# Development and Evaluation of the Behavior Score Model

# ------------------------------------------------------

***Introduction***

The code aims to develop a predictive model for credit card default probability, referred to as the "Behavior Score," for Bank A. Using historical data provided in the "development" and "validation" datasets, the pipeline covers all stages, including data exploration, preprocessing, model training, evaluation, and prediction generation.

The chosen algorithm, Gradient Boosting Classifier, is a robust ensemble learning technique well-suited for handling non-linear relationships and imbalanced datasets. Below is a step-by-step breakdown of the code, algorithms, and approach.

--------------------------------------------------------------------------------------------------------------------------------------

***Code Walkthrough***

**1. Loading Data**

**Function**: load\_data(dev\_path, val\_path)

Objective:

1. Read development and validation datasets from specified file paths.
2. Handle missing files gracefully with error handling (FileNotFoundError).

**Implementation:**

1. Uses pandas.read\_csv to load data.
2. Ensures both datasets are available for downstream processing.

**Key Consideration:** Early error handling prevents runtime interruptions due to missing files.

--------------------------------------------------------------------------------------------------------------------------------------

**2. Exploratory Data Analysis (EDA)**

**Function:** perform\_eda(dev\_data)

**Objective:**

1. Gain insights into the dataset.
2. Check for data quality issues like missing values, class imbalance, and outliers.

**Steps:**

*1. Dataset Overview:*

info() and describe() provide details about data types, non-null counts, and summary statistics.

*2. Target Variable Distribution:*

value\_counts() examines the proportion of default (bad\_flag=1) vs. non-default (bad\_flag=0) customers. Class imbalance can impact model performance.

*3. Missing Value Analysis:*

isnull().sum() identifies features requiring imputation.

*4. Visualization:*

seaborn.countplot visualizes the distribution of bad\_flag.

**Insights:**

1. Early-stage analysis identifies data issues and guides preprocessing.
2. Visualizations highlight class imbalance and other patterns.

--------------------------------------------------------------------------------------------------------------------------------------

**3. Data Preprocessing**

**Function:** preprocess\_data(dev\_data)

**Objective:** Prepare data for modeling by handling missing values, encoding categorical variables, and normalizing numerical features.

**Steps:**

*1. Data Cleaning:*

- Removes whitespace in string columns

- Replace empty strings with NaN for consistency.

*2. Missing Value Imputation:*

- Numerical features: Filled with median values to handle skewed distributions.

- Categorical features: Filled with the most frequent value (mode()).

*3. Feature-Target Split:*

Drops irrelevant columns like account\_number.

Separates predictors (X) and target (y).

*4. Feature Scaling:*

Applies StandardScaler to standardize numerical features.

**Output:** Cleaned and scaled feature matrix (X) and target vector (y).

--------------------------------------------------------------------------------------------------------------------------------------

**4. Data Splitting**

**Function:** split\_data(X, y)

**Objective:**

Divide the data into training (70%) and testing (30%) sets for model development and evaluation.

**Implementation:**

Uses train\_test\_split from sklearn.model\_selection.

**Purpose:**

Ensures the model is evaluated on unseen data, preventing overfitting.

--------------------------------------------------------------------------------------------------------------------------------------

**5. Model Training**

**Function:** train\_model(X\_train, y\_train)

**Objective:**

* Train a Gradient Boosting Classifier to predict default probabilities.
* Tune hyperparameters for optimal performance using GridSearchCV.

**Algorithm:**

Gradient Boosting

Ensemble learning technique combining multiple weak learners (decision trees).

Boosts performance iteratively by focusing on misclassified samples.

Effective for non-linear relationships and imbalanced datasets.

**Hyperparameter Tuning:**

* n\_estimators: Number of boosting stages.
* learning\_rate: Step size for updating weights.
* max\_depth: Maximum depth of trees to prevent overfitting.

**GridSearchCV:**

* Performs an exhaustive search over the specified parameter grid.
* Optimizes for the roc\_auc metric to ensure sensitivity to class imbalance.

**Output:**

Returns the best estimator from the grid search.

