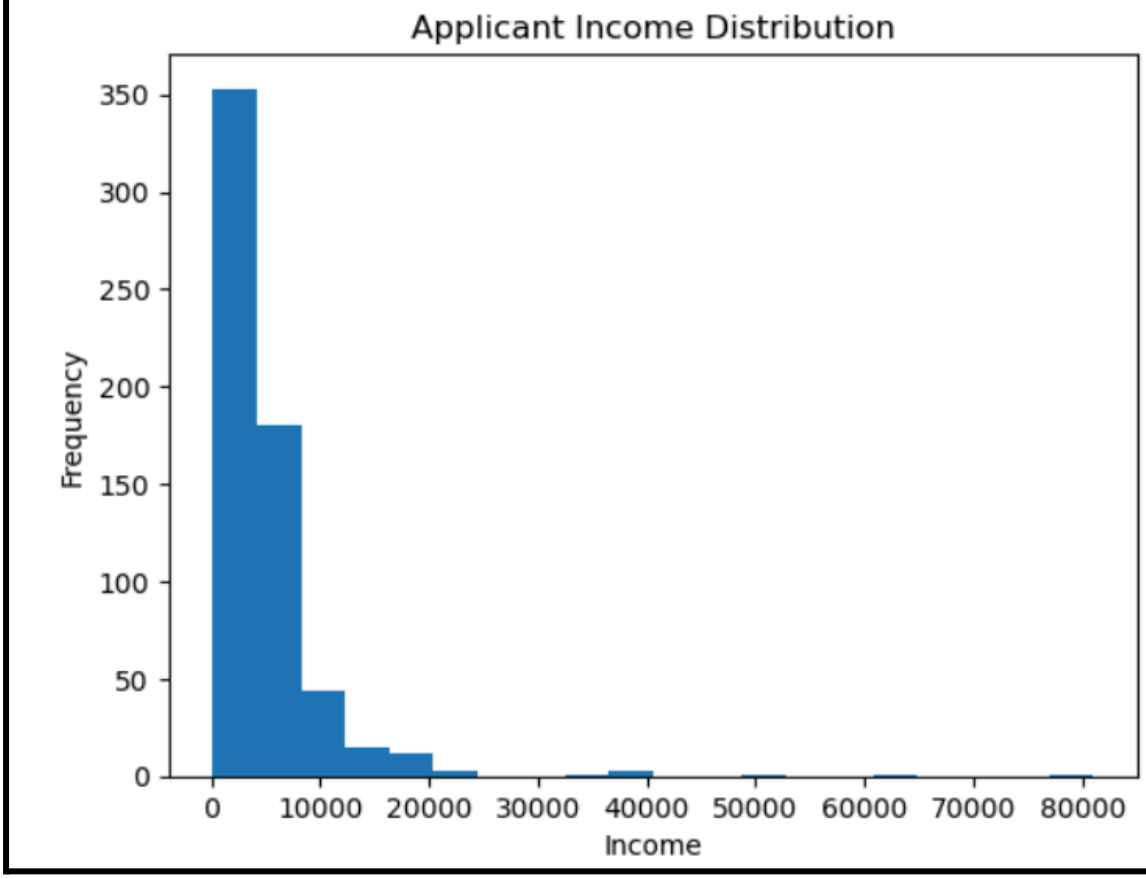
[18]:

```
pd.crosstab(
    train_clean['Credit_History'],
    train_clean['Loan_Status']
).plot(kind='bar')

plt.title('Credit History vs Loan Status')
plt.show()
```



```
plt.hist(train_clean['ApplicantIncome'], bins=20)
plt.title('Applicant Income Distribution')
plt.xlabel('Income')
plt.ylabel('Frequency')
plt.show()
```



3.5 Generation of Clean Dataset

The preprocessing stage generated two clean datasets:

- train_cleaned.csv
- test_cleaned.csv

Training Dataset:

- Records: 614
- Attributes: 13
- Missing Values After Cleaning: 0

Testing Dataset:

- Records: 367
- Attributes: 12
- Missing Values After Cleaning: 0

The cleaned datasets contain complete records and are suitable for exploratory data analysis and predictive modelling.

3.6 Data Preparation Results

The preprocessing process successfully eliminated all missing values from both datasets. Analysis of the cleaned training dataset revealed that 422 loan applications were approved while 192 applications were rejected.

Credit history was found to be a highly influential variable, with the majority of applicants (525) having a positive credit history.

The final datasets are complete, consistent, and ready for machine learning model development.

