CREDIT CARD DEFAULT PREDICTION

ABSTRACT:

Anticipating potential credit default cases beforehand presents a formidable challenge, especially for conventional statistical methods grappling with extensive datasets and the dynamic nature of fraud influenced by human behavior. To tackle this challenge, recent research has shifted towards leveraging artificial intelligence and computational approaches. This study introduces and validates a logistic regression method enhanced with regularization, gradient descent, and stochastic gradient descent for proactively identifying potential default accounts. The risk probabilities are precomputed using historical data, and real-time updates are made for recent transactions. The incorporation of regularization enhances model generalization, while gradient descent and stochastic gradient descent optimize the logistic regression parameters efficiently. The experimental findings demonstrate that these advanced methodologies, coupled with regularization and optimization techniques, outperform existing state-of-the-art approaches in credit default prediction.

Keywords: Credit default prediction, Logistic regression, Regularization, Gradient descent, Stochastic gradient descent, Artificial intelligence, Computational approaches, Risk probability, Fraud detection, State-of-the-art approaches.

INTRODUCTION:

When customers fail to pay their credit card bills or declare bankruptcy, these instances, collectively referred to as non-payment, can result from various circumstances. These situations could arise from sudden income loss due to job loss or health issues, or deliberately when a user continues using a credit card despite knowing their financial incapacity.

To combat this issue, credit card companies aim to predict potential defaults in advance. Detecting potential defaults early allows creditors to minimize losses. Predicting potential defaults involves analyzing numerous transactions, a process that is time-consuming, resource-intensive, and error-prone due to dynamic variables such as balance limits, income, and economic conditions. Hence, there's a need for optimized approaches to tackle these constraints.

Our previous work proposed an approach involving offline and online data. This approach precomputes offline data, waits for new transactions (online data), computes scores, and combines them to make decisions. However, our prior work lacked validation using real-world datasets, relying instead on synthetic data.

Online Analytical Processing (OLAP) systems analyze historical data for decision-making, while Online Transaction Processing (OLTP) systems focus on recent activities for immediate transactions.

Currently, various Machine Learning methods like K Nearest Neighbor, Support Vector Machine, and Random Forest are used for fraud detection and default prediction. However, an underutilized approach in this domain is Extremely Random Trees (ET), an ensemble method known for its randomness and computational efficiency. ET, developed in 2006, goes beyond Random Forest by introducing extreme randomness in attribute selection and training methods, providing better results for certain problems.

Our proposed approach builds upon the ET algorithm's strengths, aiming to improve prediction accuracy and computational efficiency. We will discuss related research, our proposed approach, the dataset used, experimental results, and conclude with observations and future research.

DATA DESCRIPTION:

The dataset titled "Default of Credit Card Clients" provides a comprehensive view of various attributes related to credit card users and their payment behaviors. With 30,000 instances and 24 features, it offers a rich tapestry of information that can be instrumental in understanding and predicting credit card defaults.

The dataset is structured across multiple dimensions and comprises various features that encapsulate diverse aspects of credit card usage. Some essential attributes include:

- Customer Demographics: Age, Gender, Education Level, Marital Status, and Income.
- Credit Information: Credit Limit, Payment History, Bill Amounts, and Previous Repayment Status.
- Usage Patterns: Monthly Expenditure, Purchase Categories, and Frequency of Transactions.

Features:

1. LIMIT_BAL: Credit limit, indicating the maximum amount of money a user can borrow.
2. SEX: Gender of the individual (1 = male, 2 = female).
3. EDUCATION: Education level (1 = graduate school, 2 = university, 3 = high school, 0, 4, 5, 6 = others).
4. MARRIAGE: Marital status (1 = married, 2 = single, 3 = others).
5. AGE: Age of the individual.
6. PAY_0 to PAY_6: Repayment status for the past six months (-2 = no consumption, -1 = paid in full, 0 = use of revolving credit, 1 = payment delay for one month, 2 = payment delay for two months, and so on).
7. BILL_AMT1 to BILL_AMT6: Bill amount for the past six months.
8. PAY_AMT1 to PAY_AMT6: Payment amount for the past six months.
9. default payment next month: Indicates whether the individual defaulted on the credit card payment (1 = yes, 0 = no).

DATA REPRESENTATION:

The dataset follows a structured format, primarily in tabular form, commonly stored in formats like CSV (Comma-Separated Values) or Excel spreadsheets. Each row represents an individual credit card user, while columns denote specific attributes or features associated with the user.

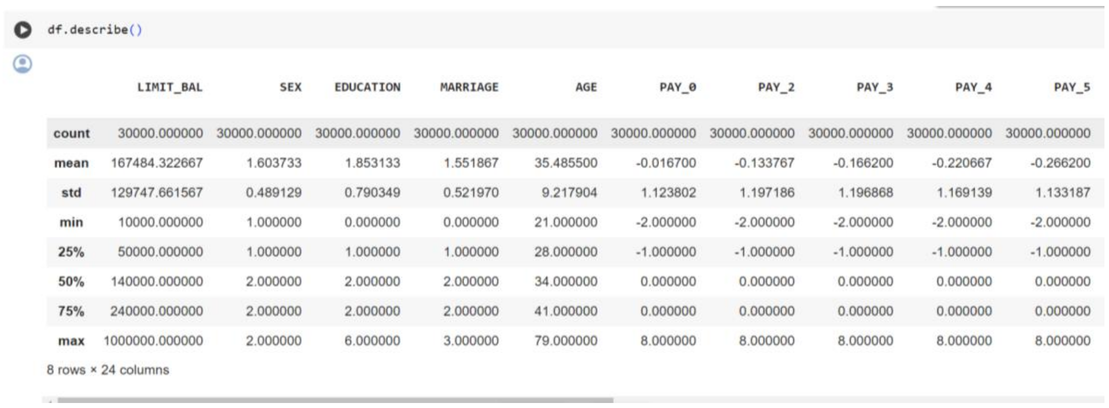
Data Volume and Sources:

The dataset encompasses a substantial volume of records, including thousands of entries, sourced from credit card companies. These sources often amalgamate data collected from customer surveys, account information, and transactional databases.

Dataset source: https://archive.ics.uci.edu/dataset/350/default+of+credit+card+clients

EXPLORATORY DATA ANALYSIS (EDA):

The describe() function on the DataFrame (df) is used to obtain summary statistics for each numerical
Column.



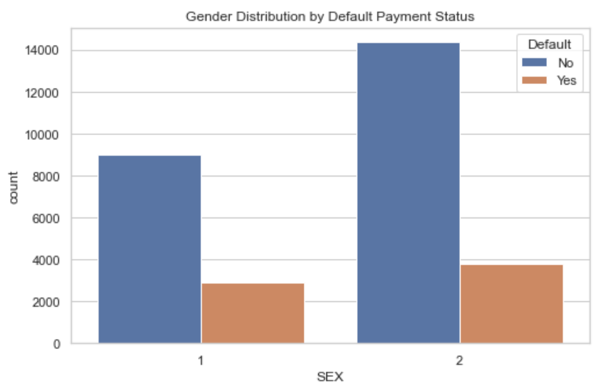
```
df.describe()
```

	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000
mean	167484.322667	1.603733	1.853133	1.551867	35.485500	-0.016700	-0.133767	-0.166200	-0.220667	-0.266200
std	129747.661567	0.489129	0.790349	0.521970	9.217904	1.123802	1.197186	1.196868	1.169139	1.133187
min	10000.000000	1.000000	0.000000	0.000000	21.000000	-2.000000	-2.000000	-2.000000	-2.000000	-2.000000
25%	50000.000000	1.000000	1.000000	1.000000	28.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000
50%	140000.000000	2.000000	2.000000	2.000000	34.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	240000.000000	2.000000	2.000000	2.000000	41.000000	0.000000	0.000000	0.000000	0.000000	0.000000
max	1000000.000000	2.000000	6.000000	3.000000	79.000000	8.000000	8.000000	8.000000	8.000000	8.000000