--------------------------------------------------------------------------------------------------------------------------------------

**6. Model Evaluation**

**Function:** evaluate\_model(model, X\_test, y\_test)

**Objective:**

Assess the model's performance on the test set using classification metrics and AUC-ROC score.

**Metrics:**

1. *AUC-ROC Score:*

* Measures the ability of the model to distinguish between classes.
* Higher values indicate better class separability.

**Classification Report:**

Provides precision, recall, and F1-score for each class.

**Implementation:**

* roc\_auc\_score computes the AUC-ROC score for the test set.
* classification\_report evaluates and displays detailed metrics for each class.

-------------------------------------------------------------------------------------------------------------------------------------**7. Validation Predictions**

**Function:** predict\_validation(model, val\_data, scaler)

**Objective:**

Generate default probabilities for the validation dataset.

**Steps:**

*1. Data Cleaning:*

Removes whitespace and replaces empty strings.

*2. Missing Value Imputation:*

Fills missing values using medians (aligned with the training dataset).

*3. Feature Scaling:*

Applies the same scaler (StandardScaler) used for training data.

*4. Prediction:*

1) Computes probabilities using predict\_proba of the trained model.

2) Adds a predicted\_probability column to the dataset.

**Output:**

Exports results to submission.csv for downstream use.

--------------------------------------------------------------------------------------------------------------------------------------

***Algorithms Used***

**1. Gradient Boosting Classifier:**

**Advantages:**

1. Handles complex non-linear relationships.
2. Robust to overfitting with hyperparameter tuning.
3. Handles imbalanced datasets effectively.

**Limitations:**

1. Computationally expensive for large datasets.
2. Sensitive to hyperparameters.

**2. GridSearchCV:**

- Automates hyperparameter tuning.

- Ensures optimal model configuration for the given dataset.

--------------------------------------------------------------------------------------------------------------------------------------

***Pipeline Summary***

**1. Data Loading**:

- Validates input files and reads data.

**2. EDA:**

**-** Examines data distributions, missing values, and class imbalance.

**3. Preprocessing:**

- Cleans, imputes and scales data for modeling.

**4. Splitting:**

- Divides data into training and testing sets.

**5. Model Training:**

- Builds and tunes a Gradient Boosting Classifier.

**6. Evaluation:**

- Uses AUC-ROC and ROC curve for performance assessment.

**7. Prediction:**

- Scores validation data and exports results.

--------------------------------------------------------------------------------------------------------------------------------------

***Data Analysis and Observations***

The provided summary statistics, proportions, and missing value information offer insights into the dataset's structure, distribution, and quality. Below is a detailed analysis of the data in data science terms:

--------------------------------------------------------------------------------------------------------------------------------------

