Random Forest Classification Analysis on Loan Approval Dataset

Taylor Cina

D214 – Data Analytics Graduate Capstone

January 6, 2025

***Research Question***

The research question for this analysis is “Can a random forest classifier be used to accurately approve loans based on the research dataset?”

Before machine learning and widespread computer use, loan approval was determined by manual review and individual decision-making. This process is subjective and leads to an increased risk of loan defaults. Machine learning algorithms are exceptional at evaluating and predicting this kind of behavior, which will reduce the risk of clients defaulting on their loans. (Saini, 2023) A random forest classification algorithm will be used to analyze the dataset. Random forests train uncorrelated decision trees independently to determine a binary outcome (IBM, 2024). Saini (2023) determined that a random forest classifier was an especially fitting choice for determining loan approval, citing an accuracy of over 98%.

The following hypotheses are to be considered:

***Null Hypothesis H*0:** A random forest classification model cannot be constructed from the research dataset with an accuracy of > 80%.

***Alternative Hypothesis H*a:** A random forest classification model can be constructed from the research dataset with an accuracy of > 80%.

In addition to accuracy, the precision, recall, and F1 scores will be considered when assessing model fitness.

***Data Collection***

The dataset used for this analysis is a synthetic dataset derived from real loan approval data without any identifying user information created specifically for the purpose of model training. The dataset is hosted [publicly on Kaggle (Lo, 2024)](https://www.kaggle.com/datasets/taweilo/loan-approval-classification-data). The dataset contains 45,000 records and 14 columns. The following table serves as a data dictionary for the dataset:

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| person\_age | Numerical | Age of the person |
| person\_gender | Categorical | Gender of the person |
| person\_education | Categorical | Highest education level |
| person\_income | Numerical | Annual income |
| person\_emp\_exp | Numerical | Years of employment experience |
| person\_home\_ownership | Categorical | Home ownership status |
| loan\_amnt | Numerical | Loan amount requested |
| loan\_intent | Categorical | Loan purpose |
| loan\_int\_rate | Numerical | Loan interest rate |
| loan\_percent\_income | Numerical | Loan amount as a percentage of annual income |
| cb\_person\_cred\_hist\_length | Numerical | Credit history length in years |
| credit\_score | Numerical | Credit score |
| previous\_loan\_defaults\_on\_file | Categorical | Indicator of previous loan defaults |
| loan\_status | Numerical | Loan approval status  0 = rejected, 1 = approved |

While this dataset is created specifically for the purpose it is being used for in this analysis, there is always the possibility that the synthetic dataset does not perfectly accurately represent a real-world set with identifying user information. A loanee’s location could very well play a role in their possibility of defaulting on a loan, and a public dataset isn’t going to be able to include real addresses of actual borrowers.

The dataset is a single downloadable CSV file that is easily imported into Python using the *pandas* library. There were no significant challenges regarding collecting the data for this analysis.

***Data Extraction and Preparation***

The first step of the extraction process was to explore the dataset and determine if there were any issues that would need to be fixed during data preparation. The initial exploration process is pictured below:

A screen shot of a computer program

Description automatically generated

No duplicate, missing, or infinite values were detected. Checking these issues is vital when training a machine learning algorithm as unclean data can completely skew the result of model training and create a useless model. If data were missing from the dataframe, it would have to either be dropped or imputed. Dropping data is the easiest way to eliminate missing data, but the dataset becomes smaller and therefore less useful. Imputing data preserves the size of the dataset, but imputation can introduce data points that are not representative of the rest of the dataset. Pictured below are the printouts of the .info(), .describe(), duplicate analysis, and missing values graph provided by the *missingno* package:

A computer screen shot of a black screen

Description automatically generatedA screenshot of a computer

Description automatically generated

A black background with white text

Description automatically generated

A graph of a number

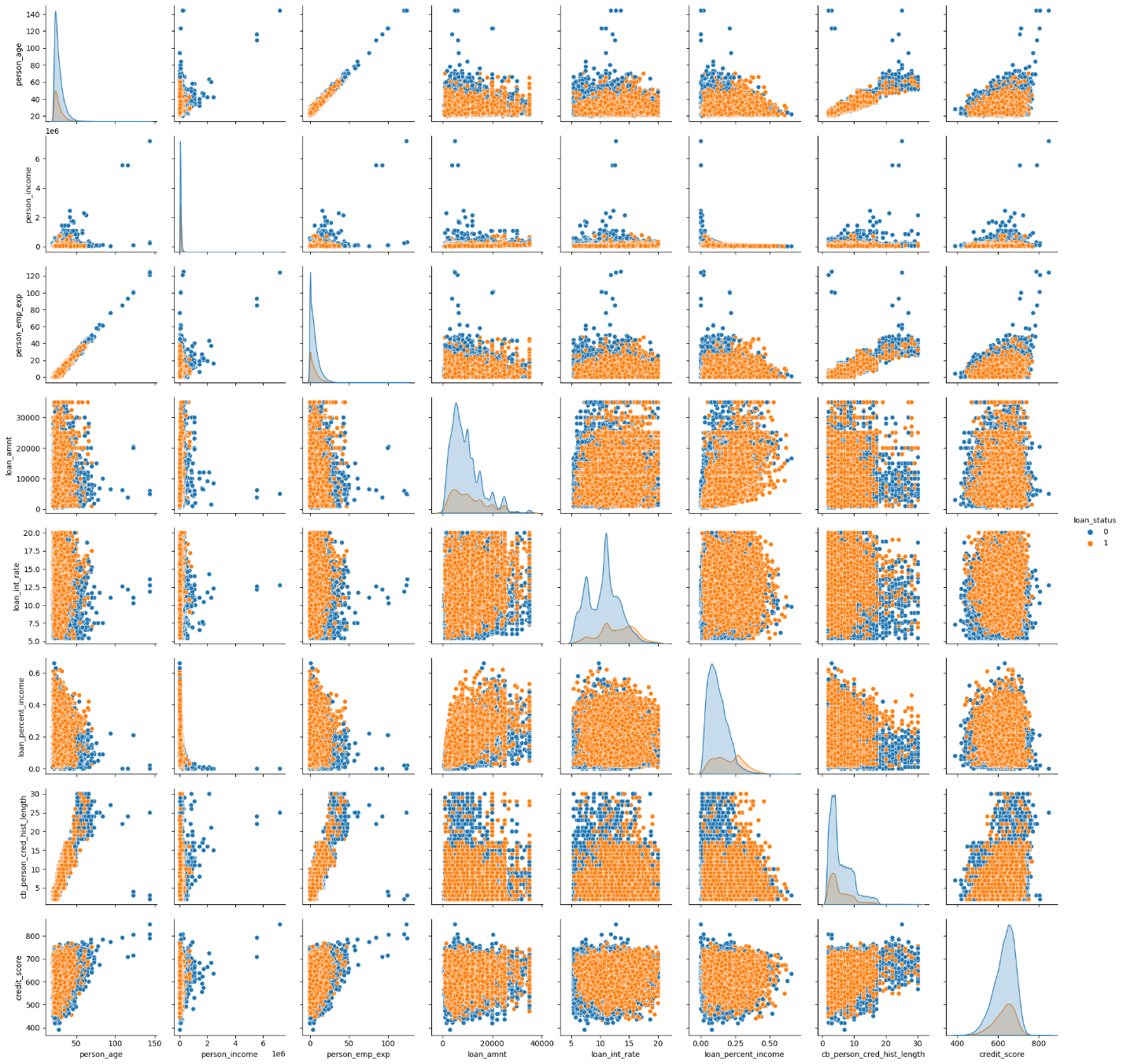
Description automatically generated with medium confidence

The next step of the extraction process was to get a better understanding of the distributions of each column and their relationships to the target variable. The code used for this is shown below:

A screen shot of a computer program

Description automatically generated

First, a pair plot was generated for the continuous variables in the dataset and colored based on the relationship with the “loan\_status” variable. The pair plot is shown below:



Subsequently, the distribution of each variable individually was visualized. The categorical variables have simple printouts of their distribution percentages for all their unique values, and the numerical variables have histograms. These are shown below:

A screenshot of a computer

Description automatically generatedA graph of a person age

Description automatically generated

A graph of a person income

Description automatically generatedA graph with numbers and lines

Description automatically generated

A graph of a long line

Description automatically generated with medium confidenceA graph of a loan rate

Description automatically generated

A graph of a long line

Description automatically generatedA graph of a person cred hist length

Description automatically generated

A graph of credit score

Description automatically generatedA graph with a blue bar

Description automatically generated

Minimal preprocessing was required for this dataset to convert all of the object variable fields into numeric or Boolean fields. Binary value fields were converted from string “Yes” and “No” values to true and false values, ordinal values were converted from string fields to integer series, and non-ordinal string fields were one-hot encoded with the original columns being dropped. Giving string fields that have ordinal relationships to each other integer values allows the model training process to learn the ordered relationship of those variables. The code to perform these operations is shown below:

A screen shot of a computer code

Description automatically generated

The last step of data preprocessing was to create the training and testing datasets. The test split used in this analysis is 70/30 for training and testing data respectively; the data was stratified on the target variable “loan\_status”. Stratifying means that the training and testing data will have a roughly equal proportion of values where loan\_status is true or false. Splitting the data allows a model to be trained on the training data and tested on data the model has not been exposed to. This provides a more realistic view of how the model performs and helps eliminate overfitting to the testing data. Additionally, a custom seed is used for reproducibility. It should be stated that - due to the synthetic nature of the dataset - it is possible that both the training and testing data are not representative of real-world scenarios and any model generated from this dataset would perform poorly on real-world data.