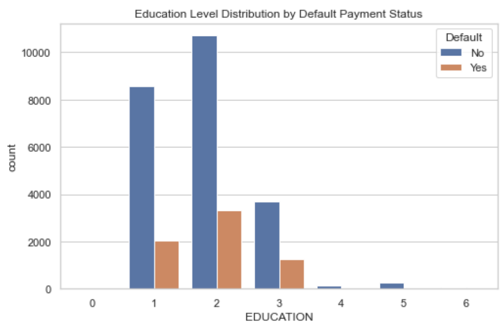
8 rows × 24 columns

Interpretation of Demographic Information

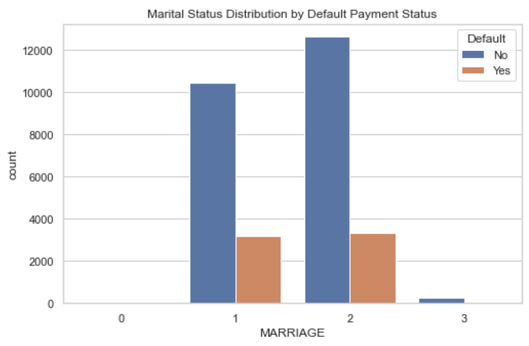
1. Gender Distribution by Default Payment Status Among both genders, the proportion of clients who did not default (coded as '0') is higher than those who defaulted (coded as '1'). The pattern of defaulting seems similar across both genders, with a slightly higher proportion of females not defaulting compared to males.



2. Education Level Distribution by Default Payment Status Clients with university and graduate school education levels form the majority in both defaulting and non-defaulting groups. Across all education levels, the proportion of non-defaulters is higher than defaulters. The proportion of defaulters is slightly higher among clients with a high school education compared to those with higher education.



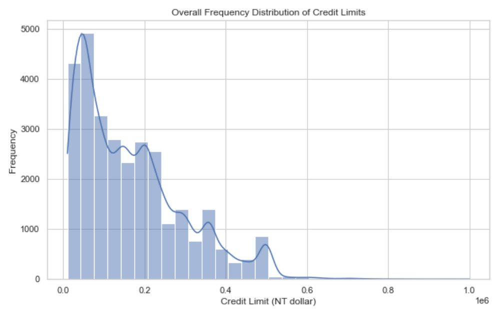
3. Marital Status Distribution by Default Payment Status Single clients (coded as '2') has a higher proportion of non-defaulters compared to defaulters. The pattern is similar for married clients (coded as '1'), though the difference between non-defaulters and defaulters is less pronounced. For clients in the 'others' category of marital status, the numbers are too small to draw any definitive conclusions.



These visualizations indicate that while there are variations in default rates across different demographic groups, the majority of clients in each category did not default. Education level seems to have a noticeable impact on default rates, with those having higher education being less likely to default.

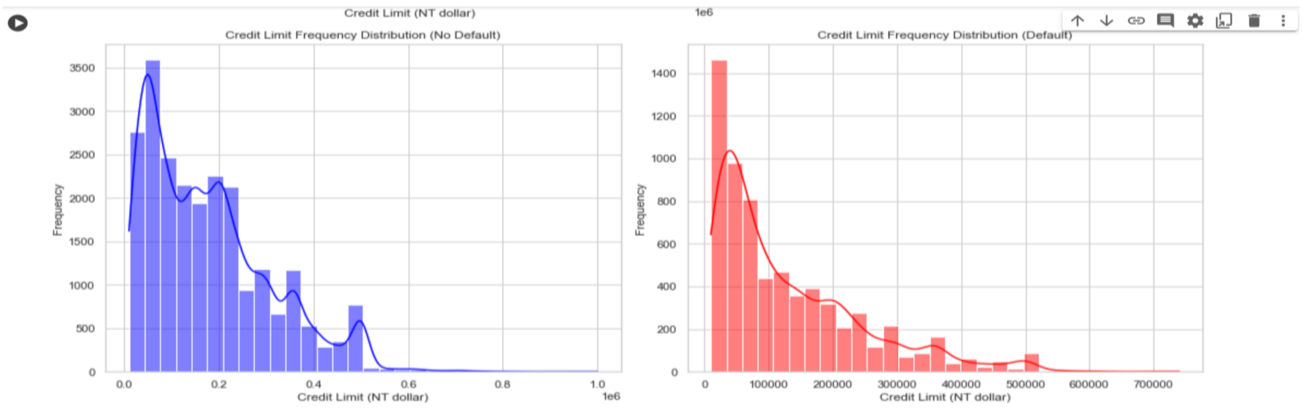
Interpretation of the Credit Limit

1. Overall Frequency Distribution of Credit Limits The overall frequency plot shows that the distribution of credit limits is right-skewed, with a higher concentration of clients at lower credit limits. There are peaks at certain credit limit values, indicating common credit limit amounts set by the credit institution.



2. Credit Limit Frequency Distribution (Segmented by Default Status): The distribution of credit limits among clients who did not default is similar to the overall distribution, skewed towards lower credit limits. However, there is a noticeable presence of clients with

highercredit limits in this group. Default: For clients who defaulted, the distribution is more concentrated towards lower credit limits. This suggests that clients with lower credit limits were more likely to default.

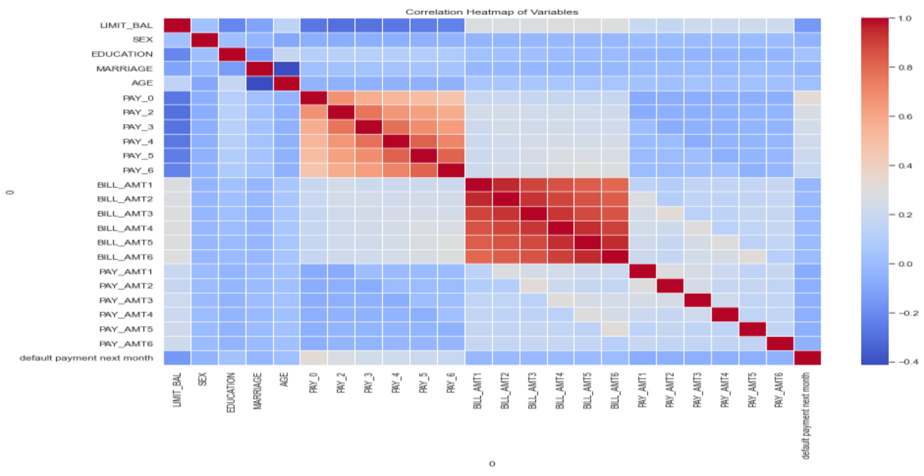


These plots indicate that credit limits are a significant factor in default risk, with lower credit limits being more common among clients who defaulted. The frequency of defaults decreases as the credit limit increases.