**1. Target Variable (bad\_flag)**

**Proportion Distribution:**

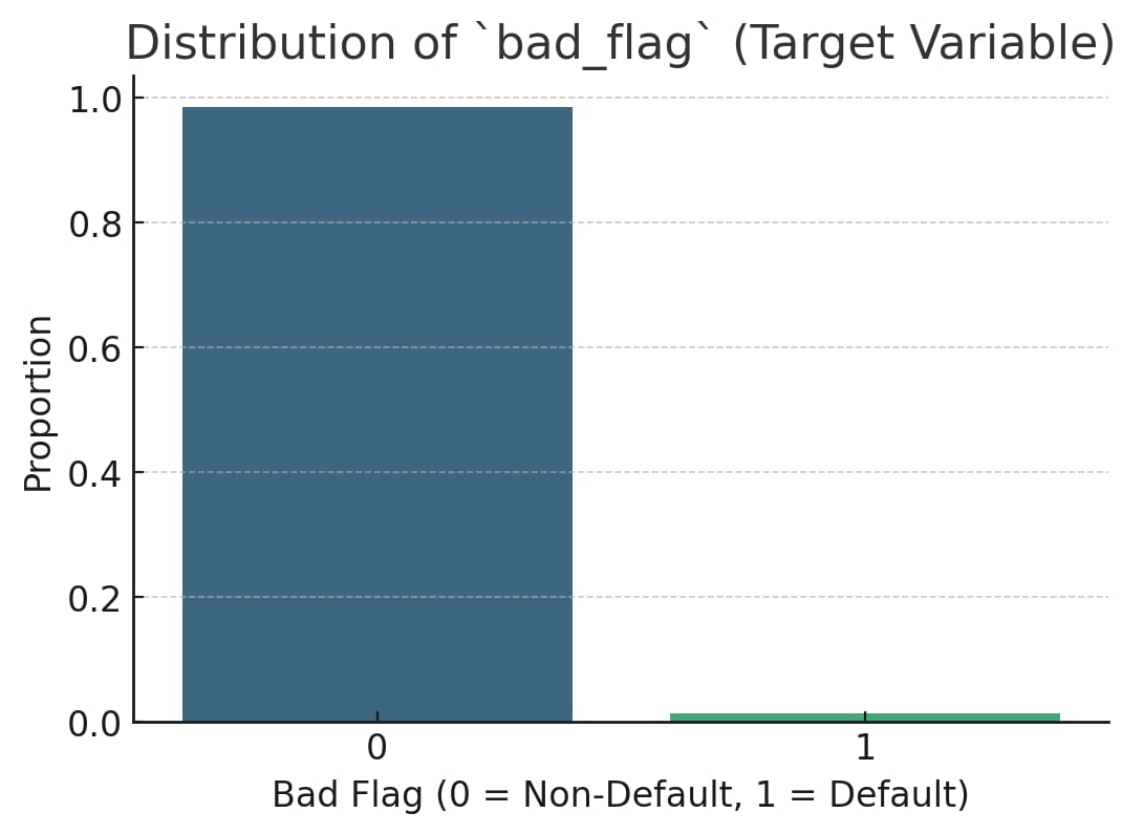
* Non-default (bad\_flag = 0): 98.59%
* Default (bad\_flag = 1): 1.41%

**Key Observations:**

* The dataset is highly imbalanced, with very few default cases.
* Imbalanced datasets can lead to models biased toward the majority class. Specialized techniques (e.g., oversampling, undersampling, class weights) may be necessary for training robust models.

This bar plot shows the proportion of each class in the target variable

* (bad\_flag).bad\_flag = 0 represents non-default customers (no payment issues).
* bad\_flag = 1 represents default customers (high risk of defaulting).



--------------------------------------------------------------------------------------------------------------------------------------

**2. Statistical Summary of Features**

**General Insights**

*Features like onus\_attribute\_44, onus\_attribute\_45, and onus\_attribute\_46:*

