Risk Prediction on Prudential Life Insurance

Milestone: Project Report

(Venkata Anantha Reddy)

arikatla.v@northeastern.edu

Submission Date: 12th April 2024

Problem Setting: In the insurance sector, accurately predicting risk is fundamental to both efficient operations and profit generation. Historically, this prediction relied on manual underwriting methods that included the use of statistical equations and life expectancy tables. While effective, these methods can be slow and sometimes lack in accuracy and precision. With the evolution of data science, there has been a trend towards adopting machine learning approaches for a more streamlined and precise risk evaluation. This project focuses on applying these advanced technologies in the realm of life insurance, where accurate risk classification is critical for making informed decisions about insurance claims. The main challenge in this endeavor is the incorporation of complex data sets and the creation of models that can effectively capture the diverse aspects of insurance risk.

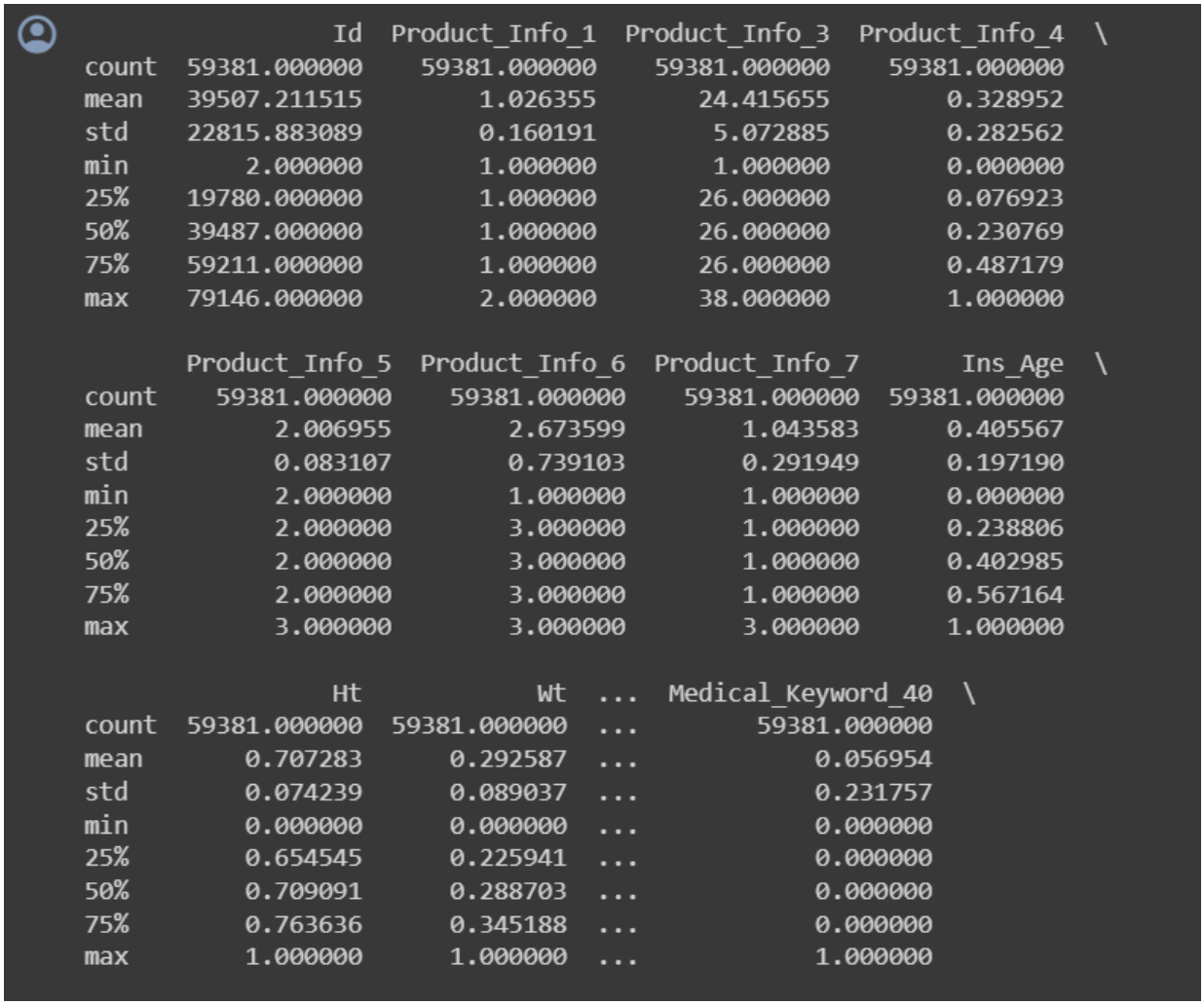
Problem definition: This project is aimed at enhancing risk evaluation in life insurance through predictive analytics. Its goal is to categorize insurance risks by leveraging historical data, while pinpointing the most effective predictive model for the task. A crucial element of the project is to improve the clarity and comprehensibility of machine learning models for all involved parties. It involves assessing different tree-based classifiers, including Decision Tree, Random Forest, and XGBoost, with the objective of identifying the model that provides the most precise risk categorization in insurance. The project faces the challenge of finding a middle ground between the sophistication of machine learning methods and the necessity for transparent and understandable outcomes that comply with both industry norms and regulatory requirements.

Data Sources: Prudential Life Insurance | Kaggle

Data Description: For this Project, the selected dataset is the Prudential Life Insurance Assessment from Kaggle. It is a comprehensive dataset featuring 59,381 rows, encompassing a broad range of 128 columns. These columns are a diverse mix of variable types, including nominal – (Product_Info_1, Product_Info_2), continuous – (Wt, BMI, Employment_Info_6), and discrete – (Medical_History_1, Medical_History_10). This rich dataset offers a robust foundation

for analyzing and predicting life insurance risk factors with a high level of granularity and precision.

Data Exploration: To thoroughly explore and visualize, we started with the exploratory data analysis (EDA) phase. We began by gaining a comprehensive understanding of the dataset's structure, features, and their types. This initial step provided us with insights into the data's overall composition and layout, setting a solid foundation for subsequent analyses.



	Id	Product_Info_1	Product_Info_3	Product_Info_4	\
count	59381.000000	59381.000000	59381.000000	59381.000000	
mean	39507.211515	1.026355	24.415655	0.328952	
std	22815.883089	0.160191	5.072885	0.282562	
min	2.000000	1.000000	1.000000	0.000000	
25%	19780.000000	1.000000	26.000000	0.076923	
50%	39487.000000	1.000000	26.000000	0.230769	
75%	59211.000000	1.000000	26.000000	0.487179	
max	79146.000000	2.000000	38.000000	1.000000	

	Product_Info_5	Product_Info_6	Product_Info_7	Ins_Age	\
count	59381.000000	59381.000000	59381.000000	59381.000000	
mean	2.006955	2.673599	1.043583	0.405567	
std	0.083107	0.739103	0.291949	0.197190	
min	2.000000	1.000000	1.000000	0.000000	
25%	2.000000	3.000000	1.000000	0.238806	
50%	2.000000	3.000000	1.000000	0.402985	
75%	2.000000	3.000000	1.000000	0.567164	
max	3.000000	3.000000	3.000000	1.000000	

	Ht	Wt	...	Medical_Keyword_40	\
count	59381.000000	59381.000000	...	59381.000000	
mean	0.707283	0.292587	...	0.056954	
std	0.074239	0.089037	...	0.231757	
min	0.000000	0.000000	...	0.000000	
25%	0.654545	0.225941	...	0.000000	
50%	0.709091	0.288703	...	0.000000	
75%	0.763636	0.345188	...	0.000000	
max	1.000000	1.000000	...	1.000000	

