Project 15 - Heart Attack Prediction

Group 13, Saumya Shah & Anas Tagui

Introduction

Task

The goal of this project is to develop a machine learning model that can predict the likelihood of a patient having an elevated risk of heart attack, based on the patient's clinical attributes. We frame this as a **binary classification** task: given input features (e.g., age, blood pressure, cholesterol, etc.), the model outputs either **0** (low risk) or **1** (high risk of heart attack). This prediction essentially corresponds to the presence or absence of heart disease, which is a strong indicator of heart attack risk.

The clinical motivation for this task is compelling cardiovascular diseases are the leading cause of death globally (responsible for about 17.9 million deaths per year).

Early identification of individuals at high risk of heart attack enables timely intervention (such as lifestyle changes or preventive medications), thereby improving patient survival rates.

In other words, detecting an increased risk *early* allows physicians to implement preventive measures before a cardiac event occurs.

This project's focus on heart attack risk prediction is thus highly relevant to public health, as it aligns with the preventive medicine paradigm in cardiology that seeks to reduce morbidity and mortality through early risk assessment.

From a machine learning perspective, heart attack risk prediction has been studied using various algorithms in clinical datasets.

The task is challenging because patient data can be noisy or heterogeneous, and **model interpretability** is often important in a healthcare setting. Therefore, while we aim for high predictive accuracy, we also consider the clinical trustworthiness of the model's decisions. Overall, this project's aim is to create a robust binary classifier that not only achieves strong performance on held-out data but also provides insight into the key risk factors for heart attacks.

Dataset

Overview

The heart attack prediction dataset consists of 8,763 patient records with 26 columns in total. Each record is identified by a unique Patient ID and includes a binary **target** column indicating heart attack risk (1 for high risk, 0 for low risk).

The dataset's purpose is to provide a comprehensive set of patient attributes for training machine learning models to predict the likelihood of a heart attack.

The data has been collected globally (patients from multiple continents) and is designed to capture a broad range of factors associated with cardiovascular risk

Feature Composition and Types

The remaining 25 columns are predictor features encompassing diverse data types, numerical measurements, categorical descriptors, and binary indicators covering demographic details, clinical readings, lifestyle habits, and more. Key types of features and examples include:

- **Demographic & Geographic Factors:** Features such as patient Age (numeric) and Sex (binary male/female) record basic demographic information. Additionally, geographic identifiers like *Country*, *Continent*, and *Hemisphere* (all categorical) indicate each patient's location, reflecting environmental and regional context.

  These factors help capture population diversity and potential geographic influences on heart health.

- **Clinical Measurements:** The dataset includes vital signs and lab results that are numerical in nature. For example, Blood Pressure (recorded as systolic/diastolic values) and Heart Rate (beats per minute) are provided for each patient.

  Key laboratory values like Cholesterol and Triglycerides levels, as well as calculated metrics such as BMI (body mass index), are included as continuous features. These clinical attributes quantify the patient's cardiovascular health status and risk factors.

- **Medical History Indicators:** Several binary features denote the presence or absence of pre-existing conditions and genetic predispositions. For instance, Diabetes status is given (yes/no), along with whether the patient has a Family History of heart disease (1 if yes).

  The dataset also notes any Previous Heart Problems (prior cardiovascular events) and current Medication Use for heart-related conditions, each as binary flags. These

variables capture personal medical history that could contribute to heart attack risk.

- **Lifestyle & Behavioral Factors:** A rich set of features describe lifestyle choices and habits, in both categorical and numeric form. Smoking status is a binary indicator (smoker or non-smoker), and Alcohol Consumption is categorized (e.g. none, light, moderate, heavy).

  Dietary habits are summarized by a Diet category (healthy, average, or non-healthy diet). Physical activity is quantified by Exercise Hours per Week (numeric total hours) and Physical Activity Days per Week (numeric count of days with exercise). Correspondingly, sedentary behavior is captured by Sedentary Hours per Day (numeric hours of inactivity).

  The dataset also logs average Sleep Hours per Day (numeric) and a self-reported Stress Level on a 1–10 scale. Together, these features reflect the patient's lifestyle and behavioral risk factors that are important for heart health.

