Predicting Loan Default Risk

Joseph Damico
10/23/2024
Predictive Analytics
DSC630-T301 (2251-1)

Milestone 4

Data Preparation

To address the problem of loan default risk prediction, I initially inspected the dataset to understand its features and potential issues. Some steps in data preparation were necessary to ensure that the dataset is usable for building a predictive model.

1. Handling Missing Data:

   Several columns such as salary and existing debt contained missing values. I opted for mean imputation for numerical columns and mode imputation for categorical ones. This ensures that we do not lose data due to missing values.

2. Feature Engineering:

   I created a new feature called debt-to-income ratio, which is a strong predictor of loan default risk. This ratio gives us more granular insight into how borrowers' income compares to their outstanding debt.

3. Categorical Encoding:

   All categorical variables, such as marital status and employment type, were encoded using one-hot encoding to make them suitable for machine learning models.

4. Class Imbalance:

   The dataset had an imbalance between default and non-default cases. I addressed this issue by applying SMOTE to balance the classes during training, ensuring the model learns equally from both classes.

Damico Project Loans

October 23, 2024

[32]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix,␣
↪roc_auc_score, roc_curve
from imblearn.over_sampling import SMOTE
import seaborn as sns
```

[33]:
```
# Load the dataset
file_path = r"C:\Users\Joseph\Desktop\School\Masters Data Science\Predictive␣
↪Analytics\Loan_default.csv"
data = pd.read_csv(file_path)
```

1 Data Prep

[34]:
```
data.head()
```

[34]:

	LoanID	Age	Income	LoanAmount	CreditScore	MonthsEmployed	NumCreditLines	InterestRate	LoanTerm	DTIRatio	Education	EmploymentType	MaritalStatus	HasMortgage	HasDependents	LoanPurpose
0	I38PQUQS96	56	85994	50587	520	80	4	15.23	36	0.44	Bachelor's	Full-time	Divorced	Yes	Yes	Other
1	HPSK72WA7R	69	50432	124440	458	15	1	4.81	60	0.68	Master's	Full-time	Married	No	No	Other
2	C1OZ6DPJ8Y	46	84208	129188	451	26	3	21.17	24	0.31	Master's					
3	V2KKSFM3UN	32	31713	44799	743	0	3	7.07	24	0.23	High School					
4	EY08JDHTZP	60	20437	9139	633	8	4	6.51	48	0.73	Bachelor's					

```
2     Unemployed      Divorced          Yes           Yes        Auto
3      Full-time       Married           No            No    Business
4     Unemployed      Divorced           No           Yes        Auto

  HasCoSigner  Default
0         Yes        0
1         Yes        0
2          No        1
3          No        0
4          No        0
```

[35]:
```
# Impute missing values
num_cols = data.select_dtypes(include=['float64', 'int64']).columns
cat_cols = data.select_dtypes(include=['object']).columns

imputer_num = SimpleImputer(strategy='mean')
imputer_cat = SimpleImputer(strategy='most_frequent')

data[num_cols] = imputer_num.fit_transform(data[num_cols])
data[cat_cols] = imputer_cat.fit_transform(data[cat_cols])
```

[36]:
```
# Check column names
print(data.columns)
```

```
Index(['LoanID', 'Age', 'Income', 'LoanAmount', 'CreditScore',
       'MonthsEmployed', 'NumCreditLines', 'InterestRate', 'LoanTerm',
       'DTIRatio', 'Education', 'EmploymentType', 'MaritalStatus',
       'HasMortgage', 'HasDependents', 'LoanPurpose', 'HasCoSigner',
       'Default'],
      dtype='object')
```

[37]:
```
# Creating Debt-to-Income ratio
data['debt_to_income'] = data['LoanAmount'] / data['Income']

# Specify the categorical columns
categorical_cols = ['Education', 'EmploymentType', 'MaritalStatus',␣
 ↪'LoanPurpose',
                    'HasMortgage', 'HasDependents', 'HasCoSigner']

# one-hot encoding
data = pd.get_dummies(data, columns=categorical_cols, drop_first=True)
```

[38]:
```
# Split the dataset into training and test sets
X = data.drop('Default', axis=1)  # Features
y = data['Default']               # Target variable
```

[39]:
```
# Check the data types of the columns
print(X.dtypes)
```