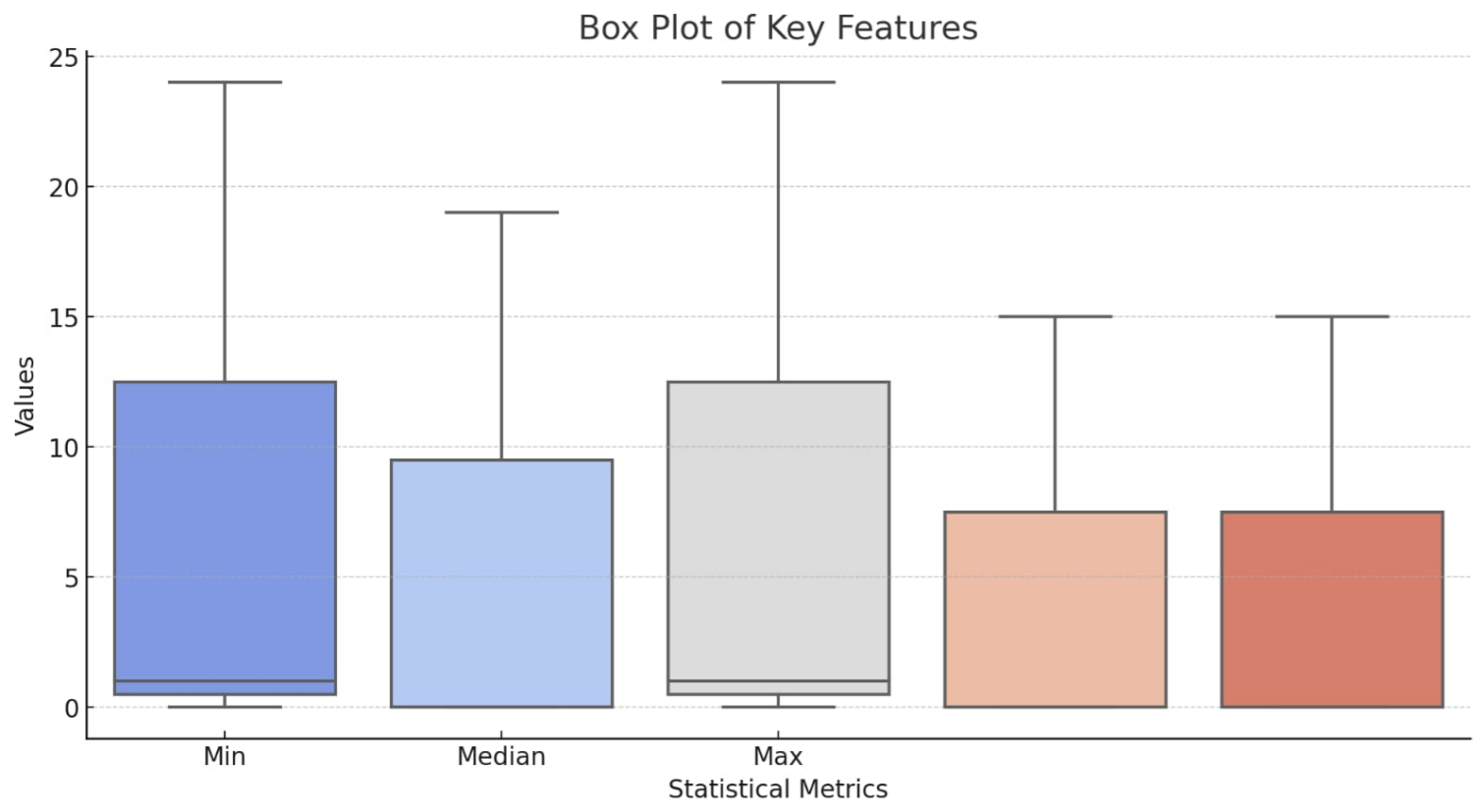
1. Mean values are relatively low compared to their ranges (e.g., maximum values of 24 and 19).
2. The median (50%) values are consistently 1 or 0, indicating skewness in the data, with many features having sparse distributions.

*Features like onus\_attribute\_47 and onus\_attribute\_48:*

1. These have very low mean and median values (both 0).
2. Sparse data with high skewness, as the maximum values (15) are much higher than the interquartile range (IQR: 0 to 0).

This box plot visualizes the statistical distribution (min, median, max) for key features (onus\_attribute\_44, onus\_attribute\_45, etc.).

Each box shows the spread of the feature values, highlighting central tendency (median) and variability.



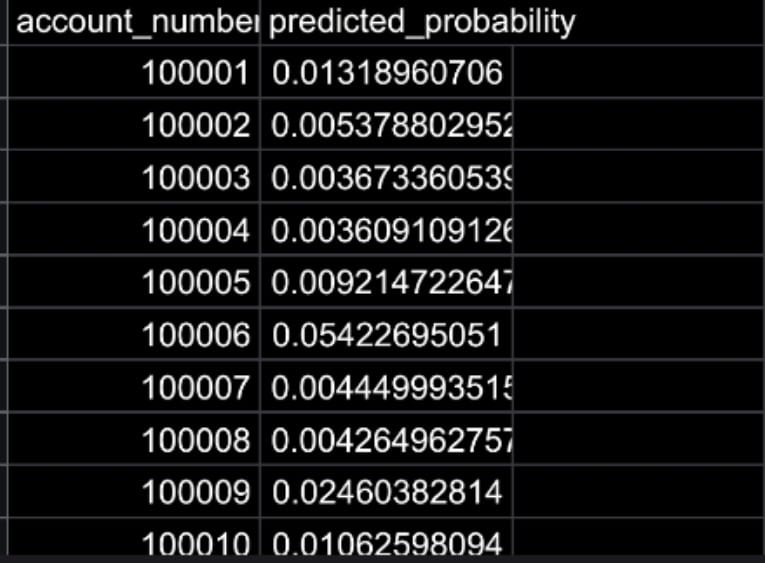
--------------------------------------------------------------------------------------------------------------------------------------

**Statistical Summary Examples**

**Key Insights:**

* High variability in some attributes, as indicated by large standard deviations relative to the mean.
* Many features have a mode of 0 (e.g., onus\_attribute\_47 and onus\_attribute\_48), showing the presence of dominant "null" or non-event values.