Dataset	Records	Attributes	Missing Values
Training	614	13	0
Testing	367	12	0

Table 3.1 Dataset Summary

Feature	Mean	Median	Min	Max
ApplicantIncome	5403.46	3812.50	150	81000
LoanAmount	145.75	128.00	9.0	700.0

Table 3.2 Numerical Feature Statistics

Key Findings from EDA

- Loan approvals (422) exceed loan rejections (192), indicating an imbalanced target variable.
- Applicant income shows substantial variation, ranging from RM 150 to RM 81,000.
- Loan amounts range from 9 to 700, suggesting the presence of high-value loans.
- Credit history appears strongly associated with loan approval outcomes.
- The cleaned dataset contains zero missing values and is suitable for predictive modelling

4.0 Model Development

4.1 Overview

This section describes the complete supervised learning pipeline applied to the loan approval prediction dataset. The goal is to train classification models that can accurately predict whether a loan application will be approved (Y) or rejected (N) based on applicant demographic and financial attributes. Six classification algorithms were implemented, evaluated, and compared using a consistent experimental setup to identify the most suitable model for this task.

4.2 Data Splitting Strategy

The cleaned dataset consists of 614 records with a binary target variable (Loan_Status: Y = Approved, N = Rejected). The dataset was split into a training set and a test set using a 80/20 ratio, resulting in 491 training samples and 123 test samples. Stratified splitting was applied to preserve the original class distribution in both splits (approximately 69% approved, 31% rejected). A fixed random seed (random_state = 42) was used throughout all experiments to ensure reproducibility.

Split	Samples	% of Total
Training Set	491	80%
Test Set	123	20%
Total	614	100%

Table 4.1: Dataset split summary

4.3 Feature Engineering and Preprocessing

Prior to model training, several preprocessing and feature engineering steps were applied to improve model performance and address the characteristics of the dataset.

4.3.1 Feature Engineering

Three engineered features were derived from existing attributes to capture meaningful financial relationships:

- TotalIncome: Sum of ApplicantIncome and CoapplicantIncome, representing the combined household income available for loan repayment.
- LogTotalIncome: Natural logarithm of TotalIncome, applied to reduce right-skewness in the income distribution.
- LogLoanAmount: Natural logarithm of LoanAmount, similarly applied to normalise the loan amount distribution.
- LoanIncomeRatio: Ratio of LoanAmount to TotalIncome, capturing the relative loan burden on the applicant.

4.3.2 Categorical Encoding

All categorical variables were converted to numerical form using Label Encoding. The variables encoded were: Gender, Married, Education, Self_Employed, and Property_Area. Label encoding

was chosen given the relatively small number of unique categories per variable and the compatibility with tree-based models.

4.3.3 Missing Value Imputation

After feature engineering, any remaining missing values in the feature matrix were imputed using the column median, a robust strategy that is not sensitive to outliers.

4.3.4 Feature Scaling

StandardScaler (zero mean, unit variance) was applied to the feature matrix for models sensitive to feature scale, specifically Logistic Regression, Support Vector Machine (SVM), and K-Nearest Neighbours (KNN). Tree-based models (Decision Tree, Random Forest, Gradient Boosting) were trained on the unscaled feature matrix as they are scale-invariant.

The final feature set used for model training comprised 11 features:

- Original features: Gender, Married, Dependents, Education, Self_Employed, Credit_History, Loan_Amount_Term, Property_Area
- Engineered features: LogTotalIncome, LogLoanAmount, LoanIncomeRatio

4.4 Classification Algorithms

Six supervised classification algorithms were selected to provide a comprehensive comparison across different model families, ranging from linear to non-linear and from simple to ensemble methods.

4.4.1 Logistic Regression

Logistic Regression serves as the interpretable linear baseline model. It models the log-odds of loan approval as a linear combination of the input features and produces well-calibrated probability estimates. Parameters: solver = lbfgs, max_iter = 1000, random_state = 42.

4.4.2 Decision Tree Classifier

A single Decision Tree was included to provide an interpretable non-linear baseline. The depth was restricted to max_depth = 5 to prevent overfitting while retaining the ability to capture non-linear decision boundaries. Parameters: max_depth = 5, random_state = 42.

4.4.3 Random Forest Classifier

Random Forest is a bagging ensemble of Decision Trees trained on bootstrap samples with random feature subsets. It reduces variance compared to a single tree and provides feature importance scores. Parameters: n_estimators = 100, random_state = 42.