- **Socioeconomic Feature:** An Income attribute (numeric or ordinal) is included as well. This provides a socioeconomic context, recognizing that income level can influence healthcare access, diet, stress, and other factors related to cardiovascular risk.

All features are encoded in appropriate formats (numerical readings for continuous variables, categorical codes for qualitative factors, and binary flags for yes/no indicators).

This diverse mix of data types captures multiple dimensions of heart attack risk from biophysical measurements and clinical history to lifestyle patterns and environmental context making the dataset a comprehensive foundation for building a predictive model of heart attack risk.

Each feature contributes a different perspective on the patient's health, thereby enriching the model's ability to learn complex relationships associated with cardiovascular events.

Models

Overview

General Preprocessing

The initial data assessment reveals no missing or null values across any columns, eliminating the need for imputation or cleaning up of those values.

The dataset exhibits a significant imbalance in the target variable, 'heart attack risk'. Class 0 (no heart attack risk) comprises 5624 instances, while Class 1 (heart attack risk) has only 3139 instances. To mitigate the bias this imbalance could introduce during model training, the Synthetic Minority Oversampling Technique (SMOTE) will be applied. SMOTE will generate synthetic samples for the minority class (Class 1), thereby increasing its representation and promoting a more balanced learning environment for the predictive model.

Beyond the target variable, several individual features also demonstrate notable class imbalances. For instance, the 'gender' feature shows an approximate 70% male to 30% female distribution. Similar imbalances in other categorical or numerical features can potentially skew model learning. To address these, various techniques will be considered, including targeted application of SMOTE for minority categories within features or downsampling of majority categories, depending on the specific feature and its impact on model performance.

Non-numerical features such as 'country', 'continent', and 'hemisphere' must be transformed into a numerical format for model training. Either label encoding or one-hot encoding will be used. The blood pressure feature, expressed as ‘<systolic>/<diastolic>’, will be separated into two columns.

To ensure the robustness of the model and prevent extreme values from disproportionately influencing results, outlier detection and removal will be performed. The Interquartile Range (IQR) will be calculated for all numerical columns, and data points falling outside the established lower and upper bounds will be removed.

Given the relatively high dimensionality of the dataset (24 features, excluding patient ID and the target variable), dimensionality reduction techniques will be explored to enhance model efficiency and mitigate the risk of overfitting.

First, a correlation matrix will be computed to identify highly correlated features. Features exhibiting a relationship will be removed, as they provide redundant information.

Subsequently, Principal Component Analysis (PCA) may be considered. The number of principal components to retain will be determined by analysing the cumulative explained variance, aiming for at least 99 per cent.

More preprocessing may be applied based on the method under consideration.

The dataset will be split into three sets: 80% of the data will be the training set, 10% will be the validation set, and the remaining will be the testing set.

Usual Methods & Metrics

There are various commonly used approaches for binary classification, categorised by their underlying approach. These include parametric models like logistic regression and naive Bayes, which assume a fixed data distribution, and non-parametric models such as K-nearest neighbours and decision trees, which adapt their complexity to the data. Classifiers can be linear (eg., logistic regression, basic SVMs) or non-linear (eg., kernel SVMs, ensemble methods, neural networks), defining the complexity of their decision boundaries. The best methods on Kaggle seem to achieve accuracies between up to 0.65, so this would be a good upper bound to aim for.

Accuracy, precision, recall, ROC-AUC, and F1-score are key metrics for evaluating binary classification models. The F1-score, being the harmonic mean of precision and recall, offers a balanced performance measure and is particularly valuable for imbalanced datasets as it equally weighs the trade-off between false positives and false negatives. Therefore, we will be using the F1 score to evaluate our model.

