**Predictive Model Plan**

**1. Model Logic**

The core logic of the predictive model for customer delinquency involves a structured process to transform raw customer data into a risk prediction. Below is a step-by-step description of this process.

**Step-by-Step Process for Delinquency Risk Prediction Model:**

1. **Data Ingestion:** Load the Delinquency\_prediction\_dataset.csv into a data structure (e.g., pandas DataFrame) to begin the analysis.
2. **Initial Data Exploration (EDA):** Perform preliminary checks on the dataset, such as viewing data types, summary statistics, and identifying columns with missing values (e.g., Income, Credit\_Score, Loan\_Balance). Also, identify inconsistencies in categorical features (e.g., Employment\_Status values like 'EMP' and 'employed').
3. **Data Preprocessing:**

* **Handle Inconsistent Categorical Values:** Standardize inconsistent entries within categorical columns (e.g., unifying 'EMP' and 'employed' to 'Employed').
* **Impute Missing Values:** Fill in missing numerical data using appropriate strategies. For this dataset, median imputation was used for Income, Credit\_Score, and Loan\_Balance. For robust analysis, Multiple Imputation by Chained Equations (MICE) is a recommended alternative.
* **Categorical Encoding:** Convert all categorical features (e.g., Employment\_Status, Credit\_Card\_Type, Location, and Month\_1 through Month\_6 payment statuses) into a numerical format using One-Hot Encoding.
* **Feature Scaling:** Standardize all numerical features using a method like StandardScaler to ensure they contribute equally to the model.
* **Feature Engineering (Optional):** Create new, more predictive features from existing ones, such as Total\_Late\_Payments or Consecutive\_Missed\_Payments derived from the monthly payment history.

1. **Handling Class Imbalance:** Address the imbalance in the Delinquent\_Account target variable (where non-delinquent accounts significantly outnumber delinquent ones). This can be done by using class\_weight='balanced' during model training or by employing resampling techniques like SMOTE on the training data.
2. **Data Splitting:** Divide the preprocessed data-set into training and testing sets (e.g., 80% for training, 20% for testing), ensuring the stratification of the target variable to maintain class proportions in both sets.
3. **Model Selection and Training:** Choose a suitable predictive model for binary classification. A **Gradient Boosting Machine (GBM)**, such as XGBoost or LightGBM, is recommended due to its strong performance in tabular data. The chosen model is then trained on the processed training data.
4. **Prediction Generation:** Use the trained model to predict the probability of delinquency for customers in the test set. These probabilities are then converted into binary risk predictions (delinquent/non-delinquent) using a predefined threshold (e.g., 0.5).
5. **Model Evaluation:** Assess the model's performance using a comprehensive set of metrics including accuracy, precision, recall, F1-score, and ROC AUC score, particularly focusing on the minority class. This stage also includes checks for potential biases.

**2. Justification for Model Choice**

For Geldium's goal of accurate delinquency forecasting to minimize financial losses and optimize lending strategies, a **Gradient Boosting Machine (GBM)**, such as XGBoost or LightGBM, is the recommended model choice. While simpler models like Logistic Regression offer high interpretability and ease of implementation, their limited ability to capture complex, non-linear relationships, as evidenced by our previous model's poor performance (ROC AUC of 0.440), makes them inadequate for Geldium's needs. Decision Trees, while providing interpretable rules, can be prone to overfitting and may not achieve the highest accuracy on their own. GBMs, conversely, excel at identifying intricate patterns in tabular data, often leading to superior predictive accuracy crucial for effective risk management. Although they are less inherently interpretable than individual decision trees or logistic regression, their robust performance in identifying high-risk customers will directly support Geldium in minimizing financial losses, allowing for more precise risk assessments that balance the need for performance with the ability to infer feature importance through techniques like SHAP values.

**3. Evaluation Strategy**

My evaluation strategy for this financial risk prediction model encompasses a comprehensive set of metrics designed to assess accuracy, robust performance for imbalanced data, and crucial fairness considerations.

**Accuracy Metrics (Overall Model Performance):**

* + **Accuracy Score:** Calculated as (TP+TN)/(TP+TN+FP+FN), it represents the proportion of correctly predicted instances out of the total. However, for imbalanced datasets, this metric can be misleading.
  + **Confusion Matrix:** A table summarizing prediction results into True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). This helps in understanding the types of errors the model makes.

**Robust Performance Metrics (for Imbalanced Data):** These are vital given the imbalanced nature of delinquency data, where the cost of different error types varies.

* + **Precision (Positive Predictive Value):** Calculated as TP/(TP+FP), it indicates the proportion of positive predictions that were actually correct. High precision is important when minimizing false alarms (e.g., wrongly flagging customers as high risk) is critical.
  + **Recall (Sensitivity, True Positive Rate):** Calculated as TP/(TP+FN), it indicates the proportion of actual positive cases (delinquent customers) that were correctly identified. High recall is crucial when minimizing missed high-risk cases is important to prevent financial loss.
  + **F1-Score:** The harmonic mean of precision and recall, providing a single score that balances both metrics and is particularly useful for uneven class distributions.
  + **ROC AUC Score (Receiver Operating Characteristic - Area Under the Curve):** Measures the model's ability to discriminate between the positive and negative classes across various thresholds. An AUC of 0.5 indicates random performance, while 1.0 is perfect discrimination.

**Fairness and Bias Checks:**

* + **Disproportionate Impact Assessment:** Analyze key performance metrics (Precision, Recall, False Positive Rate, False Negative Rate) for different subgroups defined by features like Employment\_Status, Location, or Credit\_Card\_Type. This helps identify if certain groups are unfairly over-flagged as high risk (higher False Positive Rate) or disproportionately missed (higher False Negative Rate).
  + **Interpreting Coefficients/Feature Importance for Bias:** For interpretable models like Logistic Regression, coefficients show direct influence. For complex models, tools like SHAP values or LIME can explain feature contributions to individual predictions, which can then be aggregated for group-level bias assessment.
  + **Ethical Considerations:** Recognize that even without explicit protected attributes (e.g., race, gender), features used in the model might serve as proxies, leading to indirect bias. The goal is to ensure predictions are not only accurate but also equitable, avoiding situations where specific groups are disproportionately impacted by unfavorable predictions, especially in sensitive financial decisions.
  + **Bias Mitigation Planning:** If bias is detected, plan to employ mitigation techniques such as resampling (e.g., FairSMOTE), reweighting data points, or post-processing prediction thresholds for specific subgroups to achieve fairness goals.