Date of publication 07/29/2023
Digital Object Identifier 10.1109/ACCESS.2023.0322000

Predictive Analysis Of Heart Disease Using Naïve Bayes Classifier

Amit Raj Pant[1] Arahanta Pokhrel[1]

Insitute of Engineering, Thapathali Campus, Final Year

Corresponding authors:
Amit Raj Pant (e-mail:[2]`amitrajpant7@gmail.com`),
Arahanta Pokhrel (e-mail:[1]`pokhrelarahanta5@gmail.com`).

ABSTRACT Heart diseases are a significant health concern, and accurately predicting their presence is of utmost importance. In this study, we focus on leveraging patient characteristics, including age, gender, chest pain type, blood pressure, cholesterol levels, and other relevant attributes, to develop an accurate predictive model. Utilizing the Naïve Bayes classifier, we explore different modeling approaches and preprocess the data to gain key insights through visualizations. Our data analysis yields crucial information about how various attributes are linked to the possibility of having heart disease, providing valuable risk assessment implications for heart disease prediction. The "heart disease dataset" is subjected to comprehensive analysis, and key features such as Age, Sex, RestingBP, Cholesterol, FastingBS, RestingECG, MaxHR, ExerciseAngina, Oldpeak, and ST_Slope are used as inputs to the Naïve Bayes classifier. The trained models are evaluated, and their classification results are presented with informative confusion matrices and classification reports. By employing and trying out different methods to improve the accuracy of our classifier we were able to reach an accuracy of 90%.

INDEX TERMS Heart disease, Naïve Bayes classifier, Prediction

I. INTRODUCTION

Heart disease is a critical and prevalent health issue that continues to pose significant challenges to global healthcare systems.Given the severity of cardiovascular disease and its associated risk factors like hypertension, diabetes, hyperlipidemia, or pre-existing conditions, early detection and effective management are paramount. In this context, leveraging the power of machine learning models becomes instrumental in facilitating timely intervention and care for individuals afflicted with CVD or deemed to be at high cardiovascular risk. By harnessing predictive insights from the dataset, we can potentially contribute to the development of efficient healthcare strategies to combat this significant public health issue. As the volume of medical data grows exponentially, data mining techniques have emerged as essential tools for harnessing valuable insights from large datasets. In this context, this data mining lab report aims to address the pressing issue of heart disease by leveraging the power of the Naïve Bayes classifier to predict its presence based on key features. These features, including Age, Sex, ChestPainType, RestingBP, Cholesterol, FastingBS, RestingECG, MaxHR, ExerciseAngina, Oldpeak, and ST_Slope, hold crucial information that can potentially contribute to the early detection and management of heart disease.

To achieve our primary objective, we employ rigorous preprocessing techniques to ensure the dataset's integrity and quality. Moreover, visualization tools are employed to gain deeper insights into the dataset's characteristics and uncover potential patterns or trends that may impact the classification process. By unraveling the complex relationships between the features and the target variable, we can identify significant risk factors and enhance the accuracy of our heart disease prediction model.In addition to exploring the individual contributions of features to the classification task, we delve into the performance of two Naïve Bayes models: Gaussian Naïve Bayes and Categorical Naïve Bayes. The comparison between these models allows us to identify the strengths and weaknesses of each approach and determine the most suitable classifier for heart disease prediction in our specific dataset.The implications of this study extend beyond medical diagnosis alone.

II. METHODOLOGY

A. THEORY

The Naïve Bayes classifier **[1]** a classification technique based on Bayes' Theorem, makes an important independence assumption among predictors. In simpler terms, it assumes that the presence of a particular feature in a class is entirely unrelated to the presence of any other feature. This popular supervised machine learning algorithm is widely utilized for classification tasks, particularly in text classification **[2]**. Belonging to the family of generative learning algorithms, it models the distribution of inputs for a given class or category. This approach relies on the assumption of conditional independence between input data features, given the class, enabling the algorithm to make swift and accurate predictions.