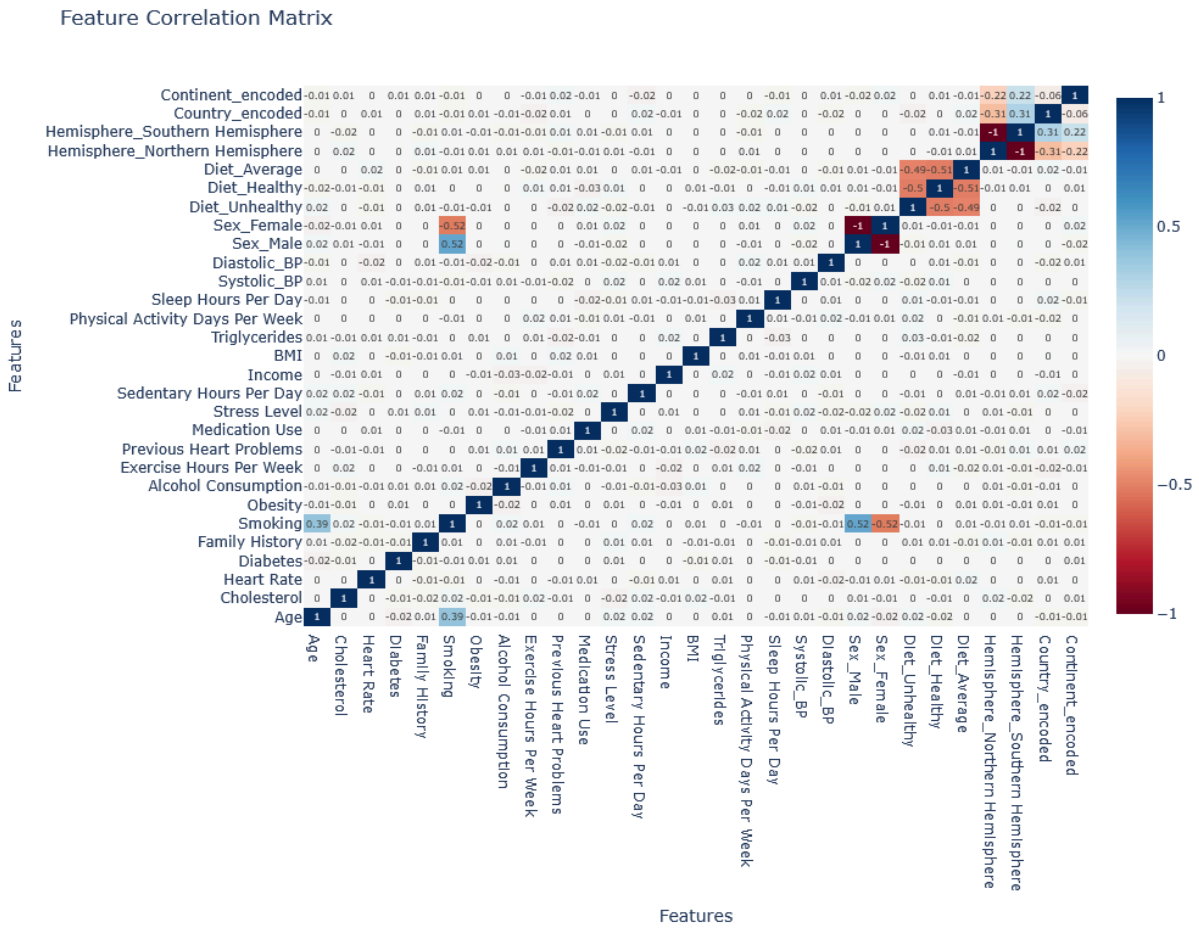
Preprocessing Results

The blood pressure column was first separated into two numerical features: systolic and diastolic blood pressure. Then, one-hot encoding was performed for low cardinality features (sex, diet, and hemisphere), while label encoding was performed for high cardinality features (country and continent).

Next, the quartiles and IQR were calculated for each feature, and the data were checked for any values outside the range $[Q_1 - 1.5\ IQR, Q_3 + 1.5\ IQR]$. However, no such values were found, so no outliers were detected and removed.

Then, the data are split into training, validation, and test sets, and SMOTE was performed to oversample the minority class (class 1) for the training set only. Finally, the values for the encoded and binary features were rounded to the nearest valid value, since SMOTE outputs continuous feature data.

Dimensionality reduction techniques were considered. First, a correlation matrix was plotted:



However, no significant correlations were observed in the numerical features, with a maximum correlation of 0.027. Therefore, no features were removed at this stage.