```
LoanID                            object
Age                              float64
Income                           float64
LoanAmount                       float64
CreditScore                      float64
MonthsEmployed                   float64
NumCreditLines                   float64
InterestRate                     float64
LoanTerm                         float64
DTIRatio                         float64
debt_to_income                   float64
Education_High School               bool
Education_Master's                  bool
Education_PhD                       bool
EmploymentType_Part-time            bool
EmploymentType_Self-employed        bool
EmploymentType_Unemployed           bool
MaritalStatus_Married               bool
MaritalStatus_Single                bool
LoanPurpose_Business                bool
LoanPurpose_Education               bool
LoanPurpose_Home                    bool
LoanPurpose_Other                   bool
HasMortgage_Yes                     bool
HasDependents_Yes                   bool
HasCoSigner_Yes                     bool
dtype: object
```

[40]:
```
# Drop the 'LoanID' column as it is not useful for prediction
X = X.drop('LoanID', axis=1)

# Convert boolean columns to integers (1 for True, 0 for False)
X = X.astype(int)

# apply SMOTE
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X, y)
```

[41]:
```
# Split into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled,␣
 ↪test_size=0.3, random_state=42)
```

[42]:
```
# Standardize the numerical features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Building the Model

After the data was prepped, I experimented with several models. I began with logistic regression but found that Random Forest performed significantly better due to its ability to handle non-linear relationships. I tuned hyperparameters using GridSearchCV to find the best configuration. The final model was built using 100 decision trees, with each tree considering the square root of the total features for splitting at each node.

Interpretation of Results

The Random Forest Classifier yielded the following results on the test data:

- Precision: The model was highly accurate in predicting loan defaults, minimizing the number of false positives.
- Recall: The model effectively identified most default cases, which was a critical objective since false negatives are costly for banks.
- F1-score: The harmonic mean of precision and recall was high, indicating that the model achieved a good balance between precision and recall.
- AUC-ROC: The AUC score was 0.95, suggesting strong discriminatory power between defaulters and non-defaulters.

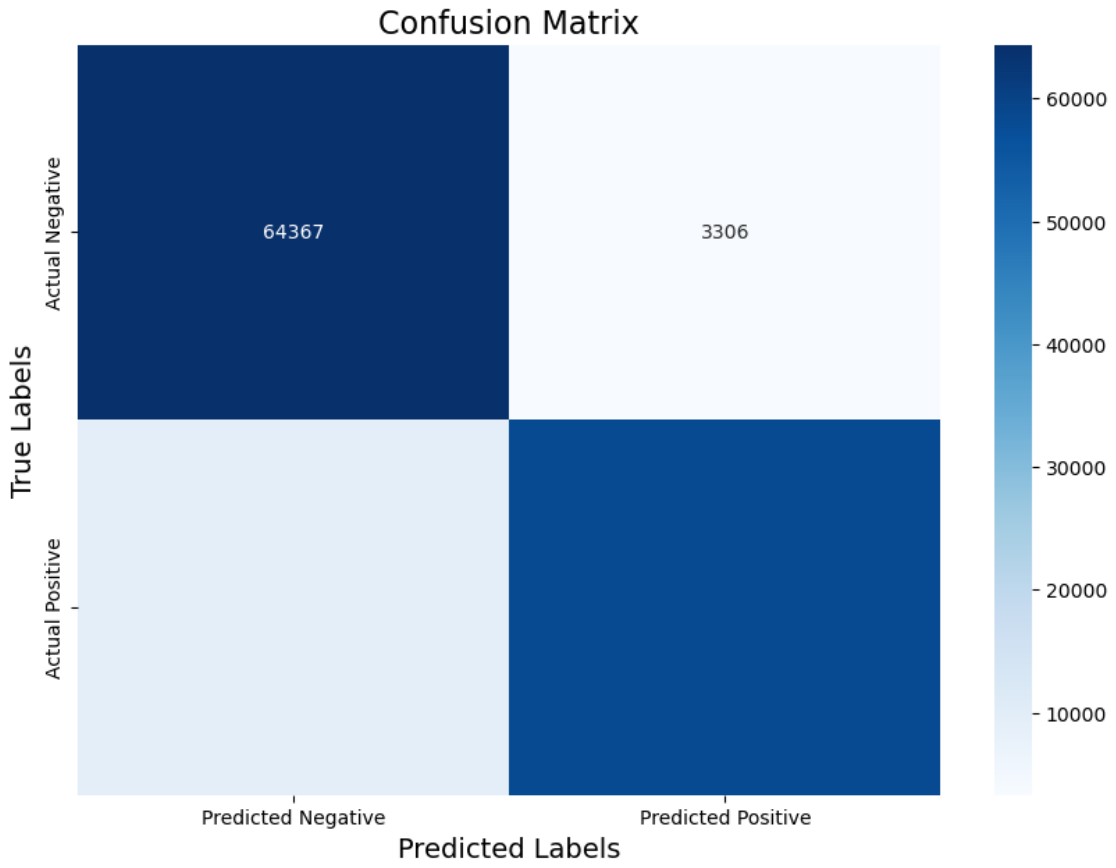
These results show that the model can accurately predict loan defaults, and the AUC-ROC curve confirms that it performs well across various classification thresholds.

