COLUMBIA UNIVERSITY DATA SCIENCE INSTITUTE

Predictive Analysis of Credit Card Application Outcomes

COMS W4995 - Topics in Computer Science - Applied Machine Learning

Project Deliverable 3

Mansi Narendra Singh, Sindhuja Rai, Kevin Noble Taylor, Joey Xia

December 9, 2023

1 Background

Financial institutions may use several factors to determine whether an applicant is accepted or rejected for a credit card. The goal of this project was to determine which factors are important in determining an applicant's status from a credit card application, and to develop and test several models to accurately predict whether an applicant was accepted or rejected.

Our project is based on the Credit Card Approval Dataset hosted on Kaggle. The dataset contains a number of numerical and categorical variables, as well as an application status encoded by 0 - "Approved" or 1 - "Rejected" which is the output variable.

2 Methodology

Analyzing the dataset consisted of four stages: an initial data exploration and deeper understanding of the dataset features, a data cleaning process during which certain attributes were dropped, missing values were imputed, and variables were encoded, a data sampling step in which the development set was resampled for training, and a model/metric proposal step where models and relevant metrics were defined, implemented and tested.

2.1 Data Exploration

Initial exploration into the dataset revealed 1,548 records with an imbalanced class distribution as detailed above. 175 records in the dataset were marked as 1 - "Rejected" and 1,373 records were marked 0 - "Accepted." Of the 18 features in the input dataset, 5 were numeric (3 discrete, 2 continuous) and the remaining 13 were categorical. The distribution of each variable, as well as the distribution of each variable with respect to the target variable was analyzed for each feature.

The distributions of the numerical variables such as number of children, annual income, birthday count (age in days), and days of employment were right-skewed and indicated they may need to be binned or log-transformed. Analyzing the categorical variables revealed that several could be dropped, as they described personal identifying information or information about the application itself, and were not features of the applicant/application that would determine acceptance/rejection. Other features would need to be imputed with missing values or encoded.

2.2 Data Cleaning

Irrelevant variables were first dropped from the dataset as they would have no predictive power in determining an applicant's outcome. Four variables were discovered to have missing values and were imputed with either "U" - unknown - for categorical variables, or with the median value for numeric variables. Multiple numeric variables were log-transformed due to their right-skewed distributions. Additionally, categorical variables were encoded depending on their number of possible values in the data. Finally, the correlation between each variable was calculated. Two sets of variables had a correlation above 0.8 - ‘Number of Children’ to ‘Number of Family Members’, and ‘Number of Employed Days’ to ‘Employment Status’ (a derived feature). One of each of these features was dropped. A summary table of data cleaning / preprocessing steps is detailed below.

Variable	Type	Preprocessing Steps
ID	Categorical	Not Relevant; Drop
Mobile Phone	Categorical	Not Relevant; Drop
Work Phone	Categorical	Not Relevant; Drop
Phone	Categorical	Not Relevant; Drop
Email ID	Categorical	Not Relevant; Drop
Gender	Categorical	Impute Missing: "U"; Binary Encoding
Annual Income	Numeric	Impute Missing: Median; Log-Transform
Birthday Count (Age)	Numeric	Impute Missing: Median
Occupation Type	Categorical	Impute Missing: "U"; Target Encoding
Car Owner	Categorical	One-Hot Encode
Property Owner	Categorical	One-Hot Encode
Employment Status	Categorical	Derived from Employed Days; Drop (High Correlation)
Type of Income	Categorical	Binary Encoding
Education	Categorical	Binary Encoding
Marital Status	Categorical	Binary Encoding
Housing Type	Categorical	Binary Encoding
Employed Days	Numeric	Log-Transform
# of Children	Numeric	None; Drop (High Correlation)
# of Family Members	Numeric	None

After these transformations were applied, all features were scaled using a standard scaler.

2.3 Data Sampling

To deal with the imbalanced dataset, data sampling was performed using random oversampling. Due to the large number of categorical variables in the data, random oversampling was chosen over other techniques such as SMOTE which is based on interpolation. A train-test split was performed (80% development, 20% test), and random oversampling was applied to the development dataset.

2.4 Models and Model Evaluation

Several models were considered for this project. Each model was tested with the base parameters, and then was hyperparameter-tuned using five-fold cross validation. The optimal hyperparameters for each model were selected based on the highest average precision with respect to the positive class (the deep neural network optimal hyperparameters were chosen based on CV accuracy). Following hyperparameter tuning, the optimal model was re-trained on the whole development set, and the performance on the test set was evaluated.

3 Results

The models considered, the base performance, the hyperparameters tuned, and the performance on the test set are detailed below:

Model Name:	**Logistic Regression**
Base Model Performance:	69% Test Accuracy, 0.20 Precision, 0.60 Recall, 0.30 F1
Hyperparameters Tuned:	*Regularization constant C; loss penalty (l1 vs. l2); solver*
Optimal Model Performance:	68% Test Accuracy, 0.19 Precision, 0.54 Recall, 0.28 F1

Model Name:	**Support Vector Machines (SVM)**
Base Model Performance:	73% Test Accuracy, 0.24 Precision, 0.63 Recall, 0.35 F1
Hyperparameters Tuned:	*C constant; kernel type; gamma*
Optimal Model Performance:	83% Test Accuracy, 0.33 Precision, 0.49 Recall, 0.40 F1

Model Name:	**Decision Tree**
Base Model Performance:	85% Test Accuracy, 0.36 Precision, 0.46 Recall, 0.40 F1
Hyperparameters Tuned:	*Maximum Depth; Min Samples Split; Min Samples Leaf*
Optimal Model Performance:	76% Test Accuracy, 0.24 Precision, 0.51 Recall, 0.33 F1

Model Name:	**Random Forest**
Base Model Performance:	94% Test Accuracy, 0.86 Precision, 0.51 Recall, 0.64 F1
Hyperparameters Tuned:	*Number of Estimators; Max Depth; Min Samples Split*
Optimal Model Performance:	93% Test Accuracy, 0.75 Precision, 0.51 Recall, 0.61 F1

Model Name:	**Gradient Boosted Trees**
Base Model Performance:	80% Test Accuracy, 0.29 Precision, 0.54 Recall, 0.38 F1
Hyperparameters Tuned:	*Learning Rate; Number of Estimators; Max Depth; Loss*
Optimal Model Performance:	89% Test Accuracy, 0.53 Precision, 0.46 Recall, 0.49 F1

Model Name:	**AdaBoost Trees**
Base Model Performance:	70% Test Accuracy, 0.20 Precision, 0.54 Recall, 0.29 F1
Hyperparameters Tuned:	*Learning Rate; Number of Estimators; Base Estimator*
Optimal Model Performance:	93% Test Accuracy, 0.84 Precision, 0.46 Recall, 0.59 F1

Model Name:	**Deep Neural Network (DNN)**
Base Model Performance:	76% Test Accuracy, 0.23 Precision, 0.46 Recall, 0.30 F1
Hyperparameters Tuned:	*Network Architecture; Number of Epochs*
Optimal Model Performance:	84% Test Accuracy, 0.36 Precision, 0.57 Recall, 0.44 F1

4 Conclusion

From the performances of the seven models on the test set, we found that Random Forest and AdaBoost trees outperformed the others with a test accuracy of 93%. These models were able to correctly predict the outcome for the imbalanced dataset with a relatively good trade-off between precision and recall. In conclusion, the ensemble methods are seen to have strong performances in the predictive analysis of the credit card applications.