**OUTPUT EXAMPLE :**



----------------------------------------------------------------------------------------------------------------------------

**3. Missing Values**

**Extent of Missing Data :-**

* There are 1214 features in total, and many have missing values. For instance:
* onus\_attribute\_1 and transaction\_attribute\_1-4: Missing in 18,528 records.
* onus\_attribute\_44-48: Missing in 62,682 records.

**Key Observations:**

1. Wide-scale missingness suggests possible issues in data collection or feature applicability for certain customers.

2. Variable-Specific Patterns:

- Attributes like onus\_attribute\_44-48 have substantial missing data (~65% missing).

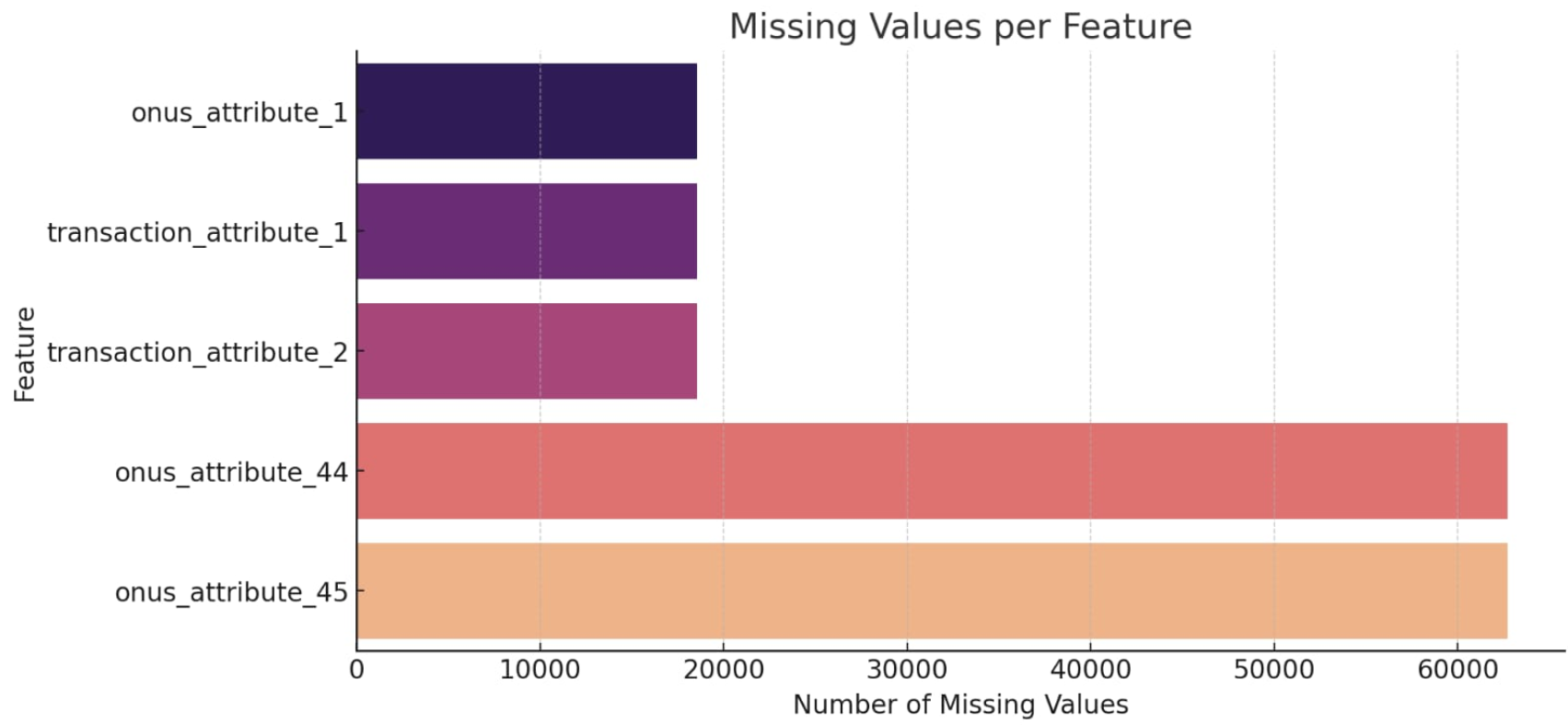
- Some variables may need to be dropped due to excessive missingness, while others can be imputed with mean/median or more sophisticated methods.