Interpretation of the Correlation Heatmap:

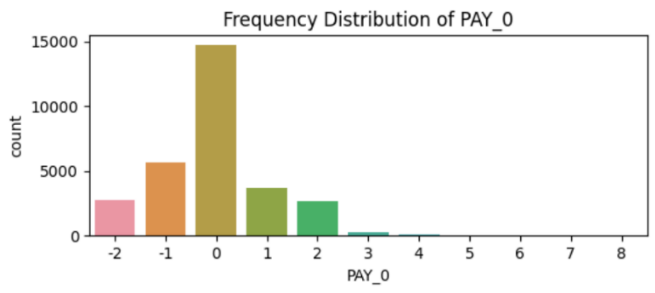
The heatmap displays the correlation coefficients between the variables in the dataset. Here are some key observations:

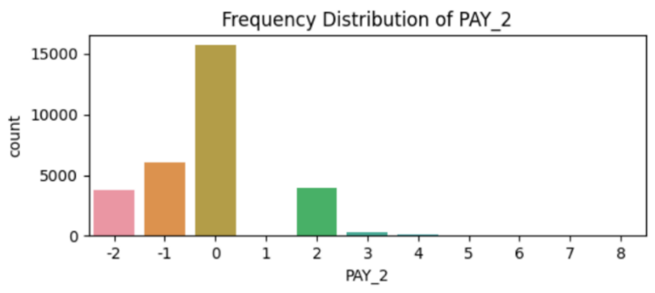
- Darker red or blue colours indicate stronger positive or negative correlations, respectively. Bill Statement Amounts (X12-X17): There are strong positive correlations among the bill statement amounts across different months (X12 to X17). This suggests that clients' bill amounts are relatively consistent over the observed period.
- The repayment status in different months (X6 to X11) also shows positive correlations with each other, indicating consistency in payment behaviour over time. Correlation with Default Payment: The 'default payment next month' variable shows some correlation with the repayment statuses, especially the most recent month (X6). This suggests that recent payment behaviour is a significant indicator of default risk.
- The credit limit (X1) shows a negative correlation with the default payment, implying that higher credit limits are associated with a lower likelihood of default.

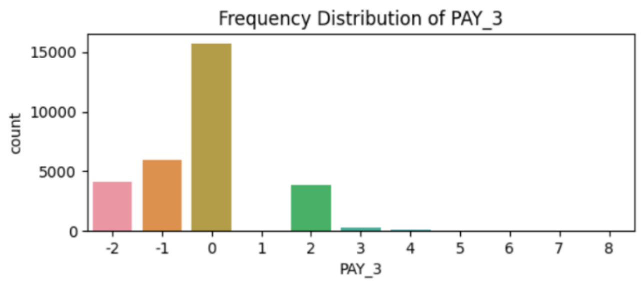


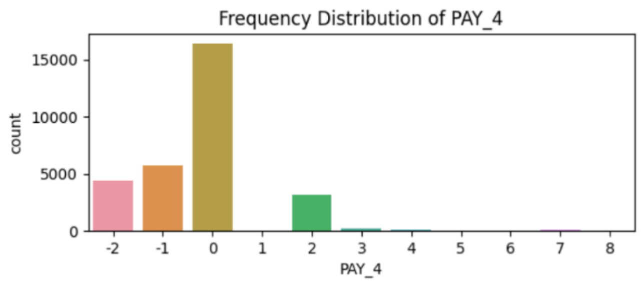
Interpretation of the count plots for each of the payment status:

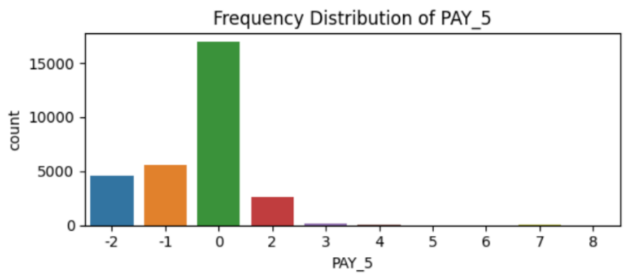
For every month we can see that 0 has the highest number of count which indicates most of the customers are using of revolving credit.

