IE7300 Statistical Learning

Classification for Early Readmission of Diabetic Patients

Project Report
Group No. 9

Nethra Narayanan (narayanan.ne@northeastern.edu)

Mansi Sain (sain.m@northeastern.edu)

Zhenan Zhuang (zhuang.zhe@northeastern.edu)

Zhaohan Zhu (zhu.zhaoha@northestern.edu)

Greeshma Janapala (janapala.g@northeastern.edu)

Abstract

This report presents a comprehensive analysis of machine learning models applied to predict diabetes patient readmission. We utilized the UCI Diabetes Dataset, focusing on Logistic Regression, Neural Networks, and K-Nearest Neighbors (K-NN) classifiers. Each model was carefully selected for its unique strengths in data analysis and pattern recognition, with an emphasis on achieving accurate predictions in a medical context. To enhance the reliability of our models, we incorporated K-Fold Cross-Validation and Principal Component Analysis (PCA), the latter reducing our data to its most informative features. Our approach was twofold: initially, we assessed the models on the original dataset, and subsequently, we evaluated their performance following the PCA dimensionality reduction. The study's goal was not just to determine the most effective model for predicting patient readmission but also to explore different machine learning techniques in a healthcare setting. The findings from this study offer valuable insights into the application of machine learning in medical data analysis, underscoring its potential in enhancing patient care and healthcare management.

Introduction

In the context of healthcare, hospital readmission refers to the scenario where a patient, after being discharged, returns to the hospital within a specified timeframe, typically within 30 days. This phenomenon is of paramount importance as it not only incurs substantial healthcare costs but also serves as a vital indicator of hospital care quality. The project's primary goal is to develop a predictive model for classifying diabetic patient readmissions, enabling the identification of influential factors behind such events. The dataset encompasses diagnostic, secondary diagnostic, and medication-related variables, along with "A1c" levels (indicating Blood Sugar Level). The model classifies patients into three categories: readmission after more than 30 days, readmission within 30 days, or no readmission. By achieving this, the project uses classification and aims to enhance our understanding of factors affecting readmissions, leading to improved diabetic patient care and outcomes, with the potential to reduce healthcare costs.

Data Source

The dataset was obtained from the UCI repository. The data has been provided by the Center for Clinical and Translational Research, Virginia. Website Link**:**

https://archive.ics.uci.edu/dataset/296/diabetes+130-us+hospitals+for+years+1999-2008

Data description

The dataset represents ten years (1999-2008) of clinical care at 130 US hospitals and integrated delivery networks. It has 101766 rows and 47 features, including both categorical and integer data types. The instances represent hospitalized patient records diagnosed with diabetes, with encounter_id and patient_nbr serving as unique identifiers for each encounter and patient, respectively. Other features and their data types are listed below.

Features	**Data Type**
encounter_id	Numeric
patient_nbr	Numeric
race	Categorical
gender	Categorical
age	Categorical
weight	Categorical
admission_type_id	Categorical
discharge_disposition_id	Categorical
admission_source_id	Categorical
payer_code	Categorical
medical_specialty	Categorical
time_in_hospital	Integer
num_lab_procedures	Integer
num_procedures	Integer

num_medications	Integer
num_emergency	Integer
num_inpatient	Integer
diag_1	Nominal
diag_2	Nominal
diag_3	Nominal
…	…
insulin	Categorical
change	Categorical
diabetesMed	Categorical
readmitted (output)	Categorical

Methods (Model Description)

1) Logistic Regression

Logistic Regression was chosen as the base model for its simplicity and effectiveness in binary classification problems. It's particularly well-suited for medical diagnostic tests where the outcome is binary i.e., if the patient gets readmitted or not. Logistic Regression works by estimating the probability of a binary outcome based on one or more predictor variables. It uses the logistic function to model a binary dependent variable. In the context of the diabetes dataset, it assesses how various features affect the probability of an individual being diabetic which we believe is pivotal.

2) Neural Network

Neural Networks was selected for its ability to model complex, non-linear relationships in data. They are particularly adept at handling large and intricate datasets, making them suitable for medical datasets with many variables. A Neural Network consists of layers of interconnected nodes or neurons. Each connection represents a weight, and each neuron applies an activation function to its input to determine its output. In our

study, we employed a Sequential Neural Network model using the Keras library to address a binary classification problem, likely focused on predicting patient readmission. This model is structured with a multi-layer architecture, starting with an input layer of 64 neurons, followed by three hidden layers with decreasing neuron counts (32, 16, and 8), all employing the ReLU activation function for non-linear processing. The final output layer, consisting of a single neuron with a sigmoid activation function, is tailored for binary classification tasks. The model is compiled with the Adam optimizer and binary cross-entropy as the loss function, emphasizing its suitability for binary classification problems. We trained the model over a specified number of epochs and batch sizes, using 20% of the data as a test set and 10% of the training data for validation.

3) K-Nearest Neighbors (K-NN) Classifier

The K-NN classifier was chosen for its simplicity and effectiveness in classification problems. It is a non-parametric method, meaning it doesn't make any underlying assumptions about the distribution of data, which is advantageous in medical datasets where such assumptions might not hold. K-NN classifies a data point based on how its neighbors are classified. It identifies the 'k' nearest data points using a distance metric (like Euclidean distance) and classifies the new point based on the majority class among these neighbors. In the diabetes dataset, it involves looking at 'k' similar patients and predicting diabetes presence based on the majority presence or absence of diabetes in these patients.

4) K-Fold Cross-Validation

K-Fold Cross-Validation is a method to assess the predictive performance of models and to mitigate the problem of overfitting. It involves dividing the dataset into 'k' subsets. In each iteration, one subset is used for testing, and the remaining 'k-1' subsets are used for training the model. This process repeats 'k' times with each subset used once as the testing set. For our Logistic Regression and K-NN models, this method provides a more reliable estimate of their performance by using different parts of the data for training and validation. We have shown the bias and variance of the model using this function to observe discrepancies.

5) Principal Component Analysis (PCA)

PCA is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of linearly uncorrelated variables called principal components. The first principal components retain most of the variation present in the original data. In our project, we reduced the feature space of the diabetes

dataset to 30 principal components to simplify the models while retaining the essential information.

6) Conclusion for Methods

These models were chosen and applied to the UCI Diabetes Dataset with the aim of accurately predicting diabetes onset. Logistic Regression provides a solid baseline with its simplicity and direct approach. Neural Networks bring the capability to model complex patterns, whereas K-NN offers a straightforward, intuitive way to classify based on similarity. The application of PCA and K-Fold Cross-Validation ensures robustness and generalizability in our modeling approach. The results of these models and the upcoming cost function analysis will further help in identifying the most effective model among these.

Exploratory data analysis (EDA)

