Methodology Report

1. Data Preprocessing & Feature Engineering

The original dataset contained over 3200 numerical features. To prepare the data for modeling:

- **Missing Values**: They were handled using `SimpleImputer(strategy='median'). Another method was also tried using strategy='mean' but the accuracy was decreased.

- **Power Transformation**: PowerTransformer(method='yeo-johnson') was tried to use to normalize skewed distributions. Some of the features were sampled and their histogram were plotted to see their distribution. Most of them were right skewed, while some were bimodal and few were normally distributed. So, Yeo-Johnson method seemed right to use. However, this method created lots of NaN and inf values which caused error for PCA. Alternatively, quantile transformer was also used but it didn't give better accuracy. Log Transformer was also used but didn't give better results.

- **Scaling**: MinMaxScaler() was used to bring all features into the [0, 1] range. StandardScaler() was also tried to see the performance but MinMaxScaler gave better results.

- **Dimensionality Reduction**: Principal Component Analysis (PCA) was used to reduce feature space and mitigate multicollinearity. The n_components was chosen in such a way that the variance is greater and equal to 90%.

Additionally, extreme outliers of few samples were visually identified using box plots and safely clipped to mitigate their effect on transformation and scaling.

2. Model Architectures & Hyperparameters

a. Logistic Regression

- `solver`: `liblinear`, `lbfgs`

- `C`: [0.01, 0.1, 1, 10, 100]

- `class_weight`: `balanced`

- `max_iter`: 1000

b. XGBoost

- `n_estimators`: 200

- `max_depth`: 6

- `learning_rate`: 0.05

- `subsample`: 0.8

- `colsample_bytree`: 0.8

c. Random Forest Classifier

- 'clf__n_estimators': [100, 200],

- 'clf__max_depth': [None, 10, 20],

- 'clf__min_samples_split': [2, 5],

- 'clf__min_samples_leaf': [1, 2],

d. SVM

- 'svm__C': [0.1, 1, 10],

- 'svm__kernel': ['linear', 'rbf'],

- 'svm__gamma': ['scale', 'auto']

Hyperparameters were tuned using `GridSearchCV`.

3. Cross-Validation Strategy

Stratified 5-Fold Cross-Validation was used throughout model training and tuning to ensure balanced class representation and stable evaluation.

Primary Evaluation Metric: ROC-AUC, Accuracy, Precision, Recall, F1-Score

4. Results

Model	Accuracy	ROC AUC	Recall	Specificity	F1 Score
Logistic Regression	0.61	0.6782	0.5238	0.6724	0.5301
Random Forest	0.67	0.6962	0.4762	0.8103	0.5479
XGBoost	0.62	0.6412	0.3095	0.8448	0.4062
SVM	0.62	0.6601	0.5238	0.6897	0.5536

5. Discussion

From the above results, Random Forest gave high accuracy of 0.67. However, the recall was low which means only 47% of actual class 1 are being detected. However, specificity was high. Class 0 can be easily detected by the model. Logistic Regression gave the low accuracy among all. However, other metrics were acceptable. The probability prediction of classes using SVM gave same value to class 0 and class 1 throughout all instances. The model might not be learning any patterns from the data. This typically happens when the model is unable to find separable hyperplanes due to lack of standardization, poor hyperparameter choice, or uninformative features. Overall, preprocessing pipeline was used to ensure robust handling of missing values and skewed features. PCA helped reduce dimensionality without major performance loss.

Limitations:

- Power Transformer become unstable due to extreme outliers.

- Outliers were not analyzed properly.

Future Improvements:

The results obtained from the current models leave room for improvement, particularly in the handling of outliers and missing data.

Outliers were initially addressed using the Interquartile Range (IQR) method. A custom function was implemented to clip values falling below the lower bound (Q1−1.5×IQR) and above the upper bound (Q3+1.5×IQR. While the intention was to bring extreme values into a reasonable range, the clipping function inadvertently removed all rows from the dataset. This happened because the clipping operation was applied in such a way that no row satisfied all feature constraints simultaneously, resulting in a train set with shape (0, 3278)—an empty dataset. This highlights the importance of applying feature-wise outlier handling, rather than filtering rows globally.

In terms of missing data handling, the current preprocessing pipeline used SimpleImputer with median strategy, which is fast and effective but may be suboptimal for complex data distributions. An alternative worth exploring is the K-Nearest Neighbors (KNN) Imputer, which imputes missing values based on the values of neighboring data points. This method can preserve local feature relationships and may lead to improved model learning, especially for algorithms sensitive to data distribution like SVM or logistic regression. With more time and computational resources, these enhancements could potentially lead to significant gains in both model performance and generalization ability.