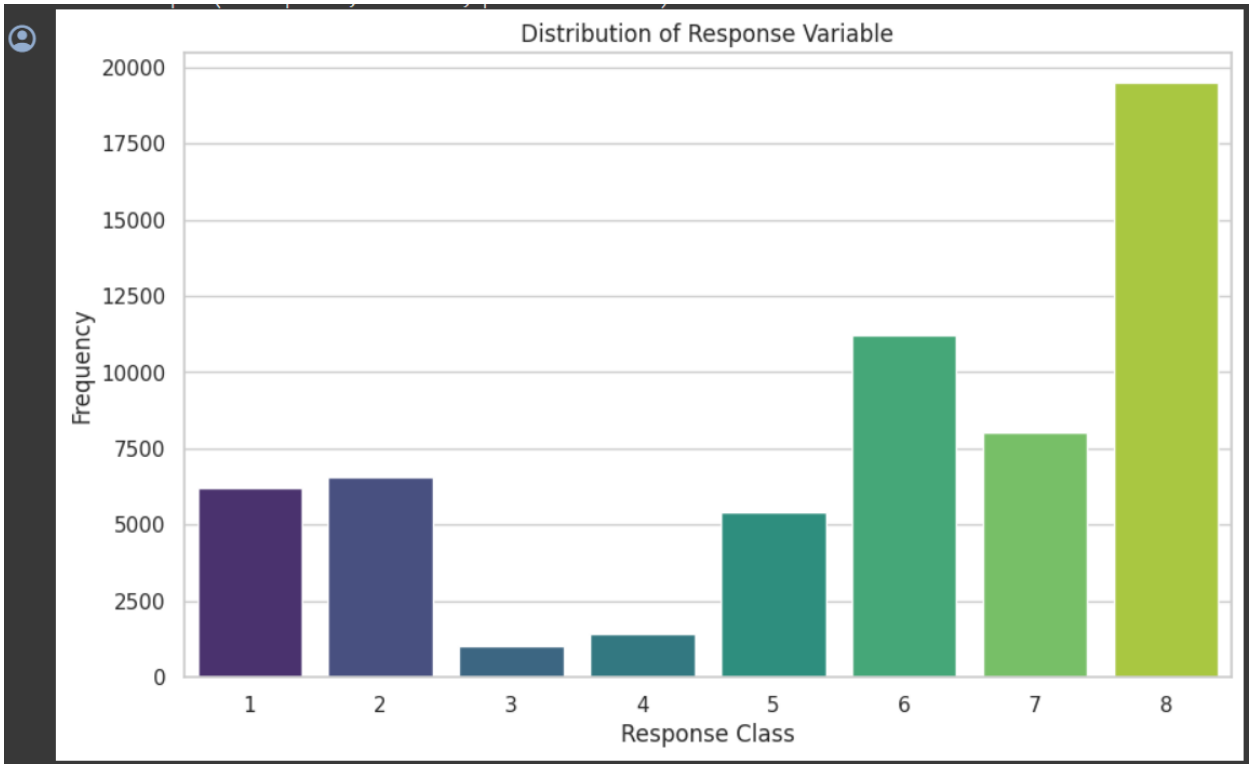
	Medical_Keyword_41	Medical_Keyword_42	Medical_Keyword_43	\
count	59381.000000	59381.000000	59381.000000	
mean	0.010054	0.045536	0.010710	
std	0.099764	0.208479	0.102937	
min	0.000000	0.000000	0.000000	
25%	0.000000	0.000000	0.000000	
50%	0.000000	0.000000	0.000000	
75%	0.000000	0.000000	0.000000	
max	1.000000	1.000000	1.000000	

	Medical_Keyword_44	Medical_Keyword_45	Medical_Keyword_46	\
count	59381.000000	59381.000000	59381.000000	
mean	0.007528	0.013691	0.008488	
std	0.086436	0.116207	0.091737	
min	0.000000	0.000000	0.000000	
25%	0.000000	0.000000	0.000000	
50%	0.000000	0.000000	0.000000	
75%	0.000000	0.000000	0.000000	
max	1.000000	1.000000	1.000000	

	Medical_Keyword_47	Medical_Keyword_48	Response
count	59381.000000	59381.000000	59381.000000
mean	0.019905	0.054496	5.636837
std	0.139676	0.226995	2.456833
min	0.000000	0.000000	1.000000
25%	0.000000	0.000000	4.000000
50%	0.000000	0.000000	6.000000
75%	0.000000	0.000000	8.000000
max	1.000000	1.000000	8.000000

- **Id:** Unique identifier for each entry in the dataset, as indicated by the mean value being around the halfway point between the minimum and maximum and a standard deviation that suggests a uniform distribution.
- **Product_Info_1, Product_Info_3, Product_Info_4:** These features represent categorical variables given that the minimum and maximum values are in a small range. The mean and percentiles indicate the distribution and skewness of these categories.
- **Product_Info_5, Product_Info_6, Product_Info_7:** Similar to the above, these are categorical variables with a limited range of values.
- **Ins_Age:** Represents the insurance age of the individuals. The data ranges from 0 to 1, which is normalized. The mean is 0.405, suggesting that the average normalized age is slightly less than the midpoint of the range.

- **Ht, Wt:** Represent height and weight, respectively. They are also normalized as their ranges are from 0 to 1.
- **Medical_Keyword_40 to Medical_Keyword_48:** These features have binary characteristics (0s and 1s), which represent the presence or absence of certain medical conditions or keywords related to a medical condition in the dataset.
- **Response:** This is the target variable which has a wider range of values (1 to 8). The mean response is approximately 5.64, suggesting that the average response is between class 5 and class 6. The standard deviation is about 2.45, indicating there's a wide spread of responses across the different classes.



The bar chart represents the frequency distribution of a response variable categorized into eight classes. The classes (1-8) correspond to risk levels. The y-axis shows the number of occurrences (frequency) of each class in the dataset.

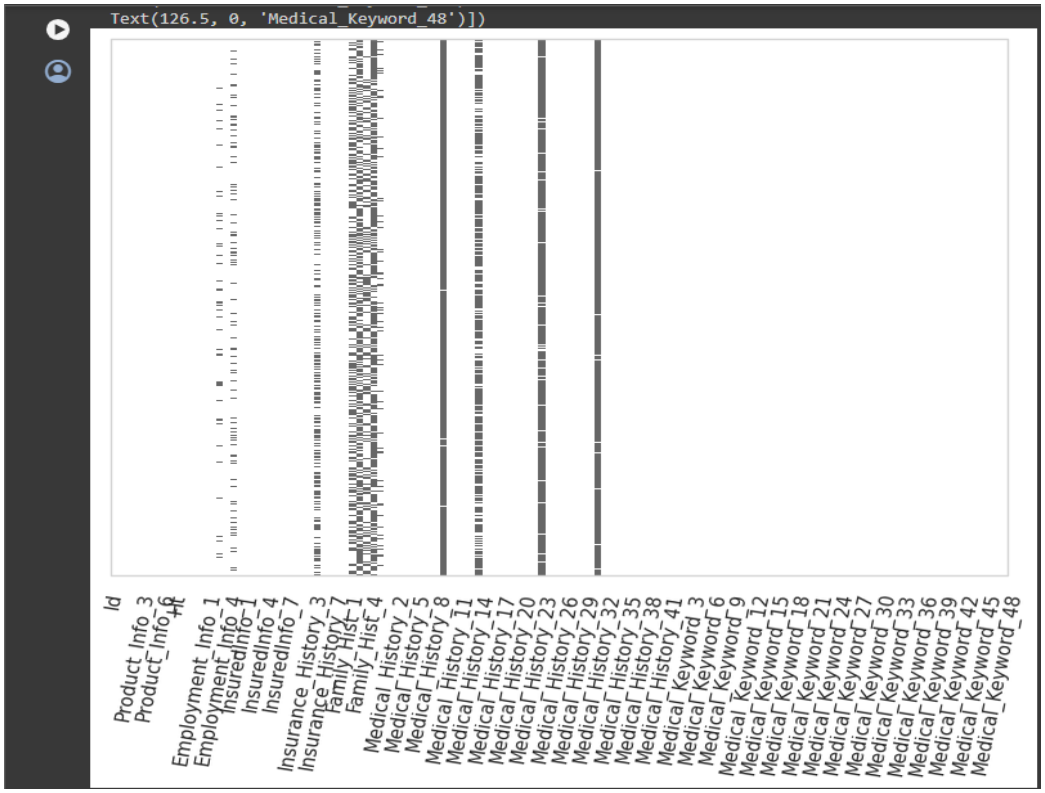
Key Observations:

- Class 8 has the highest frequency, it is the most common outcome in the dataset.
- Class 3 has the lowest frequency, indicating it is the least common outcome.
- The distribution is not uniform and suggests a skewed dataset, with some classes occurring much more frequently than others.
- Classes 1 and 2 have similar frequencies, as do classes 4 and 5, which might indicate some sort of relationship or similarity between these pairs of classes.

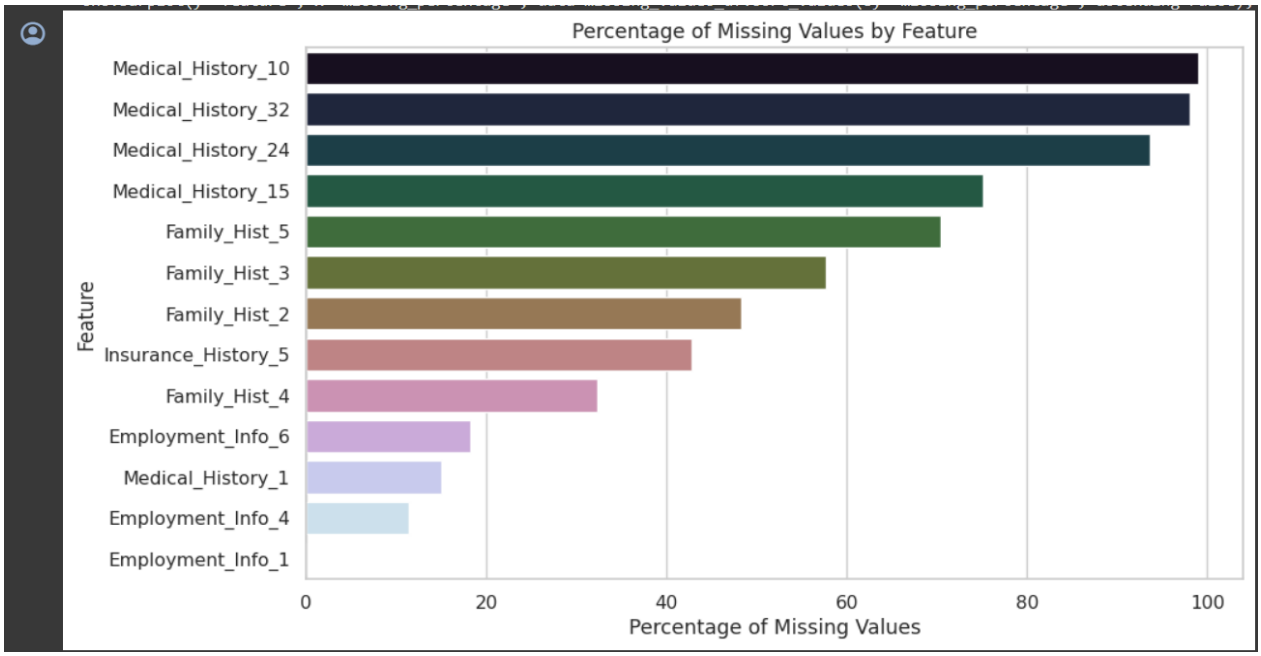
Data Mining Tasks:

Handling Missing Values:

- To handle the missing and null values, the first step was to identify them. The missing value matrix was utilized for easily identifying the features in the dataset with missing values. Once the missing values are identified, they were either removed or fill them with one of the central tendencies based on the data distribution.



