SBA : “Should This Loan be Approved or Denied?”

Background and Motivation

The U.S. SBA was founded in 1953 on the principle of promoting and assisting small enterprises in the U.S. credit market (SBA Overview and History, US Small Business Administration (2015)). Small businesses have been a primary source of job creation in the United States; therefore, fostering small business formation and growth has social benefits by creating job opportunities and reducing unemployment. One way SBA assists these small business enterprises is through a loan guarantee program which is designed to encourage banks to grant loans to small businesses. SBA acts much like an insurance provider to reduce the risk for a bank by taking on some of the risk through guaranteeing a portion of the loan. In the case that a loan goes into default, SBA then covers the amount they guaranteed. There have been many success stories of start-ups receiving SBA loan guarantees such as FedEx and Apple Computer. However, there have also been stories of small businesses and/or start-ups that have defaulted on their SBA-guaranteed loans. The rate of default on these loans has been a source of controversy for decades. Conservative economists believe that credit markets perform efficiently without government participation. Supporters of SBA guaranteed loans argue that the social benefits of job creation by those small businesses receiving government guaranteed loans far outweigh the costs incurred from defaulted loans. Since SBA loans only guarantee a portion of the entire loan balance, banks will incur some losses if a small business defaults on its SBA-guaranteed loan. Therefore, banks are still faced with a difficult choice as to whether they should grant such a loan because of the high risk of default. One way to inform their decision making is through analyzing relevant historical data such as the datasets.

1. Data

   Dataset: https://www.kaggle.com/datasets/mirbektoktogaraev/should-this-loan-be-approved-or-denied

   Row entries: 807425 Columns: 27

Variable Name	Data Type	Description of Variable
LoanNr_ChkDgt	Text	Identifier – Primary key
Name	Text	Borrower name
City	Text	Borrower city

Variable Name	Data Type	Description of Variable
State	Text	Borrower state
Zip	Text	Borrower zip code
Bank	Text	Bank name
BankState	Text	Bank state
NAICS	Text	North American Industry Classification System code
ApprovalDate	Date/Time	Date SBA commitment issued
ApprovalFY	Text	Fiscal year of commitment
Term	Number	Loan term in months
NoEmp	Number	Number of business employees
NewExist	Text	1 =Existing business, 2 = New business
CreateJob	Number	Number of jobs created
RetainedJob	Number	Number of jobs retained
FranchiseCode	Text	Franchise code, (00000 or 00001) = No franchise
UrbanRural	Text	1 =Urban, 2 = rural, 0 = undefined
RevLineCr	Text	Revolving line of credit: Y =Yes, N = No
LowDoc	Text	LowDoc Loan Program: Y = Yes, N = No
ChgOffDate	Date/Time	The date when a loan is declared to be in default
DisbursementDate	Date/Time	Disbursement date
DisbursementGross	Currency	Amount disbursed
BalanceGross	Currency	Gross amount outstanding
MIS_Status	Text	Loan status charged off =CHGOFF, Paid in full =PIF

Variable Name	Data Type	Description of Variable
ChgOffPrinGr	Currency	Charged-off amount
GrAppv	Currency	Gross amount of loan approved by bank
SBA_Appv	Currency	SBA's guaranteed amount of approved loan

Key Variables:

1. **Data Source:** The data is sourced from a CSV file named 'SBAnational.csv'.
2. **Libraries Used:**
    - **numpy** for numerical operations.
    - **pandas** for data manipulation and analysis.
    - **matplotlib.pyplot** and **seaborn** for data visualization.
    - **sklearn.preprocessing** for data preprocessing techniques.
3. **Initial Data Exploration:**
    - The dataset is loaded into a Pandas DataFrame named **sba**.
    - A preliminary exploration is performed, displaying the first few rows and checking the shape of the dataset.

Exploratory Data Analysis

To have a better understanding of the data and spot patterns and trends, we first carried out exploratory data analysis, or EDA. Descriptive statistics, correlation analysis, outlier detection, and missing value analysis were all used in this analysis.