4.4.4 Gradient Boosting Classifier

Gradient Boosting sequentially builds Decision Trees where each tree corrects the residual errors of its predecessor. It typically achieves strong predictive performance but may require careful tuning. Parameters: n_estimators = 100, learning_rate = 0.1 (default), random_state = 42.

4.4.5 Support Vector Machine (SVM)

SVM finds the maximum-margin hyperplane that separates the two classes in a high-dimensional feature space. The Radial Basis Function (RBF) kernel was used via the default settings to handle

non-linear relationships. Probability estimates were enabled (probability = True) for ROC-AUC computation. Parameters: kernel = rbf, C = 1.0, random_state = 42.

4.4.6 K-Nearest Neighbours (KNN)

KNN classifies each instance based on the majority class among its k nearest neighbours in feature space. The number of neighbours was set to k = 7 based on the dataset size. Parameters: n_neighbors = 7.

4.5 Model Validation Strategy

To obtain reliable estimates of generalisation performance, Stratified 5-Fold Cross-Validation was applied to the training set for all models. Stratification ensures that each fold maintains the same class proportion as the full training set, which is important given the class imbalance (approximately 69%/31%). The mean and standard deviation of accuracy across the five folds are reported alongside the held-out test set metrics.

Model Definition

```
models = {
    'Logistic Regression': LogisticRegression(max_iter=1000, random_state=42),
    'Decision Tree':       DecisionTreeClassifier(max_depth=5, random_state=42),
    'Random Forest':       RandomForestClassifier(n_estimators=100, random_state=42),
    'Gradient Boosting':   GradientBoostingClassifier(n_estimators=100, random_state=42),
    'SVM':                 SVC(probability=True, random_state=42),
    'KNN':                 KNeighborsClassifier(n_neighbors=7),
}

# Models that need scaled features
SCALED_MODELS = {'Logistic Regression', 'SVM', 'KNN'}
```

Training & Evaluation

```
    results[name] = {
        'model':    model,
        'cv_mean':  cv_scores.mean(),
        'cv_std':   cv_scores.std(),
        'test_acc': accuracy_score(y_test, y_pred),
        'f1':       f1_score(y_test, y_pred),
        'precision':precision_score(y_test, y_pred),
        'recall':   recall_score(y_test, y_pred),
        'roc_auc':  roc_auc_score(y_test, y_prob),
        'y_pred':   y_pred,
        'y_prob':   y_prob,
        'Xte':      Xte,
    }

    r = results[name]
    print(f"{name:<22}  {r['cv_mean']:.4f} ± {r['cv_std']:.4f}    "
          f"{r['test_acc']:.4f}  "
      f"{r['precision']:.4f}  "
      f"{r['recall']:.4f}  "
      f"{r['f1']:.4f}  "
      f"{r['roc_auc']:.4f}")
```

Train & Test Split

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
print(f"\nTraining set: {len(X_train)} samples")
print(f"Test set:     {len(X_test)} samples")

# Feature scaling for distance/margin-based models
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled  = scaler.transform(X_test)
```

