Report on Credit Application Status Prediction

1. Approach Taken

The goal of this project was to build a predictive model to classify the application status(Approved or Declined) based on various applicant features. The approach can be summarized in the following steps:

1. **Data Exploration and Preprocessing:**
   - **Duplicate Removal:** We checked for and removed duplicate rows from the dataset.
   - **Handling Missing Values:** We identified columns with a significant number of missing values and dropped columns where more than 15% of the data was missing. Rows with missing values in key features were also removed.
   - **Feature Engineering:**
     - Converted categorical features like 'GENDER' into numerical values.
     - Dropped columns that were deemed unnecessary based on the domain knowledge and their correlation with the target variable.
   - **One-Hot Encoding:** We applied one-hot encoding to categorical features to convert them into numerical format.
   - **Feature Scaling:** Standardized numerical features to ensure they were on the same scale.
   - **Handling Class Imbalance:** Used a combination of SMOTE(Synthetic Minority Over-sampling Technique) and RandomUnderSampler to balance the classes in the training set.

2. **Model Development:**
   - **Decision Tree Classifier:** I started with a simple decision tree model tuning hyperparameters like max_depth and min_sample_leafs using cross-validation to avoid overfitting.
   - **Random Forest Classifier:** To improve model performance I implemented a Random Forest model and tuned its parameters.
   - **Logistic Regression:** I also used Logistic Regression and experimented with different decision thresholds to find an optimal balance between precision and recall.

3. **Model Evaluation:**
   - Evaluated models on training data using accuracy, F1 score and cross-validation scores.
   - Used metrics like confusion matrix to understand the distribution of predictions and identify areas of improvement.

2. Insights and Conclusions from Data

- **Feature Importance:** Features like 'GENDER', 'AGE' and 'INCOME' had a significant impact on the application status. This was evident from the correlation matrix and the feature importance plots generated from the Random Forest model.

- **Class Imbalance:** The data was highly imbalanced with a majority of applications being approved. Handling this imbalance was crucial to improving the model’s performance particularly in terms of recall for the minority class(Declined applications).

- **Correlation Insights:** Some features were highly correlated which suggests potential multicollinearity. However models like Random Forest can handle such correlations well. Features that were redundant or had only one unique value across all rows were dropped to simplify the model.

3. Performance on Train Dataset

- **Decision Tree Classifier:**
    - **Training Accuracy:** 99.85%
    - **Training F1 Score:** 99.86%
    - **Cross-Validated Mean Accuracy:** 66.42%
    - **Cross-Validated Mean F1 Score:** 78.52%
    - The model showed signs of overfitting with high training accuracy but slightly lower validation scores.

- **Random Forest Classifier:**
        - **Test Accuracy:** 67.92%
        - **Test F1 Score:** 80.11%
    - The Random Forest model reduced overfitting and improved the generalization of the model.

- **Logistic Regression:**
    - **Best Threshold (0.3):**
        - **Test Accuracy:** 66.82%
        - **Test F1 Score:** 80.10%
        - Logistic regression provided a good balance between simplicity and performance particularly after tuning the decision threshold.

4. Test Predictions and Metrics

- **Final Model Chosen:** Random Forest Classifier.

- **Metrics on Test Data:**
    - **Test Accuracy:** 67.92%
    - **Test F1 Score:** 80.11%
    - **Confusion Matrix:** The model was able to correctly predict a high number of approved applications with a reasonable balance in detecting declined applications.

- **Predictions:** The final predictions on the test set were saved in a CSV file named predictions.csv containing the UID and corresponding prediction(0 or 1).

Conclusion:

The project successfully developed a model that can predict the application status with reasonable accuracy. While the model performs well further improvements can be made by exploring advanced techniques such as ensemble learning or more sophisticated preprocessing. Additionally addressing class imbalance during both training and prediction stages remains a critical factor for improving real-world performance.