From the missing value matrix where X - axis represents features from the dataset and Y - axis represents the number of instances in the dataset by analysing the missing matrix, it was discovered that there were some features with missing values. Subsequently, a list of features with missing values was compiled, along with the corresponding missing percentage.



The bar chart displays various features on the y-axis, representing different variables in the dataset, such as medical history, family history, insurance history, and employment information. The x-axis indicates the percentage of missing values for each feature.

Key Observations:

- Medical_History_10 has the highest percentage of missing values, which is nearly 100%, meaning that almost all entries for this feature are missing.
- Employment_Info_1 has the lowest percentage of missing values, indicating that this feature is the most complete in the dataset.
- There is a wide variation in the completeness of the dataset, with some features being almost entirely missing, while others are almost fully present.

Handling Missing Values:

- For columns with high missing values (e.g., more than 50%), they were considered for removal from the dataset. This decision was made because having too many missing values can compromise the integrity of the data.
- For other columns with missing values:
    - ✓ Numerical features were imputed using median imputation, which involved filling in missing values with the median of the respective column.

Outlier Detection and Treatment:

- Outliers in continuous variables were identified using the Interquartile Range (IQR) method.
- Depending on the nature of the outliers and their impact on the model, we treated it through capping technique to mitigate their influence.

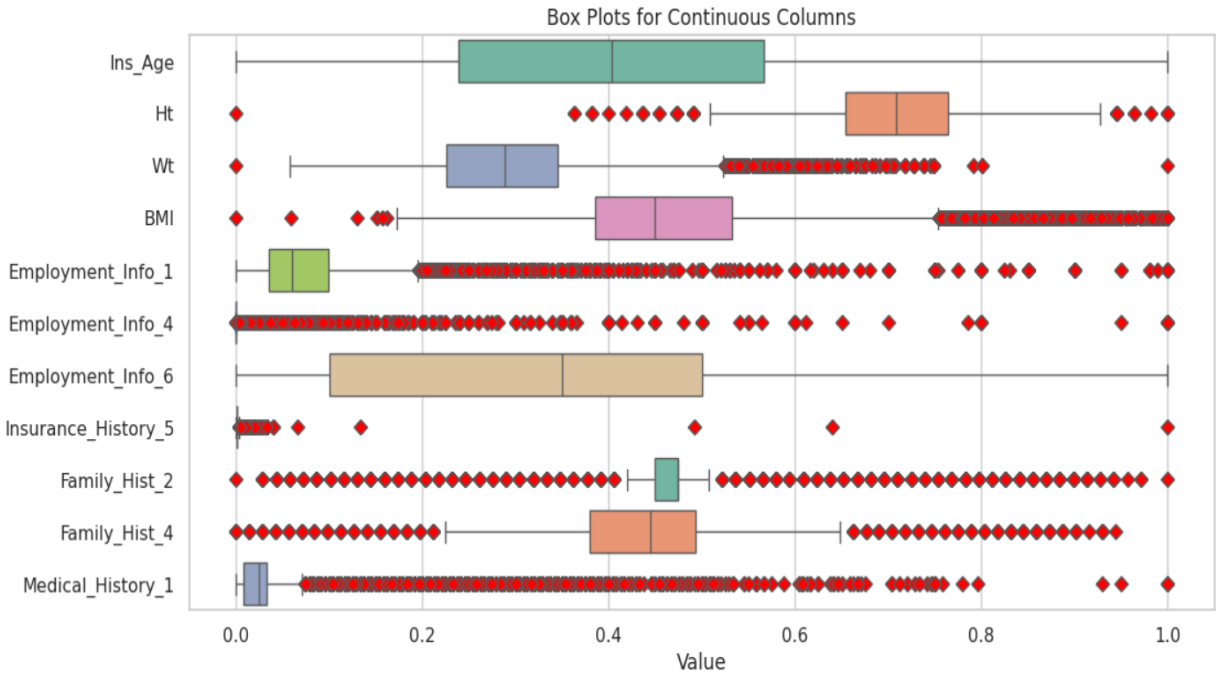
We have identified outliers in a few key numerical features (such as **Ins_Age, Ht, Wt,** and **BMI**) based on this method. We then decided on an appropriate treatment strategy, such as capping outliers.

For key numerical features, the analysis identified outliers as follows:

	lower_bound	upper_bound	outliers_count
Ins_Age	-0.253731	1.059701	0.0
Ht	0.490909	0.927273	72.0
Wt	0.047071	0.524059	854.0
BMI	0.164504	0.753870	1634.0

- **Ins_Age:** No outliers were detected based on the IQR method, indicating age data falls within a reasonable range.

- **Ht (Height):** 72 outliers detected. These represent unusually tall or short individuals relative to the general population in the dataset.
- **Wt (Weight):** 854 outliers detected. This suggests a significant number of individuals with weights outside the typical range.
- **BMI (Body Mass Index):** 1634 outliers detected, indicating a considerable number of entries with BMI values considered outside the healthy or typical range.



Given these findings, a cautious approach to manage outliers is implemented to avoid distorting the underlying distribution of these key health-related features.

- **Capping:** We have capped outliers for Ht, Wt, and BMI at their respective lower and upper bounds identified earlier. This treatment has modified only the outlier values, aligning them with the bounds, while leaving all other values unchanged.

After capping outliers for Ht (Height), Wt (Weight), and BMI (Body Mass Index), the min and max values for these features have been adjusted to fall within the identified bounds:

- **Ht:** Now ranges from 0.490909 to 0.927273, removing extreme height values.
- **Wt:** Adjusted to range from 0.047071 to 0.524059, eliminating extreme weight outliers.

- **BMI:** Modified to be between 0.164504 and 0.753870, addressing BMI outliers.

These adjustments have ensured that the data retains its original distribution characteristics while reducing the influence of extreme outlier values.

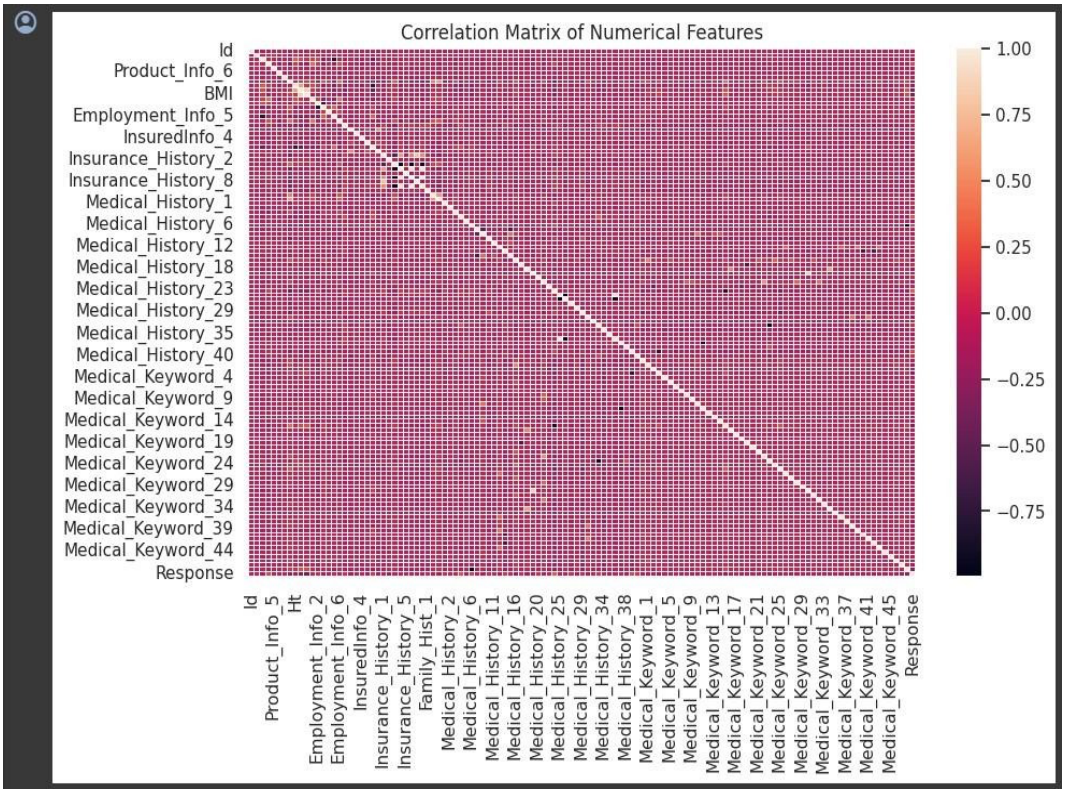
Encoding Categorical Variables:

Before moving on to dimensionality reduction, we encoded the Product_Info_2 categorical variable, as machine learning algorithms require numerical input. Given Product_Info_2 appears to be a nominal categorical feature without an inherent order, one-hot encoding is a suitable approach. Therefore, the categorical variable “Product_Info_2” was encoded using one-hot encoding to prepare it for modeling.

