ISOM 835 – Predictive Analytics and Machine Learning

Project Title: Bank Loan Default Prediction

Final Report

Thong Tao

Instructor: Hasan Arslan

1. Dataset Selection Rationale

For this project, I selected the "Bank Loan" dataset from Kaggle, which contains information on customers' demographic, financial, and credit-related attributes. The dataset is typically used to predict whether a customer will repay a loan or default. It includes variables such as income, employment status, credit score, loan amount, and more — all of which are essential factors in assessing lending risk. This dataset is well-suited for classification models and reflects a common real-world use case in the financial industry.

I chose this dataset because it offers a practical opportunity to apply predictive analytics to a high-impact business problem. Loan default prediction is crucial for banks and financial institutions to minimize risk and make informed decisions about approving loans. With the right model, organizations can not only reduce financial losses but also enhance customer targeting and operational efficiency. This project will allow me to explore different modeling techniques, evaluate performance metrics, and generate actionable insights that are directly applicable in the finance sector.

2. Introduction & Dataset Description

The goal of this project is to predict the likelihood of a loan default using customer attributes from a bank loan dataset. This dataset, sourced from Kaggle, contains 700 entries and 9 features, including age, education level, employment status, address duration, income, and various debt indicators. The target variable is default, indicating if a customer has defaulted (1) or not (0).

3. Exploratory Data Analysis (EDA)

Exploratory Data Analysis was conducted to understand the structure, completeness, and distribution of the dataset. The key points and findings are detailed below.

a. Dataset Structure and Data Types

The dataset consists of 700 rows and 9 columns, initially loaded with all columns as object types, including numeric variables. This suggests improper data type inference during import, likely due to inconsistent formatting or embedded non-numeric characters. The target variable, default, was also read as an object type, requiring manual conversion for modeling purposes.

b. Descriptive Statistics

Summary statistics were generated for key numeric fields such as income, debinc (debt-to-income ratio), and creddebt (credit card debt). These summaries showed wide ranges in values. For instance, the income column displayed high variability with some extremely large values. This indicates the presence of potential outliers and skewed distributions, which may affect the performance of certain models.

c. Missing Value Analysis

A heatmap of missing values was plotted using seaborn to visualize patterns of incompleteness in the dataset. Several rows exhibited missing data across multiple columns. Features such as income, creddebt, and othdebt showed frequent null values. These missing entries required imputation to prevent errors during model training.

d. Outlier Detection via Boxplots

Boxplots were used to inspect numerical columns for outliers. The income variable, in particular, showed extreme values above the typical range, indicating the presence of upper outliers. Similarly, creddebt exhibited a right-skewed distribution with visible outliers. These characteristics suggest the need for scaling and possibly log transformation depending on the modeling approach.

e. Class Imbalance in Target Variable

The target variable default was found to be imbalanced. The distribution was as follows:

- 0: 515 instances (non-default)
- 1: 183 instances (default)
- '0' (as a string): 1 instance
- :0 (invalid entry): 1 instance

The non-numeric and malformed entries were cleaned and converted to integers. After correction, the class imbalance remains but becomes cleaner, which is important for ensuring valid performance metrics in classification models.

f. Type Conversion

All numerical columns were explicitly converted from object to numeric types using pd.to_numeric(). This step was necessary to enable preprocessing techniques such as scaling and to prepare the features for use in machine learning models.

Visualizations such as boxplots, missing value heatmaps, and target class distribution are included in the Appendix to support these observations.

4. Preprocessing

The preprocessing phase involved several steps to ensure the dataset was clean and suitable for modeling. First, the target variable default contained inconsistent labels such as '0' (string) and :0, which were cleaned and converted to numeric values to ensure consistent binary classification. Missing values were then addressed using SimpleImputer from scikit-learn, with the strategy set to mean, replacing missing entries in numerical columns with the mean of each respective feature. This imputation preserved the overall distribution while enabling compatibility with machine learning algorithms that do not accept null values. Following imputation, all numerical features were standardized using StandardScaler, which transforms the data to have a mean of 0 and a standard deviation of 1. This step is crucial for algorithms sensitive to feature scaling, such as logistic regression. Finally, the dataset was split into training and testing sets using train_test_split, with the stratify=y parameter to maintain the original class distribution between defaulters and non-defaulters in both subsets.