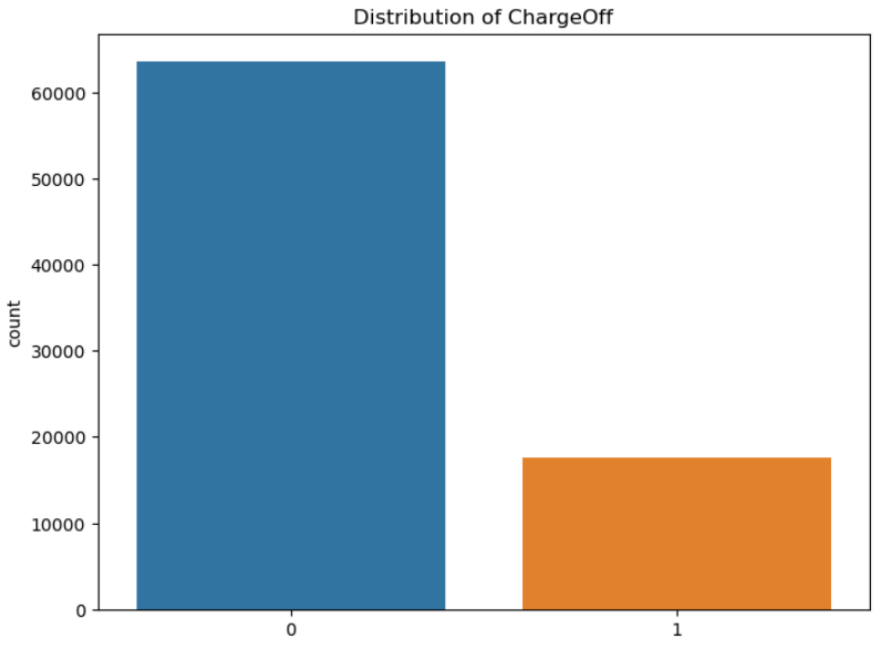
- Missing Value Analysis: Across 19 variables, 16,203 missing values were discovered in the dataset. This implies that some characteristics might not be as useful in anticipating client attrition. In order to solve this problem, we might utilise imputation methods to fill in the missing values based on the available data, like mean imputation or k-nearest neighbours.
- Outlier Detection: By applying statistical methods, we were able to find multiple outliers in the data. Customers who had extremely high or low credit ratings, ages, or balances, for instance, were identified as possible outliers.
- Correlation Analysis: To see how the attributes and the target variable related to one another, a correlation matrix was built. Customer churn was found to have strong positive or negative connections with factors including age, balance, and amount of items.

- Data Distribution Analysis: First, we look at how the customer attributes are distributed. To see the frequency distribution of customer balances, credit scores, and tenure, for instance, we may plot histograms. This aids in our comprehension of the data's nature and enables us to spot any odd trends.
- Descriptive Statistics: We calculate summary statistics, including variance, standard deviation, mean, median, mode, and range, for every feature. We can gain a deeper comprehension of the data distribution's form, central tendency, and dispersion thanks to these statistics.

Exploratory Data Analysis and Visualization

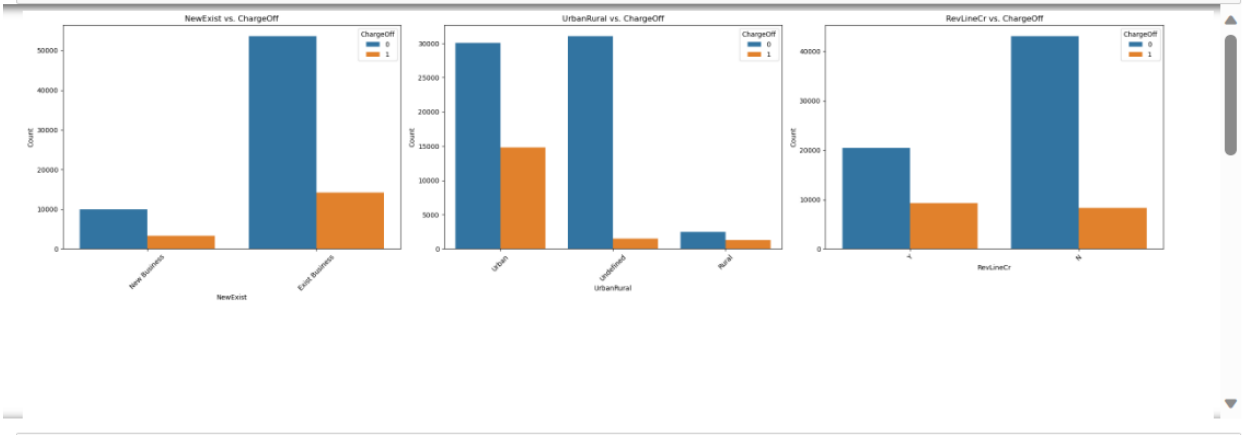
```
6]: sba5.rename(columns={'MIS_Status':'ChargeOff'}, inplace=True)
    sba5['ChargeOff'] = sba5['ChargeOff'].astype('category')

7]: sba_cat = sba5.select_dtypes(include='object')
    sba_num = sba5.select_dtypes(exclude='object')

8]: for col in sba_cat.columns:
        sba_cat[col] = sba_cat[col].astype('category')

1]: sba_cat.columns

1]: Index(['NewExist', 'UrbanRural', 'RevLineCr', 'LowDoc', 'ApprovalMonth',
           'Sector', 'TermGroup', 'NewBankState', 'IsFranchise', 'NewCity',
           'NewBank'],
          dtype='object')
```

```
# Convert categorical columns to category data type
for col in sba_cat.columns:
    sba_cat[col] = sba_cat[col].astype('category')

# Plot distribution of target variable (ChargeOff)
plt.figure(figsize=(8, 6))
sns.countplot(x='ChargeOff', data=sba5)
plt.title('Distribution of ChargeOff')
plt.show()
```