Dimensionality Reduction:

- Given the large number of features, especially medical keywords, dimensionality reduction techniques were considered.

- We'll start with a correlation analysis to identify highly correlated features that might be candidates for reduction. This step has helped us understand if Principal Component Analysis (PCA) or feature selection are necessary and beneficial for this dataset.



- The heatmap visualizes the correlation matrix for the numerical features in the dataset.

- Some features exhibit moderate to high correlations with others, suggesting potential redundancy.
- The overall pattern suggests that while there are correlations, they might not be strong enough across the board to justify aggressive dimensionality reduction for all features.
- Given the observations and the goal of preserving as much information as possible for predictive modeling, a cautious approach to dimensionality has been taken. Instead of applying broad dimensionality reduction techniques like PCA at this stage, we feel it is more beneficial to focus on feature selection based on model performance and importance rankings.

Performing PCA:

1. **Feature Selection and Exclusion:**
    - Initially, non-numeric columns and the target variable are removed from the dataset. This is crucial because PCA is sensitive to the variable types and works only with numeric data. The target variable is excluded to prevent any bias in dimension reduction aimed solely at feature explanation.
2. **Standardization of Features:**
    - Before applying PCA, it is vital to standardize the data. This step involves transforming the data so that each feature has a mean of zero and a standard deviation of one. Standardization is essential because PCA is affected by the scale of the data. Large scale differences between features can unduly influence the principal components generated, skewing the results.
3. **Implementing PCA:**
    - PCA is applied using the PCA class from sklearn.decomposition. The n_components parameter is set to 0.95, indicating that the PCA should select enough components to explain at least 95% of the total variance in the dataset. This approach balances dimensionality reduction with information retention, making the dataset simpler and often improving the performance of subsequent learning algorithms.
4. **Transforming the Data:**
    - The standardized data is transformed into a new coordinate system with the axes now being the principal components. These components are orthogonal and ordered such that the first

few retain most of the variation present in the original data. The transformation to principal components reduces the dimensionality of the data while attempting to preserve as much of the data's original information as possible.

5. **Analysis of Components:**
    - The output includes the exact number of principal components selected to cover the specified 95% variance. This is an important metric as it provides insight into the complexity of the underlying data structure.
    - A summary DataFrame, pcSummary, is created to detail the standard deviation, proportion of variance, and cumulative proportion of variance for each principal component. This summary is pivotal for understanding which components contribute most to the variance and are thus most informative.

Data Mining Models:

Logistic Regression (LR)

Logistic Regression is widely used for binary classification problems, offering a probabilistic perspective. It models the log-odds of the dependent variable as a linear combination of the independent variables. This model is particularly useful because it provides not just a prediction but also the probability of that prediction, which can be crucial for decision-making processes where you need to understand the certainty of your predictions. The logistic function, also known as the sigmoid function, ensures that the output probabilities range between 0 and 1.

Mathematically, it is expressed as:

$$P(y = 1|\varkappa) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 \varkappa_1 + \cdots + \beta_k x_k)}}$$

where $\beta_0, \beta_1, .., \beta_k$ are learned from the training data and provide the weights of importance given to the respective features.

Decision Tree (DT)

A Decision Tree is a versatile model that can be used for both classification and regression tasks. It is intuitive and easy to understand because it mirrors human decision-making more closely than other models, breaking down a dataset into smaller subsets while simultaneously developing an associated decision tree. The final model is a series of decision rules that can be easily visualized and interpreted, making Decision Trees particularly appealing for operational use where

understanding the rationale behind a prediction is important.

Although not defined by a singular equation, the construction of a decision tree can be understood as a series of branching decisions:

$$f(x) = \begin{cases} c_1 \; if \; condition_1 \\ c_2 \; if \; condition_2 \\ \quad . \\ \quad . \\ \quad . \\ c_n \; otherwise \end{cases}$$

where the conditions are derived from the training data to best separate the target classes or predict a continuous value.

Random Forest (RF)

Random Forest is an ensemble technique that builds upon the simplicity of decision trees and enhances their performance by combining the predictions of multiple trees to decide the final output. This approach significantly reduces the risk of overfitting, a common problem with individual decision trees, and provides more accurate and stable predictions. The randomness injected through features and samples when building individual trees ensures that the bias of the forest is not significantly greater than that of a single tree while reducing the variance.

For classification and regression, the Random Forest aggregates (by averaging or voting) the predictions from multiple trees:

$$f(x) = \frac{1}{T} \sum_{t=1}^{T} f_t(x)$$

Where each f_t represents an individual tree in the ensemble.

XGBoost

XGBoost stands for eXtreme Gradient Boosting and represents a cutting-edge implementation of gradient-boosted trees designed for speed and performance. It is a highly flexible and versatile tool that can address a broad spectrum of regression, classification, and ranking problems. XGBoost improves upon the standard gradient boosting framework through system optimization

and algorithmic enhancements, offering built-in handling of missing values, tree pruning, and regularized learning to prevent overfitting.

Its formulation is an ensemble of K additive functions:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i)\,, f_k \in F$$

which collectively contribute to the final prediction, with each function f_k representing an individual decision tree.

Linear Support Vector Classifier (Linear SVC)

The Linear Support Vector Classifier provides a powerful, maximum margin method for learning linear classifiers. By constructing a hyperplane in a multidimensional space that separates the different classes, the SVC seeks the hyperplane with the largest minimum distance to the training samples. This margin maximization offers an inherent regularization feature, reducing the risk of overfitting. The Linear SVC is particularly well-suited for high-dimensional datasets, where it is computationally more efficient than its kernelized counterparts.

The optimization problem for the linear SVC can be represented as:

$$\min_{w,b} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \max\,(0,1 - y_i(w^T x_i + b))$$

Where w is the weight vector, b is the bias and C provides a trade-off between achieving a low training error and a low testing error (generalization).

Multiple Linear Regression (MLR)

Multiple Linear Regression extends simple linear regression to predict an outcome based on multiple predictors. It assumes a linear relationship between the independent variables and the dependent variable, providing a quantified insight into the strength and form of these relationships. MLR is widely used in fields like economics, social sciences, and engineering for its

interpretability and simplicity.

The MLR model is mathematically expressed as:

$$y = \beta_0 + \beta_1 \varkappa_1 + \cdots + \beta_k x_k + \in$$

Where ∈ represents the error term, capturing any variation in y unexplained by the predictors.

Artificial Neural Networks (ANN)

Artificial Neural Networks are inspired by the biological neural networks that constitute animal brains. They consist of layers of interconnected nodes, each representing a neuron. The connections, or weights, between these nodes are adjusted during training to minimize the prediction error of the network. ANNs are capable of modeling complex, nonlinear relationships in the data through their layered structure and activation functions, making them extremely flexible and powerful for a wide range of tasks.

The mathematical representation of a basic ANN with a single hidden layer is:

$$y = f(\sum_{j=1}^{m} w_j^{(2)} f\left(\sum_{i=1}^{n} w_{ij}^{(1)} \varkappa_i + b_j^{(1)}\right) + b^{(2)})$$

These models, from the straightforward LR to the complex ANN, provide a toolbox for addressing various predictive modeling challenges, each with its strengths, assumptions, and contexts in which it is most effective.

Based on the project's criteria and requirements, we have decided to choose the following models:

1. **Decision Tree (DT)**: This model is ideal for its simplicity and ease of interpretation. Decision Trees can be easily visualized, which aids in explaining the decision-making process to stakeholders, satisfying the need for transparent and understandable outcomes that comply with industry norms and regulatory requirements.

2. **Random Forest (RF)**: We have selected Random Forest because it improves upon the accuracy of individual decision trees by building an ensemble. This model is less prone to overfitting and is capable of handling high-dimensional data, which is likely to result in a more precise risk categorization—a critical objective of the insurance project.

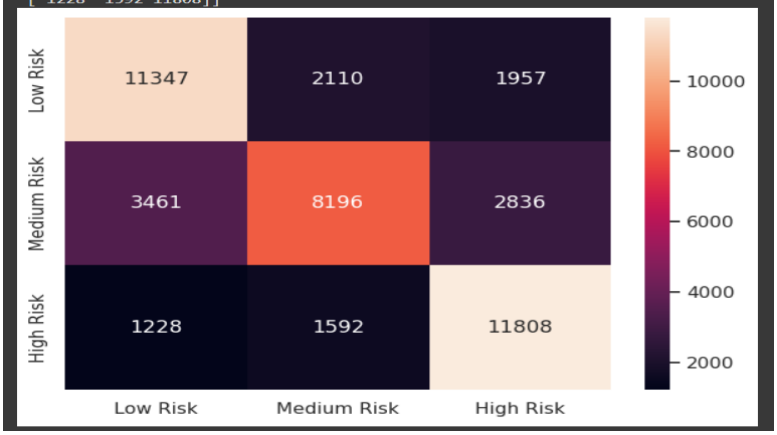
3. **XGBoost**: The inclusion of XGBoost is due to its advanced capabilities in handling large volumes of data with high efficiency and its ability to produce sophisticated models. The regularization aspect of XGBoost helps in avoiding overfitting, aligning with the project's goal of finding a balance between sophisticated machine learning methods and the need for transparency and intelligibility in the outcomes.
4. **Decision Tree**: The selection of Decision Tree is driven by its straightforward structure and intuitive decision-making process. It is particularly useful for this project because it facilitates easy interpretation and explanation of the decisions it generates, which is crucial for transparency. Moreover, Decision Trees naturally manage both numerical and categorical data and are effective even with minimal data preprocessing. Their susceptibility to overfitting can be mitigated by employing techniques like pruning, ensuring that the model remains robust and generalizable across different datasets.
5. **Gradient Boosting**: Gradient Boosting is chosen for its exceptional ability to improve predictive accuracy through the incremental correction of errors by successive models. This method optimizes a specified loss function using a gradient descent approach, which is key to refining complex models. Although computationally intensive, its application in this project is justified by the significant boost it provides in model performance, especially when dealing with varied and complex data structures. Regularization features, such as learning rate adjustments and tree depth control, contribute to balancing the model's sophistication with the necessity to prevent overfitting.