Next, PCA was considered. The features were scaled and the number of PCA components were calculated to retain 90%, 95%, and 99% variance. However, the number of PCA features required were too high for any significant dimensionality reduction.

```
Original features: 29
Components for 90% variance: 23 (79.3% of original)
Components for 95% variance: 24 (82.8% of original)
Components for 99% variance: 26 (89.7% of original)
```

K-Nearest Neighbours Classifier

Description

The K-Nearest Neighbours algorithm is a non-parametric instance-based classifier that operates without an explicit training phase; instead, it "memorises" the entire training dataset. To classify a new data point, KNN calculates its distance to every point in the training set, identifies the 'K' closest data points, and assigns the class label that represents

the majority among these 'K' neighbours, effectively making a classification decision based on local proximity within the feature space.

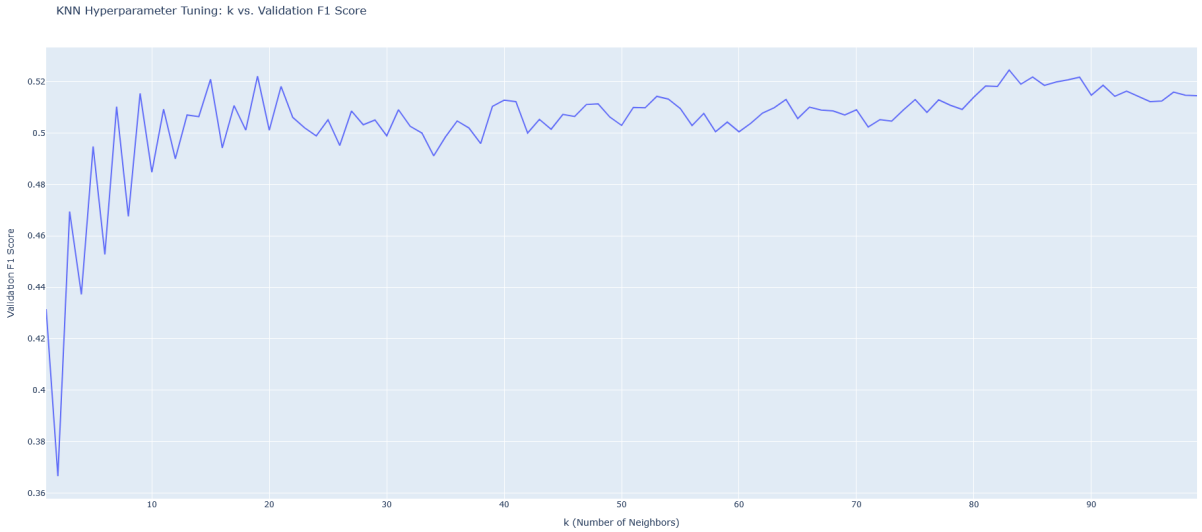
Preprocessing

Given its reliance on distance computations, feature scaling is an important preprocessing step that ensures all features contribute equally to distance calculations by bringing them to a comparable range. Furthermore, outlier removal is also required to prevent extreme values from distorting distance metrics and influencing neighbour selection. Dimensionality Reduction techniques (like PCA) are also required to mitigate the "curse of dimensionality," which can severely degrade KNN's performance and efficiency by making all data points appear sparsely distributed.

Results

KNN classification was performed by first preprocessing (scaling) the training data, and then using the same scaler to scale the validation and testing datasets (to prevent data leaks). Label-encoded features were ignored, since for distance-based algorithms like KNN, label-encoded features can be problematic because it implies an artificial ordering and magnitude difference between categories.