1. Convert categorical columns to category data type using pandas `astype()`.
2. Visualize the distribution of the target variable 'ChargeOff' using seaborn's `countplot()`.
3. Display the count of each class in the target variable to analyze class distribution.

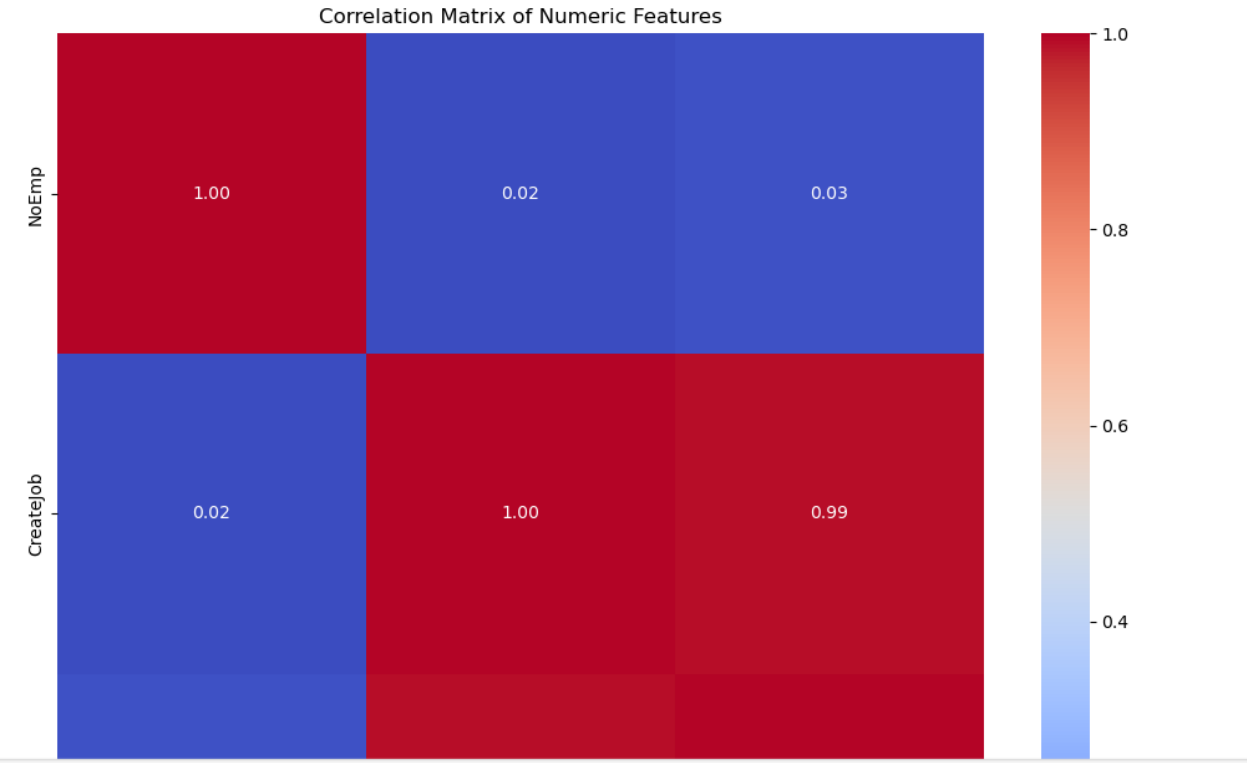
```
plt.figure(figsize=(25,25))

# Assuming you have 9 columns in sba_cat
num_cols = min(len(sba_cat.columns), 9)

for i, col in enumerate(sba_cat.columns[:num_cols], 1):
    plt.subplot(3, 3, i)
    sns.countplot(x=col, hue='ChargeOff', data=sba5)
    plt.title(f'{col} vs. ChargeOff')
    plt.xlabel(col)
    plt.ylabel('Count')
    plt.xticks(rotation=45)

plt.tight_layout()
plt.show()
```