The code for performing this split is shown below:

A black screen with white text

Description automatically generated

With this, the data is fully prepared to be used in model training.

***Analysis***

The model-building process consisted of two major phases. The first phase was the initial build of the model and analysis of its performance and most important features. The second phase was dimensionality reduction and testing how fewer features impacted the performance of the model.

Random forest classification is especially well suited to this analysis as it has been used by other researchers for a similar problem with a high degree of accuracy (Saini, 2023). However, random forests are computationally expensive, and each build of a new model took two or more minutes even with a dataset of medium size on a higher end machine.

The hyperparameters were tuned using randomized search cross-validation provided by scikit-learn’s *RandomizedSearchCV* package. This does not check every value in the listed parameters but searches parameters randomly. It performs similarly to a full grid search cross-validation while operating faster. While it is possible that the best hyperparameters could be missed using this algorithm, the given hyperparameters still perform well. The hyperparameters were optimized for high F1 scores, as it is a good metric for this analysis due to the nature of the question. Defaulting on a loan is a loss for the company that lent it. Minimizing defaults while maximizing properly approved loans should lead to higher profits. The best hyperparameters found were max\_depth = 4, n\_estimators = 318, and min\_samples\_leaf = 4. Scikit-learn’s *RandomForestClassifier* package was used to generate and train the model. The model was then used to predict values on the testing data and compared to the real values on the testing data. A classification report and confusion matrix were generated from these results. The classification report contains statistics generated from the confusion matrix like accuracy, recall, precision, and the F1 score.

The code performing the first build of the model is shown below:

*A computer screen shot of a program code

Description automatically generated*

The output of the first classification report and confusion matrix is shown below:A screenshot of a computer

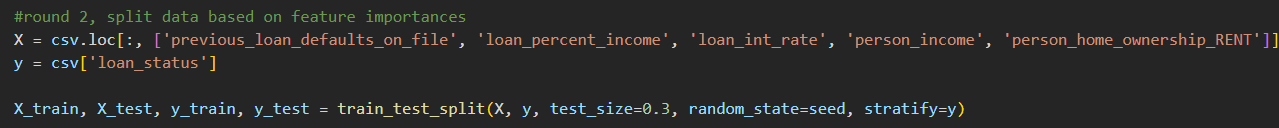
Description automatically generated

After the first build, the feature importances can be calculated. The code and output of the feature importances graph is shown below:

A screenshot of a computer

Description automatically generated

With this information, a new model can be trained using fewer features. The data is split once again using the same seed but with a smaller number of predictor variables. This was done for the top 4, 5, and 6 features. The code for splitting with only the top 5 features is shown below:



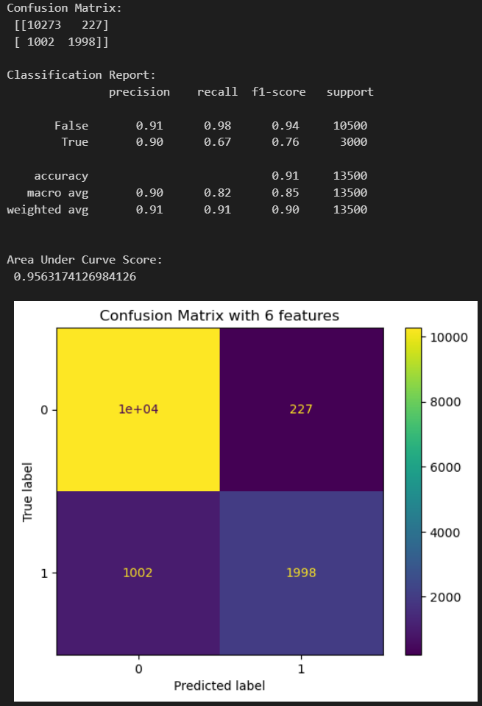
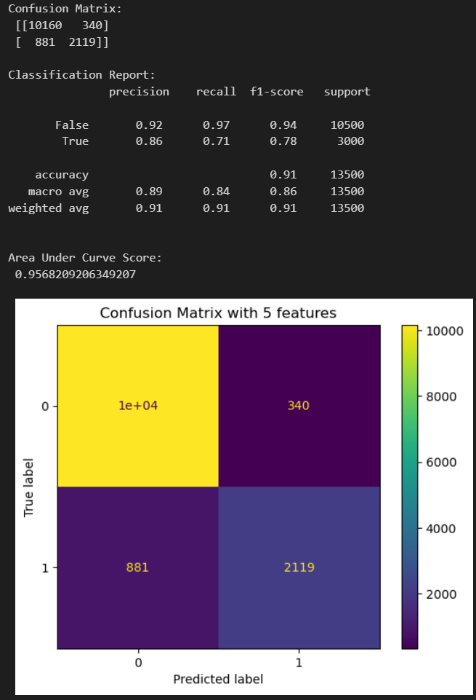
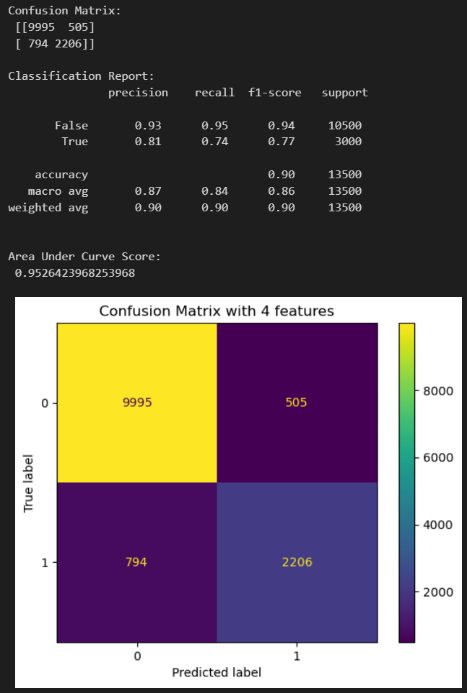
This generates the same splits as the previous model but with fewer columns in the X tables.