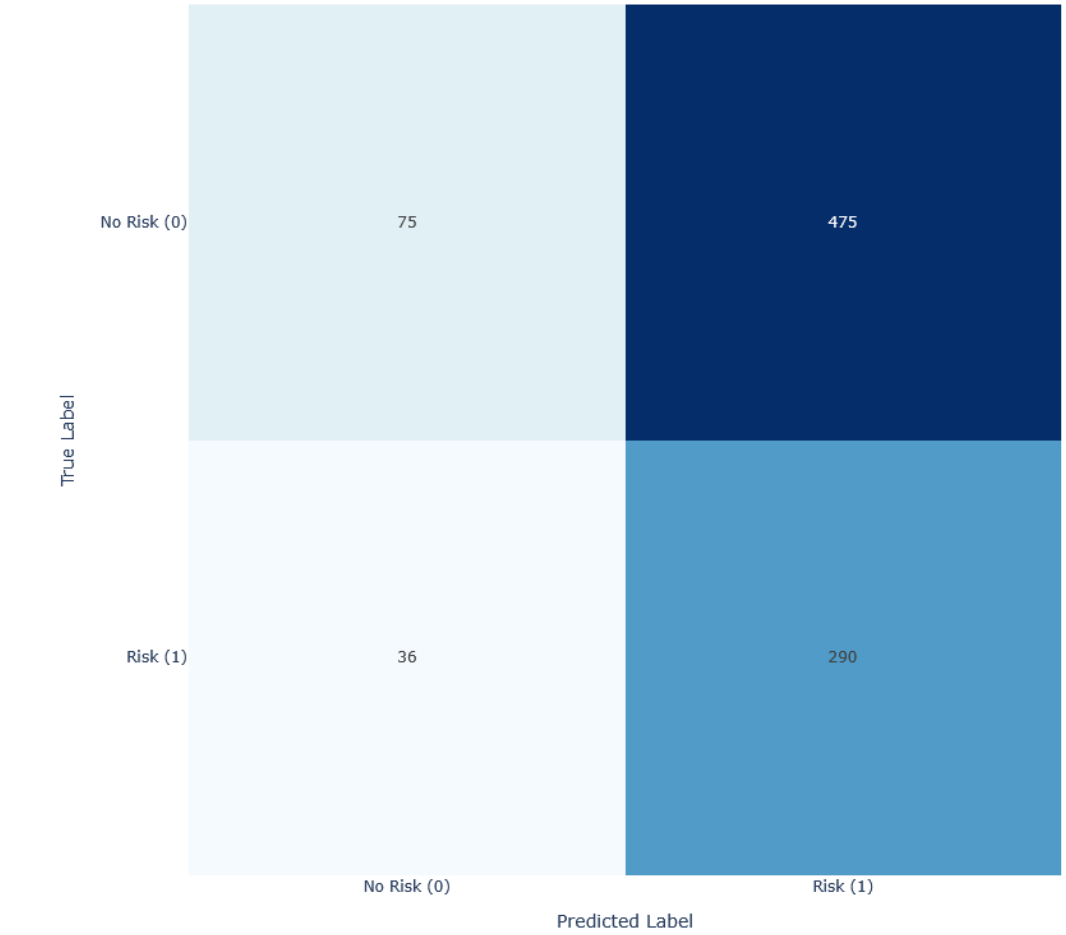
Then, automatic hyperparameter tuning was performed by fitting KNN classifiers for k-values between 1 and 100, and then calculating their F1 score on the validation set.