1) Univariate Plots

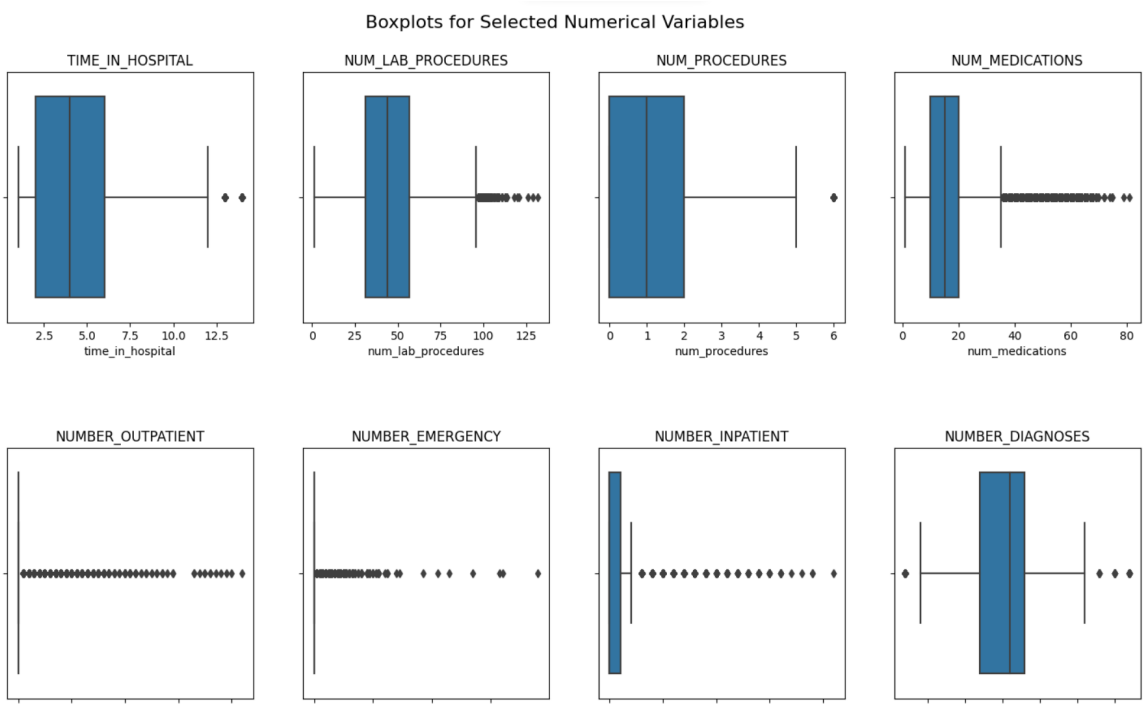


Fig 1: Box Plots for Selected Numerical Variables

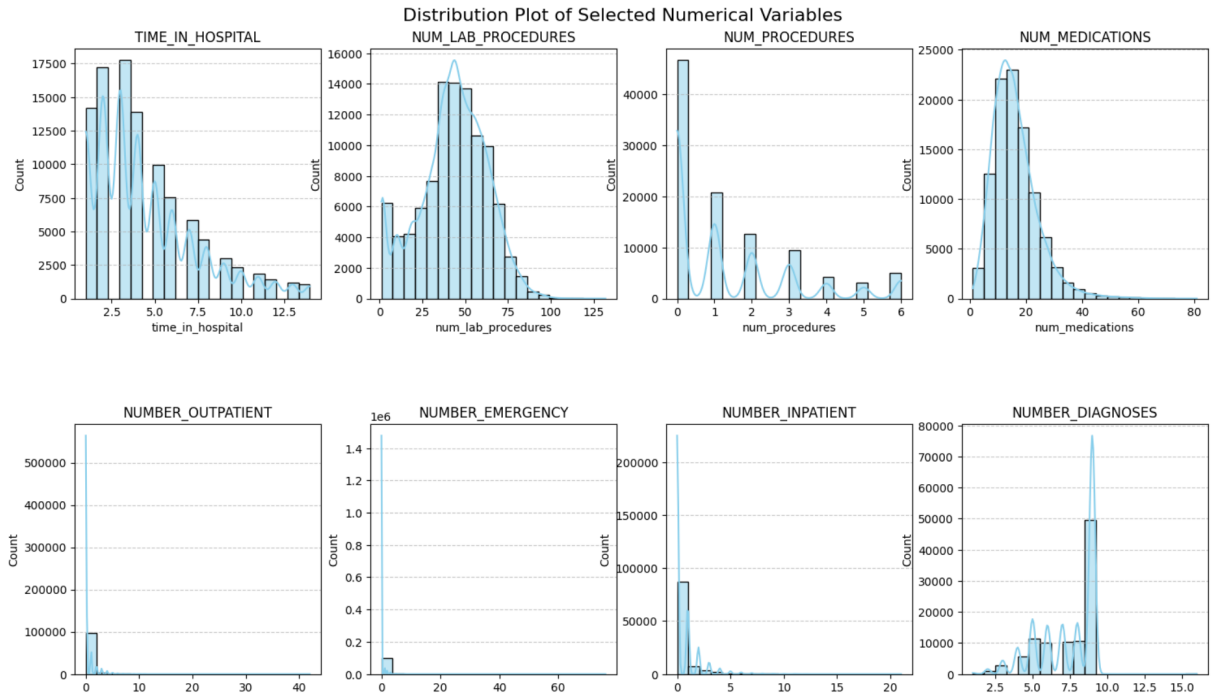


Fig 2: Distribution Plot of Selected Numerical Variables

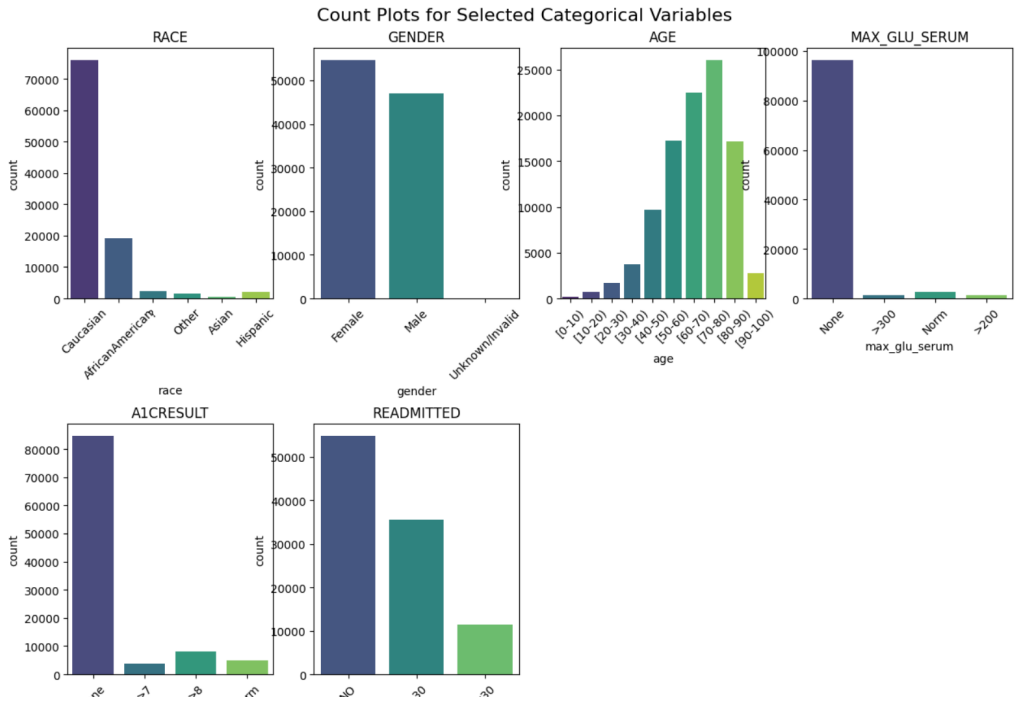


Fig 3: Count Plots for Selected Categorical Variables

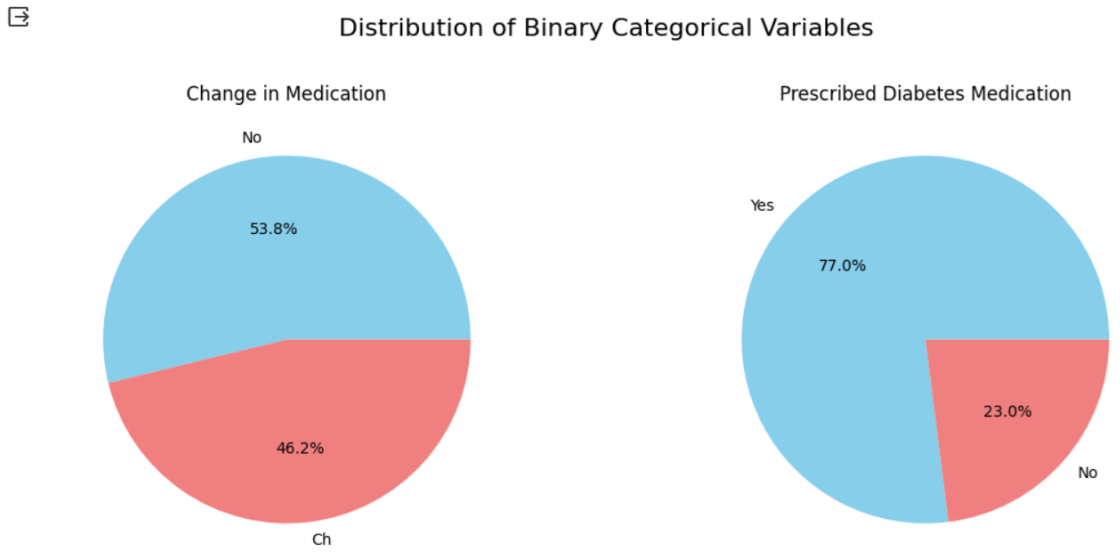