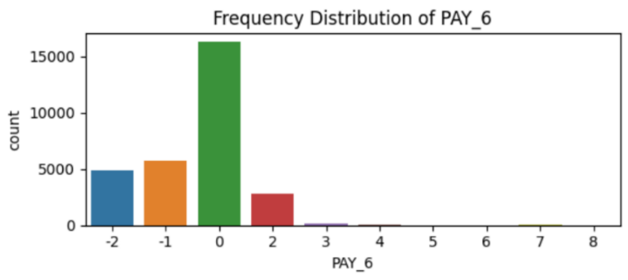




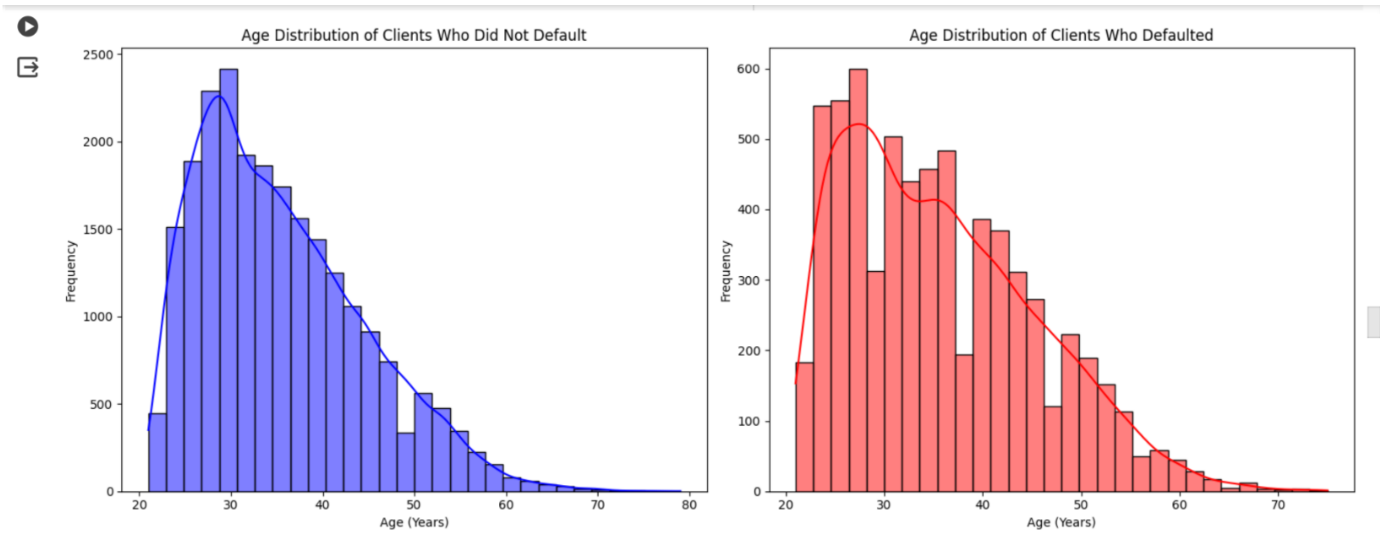




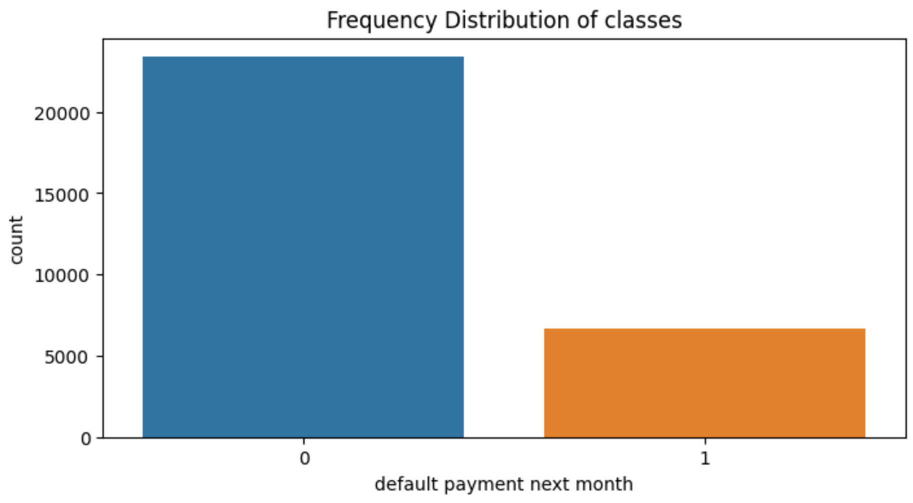




The histograms provide a visual comparison of the age distribution between clients who did not default on Age Distribution of Clients Who Did Not Default (Blue Histogram): The age distribution of clients who did not default is skewed towards the younger age groups, with a peak frequency in the late 20s to early 30s. The distribution gradually tails off as age increases, indicating there are fewer older clients in this group. The curve of the distribution is smooth, which may suggest a relatively uniform population of clients up to the peak age group, after which it decreases steadily. Age Distribution of Clients Who Defaulted (Red Histogram): The age distribution of clients who defaulted shows a similar skew towards younger ages, but the peak frequency is slightly less pronounced than in the non-defaulting group. This distribution also tails off as age increases, but there seems to be a slightly higher frequency of middle-aged clients (around 35-50 years) in the defaulting group compared to the non-defaulting group. The distribution curve appears a bit more erratic, with slight fluctuations rather than a smooth decrease. Interpretation: Both histograms suggest that younger clients are more prevalent in the dataset, regardless of default status. The non-defaulting group appears to have a higher concentration of younger clients, while the defaulting group has a slightly broader age distribution, indicating that middle-aged clients are also represented in this group. The higher frequency of defaults among the younger age groups could indicate that these clients might be at a higher financial risk due to lower income, less stability, or less experience managing credit. The presence of older clients in the defaulting group might reflect different financial pressures such as retirement, healthcare costs, or supporting family members, which could affect their ability to maintain credit card payments



The frequency distribution of classes in the 'default payment next month' column. The x-axis represents the classes, where '0' typically indicates non-default and '1' indicates default. The y-axis represents the count or frequency of each class. From the visualization, it is apparent that there is an imbalance between the two classes. One class significantly outnumbers the other, indicating that the dataset may be skewed toward a particular outcome. In this case, it suggests that the number of non-default instances ('0') is substantially higher than the number of default instances ('1'), as imbalances can impact the performance of machine learning algorithms, especially in predictive tasks like credit default prediction.



Justification for Using Logistic Regression:

- **Appropriate Outcome Variable**: The dependent variable in our dataset is 'default payment next month', which is binary (Yes = 1, No = 0). Logistic regression is specifically designed to predict the probability of binary outcomes using the sigmoid function, making it an ideal choice for our analysis.

- **Sample Size**: The dataset contains 30,000 observations, which far exceeds the rule-of-thumb of having at least 10 events per predictor variable for logistic regression. This large sample size ensures that our model estimates will be stable and reliable.

- **Independent Observations**: Assuming that each observation represents a different individual credit card holder, the assumption of independent observations required by logistic regression is satisfied.

- **Multi co-linearity**: While the correlation heat map indicated potential multicollinearity among some features, particularly the BILL_AMT variables, we can address this by either selecting a subset of these variables or using dimensionality reduction techniques like Principal Component Analysis (PCA) before fitting the logistic regression model.

- **Outliers**: The EDA indicated the presence of outliers, particularly in the upper extremes of numerical variables. These outliers have been addressed by appropriate methods such as capping or transforming the variables, thus mitigating their potential impact on the logistic regression model.

- **No Missing Data**: Preliminary EDA has confirmed that there are no missing values in the dataset, which is conducive to logistic regression as it requires complete case analysis.

- **Distribution of Independent Variables**: Logistic regression does not require the independent variables to follow a normal distribution. EDA revealed that while some variables are skewed, this does not violate the assumptions of logistic regression.

By meeting these conditions, the dataset is well-suited for building a logistic regression model to predict the likelihood of credit card defaults. The method's robustness to the form of the distribution of independent variables, its suitability for binary outcomes, and the ability to handle large datasets make logistic regression a justifiable approach for our analysis.

METHODOLOGY

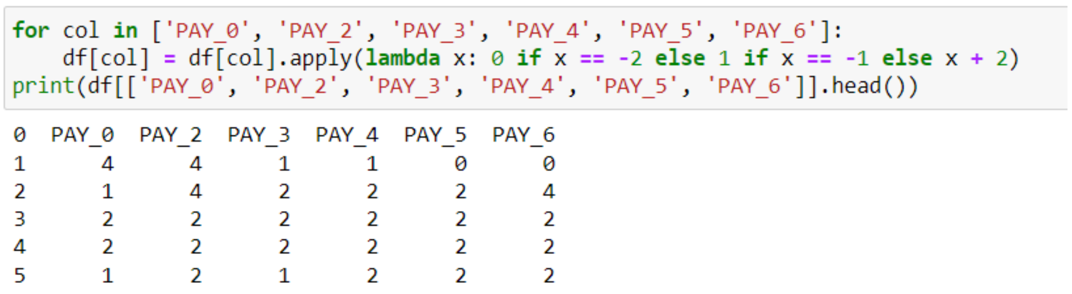
DATA PRE-PROCESSING:

1. **Data Cleaning and Formatting:**
    - Null Value Handling: Removed rows and columns with null values to ensure data integrity.
    - Dropping Unnecessary Columns: Eliminated columns that were not relevant to the analysis to focus on significant predictors.

- Data Type Conversion: Converted all variables to numeric types to facilitate mathematical operations required for the model.

2. **Treatment of Negative Values in 'PAY_0' to 'PAY_6:**

- The columns 'PAY_0' to 'PAY_6' contain negative values. In the context of logistic regression, where each feature's weight is interpreted in terms of odds, having negative values might lead to misinterpretation or skewed results.



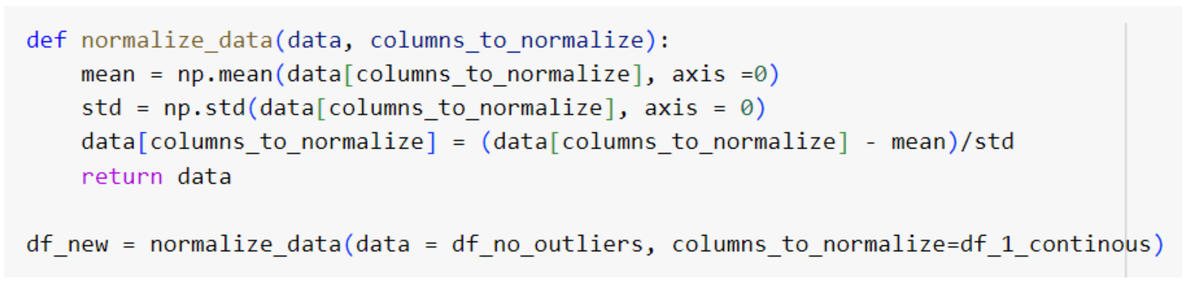
```
for col in ['PAY_0', 'PAY_2', 'PAY_3', 'PAY_4', 'PAY_5', 'PAY_6']:
    df[col] = df[col].apply(lambda x: 0 if x == -2 else 1 if x == -1 else x + 2)
print(df[['PAY_0', 'PAY_2', 'PAY_3', 'PAY_4', 'PAY_5', 'PAY_6']].head())
```

```
0  PAY_0  PAY_2  PAY_3  PAY_4  PAY_5  PAY_6
1      4      4      1      1      0      0
2      1      4      2      2      2      4
3      2      2      2      2      2      2
4      2      2      2      2      2      2
5      1      2      1      2      2      2
```

- Negative values in these columns could represent credit behavior differently from positive values. This distinction might be crucial in predicting default risk, and hence it is essential to handle them appropriately.