Performance Evaluation and Interpretation

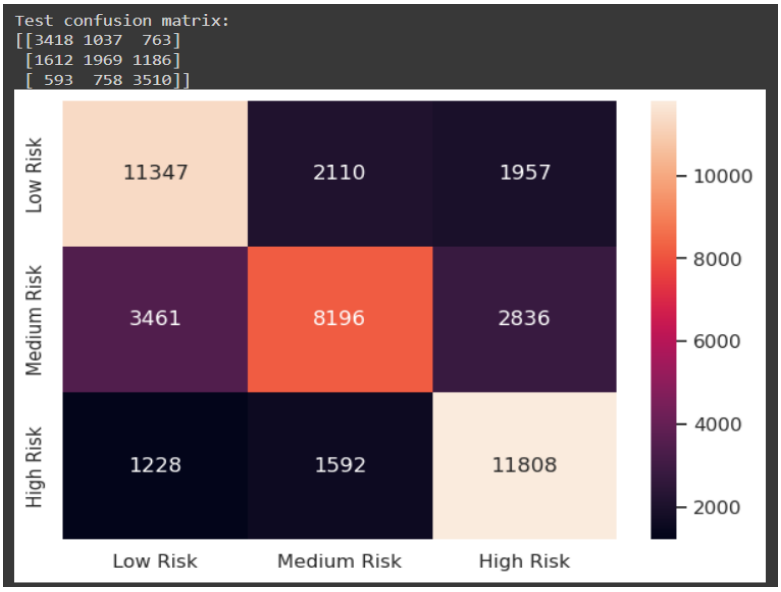
1. **Random Forest:**

   Random Forest (with PCA)

```
The accuracy on train dataset is 0.7039631750308746
The accuracy on test dataset is 0.5992860029637613

Train confusion matrix:
[[11347  2110  1957]
 [ 3461  8196  2836]
 [ 1228  1592 11808]]
```

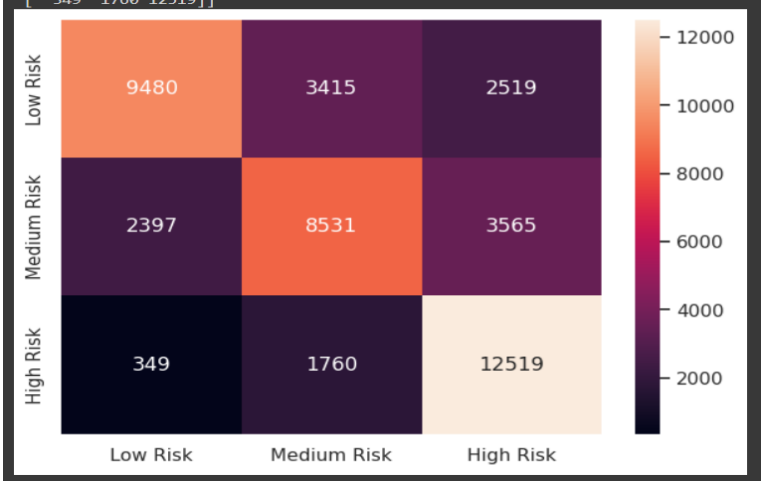



```
Test confusion matrix:
[[3418 1037  763]
 [1612 1969 1186]
 [ 593  758 3510]]
```

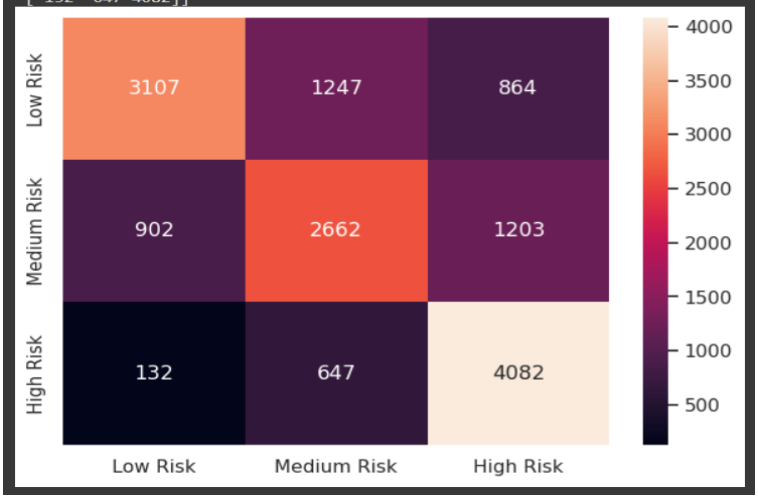


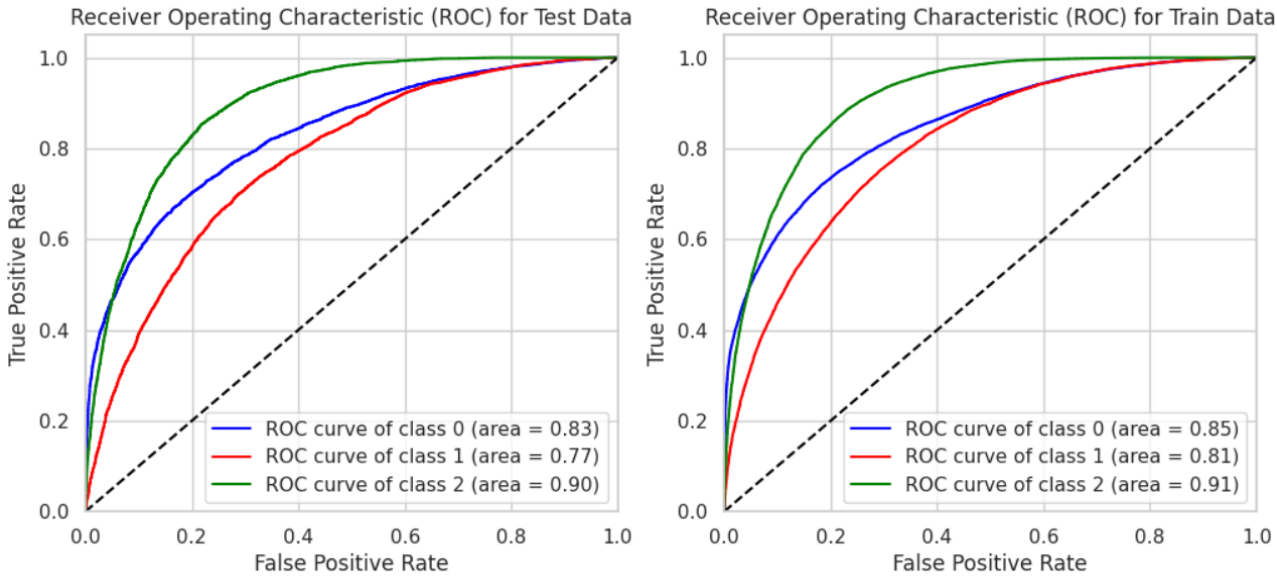
Without PCA

```
The accuracy on train dataset is 0.6855282362187044
The accuracy on test dataset is 0.663545736225245 8

Train confusion matrix:
[[ 9480  3415  2519]
 [ 2397  8531  3565]
 [  349  1760 12519]]
```



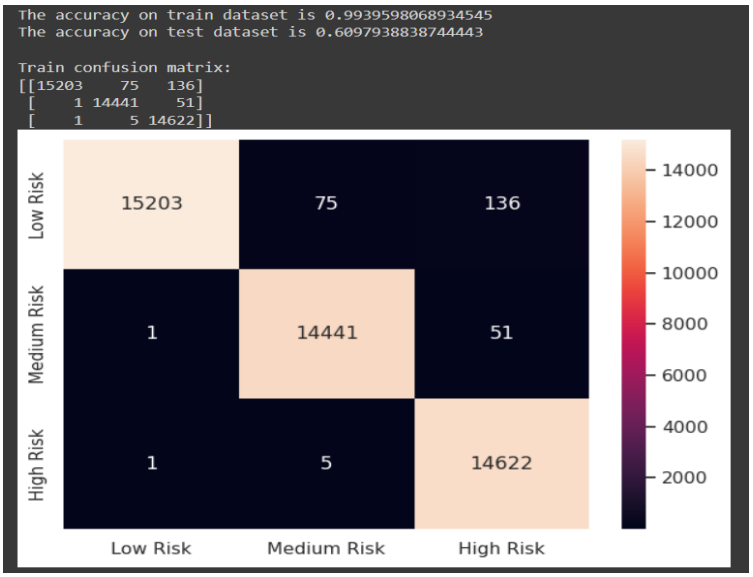

```
Test confusion matrix:
[[3107 1247  864]
 [ 902 2662 1203]
 [ 132  647 4082]]
```