Fig 4: Pie Chart for Binary Categorical Variables

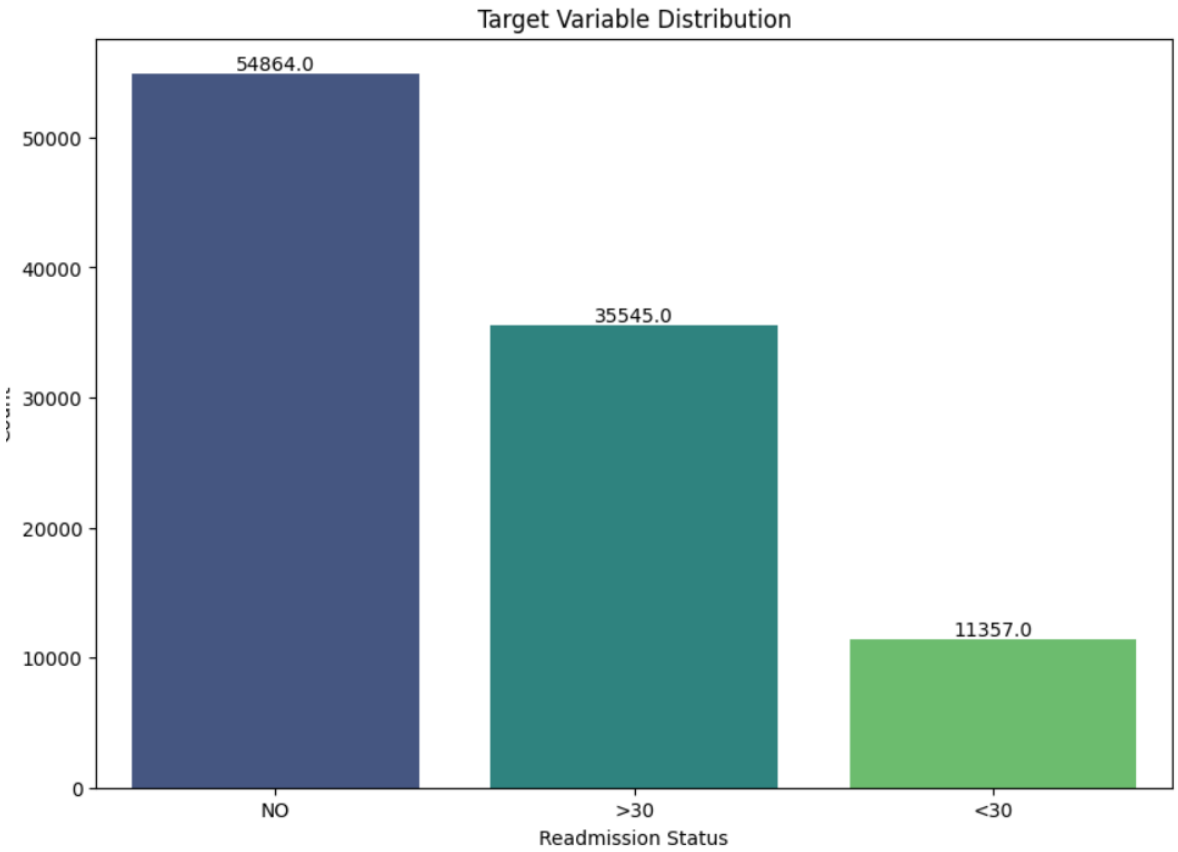


Fig 5: Count for Readmission Status

2) Bivariate Plots

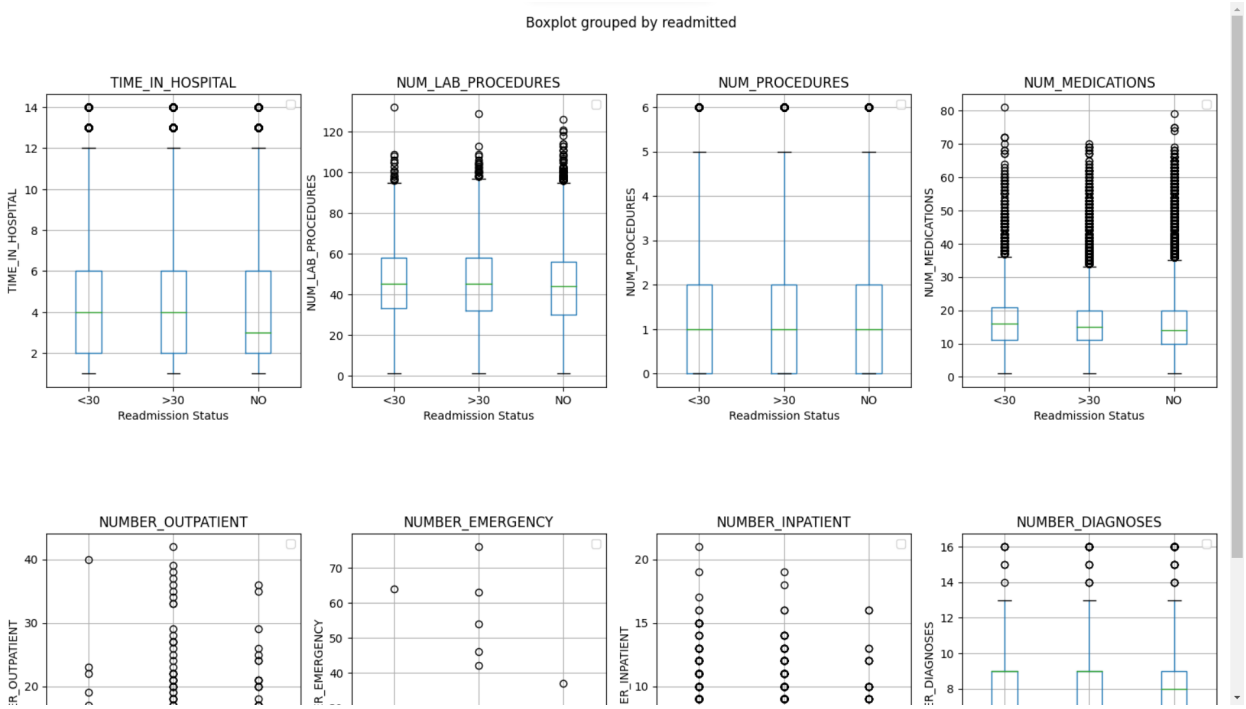


Fig 6: Box Plots for Categorical Variables grouped by readmitted

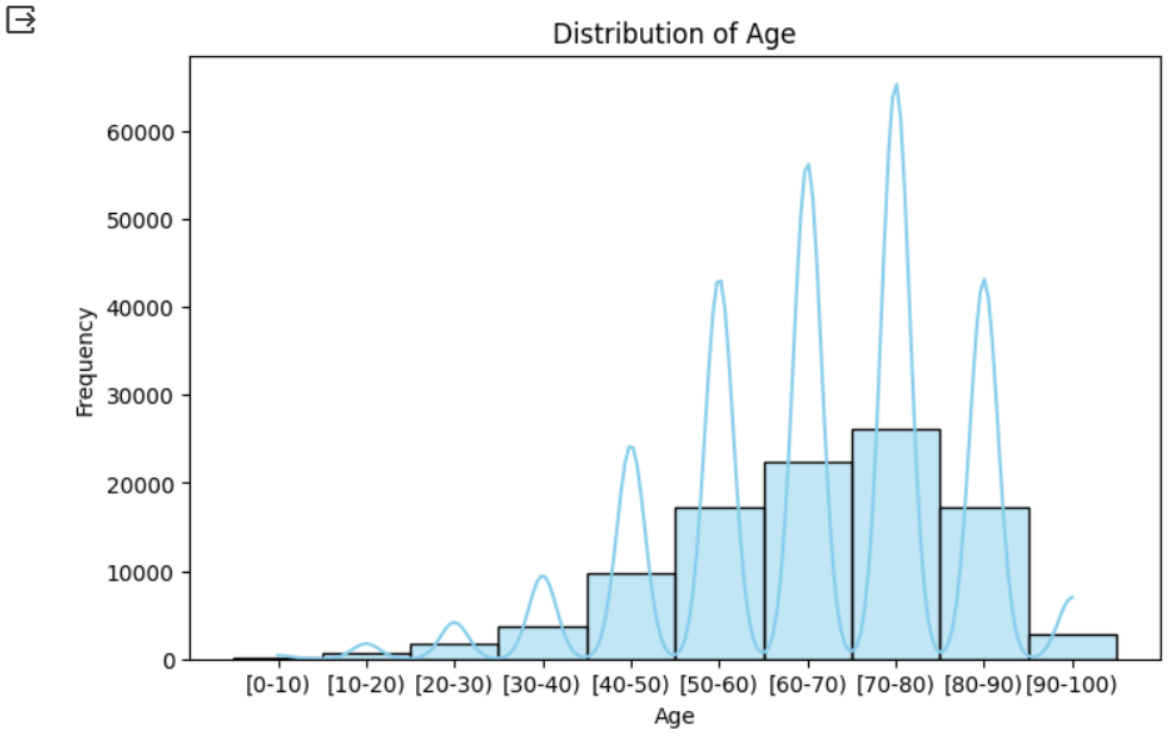