In the realm of statistics, naive Bayes classifiers are regarded as straightforward probabilistic classifiers that apply Bayes' theorem. The theorem operates on the probability of a hypothesis, considering both the data and some prior knowledge. Despite its effectiveness, the naive Bayes classifier is known for assuming independence between all input features, a condition that may not always hold true in real-world scenarios. Nevertheless, due to its efficiency and impressive performance in various practical applications, it remains widely used.

The term "Naive Bayes" comes from the fact that this classification algorithm makes a naive or simple assumption about the independence of the features in the dataset. It's based on Bayes' theorem, a fundamental principle in probability theory, which allows us to update the probability of an event based on new evidence. The "naive" part of the name refers to the assumption of feature independence, which is often not true in practice but still serves as a useful and computationally efficient approximation.

1) probability:

It refers to the possibilities or likelihood of an event occurring as a probability.

$P(A|B)$ is the conditional probability of A given B, or the likelihood of event A if event B has already occurred. Because we already know that B has happened when we experiment, the number of alternative outcomes for the event has been limited. As a result, given that B has already occurred, the unconditional probability has now changed to a conditional probability.It is a supervised machine learning algorithm for classification based on Bayes' theorem. The algorithm learns the probability of data instances belonging to a particular class. Hence, it is a probabilistic classifier.

We define the conditional probability as:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{1}$$

Where $P(A \cap B)$ represents the joint probability, i.e., probability, of both events happening together.

2) Bayes' theorem

Bayes' theorem provides a way of computing the posterior probability $P(c|x)$ from $P(c)$, $P(x)$, and $P(x|c)$. Look at the equation below:

$$P(c|x) = \frac{P(c) \cdot P(x|c)}{P(x)} \tag{2}$$

Above,

- $P(c|x)$ is the posterior probability of class (c, target) given predictor (x, attributes).
- $P(c)$ is the prior probability of class.
- $P(x|c)$ is the likelihood, which is the probability of the predictor given class.
- $P(x)$ is the prior probability of the predictor.

3) Types of Naïve Bayes used

We mainly focus on two variants of Naive Bayes classifiers: Gaussian Naive Bayes and Categorical Naive Bayes.

- **Gaussian Naive Bayes Classifier**
  The Gaussian Naive Bayes classifier is a variant of the Naive Bayes algorithm designed for continuous data attributes. It assumes that each class's feature follows a Gaussian (normal) distribution, hence the term "Gaussian." This classifier is particularly well-suited for data with continuous attributes, where the underlying assumption is that the features within each class are normally distributed.

The main advantage of Gaussian Naive Bayes lies in its simplicity and speed, making it an attractive choice for large-scale datasets with continuous features. It requires fewer training samples to estimate the parameters needed for classification compared to other more complex algorithms.

- **Categorical Naive Bayes Classifier**
  The Categorical Naive Bayes classifier, also known as Multinomial Naive Bayes, is another variant of the Naive Bayes algorithm tailored for discrete data attributes. Unlike the Gaussian variant, this classifier assumes that the features are categorical in nature and represented as discrete counts. It is particularly effective for text classification tasks where the data is often represented as word frequency vectors.
  Categorical Naive Bayes is popularly used in natural language processing applications, including sentiment analysis, spam detection, and topic categorization. Despite its simplicity and the "naive" assumption of independence between features, it often performs remarkably well in practice, especially when dealing with high-dimensional and sparse data.

4) Pros

- It is easy and fast to predict the class of the test dataset. It also performs well in multi-class prediction.
- When the assumption of independence holds, the classifier performs better compared to other machine learning models like logistic regression or decision tree, and requires less training data.
- It performs well in the case of categorical input variables compared to numerical variables. For numerical variables, a normal distribution is assumed (bell curve, which is a strong assumption).

5) Cons

- If a categorical variable has a category (in the test dataset) that was not observed in the training dataset, then the model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as "Zero Frequency." To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation.
- On the other side, Naive Bayes is also known as a bad estimator, so the probability outputs from predict_proba are not to be taken too seriously.
- Another limitation of this algorithm is the assumption of independent predictors. In real life, it is almost impossible that we get a set of predictors that are completely independent.