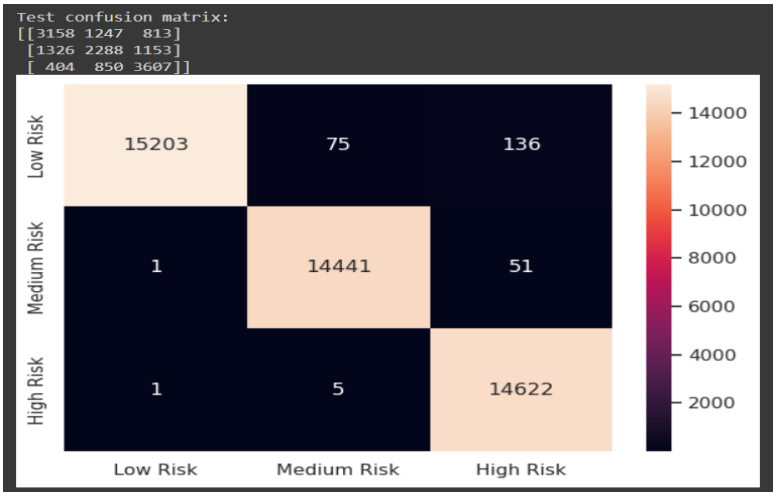




2. **XG Boosting:**

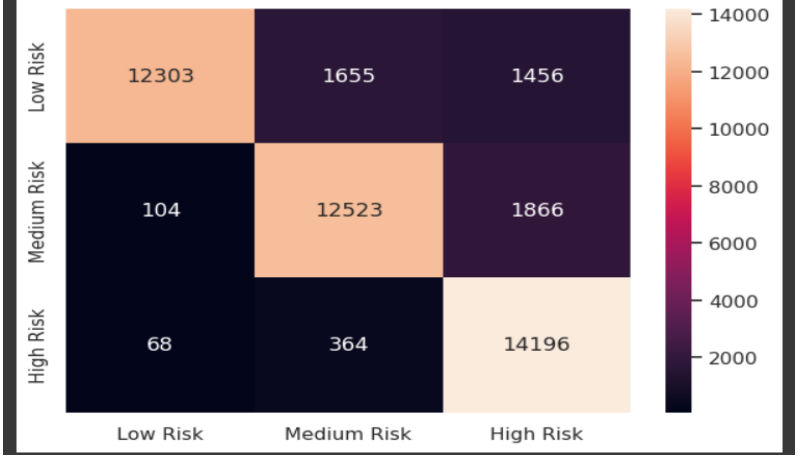
XG Boosting (with PCA)



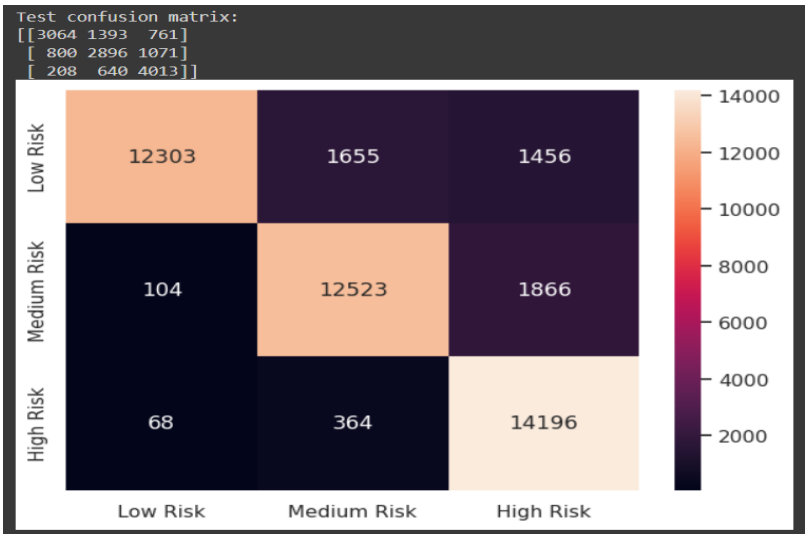


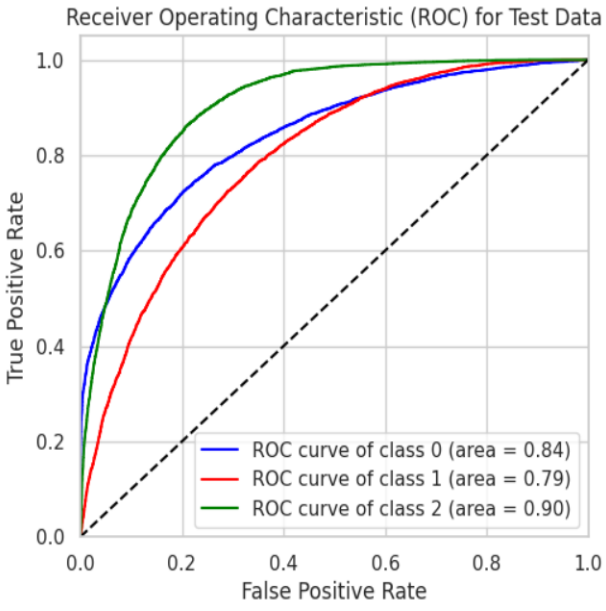
Without PCA

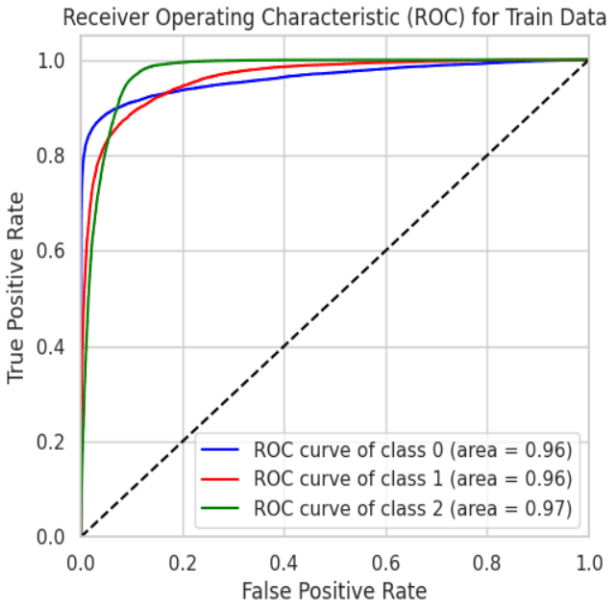
```
The accuracy on train dataset is 0.8762097226900191
The accuracy on test dataset is 0.6717634379630877

Train confusion matrix:
[[12303  1655  1456]
 [  104 12523  1866]
 [   68   364 14196]]
```




```
Test confusion matrix:
[[3064 1393  761]
 [ 800 2896 1071]
 [ 208  640 4013]]
```



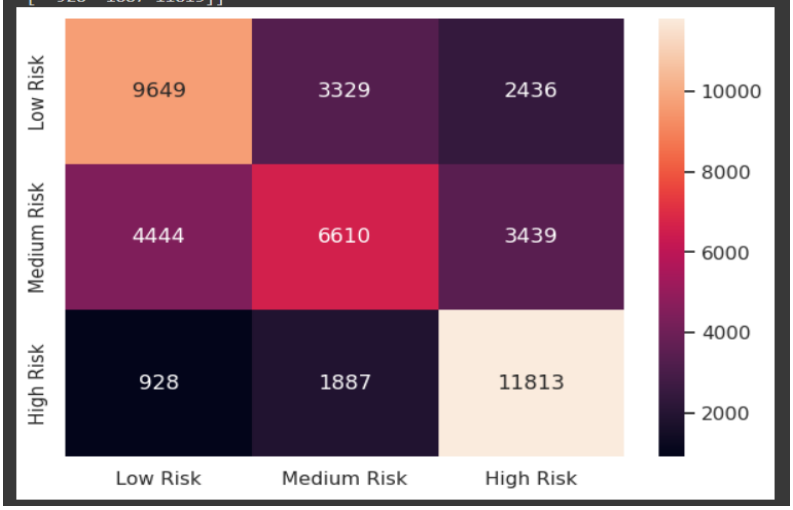




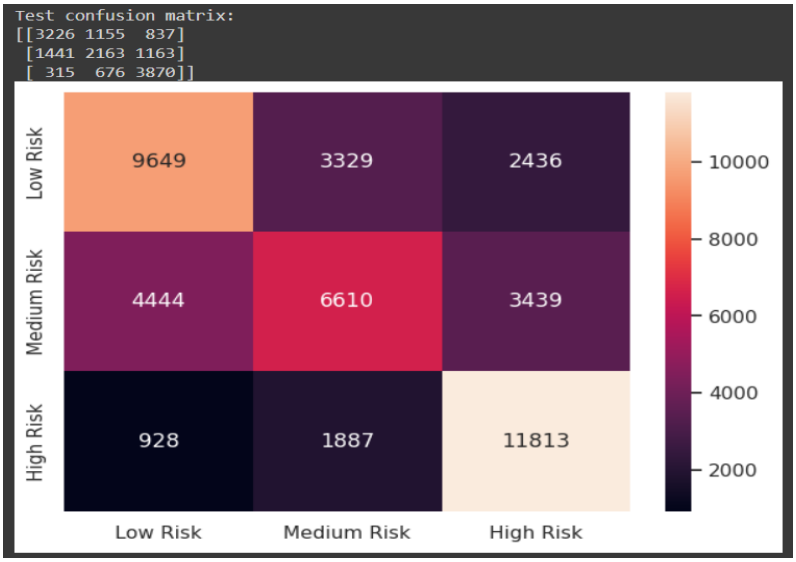
3. **Logistic Regression:**

Logistic Regression (with PCA)

```
The accuracy on train dataset is 0.6303356910295274
The accuracy on test dataset is 0.6236696753334231

Train confusion matrix:
[[ 9649  3329  2436]
 [ 4444  6610  3439]
 [  928  1887 11813]]
```