```
Training set: 491 samples
Test set:     123 samples
```

4.6 Results Summary

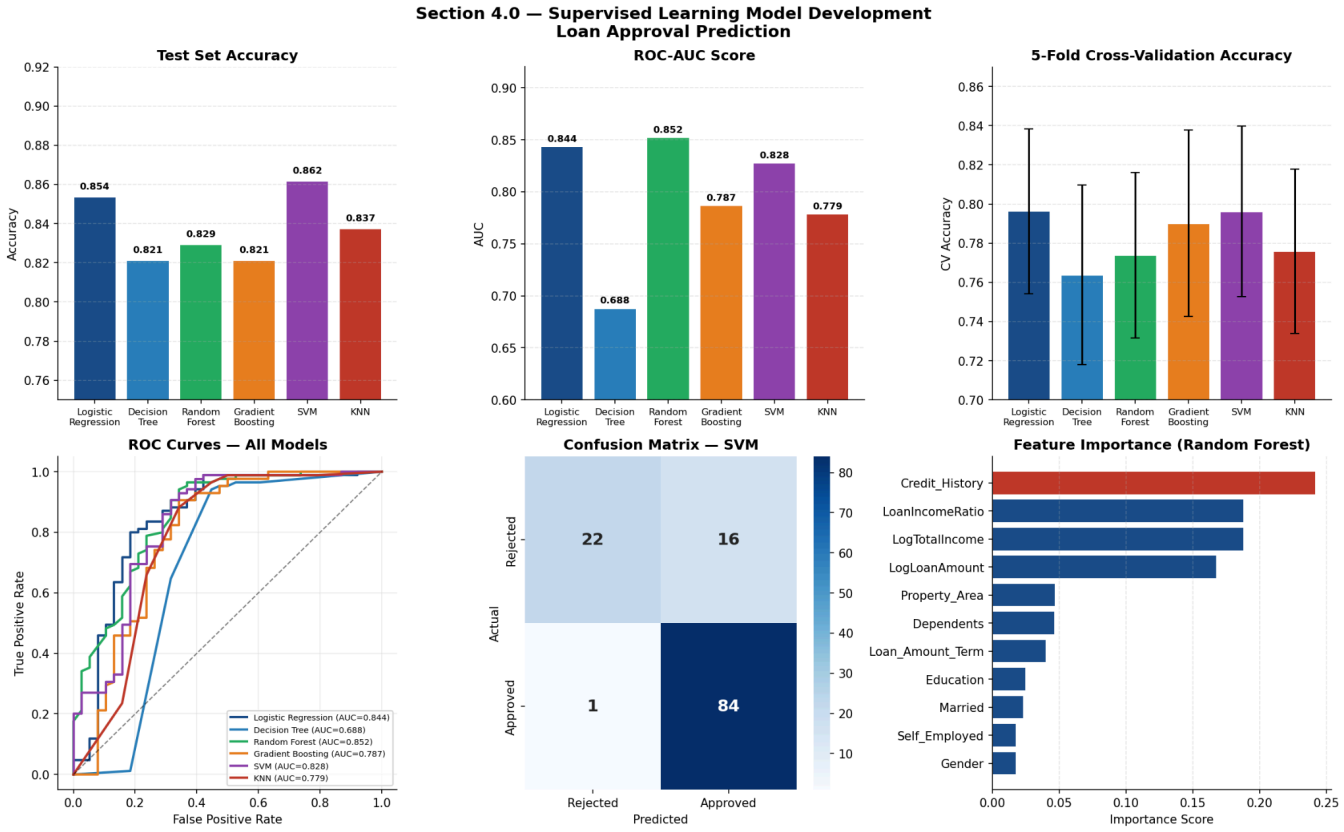


Figure 4.1: Model comparison across accuracy, ROC-AUC, cross-validation accuracy, ROC curves, confusion matrix (best model), and feature importance.

Model	CV Accuracy (mean±std)	Test Accuracy	Precision	Recall	F1	AUC
Logistic Regression	0.7963 ± 0.0421	0.8537	0.8317	0.9882	0.9032	0.8437
Decision Tree	0.7638 ± 0.0458	0.8211	0.8182	0.9529	0.8804	0.6881
Random Forest	0.7739 ± 0.0423	0.8293	0.8556	0.9059	0.8800	0.8525
Gradient Boosting	0.7902 ± 0.0475	0.8211	0.8462	0.9059	0.8750	0.7872
SVM	0.7962 ± 0.0436	0.8618	0.8400	0.9882	0.9081	0.8276
KNN	0.7759 ± 0.0419	0.8374	0.8283	0.9647	0.8913	0.7789

Table 4.2 : Performance comparison of all six classification models. CV Accuracy is expressed as mean ± standard deviation

5.0 Model Performance & Evaluation

5.1 Application of Evaluation Metrics

Several metrics were used which are accuracy, precision, recall, F1-Score and additionally Area Under the Receiver Operating Characteristic Curve (ROC-AUC) to evaluate the effectiveness of the developed classification models. These metrics were derived from the confusion matrix to assess the classification performance.

Table 5.1: Confusion Matrix of SVM

Actual Class	Predicted Rejected	Predicted Approved
Rejected	22	16
Approved	1	84

From the confusion matrix:
True Positive (TP) = 84
True Negative (TN) = 22
False Positive (FP) = 16
False Negative (FN) = 1

Accuracy
Accuracy measures the proportion of correctly classified instances among all instances in the test dataset. It provides an overall indication of model correctness.

$$Accuracy = \frac{TP+TN+FP+FN}{TP+TN}$$
= (84 + 22) / 123
= 86.18%

Precision
Precision measures the proportion of positive predictions that are actually correct. In the context of loan approval prediction, precision indicates how many applicants predicted as approved were genuinely eligible for approval.

$$Precision = \frac{TP}{TP+FP}$$
= 84 / (84 + 16)
= 84 / 100
= 0.84% = 84.00 %

Recall
Recall measures the proportion of actual positive instances that are correctly identified by the model. A high recall indicates that most eligible applicants are successfully recognised by the model.

$Recall = \frac{TP}{TP+FN}$
= 84 / (84 + 1)
= 84 / 85
= 0.9882 = 98.82%

F1-Score
The F1-Score is the harmonic mean of Precision and Recall. It provides a balanced assessment of model performance especially when class distributions are unequal.

$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$
= 2 × [(0.8400 × 0.9882) / (0.8400 + 0.9882)]
= 0.9081 = 90.81%

ROC-AUC
The Area Under the Receiver Operating Characteristic Curve (ROC-AUC) evaluates the model's ability to distinguish between approved and rejected loan applications across different classification thresholds. Higher ROC-AUC values indicate better discrimination capability.

5.2 Analysis of Model Performance

Among all evaluated models, Support Vector Machine (SVM) achieved the highest Test Accuracy of 86.18% and the highest F1-Score of 90.81%, indicating superior overall classification performance. Logistic Regression closely followed with a Test Accuracy of 85.37% and an F1-Score of 90.32%, demonstrating that a simpler linear model can still perform competitively on the loan approval dataset.

Random Forest achieved the highest precision (85.56%) and ROC-AUC (0.8525). The high precision indicates that Random Forest generated the fewest incorrect loan approvals among all models, while the highest ROC-AUC demonstrates superior discrimination capability between approved and rejected applicants.