3. **Normalization of Continuous Columns:**

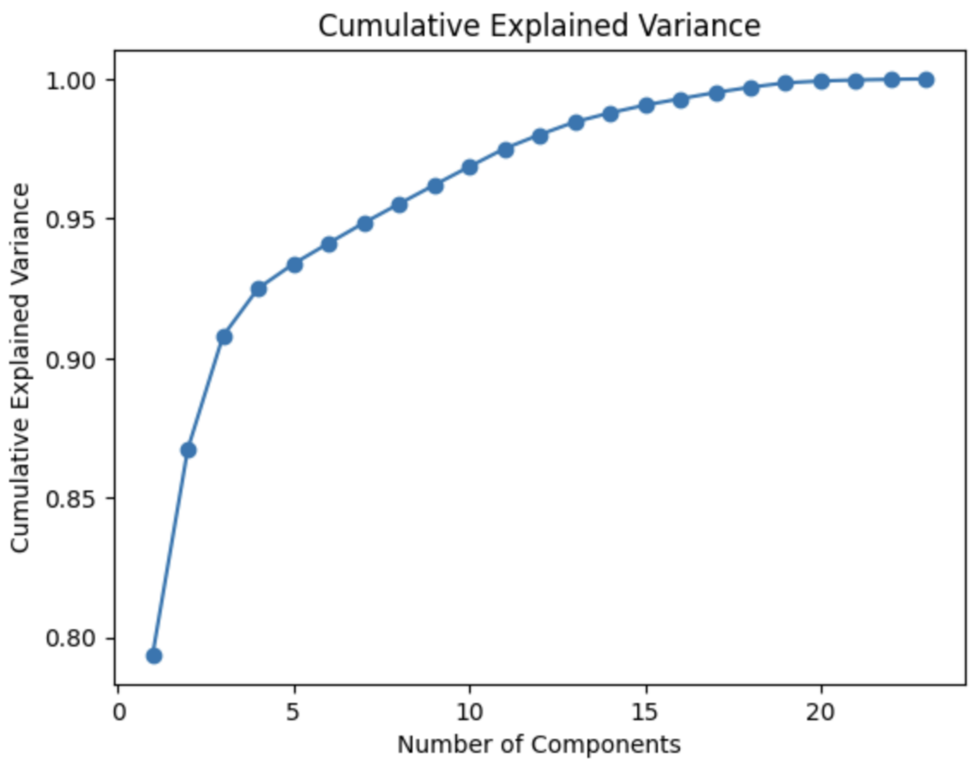
- Applied normalization to continuous variables to ensure that all features contribute equally to the model. This step is crucial because logistic regression is sensitive to the scale of input features.



```
def normalize_data(data, columns_to_normalize):
    mean = np.mean(data[columns_to_normalize], axis =0)
    std = np.std(data[columns_to_normalize], axis = 0)
    data[columns_to_normalize] = (data[columns_to_normalize] - mean)/std
    return data

df_new = normalize_data(data = df_no_outliers, columns_to_normalize=df_1_continous)
```

4. **Dimensionality Reduction Using PCA:**

- PCA Implementation: Developed a custom PCA class to reduce the dimensionality of the dataset.
- Eigenvalue Decomposition: Calculated eigenvalues and eigenvectors from the covariance matrix of the data.
- Component Selection: Selected 15 principal components based on the explained variance, ensuring a balance between dimensionality reduction and information retention.



```
The most ideal n value for 99.0% cumulative explained variance is: 15
15
```

- Cumulative Explained Variance: Used a plot of cumulative explained variance to determine the ideal number of components, aiming for a threshold (e.g., 99% variance).

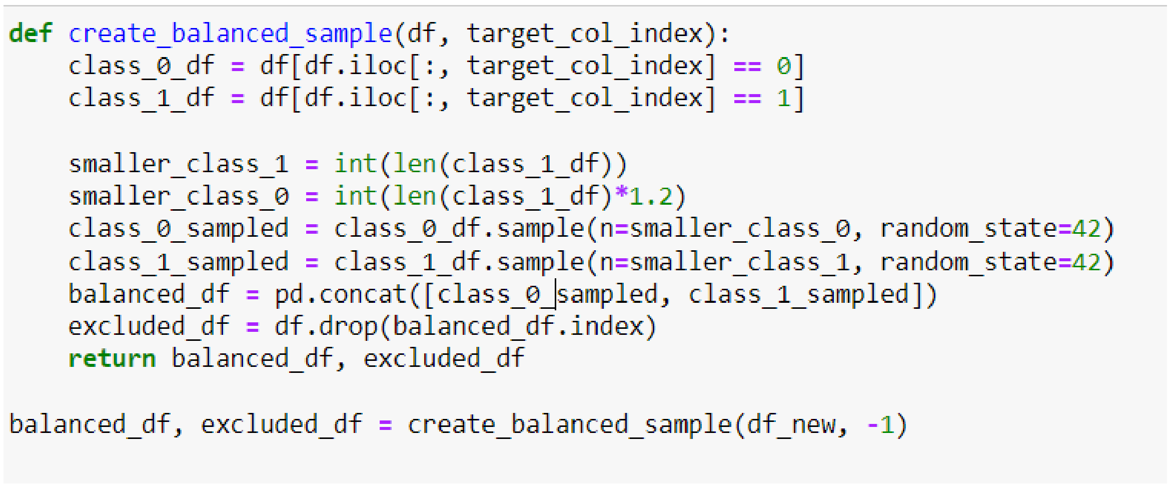
5. Balancing Classes Through Sampling:

In datasets where a significant class imbalance exists, models can become biased towards the majority class. This is particularly problematic in binary classification tasks, like credit risk or fraud detection, where the minority class is often of greater interest. To address this imbalance, undersampling aims to reduce the size of the majority class, bringing the class distribution closer to a balanced state. This helps in preventing the model from being influenced by the majority class.

Implementation of Under-sampling in the Project:

- The first step involves identifying the majority class, which, in this case, is the class with a larger number of instances in the dataset.
- The size of the minority class is noted, and the majority class is then sampled down to match this size or to a size that is deemed proportionally appropriate. For example, if the minority class constitutes 30% of the dataset, the majority class might be sampled down to constitute 70%, achieving a desired balance.
- The actual sampling is done using a method like DataFrame.sample() in pandas, which randomly selects rows from the Dataset. This randomness is crucial to ensure that the sample represents the broader characteristics of the majority class.

- After sampling, the reduced majority class dataset is concatenated with the minority class dataset, resulting in a more balanced dataset.



```python
def create_balanced_sample(df, target_col_index):
    class_0_df = df[df.iloc[:, target_col_index] == 0]
    class_1_df = df[df.iloc[:, target_col_index] == 1]

    smaller_class_1 = int(len(class_1_df))
    smaller_class_0 = int(len(class_1_df)*1.2)
    class_0_sampled = class_0_df.sample(n=smaller_class_0, random_state=42)
    class_1_sampled = class_1_df.sample(n=smaller_class_1, random_state=42)
    balanced_df = pd.concat([class_0_sampled, class_1_sampled])
    excluded_df = df.drop(balanced_df.index)
    return balanced_df, excluded_df

balanced_df, excluded_df = create_balanced_sample(df_new, -1)
```

- Excluded dataset is also created which can be used for testing an validation so as to make sure the models is not overfit and even achives higher accuracy with imbalanced data

Challenges and Considerations:

- Potential Loss of Information: One of the significant drawbacks of undersampling is the potential loss of valuable information. When instances from the majority class are removed, there's a risk that some important patterns or variations might be lost.
- Representation of the Majority Class: It's crucial to ensure that the undersampled majority class still represents the original dataset's characteristics. If the sampling is not done correctly, it might introduce biases.