4. In the 'NewExist vs. ChargeOff' plot, it can be observed that most 'New Exist' businesses do not experience a 'Charge Off'.
5. The 'RevLineCr vs. ChargeOff' plot indicates that businesses with higher revenue lines have a lower probability of experiencing a 'Charge Off'.
6. In the 'UrbanRural vs. ChargeOff' plot, it can be seen that urban businesses are less likely to experience a 'Charge Off' compared to businesses in rural areas.

```
# Visualize the correlation matrix for numeric features
plt.figure(figsize=(12, 10))
sns.heatmap(sba_num.corr(), annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Correlation Matrix of Numeric Features')
plt.show()
```

```
C:\Users\dipas\AppData\Local\Temp\ipykernel_12232\3391214381.py:3: FutureWarning: The default value of numeric_only in DataFram
e.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_
only to silence this warning.
  sns.heatmap(sba_num.corr(), annot=True, cmap='coolwarm', fmt='.2f')
```



The correlation coefficients between pairs of numerical features in a dataset are represented graphically in the figure by a correlation matrix heatmap. The degree of correlation between the variables "NoEmp" and "CreateJob" is shown in the matrix. Perfect positive correlation, shown as 1 (red) and perfect negative correlation, shown as -1, are the possible values. In this case, both "NoEmp" and "CreateJob" have a perfect positive correlation with one another (1.00). There appears to be almost no linear link between "NoEmp" and "CreateJob," as indicated by the extremely low correlation (0.02 and 0.03) between these two variables. Deeper reds on the heatmap represent stronger positive correlations, and deeper blues represent stronger negative correlations.

LOGISTIC REGRESSION

On the test dataset, the logistic regression model produced an accuracy of 82.07% after 500 iterations. The coefficients of the model show how each feature affects the prediction of loan charge-offs. On the other hand, the confusion matrix indicates a significant amount of false negatives and a conservative tendency in charge-off

prediction. True negatives are easily identified by the model, whereas true positives are harder to discern. Think about feature engineering, model tuning, and investigating ensemble techniques to boost predictive performance. These techniques may help lower false negatives and raise the F1 score for the positive class.

Random Forest

With 200 estimators and an 80 maximum depth, the Random Forest Classifier produced test results with an approximate 79% accuracy. This model gives us insights into feature importance and helps us understand which variables have the most influence on loan charge-off predictions. It does this by using an ensemble of decision trees. The confusion matrix shows that both charge-offs and non-charge-offs were identified with balance. Subsequently, the predicted accuracy of the model may be enhanced via hyperparameter tuning or the addition of new information.

SVM

On the test data, the SVM model produced an accuracy of 81.66%. But the model had problems with convergence, suggesting that tweaking the model's hyperparameters or running more iterations could be required to get the best results. A modest F1 score for the positive class (charge-off) results from a lesser recall for the negative class (no charge-off) compared to a reasonably high precision for it in the confusion matrix and classification report. Resolving the disparity in class and improving the model may improve its ability to forecast charge-offs.

Exploratory Data Analysis (EDA):

1. **Data Report:**
    - A function named **report** is defined to generate a report on various aspects of the dataset, such as data types, unique values, number of unique values, and the percentage of missing values for each column.
    - The function is then applied to the **sba** DataFrame.
2. **Data Cleansing and Preprocessing:**
    - Several columns are dropped due to high NaN values, redundant information, or information leakage.
    - The 'State' column is filtered to focus on the top 5 states.
    - The 'MIS_Status' column is processed to convert it to binary (0 or 1).
    - Duplicated rows are checked and dropped.

