Spring A 2025

CIS 508

Group Project Instructions

**Problem Definition:**

You have been engaged as a consultant by BestCard, a credit card company, to analyze a dataset they have provided. This dataset encompasses demographic and recent financial information for a sample of 30,000 of their account holders. The data is organized at the individual credit account level, ensuring each account is uniquely represented by a single row within the dataset. Each row is annotated to indicate whether the account owner defaulted in the subsequent month following a six-month historical data review period. Defaulting, in this context, means failing to meet the minimum payment requirement.

**Goal:**

Your primary objective is to construct a predictive model to accurately forecast the likelihood of an account defaulting in the next month.

The dataset you will be working with is a modified version of the dataset available from the UCI Machine Learning Repository. For detailed information on the dataset, including the data dictionary, please refer to the following link: [UCI Machine Learning Repository - Default of Credit Card Clients Dataset](https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients).

**Deliverables:**

**Python Code Documentation**: Please provide the entire Python code used for your analysis and detailed annotations for each section. These annotations must clearly describe the purpose and expected results of each part of the code. Ensure that all code is fully executable, as it will undergo testing on my system. Non-executable lines will incur a penalty of 5 points per line. [Maximum points: 30]

**Presentation Deck**: Prepare a PowerPoint presentation (up to 15 slides, with the first slide detailing the project title and all team members' names) that briefly outlines your analytical approach and findings. This presentation will be used to discuss your project during the final class meeting of the semester. Each team member must **be present and actively participate in the presentation to be eligible for credits.** [Maximum Points: 70]

**Analysis Process:**

**Data Exploration**

* Determine the total number of columns in the dataset.
* Identify the total number of samples (rows) provided.
* Classify the types of features included in the dataset (e.g., numerical, categorical).
* Examine and describe the general appearance of data within these features.
* Check for and document any instances of missing data.

**Data Cleaning**: Outline the steps taken to prepare the data for analysis, including handling missing values, removing duplicates, and any transformations applied.

**Model Building**: Describe the process of selecting and constructing the predictive models, including any algorithms used and the rationale behind their selection.

**Model Comparison**: Compare the performance of the developed models based on relevant metrics (e.g., accuracy, precision, recall) to determine the most effective model.

**Model Deployment & Strategy**: Use Streamlit to deploy your final model and then develop a plan for implementing it in a real-world environment, outlining the necessary steps for its integration and ongoing monitoring and evaluation to ensure its continued effectiveness.

**Conclusion**: Summarize the key findings from your analysis, the effectiveness of the predictive model, and any recommendations or insights for BestCard regarding future defaults.

Please ensure that your analysis is thorough and well-documented, as these factors will be crucial for both the grading of your project and the practical application of your findings by BestCard.

**1. Ratio of Previous Payments to Bill Statement**

* **Concept:**
  + This ratio helps understand how well a customer is covering their bills. A high ratio suggests they're paying off a significant portion, while a low ratio might indicate financial strain.
* **Calculation:**
  + For each of the six payment and bill statement periods (X12-X17 and X18-X23), calculate:
    - Payment Ratio = Previous Payment / Bill Statement
  + For example:
    - Payment Ratio 1 = X18 / X12
    - Payment Ratio 2 = X19 / X13
    - And so on.
  + It is very important to handle cases where the bill statement amount is zero. If a bill statement is zero, you will have a divide by zero error. You could handle this in a couple of ways.
    - You could set the ratio to zero when the bill statement is zero.
    - Or you could add a very small number to the denominator to prevent the divide by zero error.
  + You can then use these individual ratios or create an average payment ratio.

**2. Additional Feature Creation**

* **Payment Difference:**
  + Calculate the difference between each bill statement and the corresponding previous payment:
    - Payment Difference = Bill Statement - Previous Payment
  + This reveals how much of the bill remains unpaid.
* **Payment History Patterns:**
  + The "history of past payment" features (X6-X11) are categorical. You could:
    - Count the number of times a customer had a late payment.
    - Create a binary feature indicating if a customer had any late payments.
    - Convert the categorical payment history into a numerical value. For example, you could assign a numerical value to each category, representing the number of months of delayed payment.
* **Credit Utilization Ratio:**
  + Calculate the ratio of the latest bill statement (X12) to the credit limit (X1):
    - Credit Utilization = X12 / X1
  + This shows how much of their available credit a customer is using.
* **Age Groups:**
  + Instead of using the raw age (X5), create age groups (e.g., 20-30, 31-40, etc.). This can sometimes improve model performance.

**3. Average Payment Delay**

* **Concept:**
  + To create a feature that represents the average payment delay, you'll need to interpret the payment history features (X6-X11). These features typically represent payment statuses for past months.
* **Implementation:**
  + **Interpreting Payment History:**
    - The values in X6-X11 usually correspond to payment statuses (e.g., -1 = pay duly, 1 = payment delay for one month, 2 = payment delay for two months, etc.).
  + **Calculation:**
    - For each customer, extract the payment delay value from each of the X6-X11 columns.
    - Calculate the average of these delay values.
    - For example if a customer had the following values for X6-X11. (-1, 1, 2, 0, 0, -1) the average payment delay would be the average of those numbers.
* **Important Notes:**
  + Ensure you understand the exact encoding of the payment history features in your dataset.
  + Consider how to handle missing or unusual values.