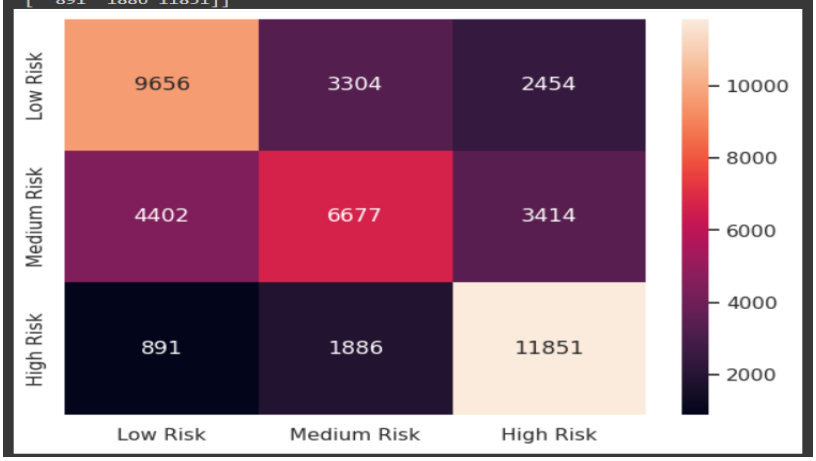



```
Test confusion matrix:
[[3226 1155  837]
 [1441 2163 1163]
 [ 315  676 3870]]
```

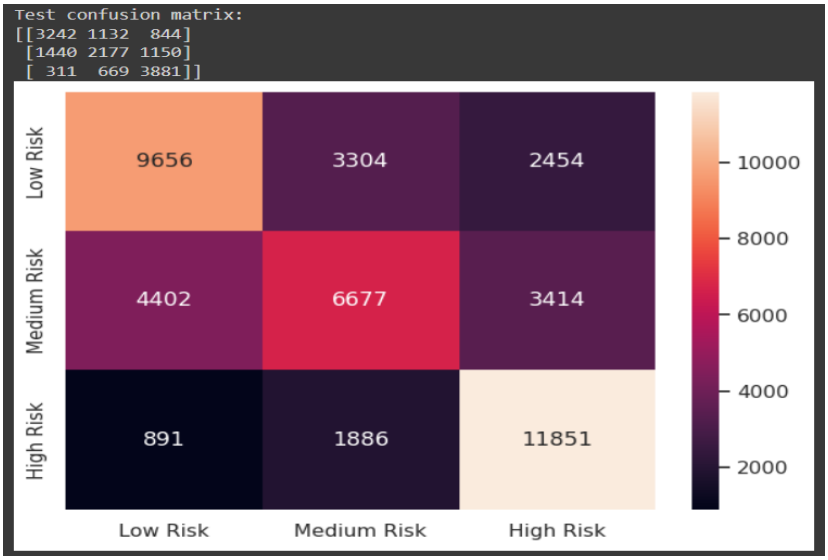


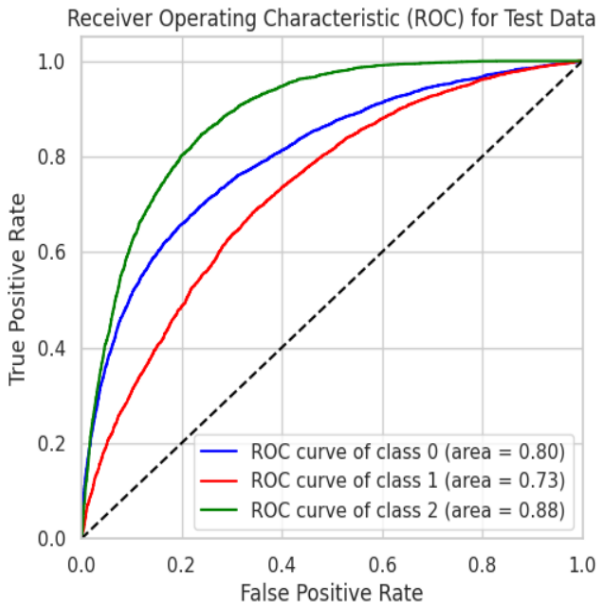
Without PCA

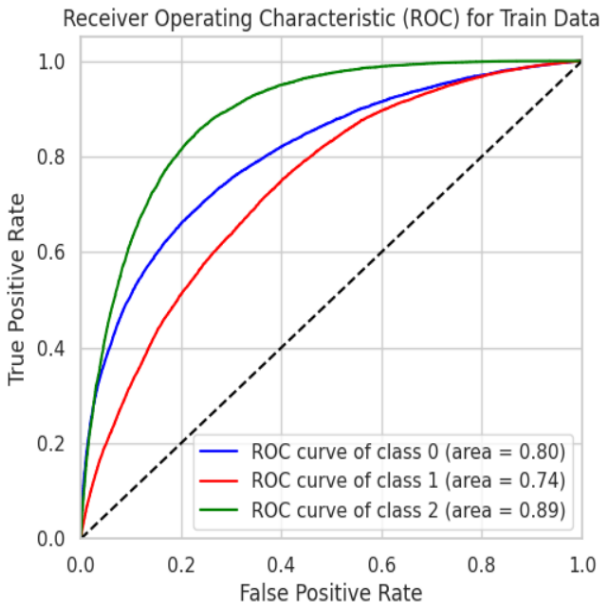
```
The accuracy on train dataset is 0.632850566969799
The accuracy on test dataset is 0.6264313619830257

Train confusion matrix:
[[ 9656  3304  2454]
 [ 4402  6677  3414]
 [  891  1886 11851]]
```




```
Test confusion matrix:
[[3242 1132  844]
 [1440 2177 1150]
 [ 311  669 3881]]
```



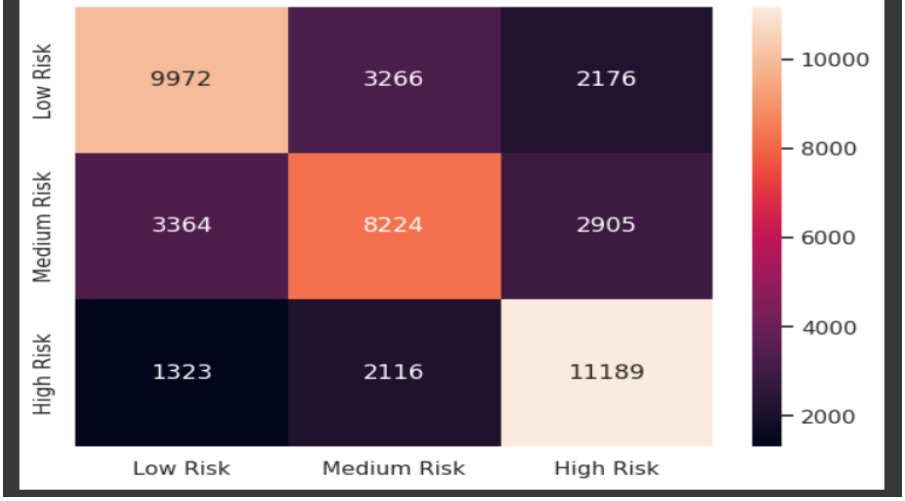




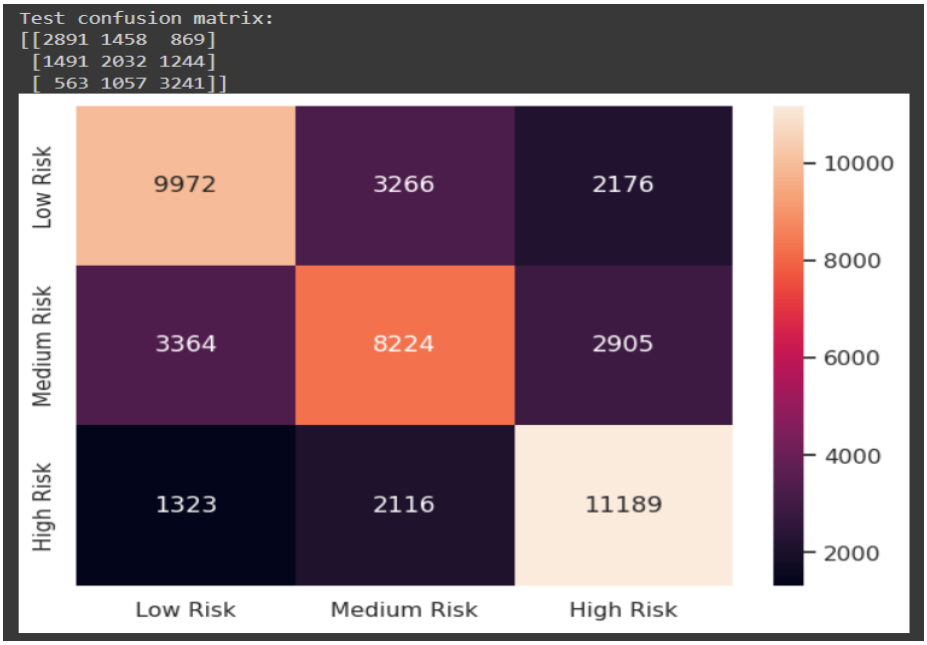
Decision Tree:

Decision Tree (with PCA)

```
The accuracy on train dataset is 0.6598181205793197
The accuracy on test dataset is 0.5499124343257443

Train confusion matrix:
[[ 9972  3266  2176]
 [ 3364  8224  2905]
 [ 1323  2116 11189]]
```

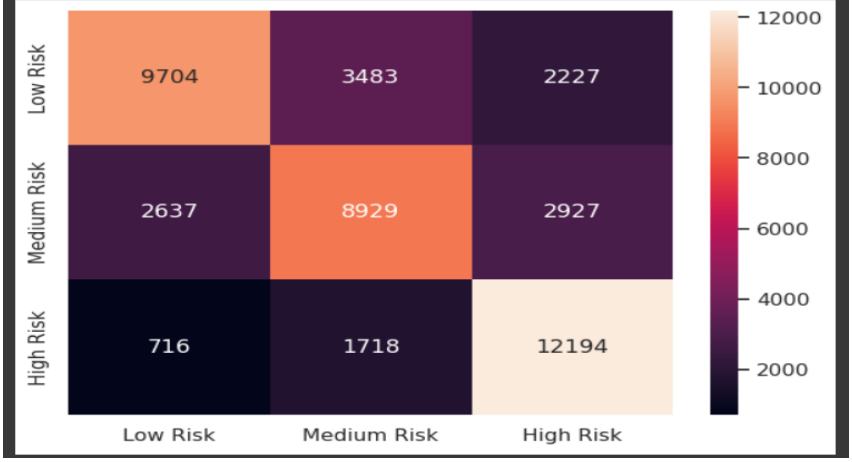



```
Test confusion matrix:
[[2891 1458  869]
 [1491 2032 1244]
 [ 563 1057 3241]]
```

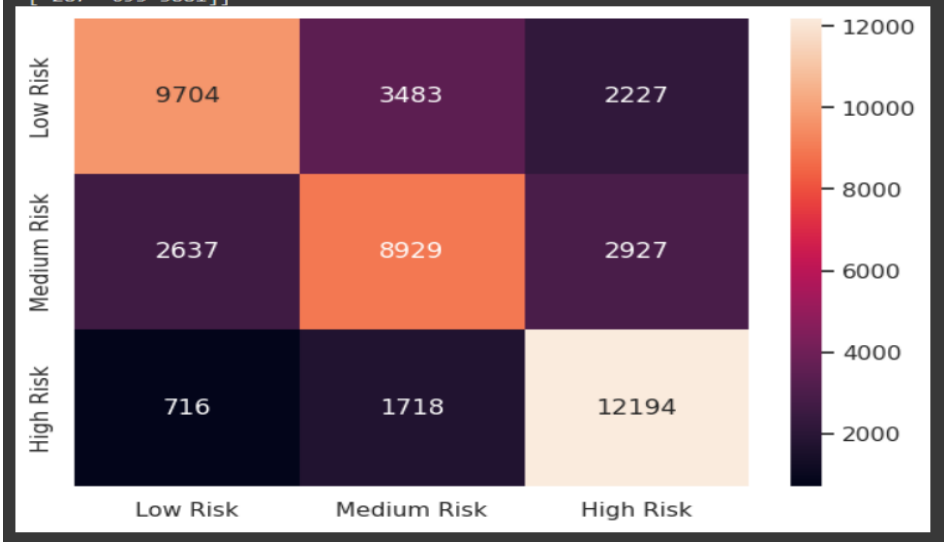


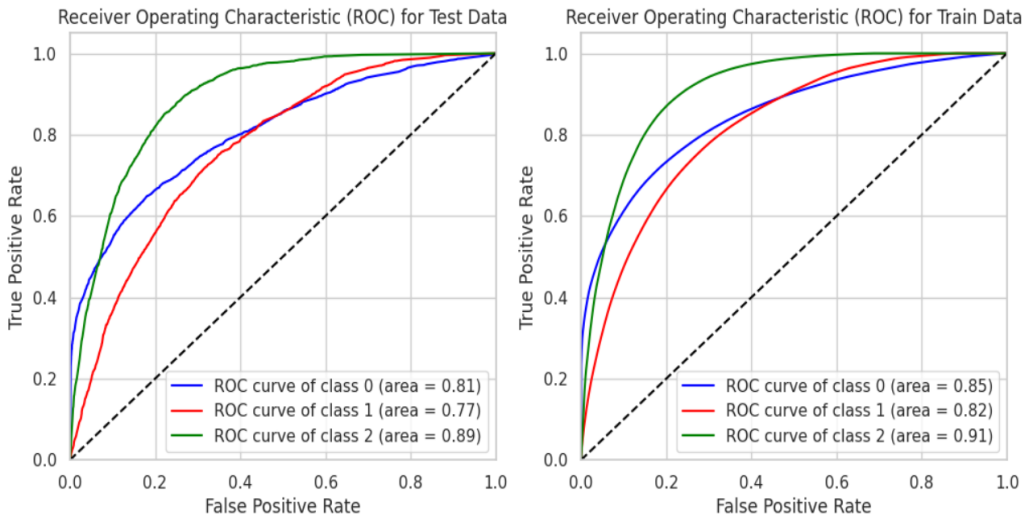
Without PCA

```
The accuracy on train dataset is 0.6921971483103178
The accuracy on test dataset is 0.6473124073824599

Train confusion matrix:
[[ 9704  3483  2227]
 [ 2637  8929  2927]
 [  716  1718 12194]]
```




```
Test confusion matrix:
[[3080 1325  813]
 [1061 2649 1057]
 [ 287  693 3881]]
```





By using PCA we observed that the models were getting overfitted hence we have proceeded further by not using PCA.

Overall Summary of all Models:

Model Name	Test ROC	Test Accuracy	F-Score	Precision	Recall
Random Forest	83.412806	66.354574	65.952164	66.868673	66.354574
XG Boost	84.618899	67.176344	66.911929	67.792993	67.176344
Logistic Regression	80.296830	62.643136	61.978298	62.023700	62.643136
Decision Tree	82.051874	64.731241	64.427050	64.769342	64.731241
Gradient Boosting	84.218032	66.927118	66.681232	67.368787	66.927118

Interpretation:

Test ROC (Receiver Operating Characteristic) AUC (Area Under the Curve):

- Random Forest has the highest ROC AUC score of about 0.819, indicating it has a good measure of separability and is able to distinguish between the classes quite well.
- XG Boost follows closely with a score of about 0.840, suggesting it also performs well in class separation.
- Logistic Regression has a significantly lower ROC AUC score of about 0.623, indicating it struggles more with class separation compared to the other two models.

Test Accuracy:

- Random Forest and XG Boost have similar accuracy scores, 0.648 and 0.664 respectively, meaning they correctly predict the class approximately 65% of the time.
- Logistic Regression lags behind with an accuracy of about 0.439, correctly predicting the class less than half the time.

F-Score:

The F-Score is a harmonic mean of precision and recall, with a higher score indicating better model performance. Random Forest and XG Boost are quite close, with scores of 0.642 and 0.623, while Logistic Regression is lower at 0.440.

Precision:

Precision measures the ratio of true positives to the sum of true and false positives. The higher the precision, the more confident we can be in the model's positive predictions. XG Boost leads with a precision of about 0.675, followed by Random Forest and then Logistic Regression.

Recall:

Recall measures the ratio of true positives to the sum of true positives and false negatives. A higher recall indicates that the model is better at catching positive cases. Random Forest and XG Boost perform similarly on recall, while Logistic Regression has a lower score.

Interpretation:

- Random Forest shows a balanced performance across all metrics. It is good at both distinguishing between classes and maintaining a balance between precision and recall.
- XG Boost has a slightly better ROC AUC score and precision, suggesting it might be making more correct positive predictions than Random Forest. However, its accuracy and F-Score are slightly lower, indicating a trade-off.
- Logistic Regression does not perform as well as the other two models across all metrics, which may suggest that the decision boundary between the classes is not linear or that the feature space does not lend itself well to logistic regression.

To improve these models, we could:

Feature Engineering: Create new features that might be more informative for the model, or perform feature selection to remove noise.

Data Preprocessing: Normalize or standardize features if not already done. Handle missing values or outliers which can affect model performance.

Ensemble Methods: Combine the predictions of multiple models to improve the overall performance.

Class Imbalance: If data is imbalanced, we can consider methods like SMOTE, adjusting class weights, or using precision-recall curves to evaluate model performance.

Model Complexity: For logistic regression, adding polynomial features might help capture more complex relationships. For tree-based models, increasing the depth or number of trees may improve performance.