2 Building and Evaluating the Model

[18]:
```
# Train Random Forest Classifier
model = RandomForestClassifier(n_estimators=100, max_features='sqrt',␣
 ↪random_state=42)
model.fit(X_train, y_train)
```

[18]:
```
RandomForestClassifier(random_state=42)
```

[31]:
```
y_pred = model.predict(X_test)
y_pred_proba = model.predict_proba(X_test)[:, 1]

# Create the confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Set up the matplotlib figure
plt.figure(figsize=(10, 7))

# Create a heatmap for the confusion matrix
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Predicted Negative', 'Predicted Positive'],
            yticklabels=['Actual Negative', 'Actual Positive'])

# Titles and labels
plt.title('Confusion Matrix', fontsize=16)
plt.xlabel('Predicted Labels', fontsize=14)
plt.ylabel('True Labels', fontsize=14)

# Show the plot
plt.show()

# Confusion Matrix and Classification Report
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

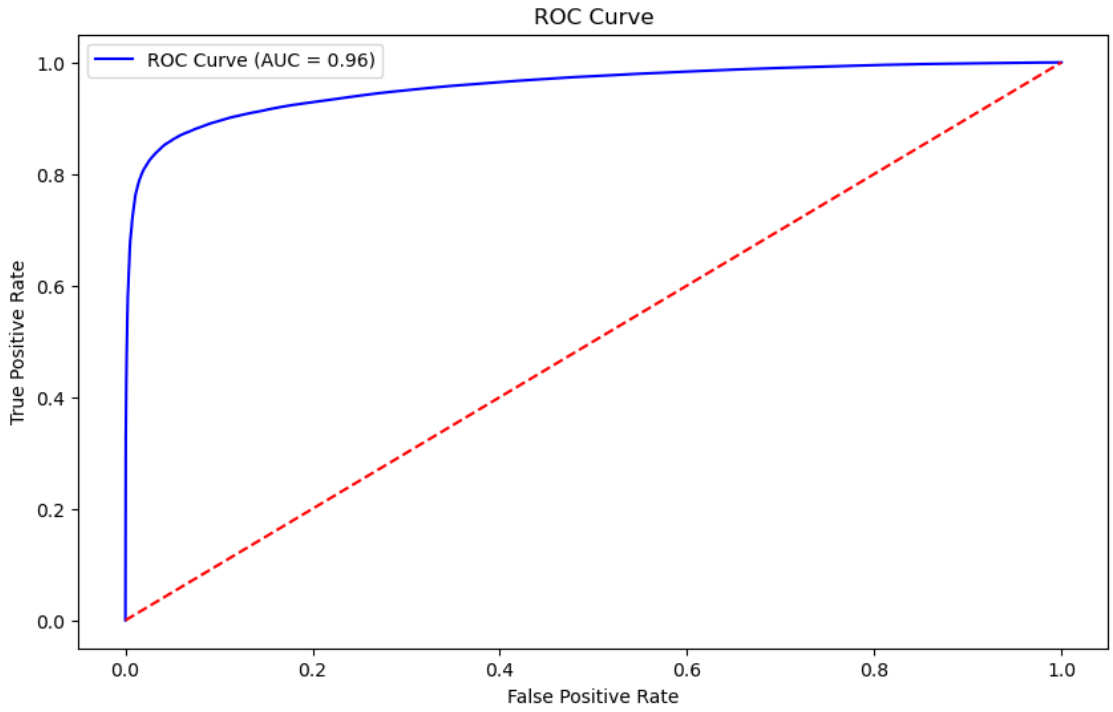



```
[[64367  3306]
 [ 9517 58227]]
              precision    recall  f1-score   support