Fig 7: Kernel Density Plot for the Age Distribution

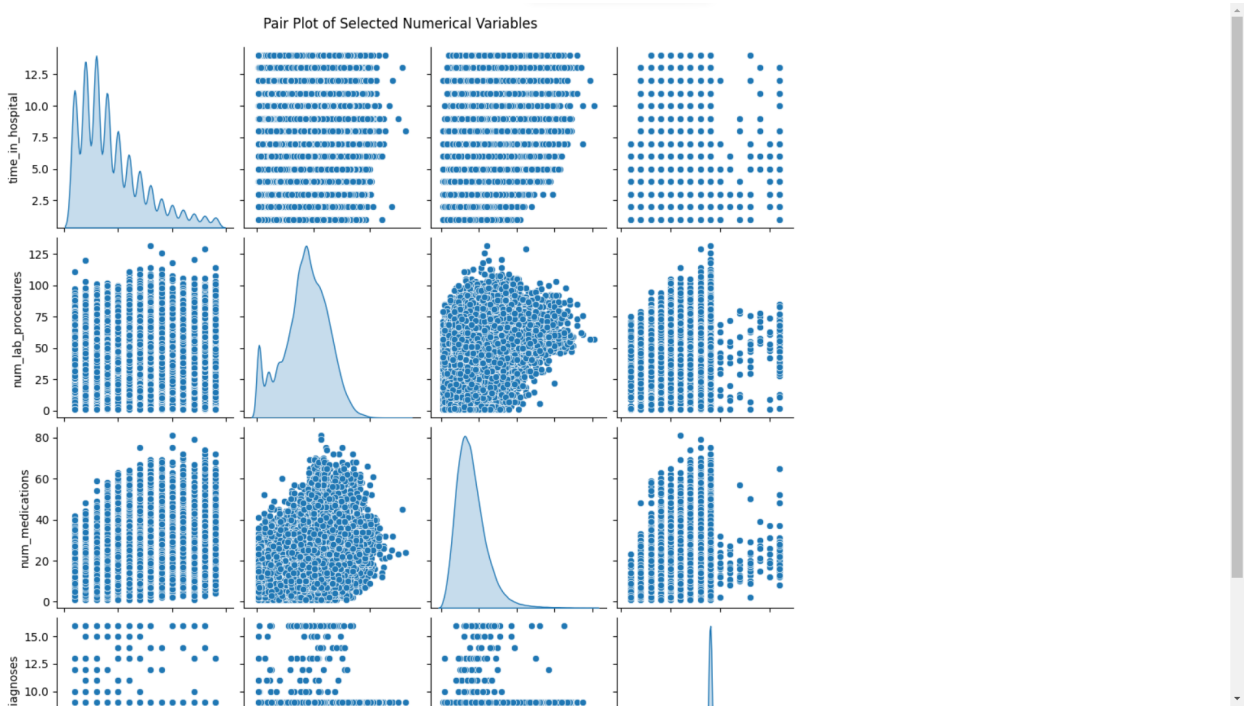


Fig 8: Pair Plot for Selected Numerical Variables

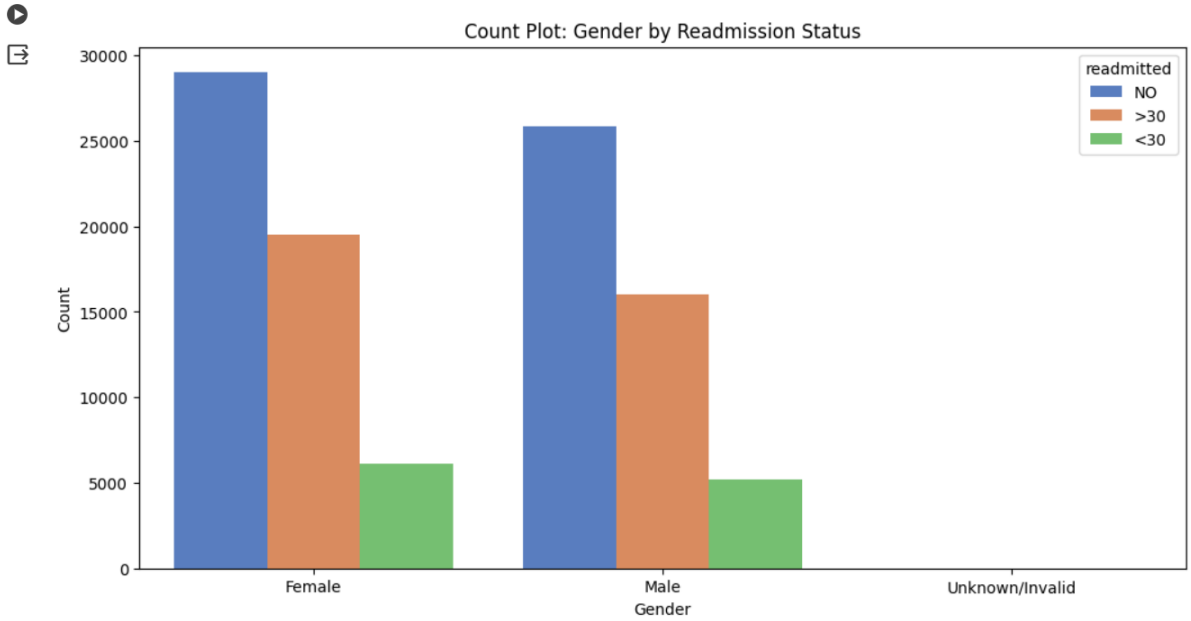


Fig 9: Categorical Plots for Gender and Readmission Status

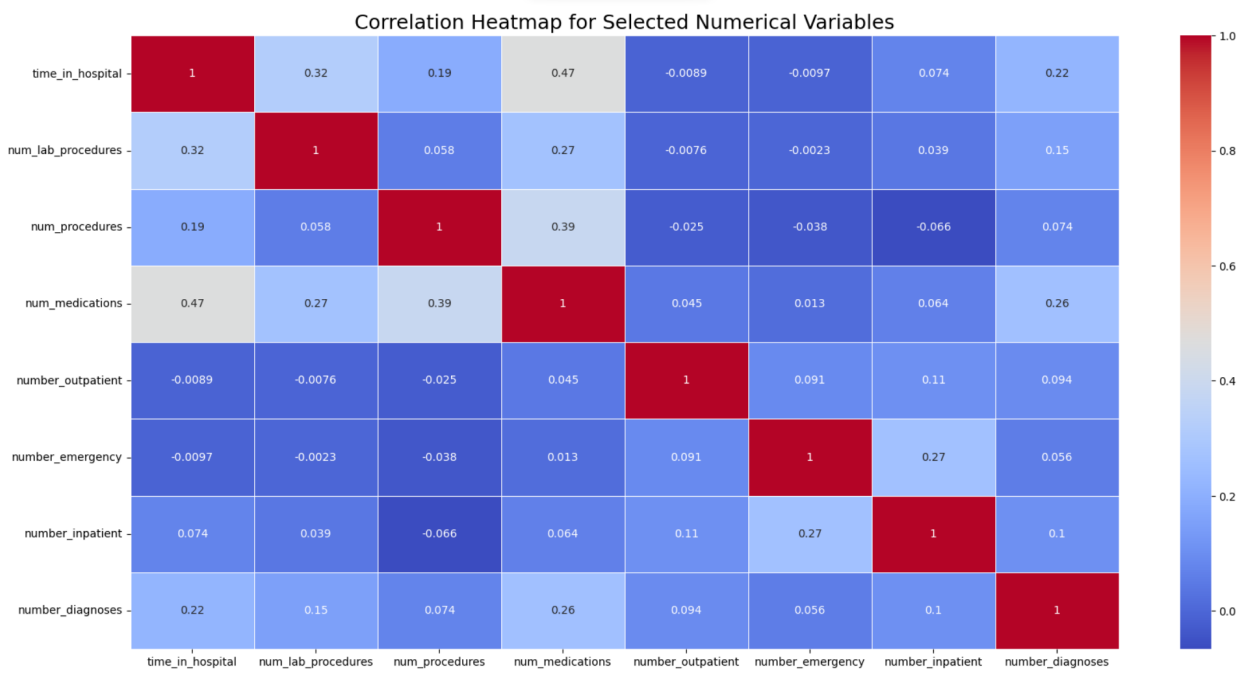


Fig 10: Correlation Heatmap for Selected Numerical Variables

[ ]