Both SVM and Logistic Regression achieved the highest Recall value of 98.82%, meaning they successfully identified almost all eligible applicants. High recall is desirable because it minimises the number of qualified applicants who are incorrectly rejected.

Decision Tree recorded the lowest ROC-AUC score (0.6881), suggesting weaker classification capability and lower ability to distinguish between approved and rejected applicants compared to the ensemble and kernel-based approaches.

The cross-validation accuracies ranged from approximately 76% to 80%, with standard deviations between ±4% and ±5%. The relatively small variation indicates stable model behaviour across different data partitions and suggests good generalisation performance.

5.3 Reporting of Model Performance Results

The confusion matrix analysis demonstrates that the SVM model correctly classified 106 out of 123 loan applications. The model produced 84 True Positives and 22 True Negatives while generating only 1 False Negative and 16 False Positives. The high Recall value of 98.82% indicates that nearly all qualified applicants were successfully identified by the model. This is advantageous because it reduces the likelihood of rejecting applicants who genuinely meet the loan approval criteria.

However, the model produced 16 False Positive predictions which indicate that some applicants who should have been rejected were classified as approved. In a real-world banking environment, such errors may increase financial risk because loans could be granted to unsuitable applicants. False Positives generally carry a higher cost than False Negatives because they may result in loan defaults and financial losses. Therefore, although SVM achieved the highest overall performance, lending institutions should carefully consider their risk tolerance when deploying the model.

The dataset shows moderate class imbalance which consist of approximately 69% approved applications and 31% rejected applications. To address this issue, evaluation was not based solely on accuracy. Additional metrics such as Precision, Recall, F1-Score, and ROC-AUC were considered to provide a more balanced assessment of model effectiveness.

Overall, the evaluation results indicate that Support Vector Machine (SVM) is the most suitable model for loan approval prediction. The model achieved the highest Test Accuracy (86.18%) and F1-Score (90.81%) and maintained strong Recall and stable cross-validation performance. These results suggest that SVM provides the best balance between predictive accuracy, applicant identification capability, and model reliability for the loan approval prediction task.

6.0 References

Data Mining Project IPYNB files [Google Drive folder]. Google Drive. Retrieved June 8, 2026, from https://drive.google.com/drive/folders/1fgnUbG3UPiQUk2uX8mH13E7kLDnhz9OJ

altruistdelhite04. (2020). *Loan prediction problem dataset* [Data set]. Kaggle. https://www.kaggle.com/datasets/altruistdelhite04/loan-prediction-problem-dataset

Végh, L., Czakóová, K., & Takáč, O. (2023). Comparing machine learning classification models on a loan approval prediction dataset. *International Journal of Advanced Natural Sciences and Engineering Researches, 7*(9), 98–103. https://doi.org/10.59287/ijanser.1516

IBM. (2024). *What is logistic regression?* https://www.ibm.com/topics/logistic-regression

IBM. (2024). *What is a support vector machine?* https://www.ibm.com/topics/support-vector-machine

Géron, A. (2023). *Supervised and unsupervised learning*. O'Reilly. https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/