By creating these new features, you can provide your machine learning model with richer information, potentially leading to more accurate predictions of credit card defaults.

The "PAY\_X" columns in the dataset (where X is 1 through 6) represent the repayment status of an individual over the past six months, where the numbers likely indicate the following:

* **-2**: No consumption or balance cleared in advance; no amount due.
* **-1**: Paid duly; the full balance was paid on time.
* **0**: The use of a revolving credit; at least the minimum was paid, but the full amount was not cleared.
* **1**: Payment delay for one month.
* **2**: Payment delay for two months.
* ... and so on for greater payment delays.

The sequence of numbers for each row indicates the payment history of an individual over the last six months. For example:

* A row with "2, 2, -1, -1, -2, -2" means the individual delayed payment for two months and then had no amount due or paid the balance in advance for the following months.
* A row with "0,0,0,0,0,0" indicates that the individual has been using revolving credit consistently for the past six months without any delay
* A row, "-1, -1, -1, -1, -1, -1," suggests the individual paid the due amount fully each month.

Also, consider the following feature engineering in the model.

* + Credit Utilization = BILL\_AMT6 / LIMIT\_BAL
  + Repayment Ratio = (sum(PAY\_AMT1-6)) / LIMIT\_BAL
  + Payment Behavior Trend = PAY\_1 - PAY\_6
* Feature Scaling: Normalize LIMIT\_BAL, BILL\_AMT1-6, and PAY\_AMT1-6 to ensure better model convergence (use StandardScaler or MinMaxScaler).

Tackling imbalanced data is crucial for building accurate predictive models, especially in scenarios like credit card default prediction where defaults are far less frequent than non-defaults. Here's a breakdown of common techniques:

**1. Resampling Techniques:**

* **Oversampling:**
  + Increases the number of instances in the minority class (defaults) to balance the class distribution.
  + **Random Oversampling:** Duplicates random samples from the minority class. Can lead to overfitting.
  + **SMOTE (Synthetic Minority Over-sampling Technique):** Generates synthetic samples for the minority class by interpolating between existing samples. More robust than random oversampling.
  + **ADASYN (Adaptive Synthetic Sampling):** Similar to SMOTE but generates more synthetic samples for minority class instances that are harder to learn.
* **Undersampling:**
  + Reduces the number of instances in the majority class (non-defaults) to balance the class distribution.
  + **Random Undersampling:** Randomly removes samples from the majority class. Can lead to loss of valuable information.
  + **Cluster Centroids:** Replaces clusters of majority class samples with their centroids.
  + **Tomek Links:** Removes pairs of instances that are close to each other but belong to different classes.

**2. Cost-Sensitive Learning:**

* Assigns different misclassification costs to different classes.
  + Gives higher penalties for misclassifying the minority class (defaults).
  + Many machine learning algorithms (e.g., support vector machines, decision trees) allow you to specify class weights.
  + This allows the model to "pay more attention" to the minority class.

**3. Ensemble Methods:**

* **Ensemble techniques** can be used to mitigate the effects of imbalanced datasets.
  + **Balanced Bagging:** Creates multiple subsets of the data with balanced class distributions and trains a model on each subset.
  + **EasyEnsemble and BalanceCascade:** Train multiple models on different undersampled subsets of the majority class.
  + **Boosting Algorithms:** Algorithms like XGBoost, LightGBM, and CatBoost have built-in mechanisms to handle imbalanced data. They can assign higher weights to misclassified minority class instances.

**4. Evaluation Metrics:**

* Don't rely solely on accuracy. Accuracy can be misleading in imbalanced datasets.
* Use metrics that are more sensitive to minority class performance:
  + **Precision:** Measures the proportion of correctly predicted defaults out of all predicted defaults.6
  + **Recall (Sensitivity):** Measures the proportion of correctly predicted defaults out of all actual defaults.
  + **F1-score:** The harmonic mean of precision and recall.
  + **Area Under the ROC Curve (AUC-ROC):** Measures the model's ability to distinguish between classes.
  + **Area Under the Precision-Recall Curve (AUC-PR):** More suitable for highly imbalanced datasets.

**5. Data Augmentation:**

* If possible, collect more data for the minority class.
* Generate synthetic data based on domain knowledge.

**6. Algorithm Selection:**

* Some algorithms are more robust to imbalanced data than others.
* Tree-based algorithms (e.g., random forests, gradient boosting) tend to perform well.

**Practical Tips:**

* Start with simple techniques like SMOTE or cost-sensitive learning.
* Experiment with different resampling ratios.
* Validate your model's performance using appropriate evaluation metrics.
* Don't oversample or undersample excessively, as this can lead to overfitting or loss of information.
* Always split the data into train and test sets *before* performing any resampling. Resampling after splitting can cause data leakage, and cause a model to perform much worse on unseen data.