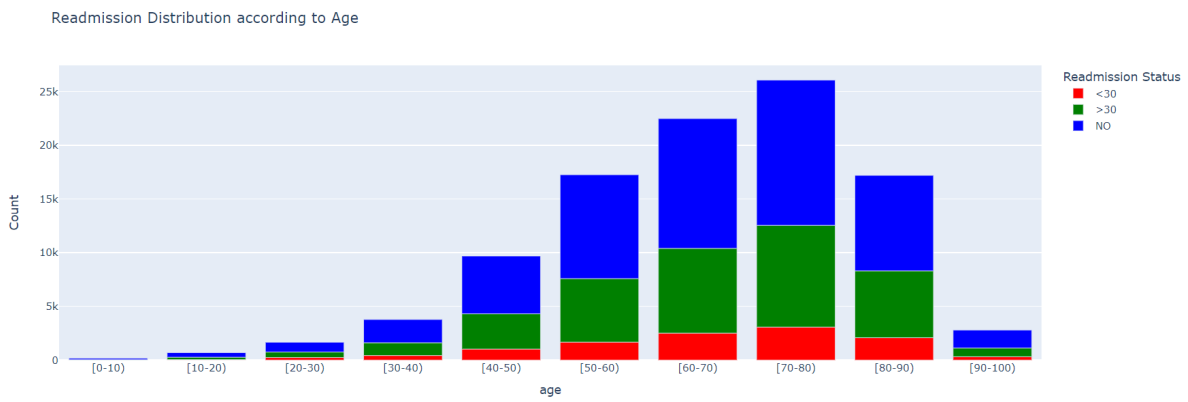


Fig 11: Readmission Distribution according to Age

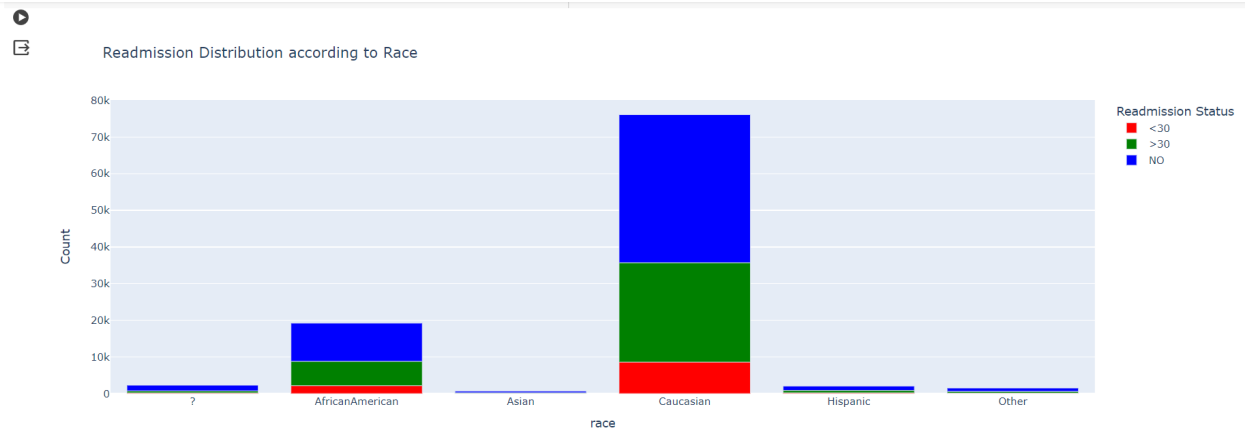


Fig 12: Readmission Distribution according to Race

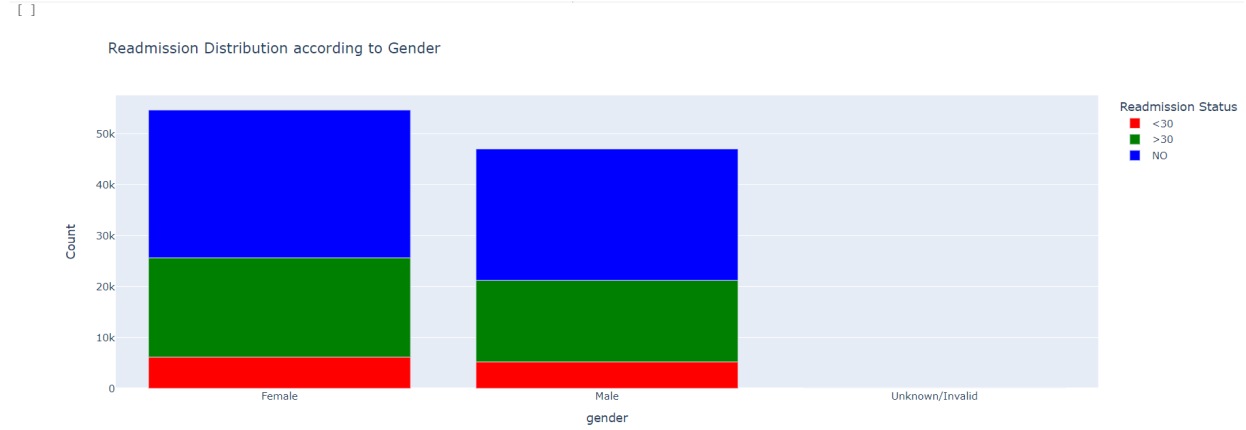


Fig 13: Readmission Distribution according to Gender

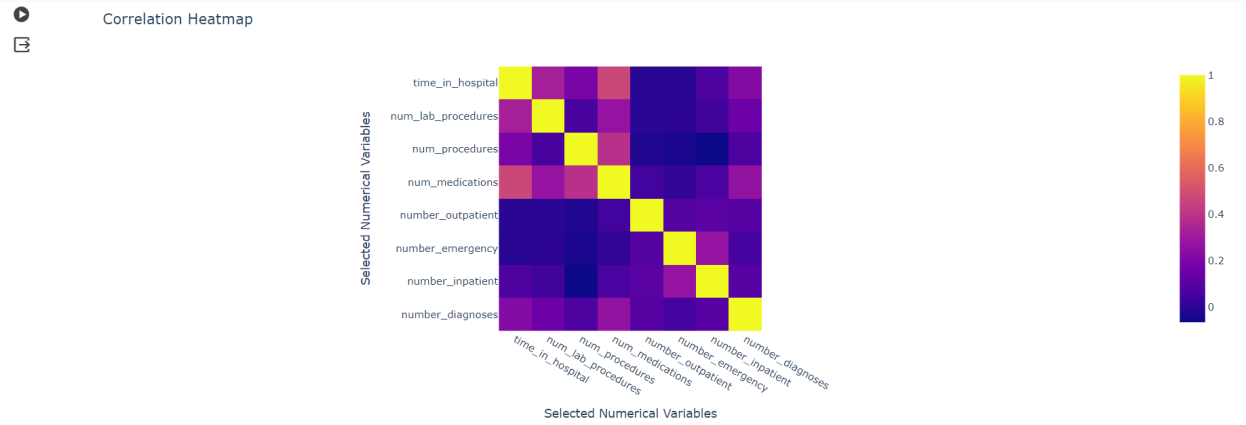


Fig 14: Correlation Matrix for Selected Numerical Variables

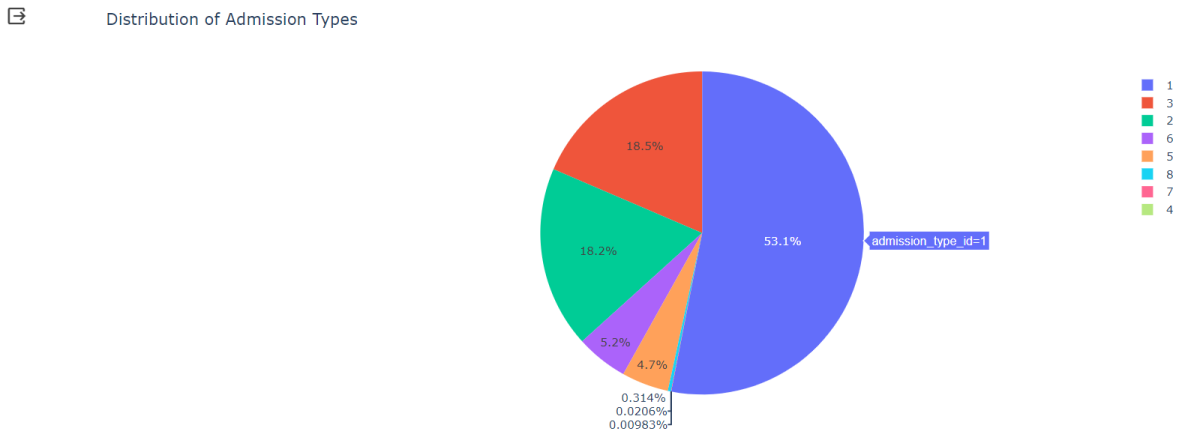


Fig 15: Pie Chart for Distribution of Admission Types

Feature Engineering Steps:

1) Handling Missing Values and Invalid Data:

- Replaced '?' and 'Unknown/Invalid' values in the dataset with NaN for a clearer representation of missing data.
- Checked for columns with a high percentage of missing values (>30%). Decided to drop columns with significant missing data (weight, payer_code, and medical_specialty) to maintain data integrity.

2) Assessing and Addressing Missing Data:

- Determined that no columns had more than 3% missing values.
- Opted to drop rows with missing values rather than imputing, as imputation can be risky in healthcare data due to the potential for inaccurate assumptions. This decision was reinforced by the fact that the loss was only about 3%, minimizing the impact on the dataset's integrity.

3) Dropping Redundant or Invariant Features:

- Removed columns examide and citoglipton since they contained only one category within the dataset, rendering them uninformative for predictive modeling.

4) Remapping and Merging Categories:

- Simplified and reduced the number of categories in admission_type_id, discharge_disposition_id, and admission_source_id by merging similar categories. This step was crucial for reducing complexity and enhancing interpretability during encoding.

5) Encoding Categorical Variables:

- Binary-encoded some categorical variables like change, gender, and diabetesMed.
- For medication columns, combined categories and encoded them into numerical values, facilitating their inclusion in machine learning models.

6) Handling Test Result Columns:

- Revised test result columns like A1Cresult and max_glu_serum, assigning a distinct negative numerical value to 'None' for differentiation.

7) Encoding Age as a Numerical Variable:

- Converted age ranges to midpoints, allowing for a more precise and quantifiable analysis of age-related patterns.

8) Target Variable Encoding for Binary Classification:

- Encoded the readmitted variable to reflect a binary classification goal (readmitted within 30 days vs not readmitted or readmitted after 30 days).

9) Combining Diagnosis Codes:

- Created a new column diag_cat to classify patients as 'Diabetes' or 'Other', based on whether any of the three diagnosis columns contained a diabetes-related code.
- Dropped the original diagnosis columns (diag_1, diag_2, diag_3) to avoid redundancy.

10) Applying One Hot Encoding:

- Performed One Hot Encoding on several categorical variables to transform them into a format suitable for machine learning algorithms.

- ❖ Generated dummy variables for race and merged them with the main dataset.

11) Final Data Cleaning:

- ❖ Removed duplicate entries based on patient_nbr to ensure each patient's data was represented only once, maintaining the uniqueness of each patient's information.

Explanation:

- These steps were taken to ensure the dataset's quality and relevance for predictive modeling. Handling missing data and simplifying categorical variables reduces model complexity and potential biases.
- Encoding categorical variables and merging similar categories streamline the dataset, making it more manageable for machine learning algorithms.
- The focus on maintaining data integrity, especially in a healthcare context, guided decisions like dropping rows with missing data instead of imputing them.
- These preprocessing steps are crucial for building robust and reliable predictive models, particularly in healthcare analytics where data accuracy and integrity are paramount.

Results

Logistic Regression before PCA:

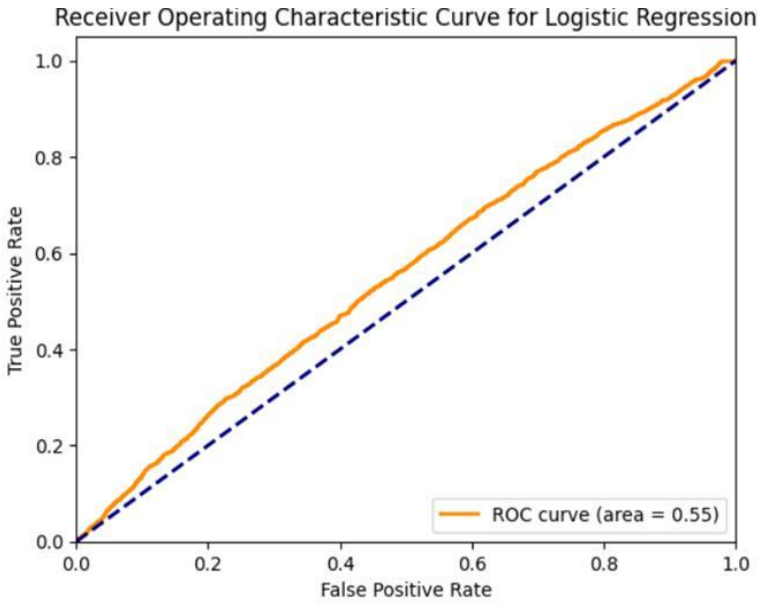
Our initial approach was to use Logistic Regression as a baseline model. Logistic Regression is a powerful and straightforward method for binary classification. We applied this model to our preprocessed dataset to establish a foundation for comparison with more complex models. The primary aim was to understand how well Logistic Regression could handle our dataset without any dimensionality reduction.