METHODS AND MODEL IMPLEMENTATION:

Logistic Regression Model outline
Inputs:

- X: Feature matrix
- y: Target variable
- learning_rate: Learning rate for gradient descent
- max_iteration: Maximum number of iterations for the optimization algorithm
- epsilon: Convergence threshold for stopping the algorithm
- lamda: Coefficient for L2 regularization (Ridge)
- alpha: Coefficient for L1 regularization (Lasso)
- SGD: Boolean flag to indicate whether to use Stochastic Gradient Descent (SGD) or not

Initializes the logistic regression model with the given parameters.

- **Data Splitting :** Splits the dataset into training and testing sets using a 70-30 ratio.
- **Adding Bias Term** (add_X0 method)**:** Adds a column of ones to the feature matrix to incorporate the bias term in the model.
- **Sigmoid Function:** Calculates the sigmoid of the given input, which is used in the logistic regression model.
- **Cost Function:** Calculates the cost function of the logistic regression model, including the regularization term if lamda or alpha is non-zero.
- **Cost Derivative:** Computes the gradient of the cost function, which is used in gradient descent.
- **Gradient Descent** : Performs gradient descent to optimize the model parameters. Stops if the change in cost function is less than epsilon.
- **Stochastic Gradient Descent:** An alternative optimization method that performs updates with a random subset of data at each iteration, Stops if the change in cost function is less than epsilon.
- **Model Fitting:** Splits the data, adds the bias term, initializes weights, and then runs either gradient descent or SGD based on the SGD flag.
- **Cost Plotting:**Plots the cost function over iterations to visualize the learning process.
- **Prediction:** Makes predictions using the trained model on new data.
- **Accuracy Calculation**: Calculates the accuracy of the model based on predictions and true labels.
- **Model Evaluation**: Evaluates the model on the test set and returns the accuracy.

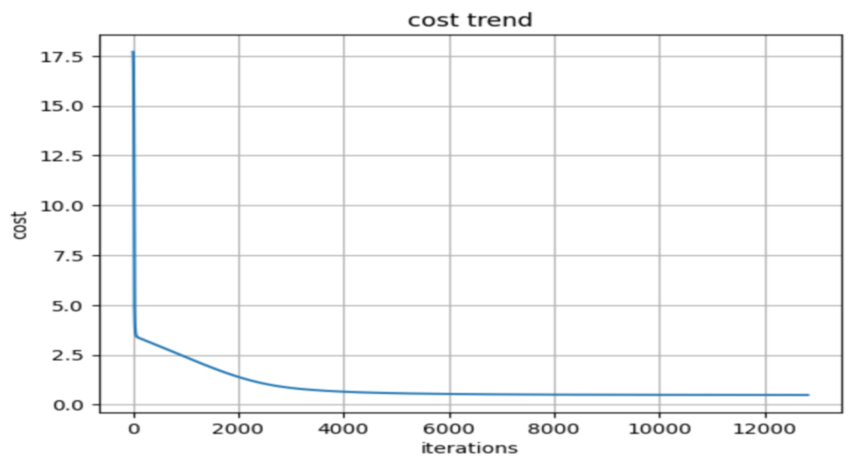
MODEL FITTING, TESTING AND RESULTS

1. **Implementing using Raw Data & Testing:**

   The logisitic regression model is fit on the normalized data for all the variables without any dimesnionality and with gardient descent optimisation where the learning rate is 0.0000001 with a maximum iterations as 10000.

   - The logistic regression model seems to be performing well in terms of accuracy and identifying the negative class but is not performing well in identifying the positive class, as evidenced by the low recall (18%) and low F1-score (29%) for the positive class.\
   - The low recall for the positive class (1) indicates a high number of false negatives, meaning the model frequently predicts the negative class when it should predict the positive one.

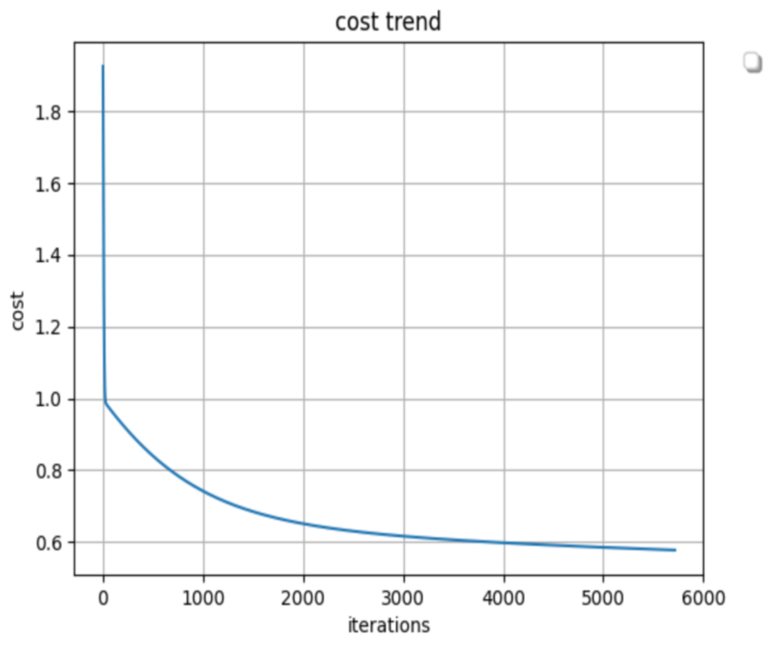
- The disparity between performance on the two classes might suggest that the model is biased towards the more prevalent class in the dataset (class imbalance), or that there is a lack of distinguishing features that can separate the positive class effectively.
- The overall accuracy might be misleading in the presence of class imbalance.



```
Training accuracy =  0.7914727843307001
accuracy =  0.7936127240595809
Confusion Matrix:
 [[5932  173]
 [1462  355]]
Classification Report:
               precision    recall  f1-score   support

           0       0.80      0.97      0.88      6105
           1       0.67      0.20      0.30      1817

    accuracy                           0.79      7922
   macro avg       0.74      0.58      0.59      7922
weighted avg       0.77      0.79      0.75      7922
```

2. **Implementation of PCA and Testing-2:**

   The logistic regression mode is fit after application of dimensionality reduction on the data with 15 principal components carrying 99% of the variance explained, where the learning rate is 0.0000001 with a maximum iterations as 10000.