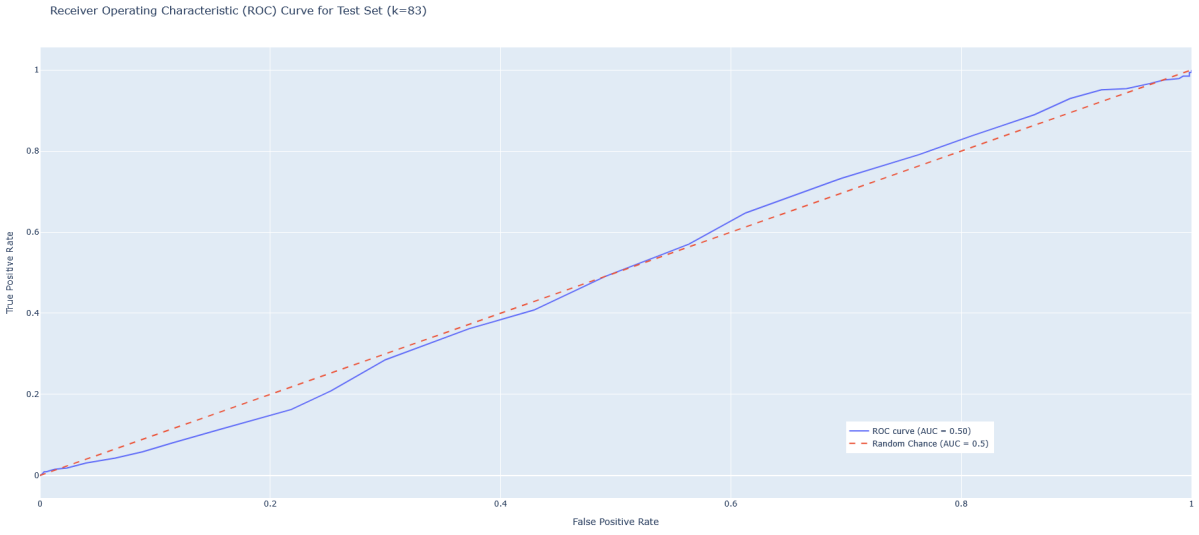


It was found that k = 83 was optimal, with a validation F1 score of 0.5245. The classifier corresponding to this k-value was then evaluated against the test set, which gave us our final F1 score of 0.5316.

```
Best k found: 83 with Validation F1 Score: 0.5245
Final KNN Model - Validation F1 Score (k=83): 0.5245
Final KNN Model - Test F1 Score (k=83): 0.5316
```

The confusion matrix and ROC curve were also plotted for evaluation:





From these plots, it is evident that the KNN classifier is not much better than random chance. This may be due to reasons such as the curse of dimensionality (KNN performance may degrade with increasing input features) and the presence of noisy/uninformative features.

A challenge faced for this model was that none of the dimensionality reduction techniques we tried (removing highly-correlated features and PCA) were useful, as discussed in the

preprocessing results section. Therefore, the model was subject to the curse of dimensionality, which may have degraded model performance. Another challenge was whether label-encoded features should be included. We could lose valuable information that could decrease model performance, but label encoding for countries and continents implies an artifical ordering that may have been uninformative at best and noisy at worst.

Gaussian Naive Bayes Classifier

Description

The naive Bayes classifier is a probabilistic classifier operating under the naive assumption that all features are conditionally independent of each other given the class label, and the data are independent and identically distributed. For each class, Gaussian Naive Bayes models the distribution of each continuous feature as a Gaussian (normal) distribution, characterised by its mean and variance within that class. When classifying a new data point, the algorithm calculates the likelihood of observing its feature values given each class, using the Gaussian probability density function. These likelihoods are then combined with the prior probabilities of each class, and the data point is assigned to the class with the highest resulting posterior probability.

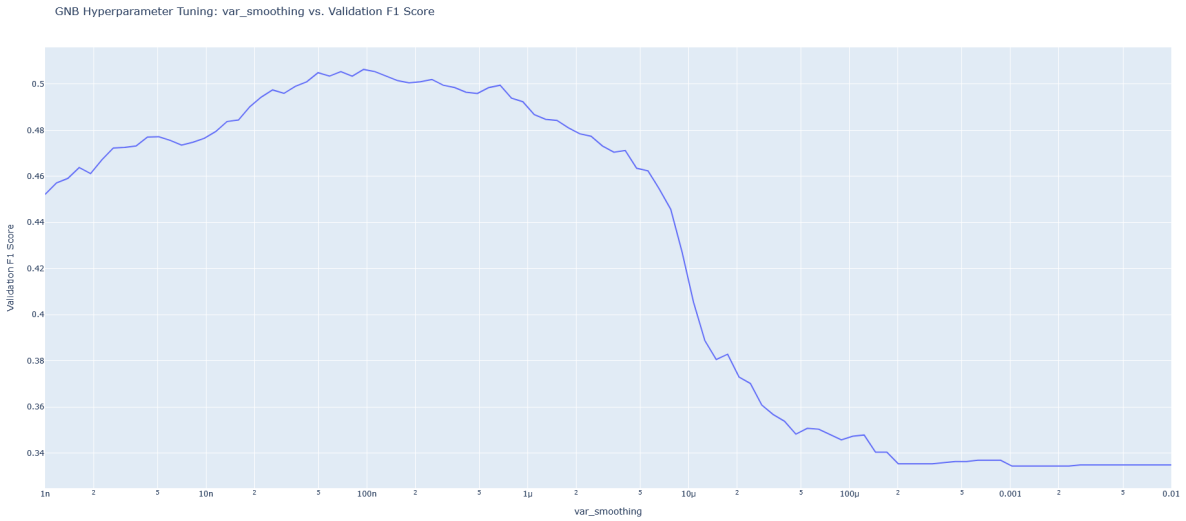
Preprocessing

Outliers can significantly impact the estimation of means and variances for the Gaussian distributions, potentially leading to inaccurate probability calculations. Therefore, outliers need to be identified and removed. In addition, highly correlated features are double-counted in the model, leading to overestimation of the importance of those features. Therefore, correlated features must be identified and removed.

