**Problem Statement 2: Real-Time Credit Risk Assessment Using Alternative Data**

**Team : The Boys**

**Madhav Kumar**

**Shashidhar R**

**Adhithiya M S**

**Nitheesh Prakash JD**

**1. Combine and Prepare the Dataset**

The first step is gathering data from multiple sources and preparing it for machine learning.

1. **Combining Data**: Often, credit risk prediction models combine different types of data, such as traditional financial data (income, credit history) and alternative data sources (social media sentiment, utility payments, etc.). It's crucial to ensure that these different data sources are combined in a consistent format (e.g., merging datasets using a unique ID for customers or transactions).
   1. **Example**: If you have social media sentiment in textual form and financial data as numeric, you'll need to preprocess both and align them based on a common attribute (e.g., customer ID).
2. **Data Preprocessing**: This includes transforming the data into formats suitable for machine learning. This may involve:
   1. **Feature Engineering**: Creating new features from the existing ones, such as aggregating transaction history into a monthly spending pattern.
   2. **Data Cleaning**: Removing or correcting inconsistencies in the dataset (e.g., handling duplicates, or correcting invalid data points).

**2. Feature Selection**

Feature selection is crucial in reducing the dimensionality of your dataset and selecting the most important features for model performance.

1. **Heatmap for Correlation**: A **correlation matrix** is a way to measure the relationship between different numerical features. A **heatmap** is a visual representation of this matrix, which helps in spotting highly correlated features.
   1. **Multicollinearity**: If two features are highly correlated (e.g., 0.9 or more), keeping both might result in **multicollinearity**, which can hurt certain models (e.g., linear regression). Ideally, you should drop one of the correlated features.
2. **Dimension Reduction**: This involves reducing the number of features while retaining as much information as possible.
   1. **PCA (Principal Component Analysis)**: PCA is a technique to reduce the number of dimensions (features) by transforming the data into principal components (linear combinations of the original features). PCA helps in reducing noise and multicollinearity, but some interpretability might be lost as the new features (principal components) are combinations of the originals.
   2. **RFE (Recursive Feature Elimination)**: RFE is a method of feature selection where features are recursively removed based on their importance, and the model is retrained to determine the most significant features.
3. **Feature Importance**: Once dimensionality is reduced, you can use techniques like **Random Forest**, **XGBoost**, or **Lasso regression** to rank the importance of features. This can help in selecting the most significant features for predicting credit risk.

**3. Data Preprocessing**

Data preprocessing ensures that the data is in a suitable format and scale for machine learning algorithms.

1. **Encoding**: Machine learning models often require numerical input, but features like categorical data (e.g., customer region, social media sentiment) need to be encoded into numbers.
   1. **One-Hot Encoding**: This creates binary columns for each category. For example, if a categorical variable "Region" has three categories: "North", "South", and "East", one-hot encoding would create three new columns (North, South, East), with 0 or 1 indicating presence.
   2. **Label Encoding**: This method assigns a unique integer to each category. For instance, "North" could be 0, "South" could be 1, and "East" could be 2.
2. **Numerical Features**: Ensure that all numerical features are properly formatted, for example converting integer values to float when required.
3. **Normalization**: Many machine learning models (especially distance-based models like k-NN, and gradient descent-based models like logistic regression) work better when the features are on the same scale.
   1. **StandardScaler**: Standardizes features by subtracting the mean and dividing by the standard deviation, resulting in features with a mean of 0 and a standard deviation of 1.
   2. **MinMaxScaler**: Scales features to a range between 0 and 1.
4. **Handling Missing Values**:
   1. **Imputation**: If some values are missing, you can use imputation techniques to fill them in. Common methods include using the mean, median, or mode for numerical data, or predicting missing values using algorithms like **KNN imputation**.

**4. Train-Test Split & Cross-Validation**

1. The goal of splitting data into training and testing subsets is to assess model performance while avoiding overfitting.
2. **Data Split**: The dataset is usually split into training and test sets, commonly in a 70/30 or 80/20 ratio. The **test set** is used to evaluate the model's generalization ability, while the **training set** is used to train the model.
   1. **Stratified Split**: In classification tasks, ensure that the class distribution is maintained across both training and test sets.
3. **K-Fold Cross-Validation**: Instead of relying on a single train-test split, **K-Fold Cross-Validation** splits the data into **K** subsets (folds). The model is trained on K-1 folds and tested on the remaining fold. This is repeated K times, and the results are averaged to reduce variance and ensure the model generalizes well.

**5. Hyperparameter Tuning**

1. Hyperparameter tuning is about finding the best set of parameters for your model to achieve optimal performance.
2. **Grid Search**: Exhaustively tests a predefined set of hyperparameters for each model and evaluates performance for each combination. This is computationally expensive but thorough.
3. **Randomized Search**: Instead of testing every combination, it samples random combinations of hyperparameters. This is faster than Grid Search, especially for large parameter spaces.
4. **Optuna**: Optuna uses **Bayesian optimization**, a more advanced technique to explore the hyperparameter space more efficiently by focusing on promising areas.

**6. Model Training and Evaluation**

1. Once the data is pre-processed and the hyperparameters are tuned, you can train the model.
2. **Train Models**: Fit the model to your training data using the optimal hyperparameters.
3. **Evaluation Metrics**: The choice of evaluation metrics depends on whether the task is classification or regression:
   1. **Classification**: Metrics like **accuracy**, **precision**, **recall**, **F1-score**, and **AUC-ROC** are used to evaluate how well the model is classifying credit risk.
   2. **Regression**: Metrics like **RMSE** (Root Mean Squared Error), **MAE** (Mean Absolute Error), and **R-squared** are used to assess how well the model is predicting continuous credit scores.

**7. Final Prediction**

The trained model is then used to make predictions on unseen data.

1. **Class Label (Classification)**: If it's a classification problem (e.g., predicting whether a person will default on a loan), the model predicts a **creditworthiness class** (e.g., "Low Risk" or "High Risk").
2. **Regression Prediction**: If it's a regression problem (e.g., predicting a customer's credit score), the model predicts a **continuous value** like a FICO score or a custom credit score.