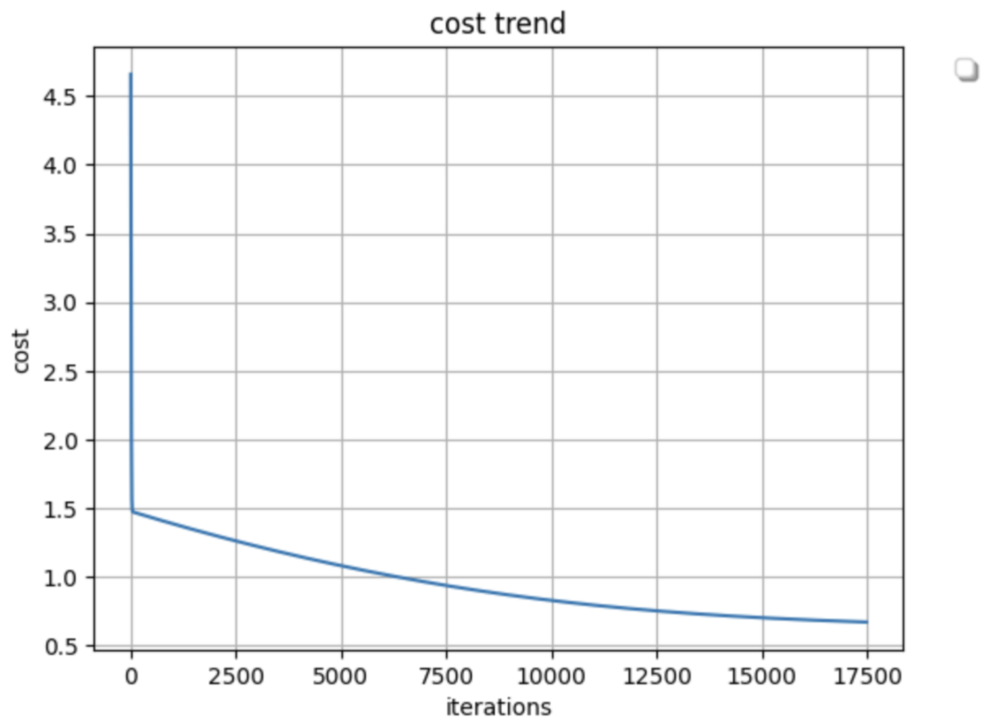


```
Training accuracy =  0.7677199437290336
accuracy =  0.7687452663468821
Confusion Matrix:
 [[5519  538]
 [1294  571]]
Classification Report:
               precision    recall  f1-score   support

           0       0.81      0.91      0.86      6057
           1       0.51      0.31      0.38      1865

    accuracy                           0.77      7922
   macro avg       0.66      0.61      0.62      7922
weighted avg       0.74      0.77      0.75      7922
```

- The use of PCA seems to have a mixed impact on the model's performance. While the accuracy is relatively high, the performance metrics indicate that the model is significantly better at predicting the negative class than the positive one.
- The recall for the positive class is low, suggesting that the model misses a substantial number of positive instances (high false negatives).
- The precision for the positive class is also relatively low, which means there are a considerable number of false positives. This could indicate that the model has difficulty distinguishing between the classes, possibly due to the reduced feature space after PCA or the nature of the data itself.
- The F1-score for the positive class is much lower than for the negative class, reflecting the model's imbalanced performance across the classes.
- The macro average F1-score is moderate, but this figure does not reflect the imbalance in class distribution.
- The weighted average metrics are closer to those of the negative class, likely because the negative class is the majority class.

3. **Implementation Sampling method and Testing 3:**

As obserbved in the abvoe tests, the imblanced nature of the data seems to effect the models prediction performance. The custom-defined sampling function, where the size of class_0 is 1.2 times the size of class_1. The Logistic regression model is fit into the data, also dimensionality reduction is performed using PCA with 15 principal components explaining 99% of variance.

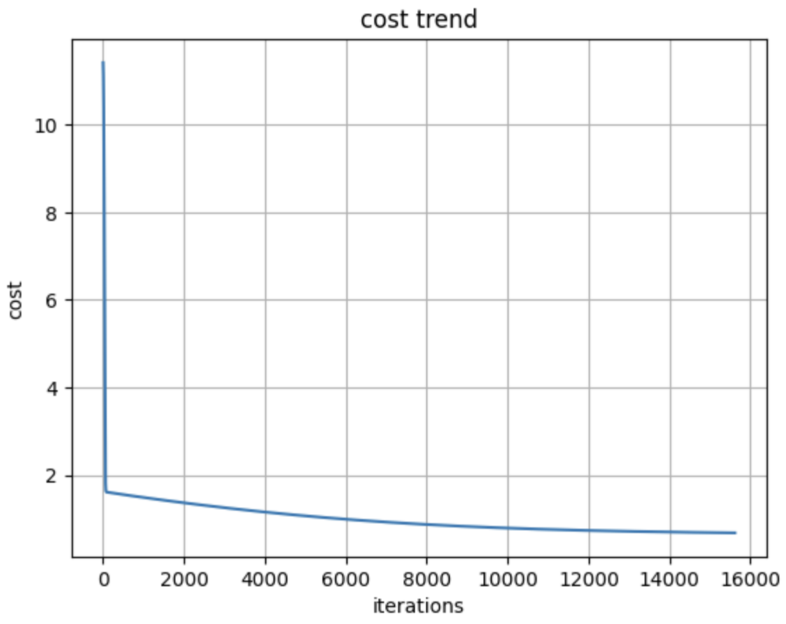


```
Training accuracy =  0.6013874066168623
accuracy =  0.5932287776947971
Confusion Matrix:
 [[1309  889]
 [ 745 1074]]
Classification Report:
               precision    recall  f1-score   support

           0       0.64      0.60      0.62      2198
           1       0.55      0.59      0.57      1819

    accuracy                           0.59      4017
   macro avg       0.59      0.59      0.59      4017
weighted avg       0.60      0.59      0.59      4017
```

- The accuracy is relatively low, suggesting the model does not perform well overall. The balanced distribution of accuracy metrics indicates that the model does not strongly favor either class, which is a positive outcome of the sampling strategy used to balance the classes.
- However, the precision and recall rates for both classes are also moderate, indicating the model has room for improvement in distinguishing between classes.
- The F1-scores being close for both classes suggest that the model is relatively consistent across both classes, likely due to the re-sampling strategy employed to balance the dataset.

4. Implementation balanced class and Testing-4:

As obserbved in the abvoe tests, the imblanced nature of the data seems to effect the models prediction performance. The custom-defined sampling function, where the sample size of both classses is same. Logistic regression model is fit into the data, also dimensionality reduction is performed using PCA with 15 principal components explaining 99% of variance.



```
Training accuracy =  0.6003052001408616
accuracy =  0.6064092029580936
Confusion Matrix:
 [[1031  779]
 [ 658 1183]]
Classification Report:
              precision    recall  f1-score   support

           0       0.61      0.57      0.59      1810
           1       0.60      0.64      0.62      1841

    accuracy                           0.61      3651
   macro avg       0.61      0.61      0.61      3651
weighted avg       0.61      0.61      0.61      3651
```

- The accuracy of 66.26% and the balanced F1-scores for both classes suggest that the model has a relatively equal performance for both classes, which is an improvement from a model trained on imbalanced data.
- The balanced class distribution achieved by the custom-defined sampling function seems to have mitigated the bias towards the majority class that observed in models trained on imbalanced dataset.
- The model might still be underfitting, given that the accuracy and F1-scores are moderately high but not close to the desired level. This could be due to the inherent complexity of the data, possible over-regularization, or the loss of some discriminative features during PCA.

5. Implementation of Regualrization and Testing-5

As the balanced class seems to produce balances F1-scores, by finding the optimal values for either Gradient descent and sochastic Gradient descent, tuning the hyperparameters for the regularisation term can improve the models accuracy. This can make the model perfectly fit, to the model and avoid underfitting like the above scenerio.

The lamda Values are taken in the list of 0.01, 0.1, 1, 10, this is the regualarisation parameter for Ridge regression. The Logistic regression model is fit to sampled data with balanced classes.

lamda	Class	Precision	Recall	F1-score	Accuracy
0.01	Class 0	0.64	0.61	0.62	0.598
	Class 1	0.56	0.59	0.57	
0.1	Class 0	0.66	0.59	0.62	0.602
	Class 1	0.55	0.62	0.58	
1	Class 0	0.64	0.62	0.63	0.602
	Class 1	0.57	0.59	0.58	
10	Class 0	0.62	0.58	0.60	0.588
	Class 1	0.55	0.6	0.57	

The Alpha values are taken in the list of 0.001, 0.01, 0.1, this is the regualarisation parameter for Lasso regression.