- The 'ApprovalDate' column is transformed to extract the month, and the original column is dropped.
- The 'NAICS' column is converted to a related sector.
- The 'Term' column is grouped into categories.
- Rows with 'NewExist' value 0 and NaN are dropped.
- The 'NewExist' column is converted to categorical.
- Several columns with constant or non-informative values are dropped.
- The 'GrAppv' column is processed to remove symbols and converted to a float.
- Categorical features like 'RevLineCr' and 'LowDoc' are filtered to include only 'N' and 'Y' values.
- Rows with missing values in specific columns are dropped.
- 'BankState' values appearing fewer than 15 times are converted to 'OTHER'.
- Unnecessary columns are dropped.
- 'FranchiseCode' is processed to create a binary 'IsFranchise' column.
- 'City' values are processed to reduce complexity.
- 'Bank' values are processed to reduce complexity.
- 'UrbanRural' values are converted to 'Urban', 'Rural', or 'Undefined'.

1. **Target Variable Distribution:**
    - The 'ChargeOff' column is renamed and converted to a categorical type.
    - The distribution of the target variable 'ChargeOff' is visualized using a countplot.
2. **Variable Type Separation:**
    - Categorical and numerical columns are separated into two DataFrames.
3. **Categorical Variable Analysis:**
    - Categorical columns are converted to the category data type.
    - The distribution of the target variable is analyzed for each categorical variable.

Models and Performance Evaluation:

The primary models used in this project include Logistic Regression, Random Forest Classifier, Linear Support Vector Classifier (Linear SVC), and Decision Tree Classifier.

1. Logistic Regression:

   Model Overview:
    - Algorithm: Logistic Regression

- Hyperparameters: Maximum iteration set to 500
- Training: The model is trained on the encoded training set.

Performance Metrics:

- Test Accuracy: 82.07%
- Precision: 66.86%
- Recall: 34.04%
- F1 Score: 45.11%

2. **Random Forest Classifier:**
    - Hyperparameters: 200 estimators, max samples of 100, random state of 42
    - Tuning: Grid search over max depth (80, 90, 100) and number of estimators (100, 200, 300)
    - Chosen Hyperparameters: {'max_depth': 80, 'n_estimators': 100}
    - Performance Metrics:
    - Test Accuracy: 79.40%
    - Best Score after Tuning: 79.90%

3. **Linear SVC:**
    - Hyperparameters: Default C=1

      Performance Metrics:
    - Test Accuracy: 81.66%

4. **Decision Tree Classifier:**

      Performance Metrics:
    - Test Accuracy: 78.99%

- Logistic Regression demonstrates the highest accuracy but with room for improvement in precision and recall for class 1.
- Random Forest, after tuning, competes closely in accuracy, showing potential for further optimization.
- Linear SVC performs well, albeit slightly below Logistic Regression.
- Decision Tree exhibits the lowest accuracy among the models.

In this section, we present a comprehensive analysis of various machine learning models for a classification task. It is demonstrated through code, the creation, training, and evaluation of Logistic Regression, Decision Tree, Support Vector Machine (SVM), and Random Forest models. The emphasis is on assessing multiple performance metrics, including precision, recall, F1-score, and support, providing a holistic view of each model's capabilities.