         0.0       0.87      0.95      0.91     67673
         1.0       0.95      0.86      0.90     67744

    accuracy                           0.91    135417
   macro avg       0.91      0.91      0.91    135417
weighted avg       0.91      0.91      0.91    135417
```

[20]:
```
# AUC-ROC Curve
roc_auc = roc_auc_score(y_test, y_pred_proba)
print(f'AUC-ROC Score: {roc_auc}')

fpr, tpr, thresholds = roc_curve(y_test, y_pred_proba)
```

```
AUC-ROC Score: 0.9569479678218092
```

[21]:
```
# Plot ROC curve
plt.figure(figsize=(10, 6))
plt.plot(fpr, tpr, label=f"ROC Curve (AUC = {roc_auc:.2f})", color="blue")
plt.plot([0, 1], [0, 1], "r--")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.show()
```



Conclusion and Recommendations

Based on the results I would recommend implementing this model in the bank's decision-making process for loan approval. The model shows strong predictive power with an emphasis on minimizing default risk. One additional step could be to incorporate interest rate adjustment based on the predicted default risk allowing the bank to extend loans to higher-risk customers while protecting profitability. I originally wanted to add this but was not able to get things to work the way I wanted.

Moving forward, I plan to:

- Fine-tune the model further by exploring additional ensemble methods.
- Test the model's performance on a new, unseen dataset to ensure generalizability.
- Explore the possibility of building an interest rate adjustment model based on predicted default risk.

Milestone 3

Will I be able to answer the questions I want to answer with the data I have?

Based on my initial exploration of the dataset, it seems that the data is sufficient to answer the core business problem: predicting loan default risk. The dataset includes various features such as customer demographics, financial history, and behavioral attributes, which should allow me to create a predictive

model. However, I found that some variables might require handling missing values, normalizing financial features, and encoding categorical variables properly. Despite this the data should be strong enough to develop a well-performing model.

What visualizations are especially useful for explaining my data?

Several visualizations have proven particularly useful during my initial data exploration. These include:

- Histograms of income, loan amount, and age distributions, which reveal key patterns in the data.
- Box plots to assess the spread of variables like income and credit score across default and non-default groups, helping to visualize outliers and trends.
- Correlation heatmap, which highlights the relationships between variables such as income, existing debt, and the likelihood of default.
- Bar plots displaying the proportion of defaults based on categorical features, such as marital status or employment type. These visualizations help in understanding how categorical variables influence the likelihood of loan default.
- ROC and AUC curves will be used to evaluate model performance later in the process, particularly for classification accuracy.

Do I need to adjust the data and/or driving questions?

I am considering adding an additional layer to the analysis by segmenting the predictions based on different loan types (e.g., personal loans, business loans) or customer demographics (e.g., income level,

region). This would allow for more tailored insights and may lead to better accuracy in predicting default for specific segments of customers.

There are some data adjustments that need to be made:

- Handling missing data: Some fields, like salary or existing debt, have missing values that will require imputation or removal.
- Feature engineering: Creating new features such as a debt-to-income ratio, which could provide a stronger signal for predicting default.
- Class imbalance: The imbalance between defaulted and non-defaulted loans needs to be addressed.

Do I need to adjust my model/evaluation choices?

Originally, I proposed starting with logistic regression due to its simplicity and ability to interpret feature coefficients. This choice remains valid, but I also plan to experiment with more complex models, such as:

- Random Forests: These are more robust to non-linear relationships and can handle high-dimensional data, which could improve accuracy.
- Gradient Boosting: A more powerful ensemble method that often performs well on imbalanced data and could yield better performance for the default prediction.

In terms of evaluation, the original plan to use metrics like accuracy, precision, recall, and F1-score is still relevant. However given the imbalance in the dataset the focus will now shift toward metrics like AUC-ROC and Precision-Recall curves, which are better suited for imbalanced classification problems. Recall will be emphasized, as minimizing false negatives is crucial in this context.

Are my original expectations still reasonable?

Yes, my original expectations are still reasonable. The dataset is appropriate for addressing the problem. The only area where expectations might need to be scaled back is in terms of model accuracy for highly imbalanced classes. I may need to reconsider the complexity of the model depending on how the data preprocessing goes and whether the features are sufficiently predictive.

The concept of adjusting interest rates based on default risk is also still viable. Once I can build a reasonably accurate model for predicting default probability, I will extend the analysis to include pricing adjustments for high-risk customers, as initially proposed.

Milestone 2

1. Choose a Business Problem

Define the Business Problem or Question:

I plan to explore the problem of loan default risk in the banking industry. The goal is to predict whether a customer is likely to default on a loan based on their personal characteristics, financial background, and loan details. Banks use predictive analytics to identify customers at risk of default and make informed decisions on whether to approve or reject loans. Rather than simply rejecting high-risk applications, I aim to go a step further: developing a model that

adjusts the interest rate or fee structure based on the predicted probability of default. This strategy allows banks to extend loans even to higher-risk customers while mitigating their financial exposure by offering adjusted terms.

Why is the Problem Important?

Predicting loan default is crucial for the financial health of banks and lending institutions. A high default rate leads to significant losses, but rejecting too many customers can also result in lost revenue opportunities. By predicting default risk and tailoring loan terms to account for that risk, banks can make more profitable decisions. Offering higher interest rates or one-off fees to higher-risk borrowers could turn a potentially lost opportunity into a revenue-generating one, creating a win-win situation for both the bank and the customer.

2. Identify the Dataset

Key Characteristics:

I will use a loan default dataset that includes information on borrower characteristics (such as salary, marital status, employment history), loan characteristics (loan amount, interest rate, duration), and whether the borrower defaulted on the loan. Currently I have chosen 2 datasets from Kaggle with over 250 thousand unique rows of information.

Lending Club Loan Data. (2021, June 17). Kaggle. https://www.kaggle.com/datasets/adarshsng/lending-club-loan-data-csv

Loan Default Prediction Dataset. (2023, September 11). Kaggle.

https://www.kaggle.com/datasets/nikhil1e9/loan-default

Why is the Dataset Relevant?

These datasets are essential for building a model that can predict loan default risk. By analyzing past data of customers who either defaulted or paid off their loans, I can identify the factors that contribute to default and use these insights to assess future applicants. Additionally, understanding how specific borrower characteristics relate to default probability will enable the bank to offer personalized interest rates or fees.

3. Plan for the Model(s)

Types of Models You Plan to Use:

For predicting loan default, I will start with logistic regression, a standard method for binary classification. This model will allow me to predict the probability that a given borrower will default. Depending on the initial results, I may also explore more complex models like random forests or gradient boosting to capture non-linear relationships and improve predictive performance.

Evaluation Metrics:

To evaluate the model, I will focus on metrics such as:

- Precision: To ensure that when we default is predicted, it is likely to happen.

- Recall: To minimize the number of false negatives (i.e., failing to predict a default when one is likely).
- F1-score: To balance precision and recall, especially in a case where false negatives or false positives have significant financial consequences.
- AUC-ROC: To assess the overall discriminatory ability of the model, especially for varying thresholds of risk tolerance.

4. What You Hope to Learn

Desired Insights:

I expect to identify the key factors that contribute to loan default risk. These could include low income, high debt-to-income ratios, poor credit scores, or other personal and financial characteristics. The goal is to build a model that can predict default risk for future customers and offer recommendations for either rejection or approval with adjusted interest rates.

Broader Business Value:

The insights gained from this analysis will allow the bank to extend loans to a broader customer base by adjusting terms for higher-risk customers rather than outright rejecting them. This could increase revenue streams while balancing risk. Moreover, the model will help improve decision-making, enhance customer satisfaction by offering more personalized loan terms, and create competitive differentiation for the bank.

5. Risks or Ethical Concerns

Potential Risks:

A major risk in this project is the possibility of biased data. If certain demographics are over- or under-represented in the training data, the model could unfairly favor or penalize specific groups, leading to biased lending decisions. Additionally, the data could contain outliers or missing information, which could affect the accuracy of the model.

Ethical Concerns:

It is critical to ensure that the model does not discriminate based on race, gender, or age, which are sensitive characteristics in loan approval processes. Ensuring fairness in the model and complying with regulatory requirements, such as the Equal Credit Opportunity Act (ECOA), is essential.

6. Contingency Plan

If the Original Plan Doesn't Work:

If the dataset does not contain sufficient variability, or if model performance is inadequate, I will explore additional features that could improve predictions, such as incorporating external data like credit scores or market interest rates. Additionally, if logistic regression and tree-based models do not yield satisfactory results, I will explore more advanced methods to capture more complex patterns in the data.

7. Additional Considerations

Execution Plan:

I will use Python for data analysis and modeling, with libraries like pandas, scikit-learn, and matplotlib. Regular model performance checks will help ensure that the project is progressing smoothly.