Results

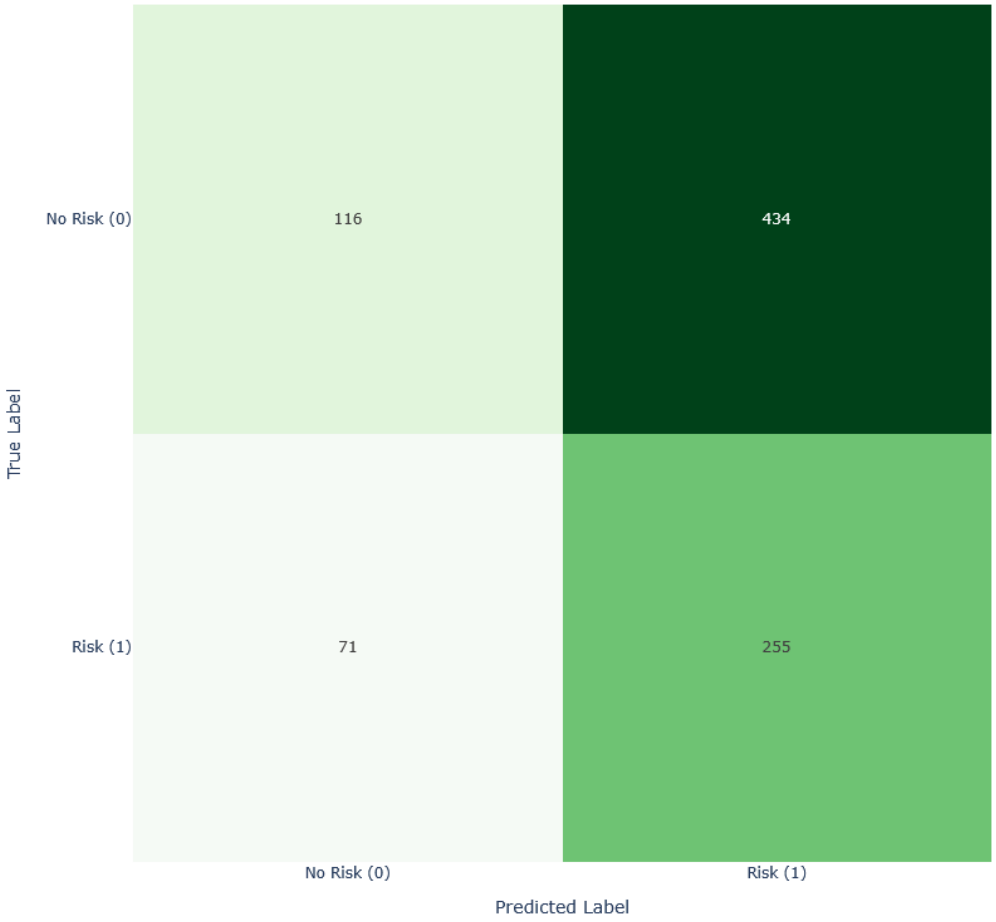
Gaussian Naive Bayes (GNB) classification was performed by first preprocessing the training data to create a correlation matrix and remove highly-correlated features, since they may lead to higher emphasis on those features.