5. Business Questions

a. **Which customer traits are most predictive of default?**

This question aims to identify the key financial and demographic indicators that influence loan default. Understanding these traits can help financial institutions design more effective risk assessment models and develop fairer lending policies by focusing on variables that have a statistically significant relationship with default behavior.

b. **How accurately can default risk be classified?**

 Evaluating the predictive power of a classification model is essential for determining its usefulness in real-world applications. This question is relevant for organizations that want to automate or enhance their loan approval processes while minimizing the risk of approving loans to high-risk individuals.

c. **Is a complex model like Random Forest better than Logistic Regression for this task?**

 Comparing a simple, interpretable model with a more complex, higher capacity one provides insight into the trade-offs between accuracy and transparency. Logistic Regression offers clear explanations for decisions, which is valuable in regulated environments, while Random Forest may yield higher performance by capturing non-linear interactions. This question helps determine which approach is more appropriate given the needs of the stakeholders—whether they prioritize interpretability, performance, or a balance of both.

6. Modeling

Two predictive models were implemented to evaluate loan default risk: Logistic Regression and Random Forest Classifier. Both models were trained on the same clean, imputed, and standardized dataset to ensure a fair comparison. Logistic Regression was chosen as a baseline model due to its simplicity and interpretability. It is computationally efficient and allows for direct insight into the impact of individual features on the target variable. The model's performance was assessed using classification metrics such as precision, recall, F1-score, and ROC AUC, providing a comprehensive view of its strengths and limitations.

In contrast, the Random Forest Classifier was employed as a more complex ensemble method capable of capturing non-linear relationships and interactions between features. It demonstrated stronger performance, particularly in recall and ROC AUC, which are critical when the goal is to correctly identify potential defaulters. By improving recall, Random Forest reduces the risk of misclassifying high-risk borrowers as safe, which is essential in financial decision-making. Overall, both models were evaluated using the same metrics and testing data, allowing for an objective comparison that highlights the trade-offs between model accuracy and interpretability.

7. Insights

The analysis revealed that features such as employment duration, income level, and credit-to-debt ratios play a significant role in predicting loan default. These financial indicators are closely tied to a borrower’s ability to manage debt and make timely payments, making them critical variables in risk assessment models. When comparing model performance, the Random Forest Classifier demonstrated higher accuracy and recall than Logistic Regression, making it a more powerful tool for identifying potential defaulters. However, Logistic Regression remains a valuable model due to its simplicity and interpretability, which is often preferred in regulated financial environments where transparency in decision-making is required. Despite the models' effectiveness, class imbalance in the target variable continues to pose a challenge. The relatively small number of default cases can skew performance metrics, particularly accuracy, and may cause models to underperform in detecting defaults. To address this, future iterations could incorporate resampling techniques such as SMOTE or use cost-sensitive learning approaches to improve model balance and reliability.

