Bureau_Assignment

Report: Loan Approved / Not Approved.

1. Approach Taken:

- The first step was of loading and examining the datasets (Assignment_Train.csv and Assignment_Test.csv) to understand the structure and content of the data.
- Columns with more than 50% missing values were dropped to avoid excessive imputation, focusing on retaining features with sufficient data.
- Then we have divided the data into 2 set
    - Numerical Features
    - Categorial Feature

Now Feature training and Selection

- I have used exixting feature to evaluate the model
- Then the cleaned dataset was split into raining and validation sets to evaluate model performance.

Now Model Selection and Training:

- A Random Forest Classifier was chosen for its robustness and ability to handle a mix of numerical and categorical features without extensive preprocessing.
- Model was trained using a pipeline that combined preprocessing steps and model training, ensuring a streamlined process.

Predictions:

- The trained model was used to predict the application status for the test dataset, and the results were saved in a CSV file nammed Predictions.csv

2. Insights and Conclusions from Data:

- ***Missing Data:*** A significant portion of the data had missing values, especially in social media-related features and asset cost details. So we have clean the data to get best possible results.

- ***Feature Relevance:*** Features like Cibil Score, mobile verification, and AADHAR verification likely play crucial roles in determining the loan approval status, given their direct relationship with an applicant's creditworthiness.

- ***Data Imbalance:*** Without the exact distribution of the target variable (Application Status), it's challenging to determine if the data is balanced or if there is a bias toward loan approvals or rejections. This could impact model performance.

3. Performance on Train Dataset

- Training Accuracy: The model achieved high accuracy on the training dataset, suggesting it could learn from the provided data effectively.

- Validation Accuracy: The model's performance on the validation set was slightly lower than on the training set, indicating some generalization ability but also suggesting the potential for slight overfitting. This is typical for an initial model without hyperparameter tuning.

4.Classification Report:

[37]:
```
# evaluateing  the model
y_train_pred = model_pipeline.predict(X_train)
y_val_pred = model_pipeline.predict(X_val)

train_accuracy = accuracy_score(y_train, y_train_pred)
val_accuracy = accuracy_score(y_val, y_val_pred)
classification_rep = classification_report(y_val, y_val_pred)

print("Training Accuracy:", train_accuracy)
print("Validation Accuracy:", val_accuracy)
print("Classification Report:\n", classification_rep)
```

```
Training Accuracy: 0.99975
Validation Accuracy: 0.7695
Classification Report:
              precision    recall  f1-score   support

    APPROVED       0.76      0.96      0.85      1327
    DECLINED       0.84      0.39      0.53       673

    accuracy                           0.77      2000
   macro avg       0.80      0.68      0.69      2000
weighted avg       0.78      0.77      0.74      2000
```

RESULTS :

Training Accuracy: 0.99975

Interpretation: The model performs almost perfectly on the training data. .

Validation Accuracy: 0.7695

Interpretation: The model performs much worse on the validation data. This drop in accuracy indicates that the model may not generalize well and has likely overfitted to the training data.

Classification Report

1. **Precision:** The precision for each class indicates how many of the predicted positives for that class are true positives.

    APPROVED: 0.76

    DECLINED: 0.84

2. **Recall:** The recall for each class shows how many actual positives for that class were correctly predicted.

    APPROVED: 0.96

    DECLINED: 0.39

3. **F1-Score:** The F1-score is the harmonic mean of precision and recall, providing a single metric that balances both.

    APPROVED: 0.85

    DECLINED: 0.53

4. **Macro Average:** Averages the performance across classes, treating all classes equally.

    Precision: 0.80

    Recall: 0.68

    F1-Score: 0.69

5. **Weighted Average:** Averages the performance across classes, weighted by the number of true instances for each class.

    Precision: 0.78

    Recall: 0.77

    F1-Score: 0.74

Conclusion:

The model is a good starting point, providing a solid baseline for predicting loan application status. Future improvements could include more advanced feature engineering, hyperparameter tuning, and addressing any class imbalance issues.