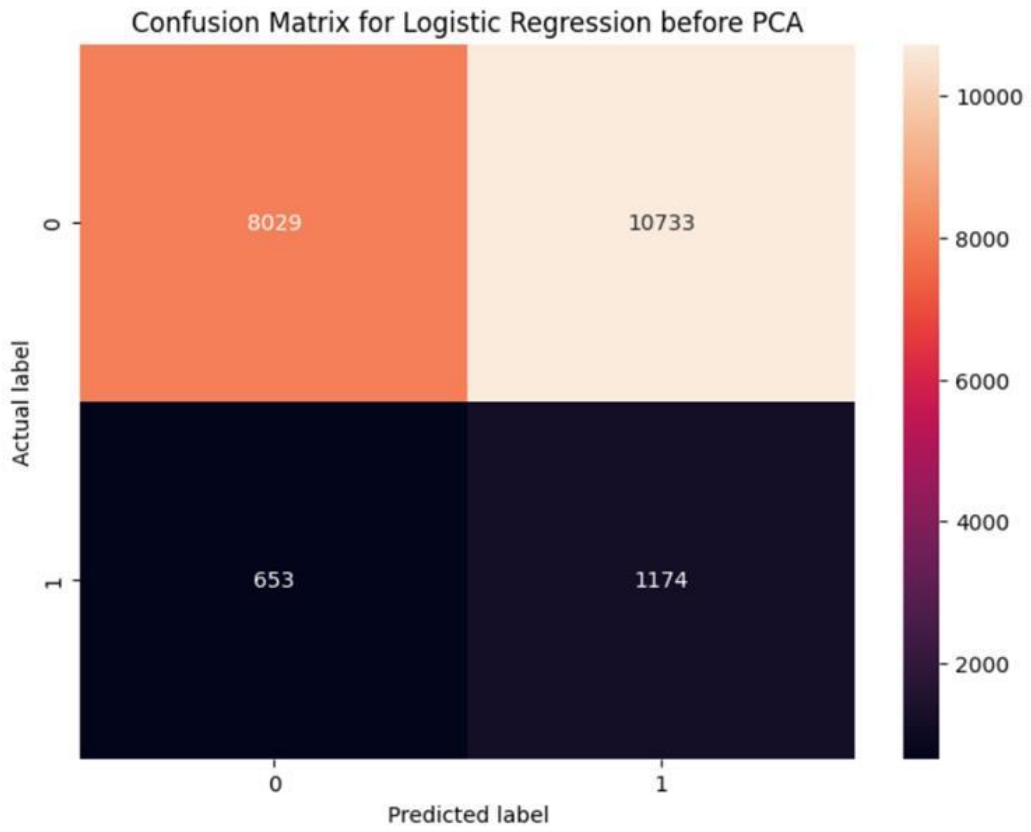
- The Logistic Regression model without PCA yielded an accuracy of 44.8%, a recall of 65.15%, a precision of 10.07%, and an F1 score of 17.44%. The cost value calculated was approximately 0.89, which indicated the model's general performance on the given data.
- We included a confusion matrix to visualize the performance of the Logistic Regression model in terms of true positives, true negatives, false positives, and

false negatives. The ROC (Receiver Operating Characteristic) curve was plotted to evaluate the model's diagnostic ability. The ROC curve is a graphical

representation that illustrates the trade-off between the true positive rate and the false positive rate at various threshold settings.

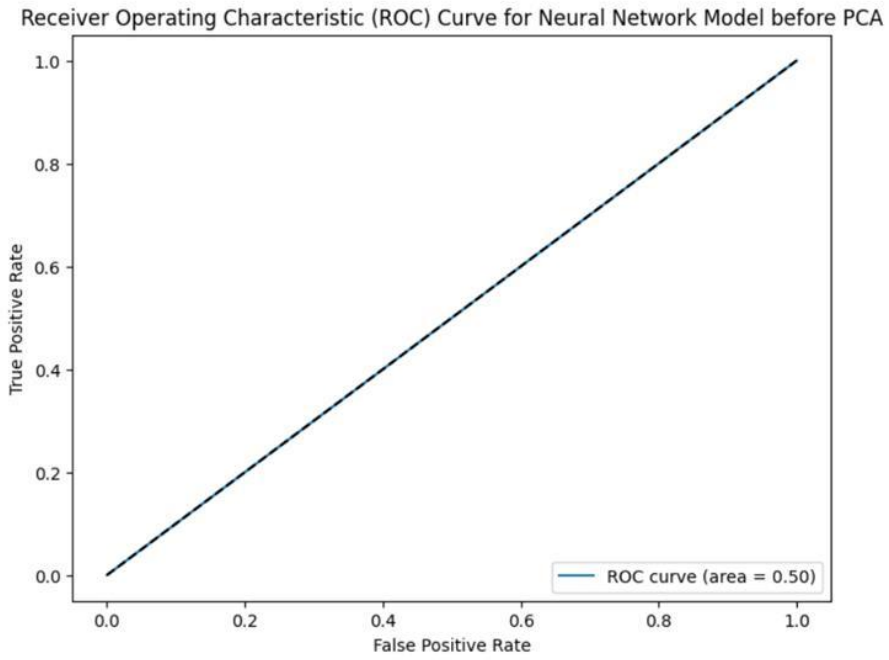
```
Training accuracy: 0.448022481265612
Training recall: 0.6515468713654339
Training precision: 0.10068296189791517
Training F1 score: 0.17441389831563875
     Iteration      Cost
0            0  2.945523
1            1  2.938046
2            2  2.930580
3            3  2.923123
4            4  2.915677
..         ...       ...
495        495  0.903801
496        496  0.902753
497        497  0.901710
498        498  0.900674
499        499  0.899645
```



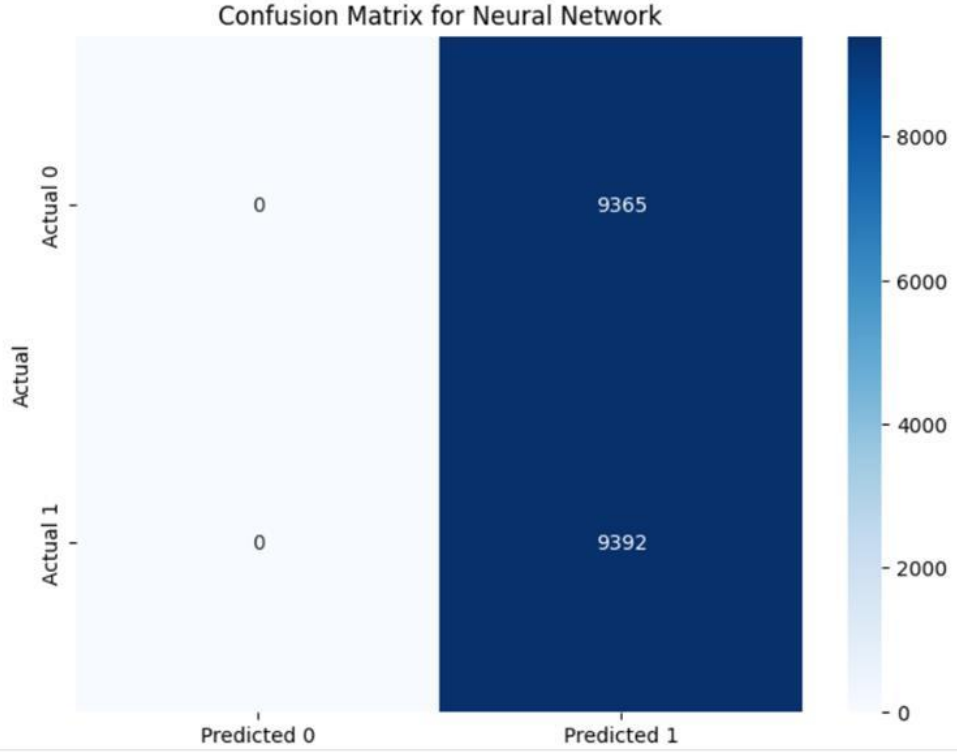


Neural network before PCA:

Neural Networks are known for handling complex patterns in data. We utilized this model to see if a more intricate approach would better capture the nuances in our dataset. The Neural Network showed an accuracy of 50.07%, which is only slightly better than random guessing in a binary classification problem. The model achieved a recall of 100%, indicating it identified all positive cases but failed to correctly predict any negative cases (as precision is also 50.07%). This might suggest the model was biased towards predicting one class.




```
Test Accuracy: 0.5007197313003146
Precision: 0.5007197313003146
Recall: 1.0
F1 Score: 0.6673061209989697
AUC: 0.5
Confusion Matrix:
[[   0 9365]
 [   0 9392]]
```



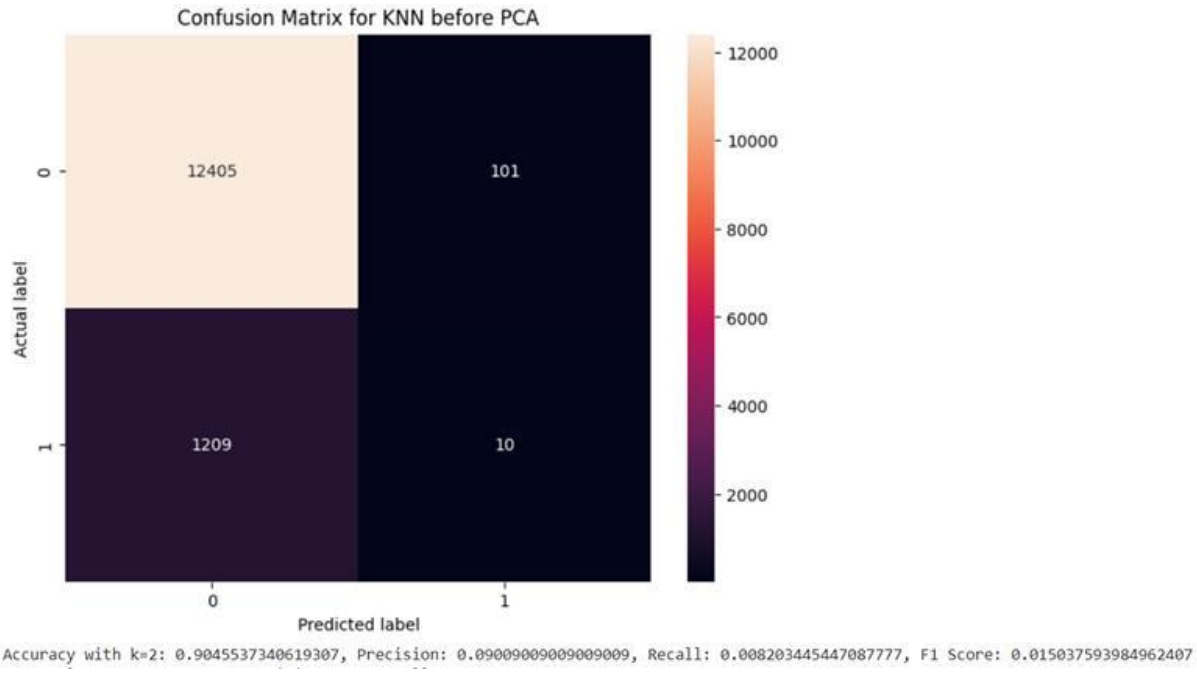
	epoch	loss	val_loss
0	1	41771.187500	13963.213867
1	2	12606.761719	2340.227051
2	3	2641.900391	166.107208
3	4	516.226379	1551.348755
4	5	499.093597	1.757422
5	6	1.181137	0.701160
6	7	1.201944	0.700665
7	8	0.693576	0.693272
8	9	0.693158	0.693070
9	10	0.693179	0.693169

K-NN Classifier before PCA:

K-Nearest Neighbors (KNN) operates on a simple principle of classifying a data point based on how its neighbors are classified. This model doesn't make explicit assumptions about the form of the mapping function from inputs to outputs, making it a good choice for our dataset where such mappings might be non-linear or complex.