8. Ethics & Interpretability

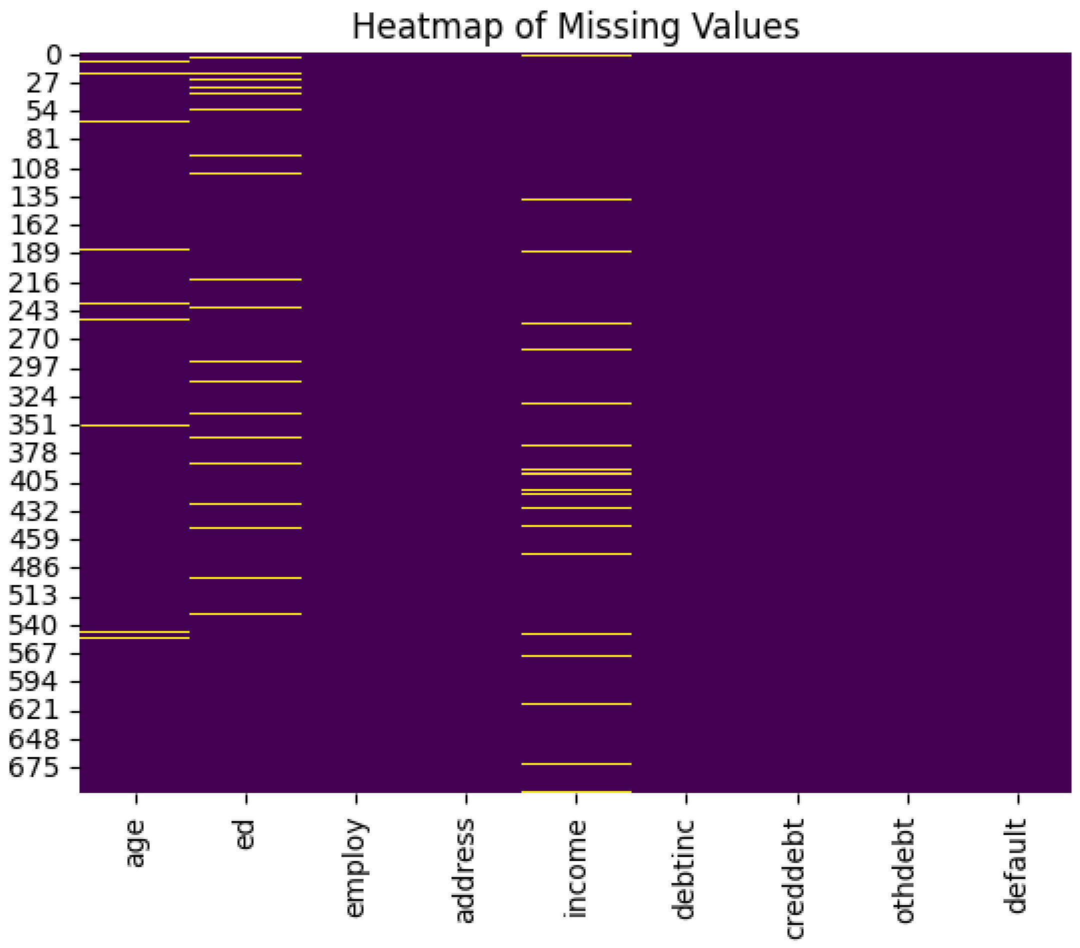
The analysis considered ethical implications and model interpretability throughout the predictive modeling process. Particular attention was paid to the risk of bias arising from potentially sensitive features such as income or employment history. To promote transparency, logistic regression was used alongside Random Forest, as it offers interpretable coefficients that can help explain how individual features contribute to predictions. This transparency is essential for stakeholder trust, especially in high-impact domains like credit decisions. While Random Forest achieved better performance, it is more complex and less explainable. Therefore, it is recommended that future model deployment include routine audits for fairness and interpretability, ensuring responsible use of predictive analytics.