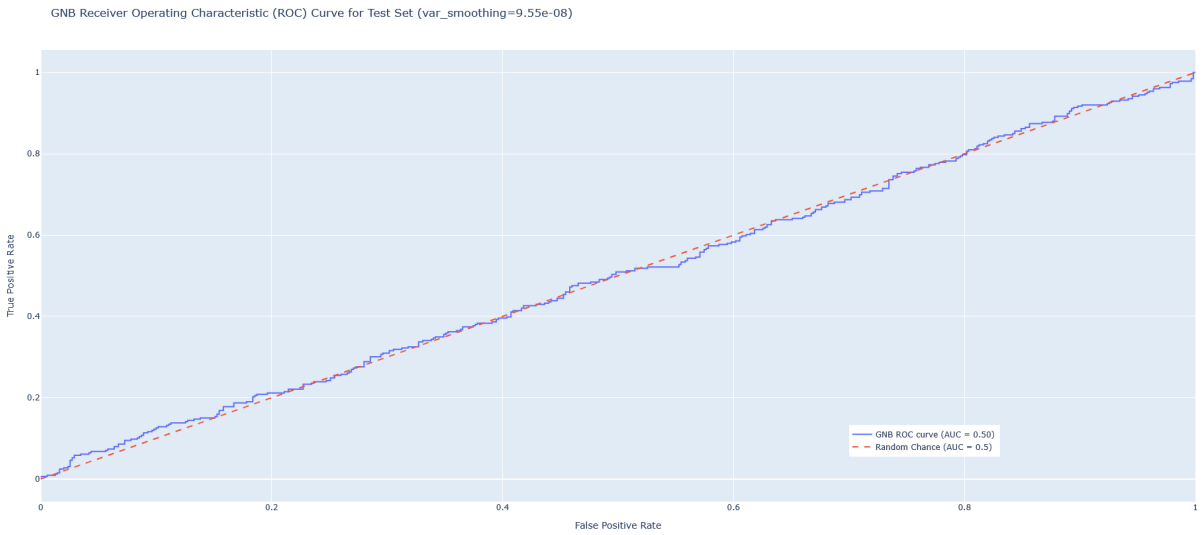
Then, automatic hyperparameter tuning was performed by fitting GNB classifiers for 100 `var_smoothing` values between 10^{-9} and 10^{-2}, and then calculating their F1 score on the validation set.



It was found that `var_smoothing` = 9.55 * 10^{-8} was optimal, with a validation F1 score of 0.5064. The classifier corresponding to this k-value was then evaluated against the test set, which gave us our final F1 score of 0.5025.

```
Best var_smoothing found: 9.55e-08 with Validation F1 Score: 0.5064
Final GNB Model - Validation F1 Score (var_smoothing=9.55e-08): 0.5064
Final GNB Model - Test F1 Score (var_smoothing=9.55e-08): 0.5025
```





From these plots, it is evident that the Naive Bayes model is worse than the earlier KNN model, and not much better than random chance. This may be because the model's base assumption of the conditional independence of features may have been violated, compromising its ability to learn and make accurate predictions. Another reason may be that Gaussian functions may not be good candidate functions for modelling the features in this dataset, especially the binary ones, which may be better modelled using Bernoulli functions.

A challenge faced when developing this model was selecting the probability distribution to model the features. Since most of the features were numerical and continuous, we decided to use Gaussian functions for simplicity. However, we could have used a mixture of distributions depending on the type of the feature (Bernoulli for binary features, multinomial for encoded features, and Gaussian for continuous features), which may have improved model performance.

Neural Network Classifier

Description

Neural Networks (NNs), especially deep learning architectures, are non-linear models composed of interconnected layers of neurons that learn complex patterns. For binary classification, an NN typically consists of an input layer (receiving the features), one or more hidden layers (where non-linear transformations occur via activation functions like ReLU), and an output layer. In binary classification, the output layer has a single neuron with a sigmoid activation function, which squashes its output to a value between 0 and 1, directly interpretable as the probability of belonging to the positive class. The network learns by iteratively adjusting its internal weights and biases through gradient descent to minimise a specified loss function.

Preprocessing

Feature scaling is essential to help stabilise and accelerate the convergence of the optimisation algorithm, preventing larger feature values from dominating the learning

process. Dimensionality reduction techniques may also be required for very high-dimensional datasets to reduce noise, prevent overfitting, and potentially speed up training, particularly when data is sparse.

Future Plans

In the final report, we would like to show the following results:

- F1 Scores and ROC curves of the three above models after hyperparameter tuning.
- Comparison of the models based on these metrics.

To do this, we will work on implementing the neural network classifier with embeddings for encoded data, and then tuning its hyperparameters to minimise loss and find an optimal classifier.