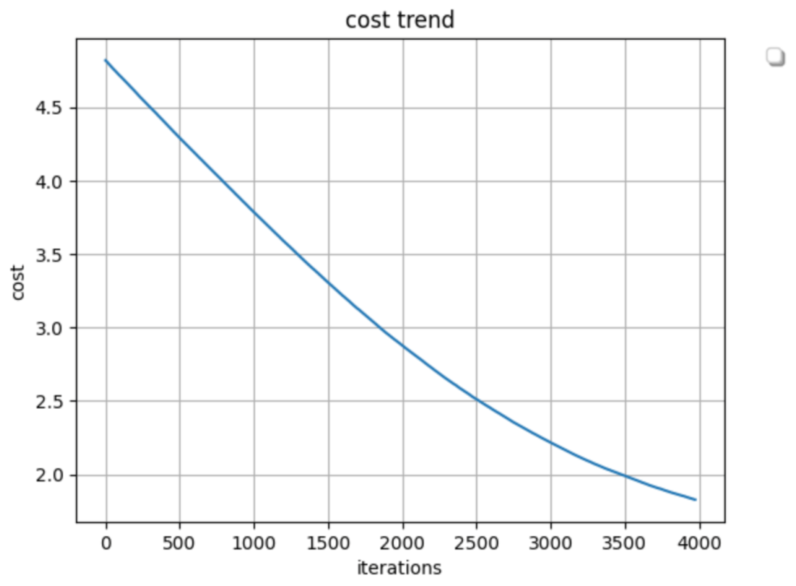
Alpha	Class	Precision	Recall	F1-score	Accuracy
0.001	Class 0	0.63	0.6	0.62	0.594
	Class 1	0.55	0.59	0.57	
0.01	Class 0	0.64	0.58	0.61	0.594
	Class 1	0.55	0.61	0.58	
0.1	Class 0	0.57	0.47	0.51	0.51
	Class 1	0.46	0.56	0.51	

- The best performing Ridge regularization parameter from the given options is $\lambda = 0.1$, providing a balance between precision and recall with a slight improvement in overall accuracy.
- For Lasso regularization, $\alpha = 0.01$ seems to balance precision and recall reasonably well without compromising accuracy too much.

- In both cases, as regularization strength increases (larger λ or α), performance tends to decrease, indicating a potential for over-regularization that strips the model of its capacity to adequately fit the data.
- The drop in performance at higher levels of regularization (especially with Lasso at α = 0.1) is an indication of underfitting, where the model is too simple to capture the underlying pattern in the data.
- Regularization is a trade-off between bias and variance. The optimal regularization parameter should minimize bias without incurring too much variance, or vice versa. The results indicate that neither extreme regularization (too high values of λ or α) nor too little (resulting in lower accuracies) is beneficial.
- Tuning the hyperparameters further, potentially with values between the tested ones, could lead to better model performance. Using cross-validation could help in identifying the optimal regularization strength that maximizes the model's accuracy while maintaining a balance between precision and recall.

6. Implementation of Stochastic Gradient Descent and Testing-6:

As the best performing Regularisation term is Ridge regression with λ = 0.1, The stochastic Gradient descent is applied in the logistic regression model to the balanced class data.



```
Training accuracy =  0.5422625400213447
accuracy =  0.5322379885486681
Confusion Matrix:
 [[1645  529]
 [1350  493]]
Classification Report:
              precision    recall  f1-score   support

           0       0.55      0.76      0.64      2174
           1       0.48      0.27      0.34      1843

    accuracy                           0.53      4017
   macro avg       0.52      0.51      0.49      4017
weighted avg       0.52      0.53      0.50      4017
```

- The overall performance of the model is quite poor, with low accuracy and low F1-scores, especially for the positive class. This could be due to several factors, including not enough iterations for SGD to converge, a learning rate that's too high or too low, or issues with the feature representation, possibly due to PCA reducing too much information.
- The precision and recall for the positive class are particularly low, suggesting that the model struggles to identify the positive instances correctly.
- The significant number of false negatives indicates that the model tends to predict the negative class more often, which could be a result of the remaining class imbalance or the model's inability to capture the complexity of the positive class.
- Stochastic Gradient Descent's performance can vary significantly depending on the batch size, learning rate, and other hyperparameters. In this case, tuning these parameters could potentially improve model performance.
- The moderate F1-score for the negative class and the low F1-score for the positive class suggest that there might be an issue with the model's fit to the data. It could be underfitting, where the model is too simple to capture the underlying structure of the data, or it could be a result of inappropriate regularization strength.

CONCLUSION & DISCUSSION:

The application of logistic regression for credit card default prediction has illuminated several crucial factors that influence model performance. Central to these are the issues of class imbalance and the dimensionality of the feature space. The initial models faced significant challenges due to imbalanced classes, a common problem in predictive modeling. The majority class (non-defaults) overshadowed the minority class (defaults), leading to a model bias towards predicting non-defaults. To mitigate this, we employed a custom-defined sampling strategy, adjusting the class distribution to have class 0 at 1.2 times the size of class 1, so as to balance the classess and also not loose much information associated with observation in class 0. This approach aimed to balance the class distribution, and the results confirmed its effectiveness; the F1-scores for both classes improved, becoming more balanced. However, precision and recall remained moderate, hinting at the persistent complexity of accurately predicting defaults. Also explored dimensionality reduction using PCA, reducing the feature space to 15 principal components that accounted for 99% of the variance. While this helped simplify the model and potentially reduce overfitting, it may have also led to the loss of critical information. The moderate accuracies post-PCA suggest that some discriminative features may have been discarded, hindering the model's ability to distinguish between classes effectively.

Hyperparameter tuning, particularly the regularization strength in logistic regression, was pivotal. Ridge regularization with $\lambda = 0.1$ and Lasso regularization with $\alpha = 0.01$ emerged as the optimal parameters, providing the best balance between bias and variance, precision, and recall. However, as regularization increased, we observed a decline in performance, indicative of over-regularization and potential underfitting — the model became too simplistic, unable to capture the nuances of the data.

The experiments has resulted in fitting the model in ideal condition with balance between underfitting and overfitting, as the Test set accuracires and Train set accuracies are similar in most cases. Regularization parameters that were too extreme in either direction proved detrimental. Values that were too low led to lower accuracies, while values that were too high stripped the model of its complexity, necessary for capturing the intricate patterns within the data. This trade-off between bias and variance is a recurring theme in machine learning and was especially pronounced in our analysis due to the intricacies of financial data.

The implementation of SGD provided a means to efficiently optimize our logistic regression model, especially useful given the large dataset. However, the results from SGD were mixed, with accuracies hovering around the 50% mark, only marginally better than a random guess in a balanced dataset. This points to the need for a more nuanced approach to SGD's hyperparameter optimization, including learning rate adjustments and batch size considerations.

The journey from an imbalanced dataset to one that is carefully preprocessed and balanced, with dimensionality reduced and regularized, has been instructive.Its evident that there is no one-size-fits-all solution. Each dataset and predictive modeling task requires a tailored approach, considering the nature of the data, the problem at hand, and the balance between complexity and generalization. Further model refinement through cross-validation and exploration of hyperparameters within a narrower range may yield improvements. This iterative process of optimization and validation is essential for advancing towards a model that is not only accurate but also robust and reliable in its predictions.