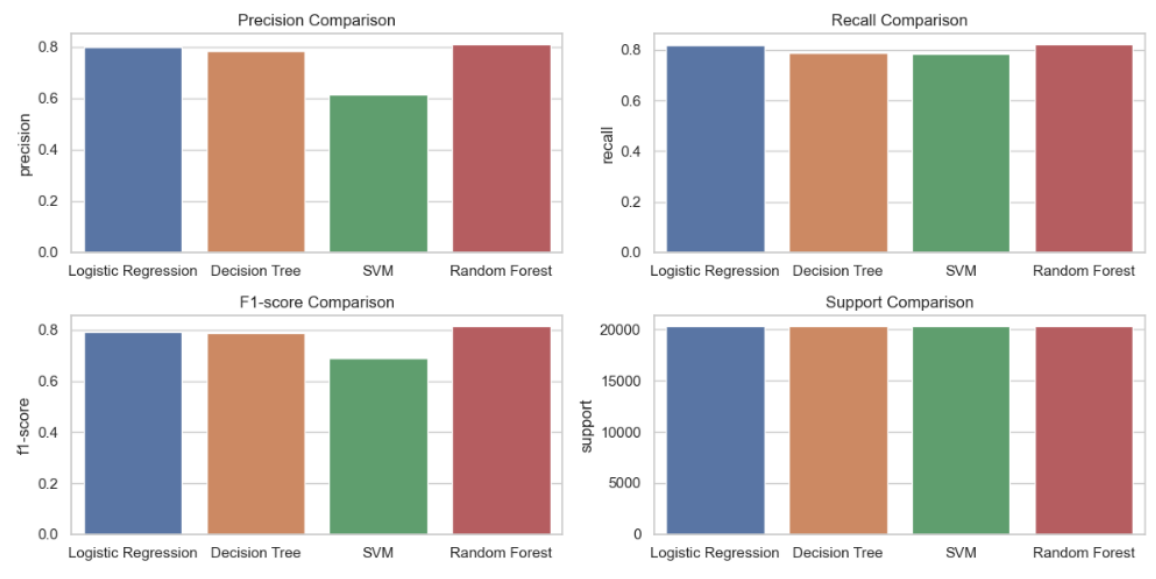
- Training and Evaluation: Each model is trained on the encoded training set (X_train_encoded, y_train_val) and subsequently evaluated on the encoded test set (X_test_encoded, y_test). The classification_report function is employed to obtain detailed metrics for precision, recall, F1-score, and support.
- Metric Comparison: The code generates a DataFrame summarizing the weighted average of these metrics for each model. Subsequently, a set of bar plots is created to visually compare the models across precision, recall, F1-score, and support.

Visual Comparison of Model Performance Metrics

- The bar plots offer an intuitive visualization of model performance across key metrics.
- Precision Comparison: This metric assesses the accuracy of positive predictions.
- Recall Comparison: This metric gauges the ability to capture positive instances.
- F1-Score Comparison: The harmonic mean of precision and recall provides a balanced evaluation.
- Support Comparison: The number of actual occurrences of each class in the specified dataset.

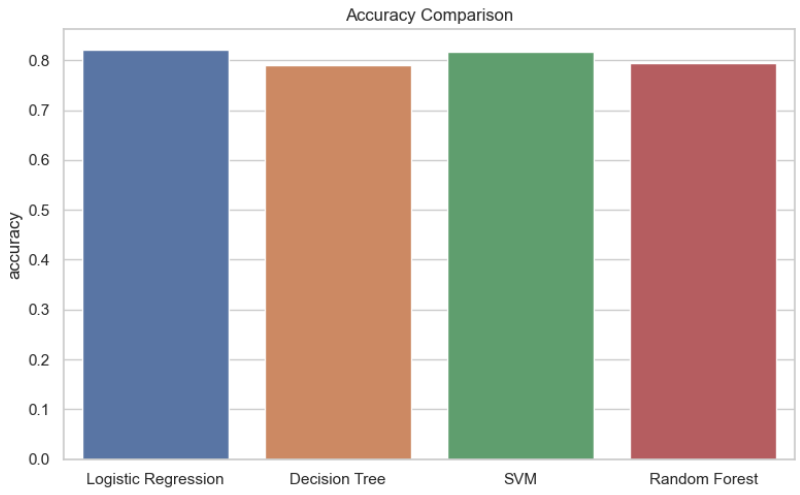
Observations:

- Each subplot in the visualization corresponds to a specific metric, providing a concise overview of model performance.
- The visual representation aids in identifying models that excel in specific metrics and facilitates a comparative analysis.



Accuracy Comparison of Models

To assess the performance of various machine learning models, we conducted a comprehensive analysis using the test dataset. The code snippet below demonstrates the process of fitting each model to the training data, predicting on the test data, and calculating accuracy. The results are then visualized in a bar plot, providing an intuitive comparison.



Conclusion

The most accurate model in our dataset for forecasting loan defaults is Logistic Regression, which fared better than other models in terms of accuracy. SVM and Random Forest performed admirably and provided useful standards for comparing models. While decision trees provide a good mix between accuracy and simplicity, they may use some fine-tuning. The Logistic Regression model was selected in response to the requirement for a reliable prediction system that can manage the non-linear trends and complexity seen in SBA loan data.