APPENDIX

A. Exploratory Data Analysis (EDA)

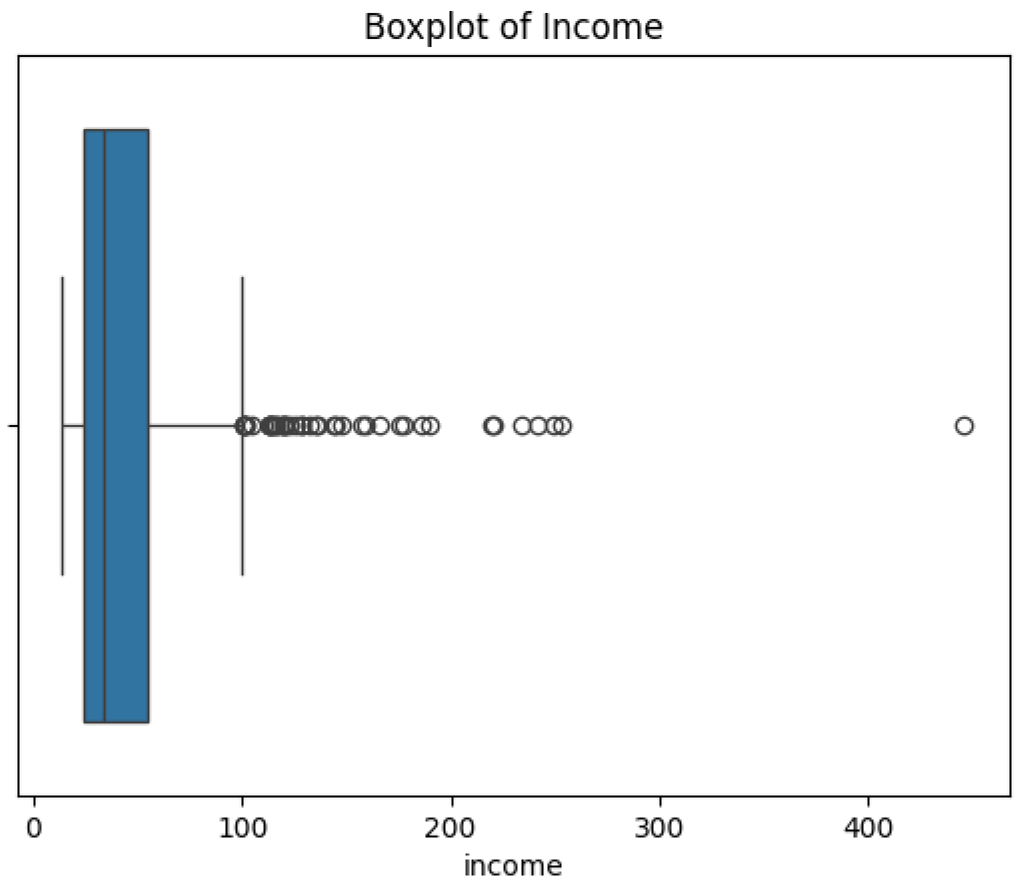
1. Missing Value Heatmap

```
sns.heatmap(df.isnull(), cbar=False, cmap='viridis')
plt.title('Heatmap of Missing Values')
plt.show()
```



2. Boxplot for Outliers (Income)

```
sns.boxplot(x=df['income'])
plt.title('Boxplot of Income')
plt.show()
```



B. Data Preprocessing

3. Train-Test Split

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

X = df.drop('default', axis=1)
y = df['default']

# Scale
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Split with stratify
```

```
X_train, X_test, y_train, y_test = train_test_split(
X_scaled, y, test_size=0.2, stratify=y, random_state=42
)
```

4. Missing Value Imputation

```
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler

# 1. Impute missing values
imputer = SimpleImputer(strategy='mean')
X_train_imputed = imputer.fit_transform(X_train)
X_test_imputed = imputer.transform(X_test)
```

5. Feature Scaling

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train_imputed)
X_test_scaled = scaler.transform(X_test_imputed)
```

C. Predictive Modeling

6. Logistic Regression Model

```
# 3. Train logistic regression
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, roc_auc_score

lr_model = LogisticRegression()
lr_model.fit(X_train_scaled, y_train)
lr_preds = lr_model.predict(X_test_scaled)

# 4. Print results
print("Logistic Regression Results")
print(classification_report(y_test, lr_preds))
print("ROC AUC Score:", roc_auc_score(y_test, lr_preds))
```

7. Random Forest Model

```
# Random Forest
```

```
rf_model = RandomForestClassifier(random_state=42)
rf_model.fit(X_train_scaled, y_train)
rf_preds = rf_model.predict(X_test_scaled)

print("\n ◇  Random Forest Results")
print(classification_report(y_test, rf_preds))
print("ROC AUC:", roc_auc_score(y_test, rf_preds))
```