Hyperparameter tuning was done for each new model to ensure the most useful results. The process is the same as the initial model and is shown below:

A computer screen shot of a program code

Description automatically generated

The confusion matrices and classification reports for each of the 3 new models is shown below:



***Data Summary and Implications***

According to the results of the analysis, it is conclusive that random forest classification is capable of producing a model with an accuracy greater than 80%. Therefore, the null hypothesis is rejected, and the alternative hypothesis is accepted. Even the initial unoptimized model had an accuracy of greater than 80%, though this metric is not as impressive when one realizes that only 22% of the dataset contained approved loans. Even a model that denied all loans unambiguously would have an accuracy of 78%. In the case of approving loans, the initial model has a terrible recall of 56%, which is barely better than random guessing. However, it was very capable of correctly predicting unapproved loans, with a recall of 99% when the true value was unapproved. This left much to be desired, which was rectified with dimensionality reduction and better-optimized models. The initial model was likely overfitted to the training data due to having 13 indicator variables and could not handle testing data well.

The second round of models performed much better, with the 5-feature model having the most desirable traits for this specific business case. The new models were evaluated primarily by their F1 score due to it being a balance between correctly rejecting clients who would default on their loans and correctly accepting clients who would not. Too many defaults or rejections of loanees in one direction or the other would lead to a loss of profits, so the F1 score was a sensible evaluation metric.

The 5-feature model performed best, so it will be the one discussed here. This model has the highest F1 score of 78%. Although it does not have the highest recall, it doesn’t score much lower than the model that does. An F1 score of 78% is significantly more indicative of a functional model than accuracy is in this case as the dataset is skewed heavily toward unapproved loans.

Although this analysis found that a 5-feature model had the most desirable traits, what those 5 features are could be different than the ones listed here as they were based on an initial model that did not perform as well. A more thorough analysis of correlated features could be performed to create an initial model that may have different feature importances.

Because it has been shown that the top 5 features of the initial model can satisfactorily predict loan defaults, each of those aspects should be researched further to learn in what way they affect the probability of loan defaults. Changes can be made within the company regarding the variables it has direct control over (interest rate, loan amounts, etc.) to minimize defaulting.

There are other analyses that are worth performing on this dataset besides the one shown here. Separating the data based on age groups, education level, or loan purpose for example could reveal patterns that aren’t present within a full range analysis. Another example is the fact that loanee credit score appeared to play a very minimal role in predicting loan defaults according to the initial model’s feature importance list, which doesn’t initially appear to make intuitive sense. It would also be worthwhile to analyze a real-world dataset with similar fields and see if the findings of this analysis correspond with the findings of an analysis conducted on non-synthetic data.

***References and Sources***

[IBM. (2024, December 19). *What is Random Forest?* https://www.ibm.com/think/topics/random-forest](https://www.ibm.com/think/topics/random-forest)

[Lo, T. (2024, October 29). *Loan approval classification dataset*. Kaggle. https://www.kaggle.com/datasets/taweilo/loan-approval-classification-data](https://www.kaggle.com/datasets/taweilo/loan-approval-classification-data)

[P. S. Saini, A. Bhatnagar and L. Rani, "Loan Approval Prediction using Machine Learning: A Comparative Analysis of Classification Algorithms," *2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, Greater Noida, India, 2023](https://ieeexplore.ieee.org/document/10182799/metrics#metrics)