- Despite the high accuracy (90.46%) with k=2, the model's precision and recall were notably low. The high accuracy in the presence of low precision and recall is an example of the accuracy paradox. It happens when a model has a majority class that it predicts very well, but it fails at or ignores the minority class. In binary classification like ours, this often means the model is good at predicting the more frequent class but not the less frequent one.
- Choice of k (k=2): The choice of 'k' in KNN is crucial. A small value of k means that noise will have a higher influence on the result. In our case, using k=2 was optimal and was decided based on 5 iterations amongst which we got the best accuracy at k=2.
- The performance of KNN is heavily reliant on the distance metric used (e.g., Euclidean distance) and whether the features were scaled appropriately. Without proper scaling, features with larger ranges could disproportionately influence the distance computations, leading to biased classifications.

With k=2, KNN achieved 90.46% accuracy, which looks impressive. However, the precision and recall were significantly low. This implies that while the model was accurate overall, it struggled to correctly classify positive cases and mostly predicted the majority class.



K-Fold Cross Validation:

The cross-validation process was crucial for testing the models' robustness. It helped us ensure that the models were generalizable and not just tailored to a specific subset of data. It also helped us understand the bias and variance of the data better. Through this technique, we saw that the variance was significantly low in our dataset.

PCA Application:

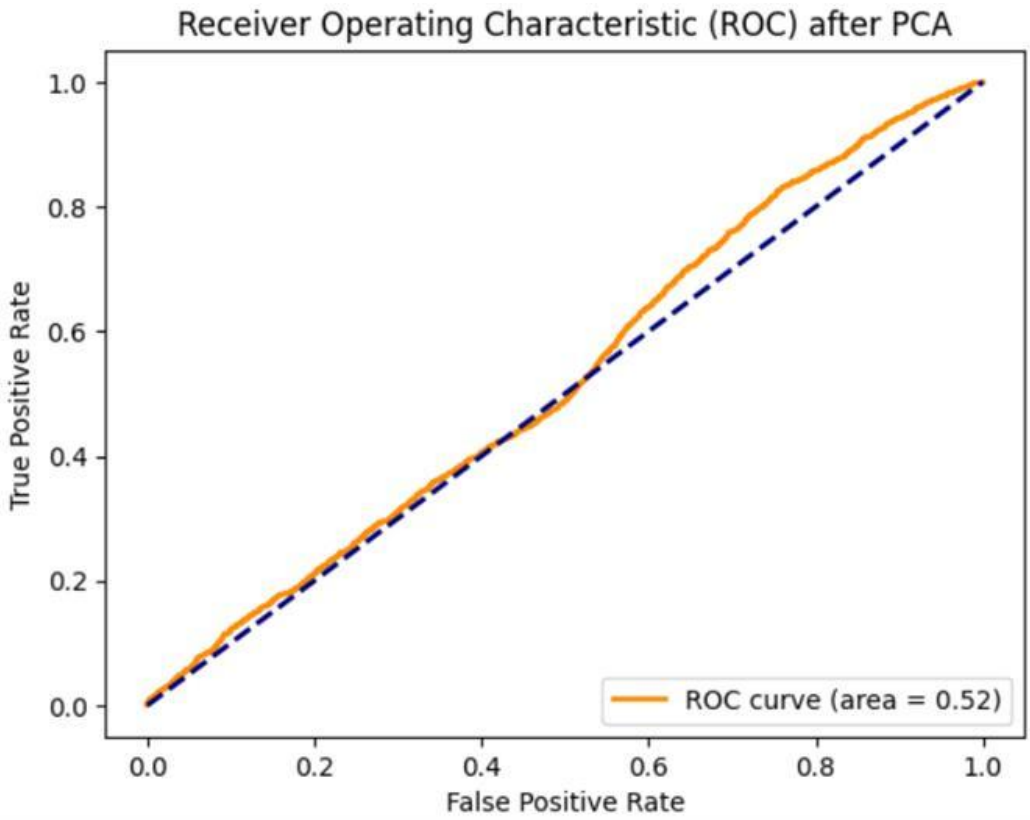
PCA reduces dimensionality by transforming features into fewer principal components. This step aimed to enhance model performance by reducing noise and computational complexity. Although the method does not guarantee better results we were able to see visible improvements in the model after dimensionality reduction.

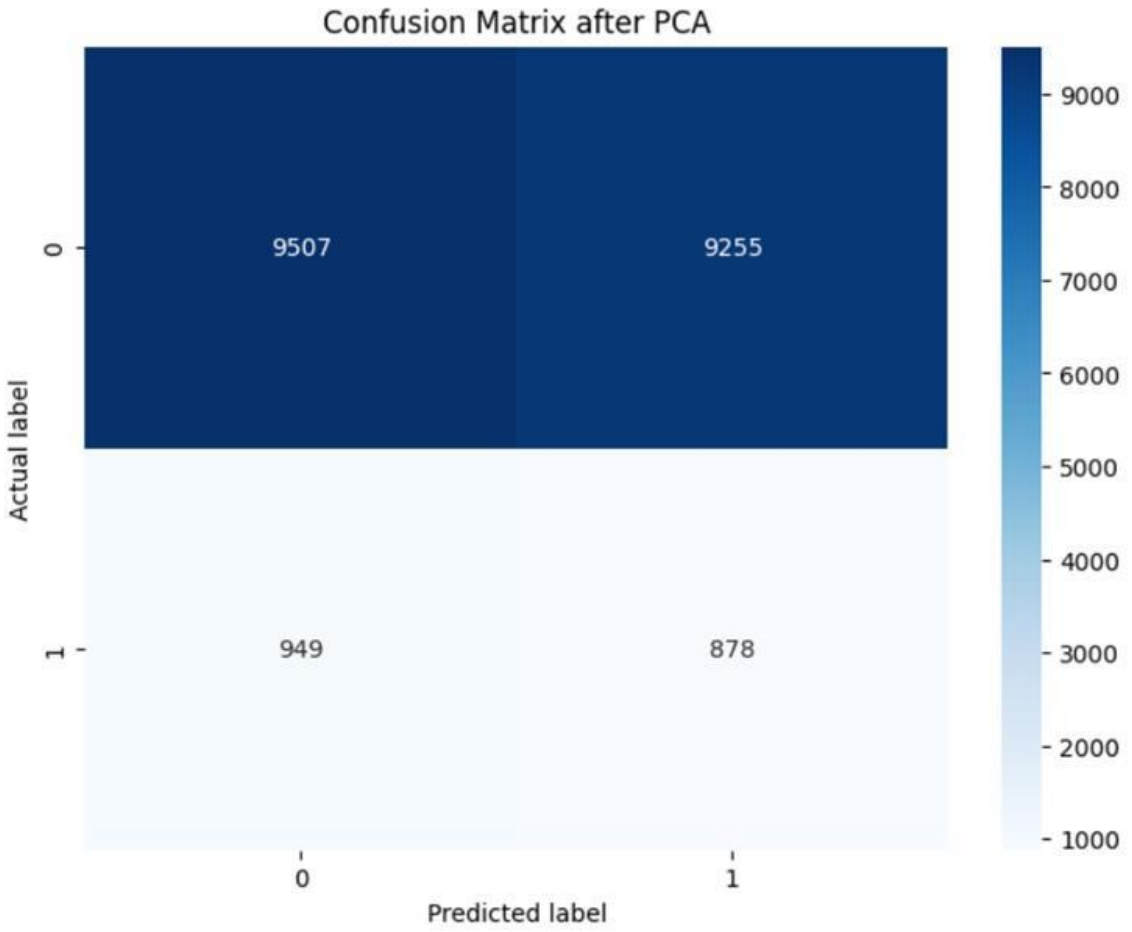
Logistic Regression after PCA:

We reapplied Logistic Regression on the PCA-transformed data to evaluate the impact of reduced dimensionality on model performance.