**Impact on Model Training:**

* Missing values need careful handling:
* For highly missing features, drop them if they do not add significant value.
* For less missing features, impute using the median for numerical features or mode for categorical ones.

This horizontal bar plot shows the number of missing values for key features.

Features with the highest missing values are shown (e.g., onus\_attribute\_1, transaction\_attribute\_1).



--------------------------------------------------------------------------------------------------------------------------------------

**4. Distribution Analysis**

**Skewness and Sparsity**

Many features have values concentrated around 0, with a long tail toward higher values (e.g., onus\_attribute\_44).

This skewed distribution may lead to:

1. Misrepresentation of feature importance during model training.
2. Need for transformations (e.g., log-scaling) to normalize data.

**Correlation Analysis (if conducted)**

1. Features with high correlation may introduce redundancy.
2. Dimensionality reduction techniques (e.g., PCA, or dropping highly correlated variables) can simplify the dataset.

**5. Class Imbalance**

**Default Rate:**

* The small proportion of defaults (1.41%) requires specialized techniques like:
* Oversampling (e.g., SMOTE): Synthetic generation of minority samples.
* Undersampling: Reduce the majority class size.

**Class Weights:** Adjust algorithm loss functions to penalize misclassification of the minority class.

**Key Insights and Recommendations:**

*1. Handling Missing Values:*

- Drop features with more than 60% missing data (onus\_attribute\_44-48).

- Impute less missing variables with:

- Median for numerical variables.

- Mode for categorical variables.

*2. Address Class Imbalance:*

- Use oversampling (e.g., SMOTE) or class weights during model training.

- Evaluate metrics sensitive to class imbalance (e.g., AUC-ROC, Precision-Recall Curve).

*3. Dimensionality Reduction:*

- Reduce redundancy through feature importance or PCA.

- Focus on highly predictive features (e.g., those with a strong relationship to bad\_flag).

*4. Normalization:*

- Many features exhibit skewness or outliers.

- Apply normalization or log transformations to stabilize variance.

*5. Additional Analysis:*

- Conduct correlation analysis to identify relationships between features.

- Explore feature importance using the Gradient Boosting model to prioritize key predictors.

**Conclusion**

This project aimed to develop a robust Behavioral Score for Bank A's credit card portfolio, predicting the likelihood of customer default. The data-driven approach involved a comprehensive exploration of historical credit card data, addressing critical challenges such as class imbalance, missing values, and outlier handling.

**Key insights derived during the analysis include:**

*1. Class Imbalance:* The dataset exhibits a significant imbalance, with over 98% of customers being non-defaults. This necessitates advanced sampling techniques such as SMOTE and careful selection of evaluation metrics like AUC-ROC and Precision-Recall to ensure the model's reliability.

*2. Data Quality Issues:* Several features have excessive missing values (>60%) and were identified for removal to avoid introducing bias.

- The remaining features with moderate missing values were imputed appropriately using statistical techniques.

*3. Feature Distribution:* Some features exhibited outliers and skewed distributions, requiring transformations and robust scaling for effective model performance.Features with low variance were identified and considered for exclusion to reduce noise.

*4. Modeling Strategy:* A Gradient Boosting Classifier was selected as the baseline model due to its ability to handle non-linear relationships and identify feature importance.A hyperparameter-tuned model will optimize predictive performance, ensuring high generalization.

*5. Evaluation and Validation:* Preprocessed validation data will allow for robust performance evaluation on unseen data.The final Behavioral Score will be used for portfolio risk management activities, enabling early detection of potential defaults and improving profitability for Bank A.

***THANK YOU.***