Post-PCA, accuracy slightly increased to 50.44%, and recall decreased. This suggests a marginal improvement in model balance. The lower cost value post-PCA indicates a more efficient model, likely due to the reduced complexity of the data.

```
Accuracy: 0.5043955510223906
Precision: 0.08664758709168065
Recall: 0.480569239189928883
F1 Score: 0.14682274247491636
```



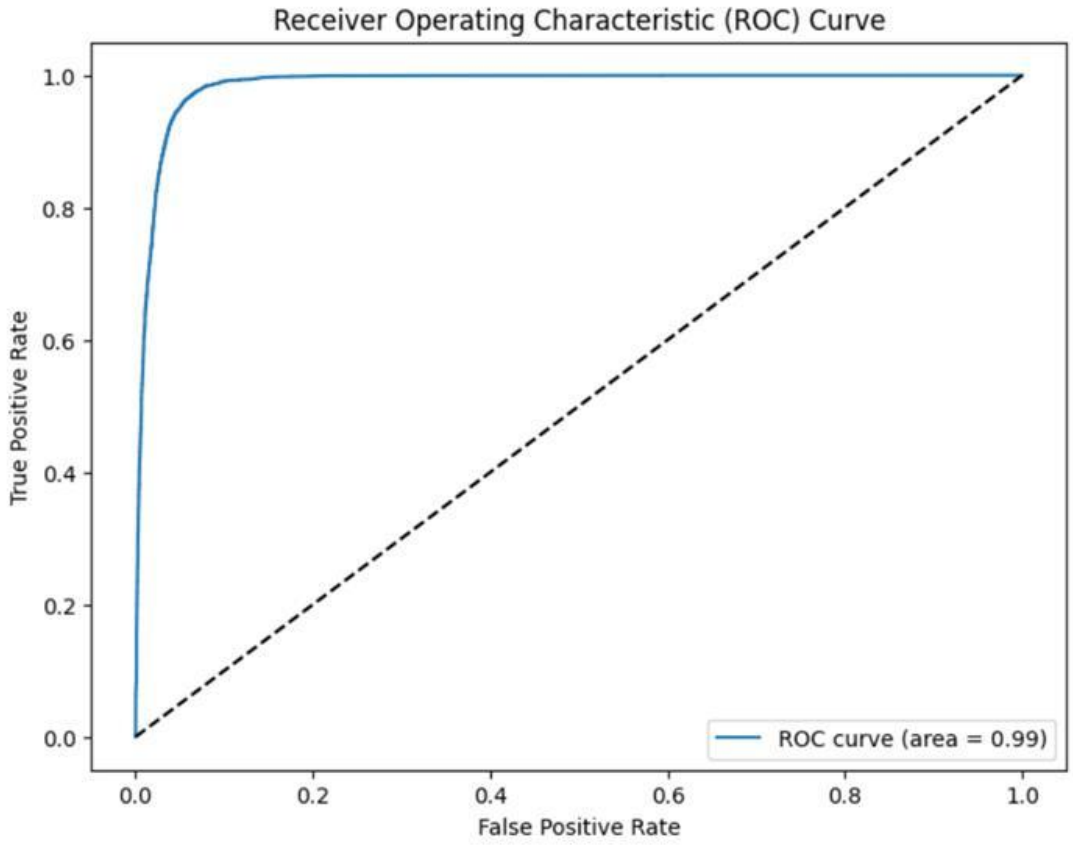


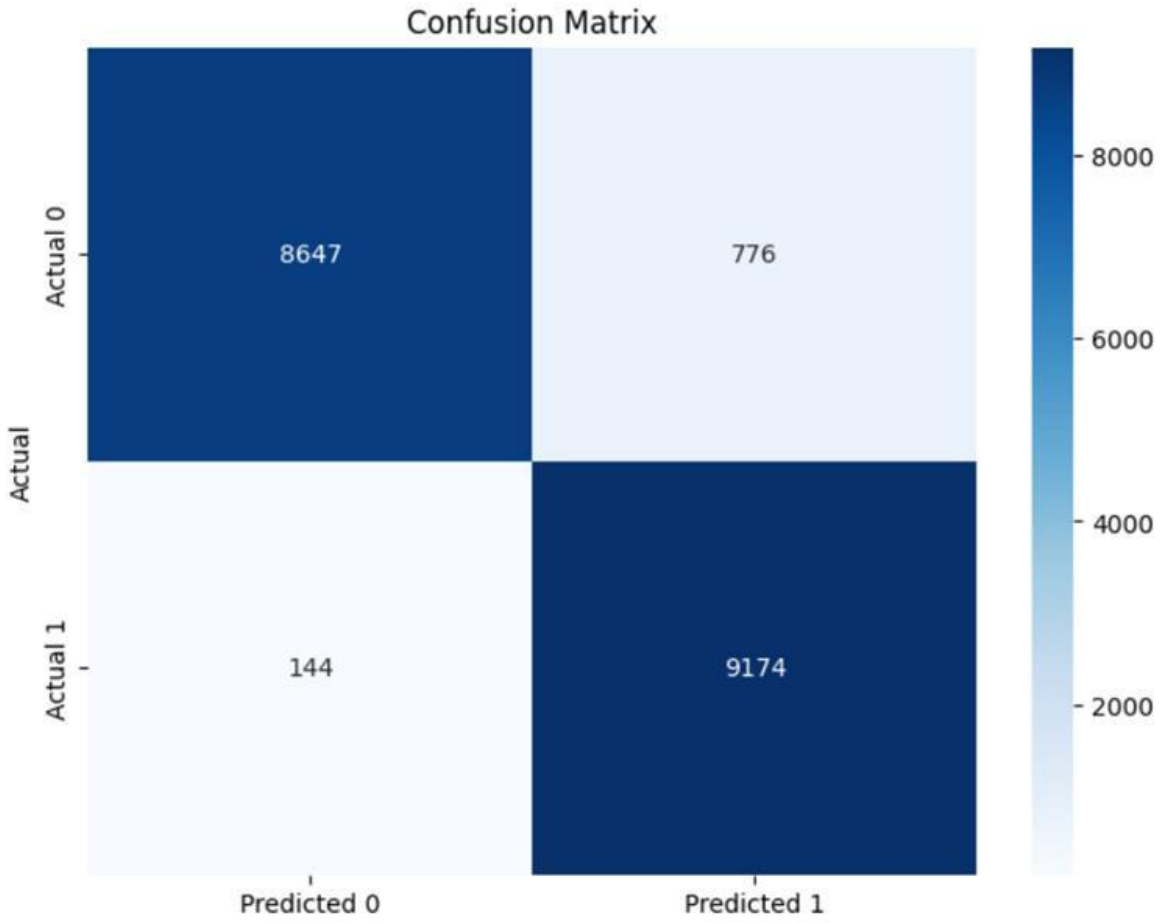
Neural Network after PCA:

Re-evaluating the Neural Network with PCA-transformed data was crucial to determine if dimensionality reduction would aid a complex model.

The model's performance improved drastically, with high accuracy, precision, recall, and an excellent AUC score. This enhancement indicates that the Neural Network could effectively learn patterns in the reduced feature space provided by PCA.

```
Test Accuracy: 0.9509097700229443
Precision: 0.9220100502512563
Recall: 0.9845460399227302
F1 Score: 0.9522524392775586
AUC: 0.9860791562396923
Confusion Matrix:
[[8647  776]
 [ 144 9174]]
```





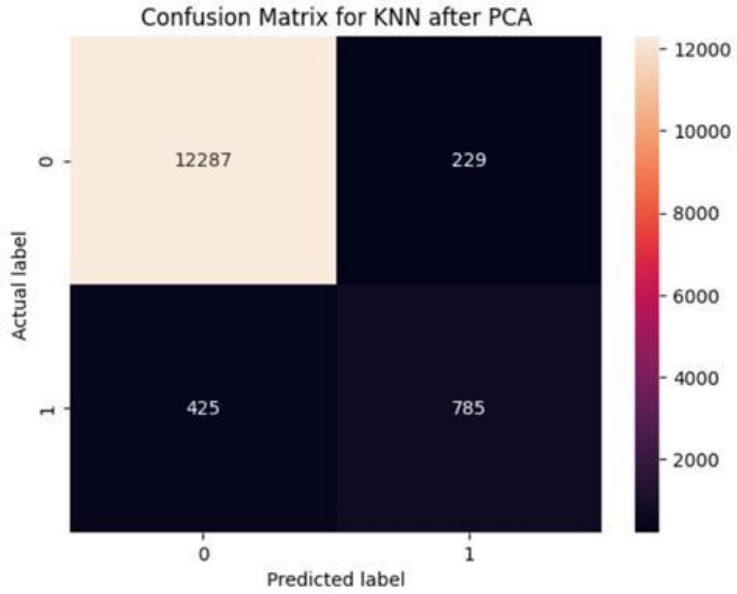
	epoch	loss	val_loss
0	1	0.273622	0.207464
1	2	0.185756	0.167117
2	3	0.158392	0.228889
3	4	0.152999	0.166621
4	5	0.150084	0.168191
5	6	0.145550	0.170129
6	7	0.142817	0.150431
7	8	0.141292	0.145095
8	9	0.138459	0.141228
9	10	0.134781	0.142933

K-NN after PCA:

Finally, applying KNN to the PCA-transformed data helped us assess the impact of PCA on a distance-based algorithm.

There was a significant improvement in all metrics, suggesting that KNN benefited from the clearer structure and reduced noise in the PCA-transformed data.

```
Accuracy: 0.952353198309777, Precision: 0.7741617357001972, Recall: 0.6487603305785123, F1 Score: 0.7059352517985612
```



Model Evaluation:

```
Model Evaluation Metrics:
          Model  Accuracy  Precision    Recall  F1 Score  Cost Value
0     LR no PCA   4480.22    1006.83   6515.47   1744.14      0.8900
1     NN no PCA   5007.19    5007.19  10000.00   6673.06      0.6932
2    KNN no PCA   9045.54     900.90     82.03    150.38      0.1000
3   LR with PCA   5043.95     866.48   4805.69   1468.23      0.7300
4   NN with PCA   9509.09    9220.10   9845.46   9522.52      0.1429
5  KNN with PCA   9523.53    7741.62   6487.60   7059.35      0.0500

Based on the evaluation, the best model for our dataset is the 'KNN with PCA' due to its highest accuracy.
```

Discussion

After reviewing these results, we believe that the Neural Network model with PCA (NN with PCA) is the best model for our dataset. It not only has the highest accuracy but

also maintains an impressive balance between precision and recall, as evidenced by its high F1 score. The low cost value further reinforces its effectiveness in classification tasks. While the KNN with PCA also shows high accuracy, its precision and recall are not as well-balanced as the NN with PCA, making the latter a more robust choice for our needs.

Taking the cost value into account, the Neural Network with PCA (NN with PCA) still appears to be the best model. It achieves a remarkable balance between high accuracy, precision, recall, and a low cost value, indicating it’s not only making accurate predictions but also doing so in a way that minimizes the cost associated with misclassifications. The KNN with PCA also performs well in terms of cost, but it's slightly lower precision and recall compared to the NN with PCA make the latter a more balanced choice overall.