6) Ways to Improve Naive Bayes Classification Performance

1) **Remove Correlated Features**: Removing highly correlated features reduces redundancy and potential overfitting.
2) **Use Log Probabilities in Calculations**: Logarithmic probabilities prevent underflow and ensure stable computations.
3) **Eliminate Zero Observations**: Applying smoothing techniques handles zero-frequency observations to avoid zero probabilities.
4) **Handle Continuous Variables**: Using appropriate probability distribution models (e.g., Gaussian Naive Bayes) effectively handles continuous feature values.
5) **Handle Text Data**: Preprocessing text data (tokenization, stemming, stop word removal) and converting it into numerical representations enables Naive Bayes to work with text-based features.
6) **Re-train the Model with New Data**: Periodic retraining with updated data helps the model adapt to changing patterns and improve performance.
7) **Parallelize Calculations**: Parallelizing computations across multiple processors enhances model training and prediction speed for large datasets.

The effectiveness of these techniques may vary based on the nature of the dataset and the specific problem . We have used Remove correlated feature technique and got some improvement.

B. DATASET

This dataset contains valuable information related to cardiovascular diseases (CVDs), which are the leading cause of global mortality. The dataset comprises 11 features that can be utilized to predict the likelihood of heart disease. These features include the patient's age, sex, chest pain type, resting blood pressure, serum cholesterol level, fasting blood sugar, resting electrocardiogram results, maximum heart rate achieved, presence of exercise-induced angina, ST segment depression, slope of the peak exercise ST segment, and the output class indicating the presence of heart disease or a normal state. Early detection and management of cardiovascular risks are crucial, and this dataset can be utilized for developing machine learning models to assist in such efforts. The dataset is an amalgamation of five separate heart datasets, consisting of 918 observations after removing duplicates, making it the most comprehensive heart disease dataset available for research purposes.

Attribute Information:

1) **Age:**The age of the patient in years, representing the individual's age at the time of data recording.. The ages range from 28 to 77. The most common age is 54, occurring 51 times, followed by 58 and 55 with 42 and 41 occurrences, respectively. The distribution provides insights into the prevalence of different age groups, with several individuals falling within their 50s and 60s

2) **Sex:**The sex of the patient, classified as "M" for Male and "F" for Female, denoting the gender of the individual.There are 725 male patients and 193 female patients in the dataset.

3) **ChestPainType:** In the dataset, the "ChestPainType" column represents the different types of chest pain experienced by patients. It is a categorical variable with four unique values: 'ATA', 'NAP', 'ASY', and 'TA', each corresponding to a specific type of chest pain.
   - **ATA - Atypical Angina:**
     This category refers to chest pain that is not typical of angina. Atypical angina is characterized by chest discomfort that differs from the usual symptoms associated with angina, such as pressure or tightness in the chest.
   - **NAP - Non-Anginal Pain:**
     The 'NAP' category represents chest pain that is not related to angina. It includes other types of chest discomfort that are not caused by reduced blood flow to the heart.
   - **ASY - Asymptomatic:**
     Asymptomatic means that the patient does not experience any chest pain or discomfort. In this category, patients have no apparent symptoms of chest pain related to heart issues.
   - **TA - Typical Angina:**
     'TA' stands for Typical Angina, which is chest pain caused by reduced blood flow to the heart muscle. It is usually described as a squeezing or pressure-like discomfort in the chest, often brought on by physical exertion or emotional stress.

   The frequency of each chest pain type in the dataset is different. The most common type is 'ASY' (Asymptomatic), occurring 496 times, followed by 'NAP' (Non-Anginal Pain) with 203 occurrences, 'ATA' (Atypical Angina) with 173 occurrences, and 'TA' (Typical Angina) with 46 occurrences.

4) **RestingBP:** The "RestingBP" column in the dataset represents the resting blood pressure of patients, measured in millimeters of mercury (mm Hg). This value indicates the level of pressure exerted by the blood against the walls of the arteries when the individual is at rest, without any physical or emotional exertion. The dataset contains a range of unique blood pressure readings, spanning from 0 mm Hg to 200 mm Hg. The frequency of each blood pressure value in the dataset varies. For instance, there are 132 individuals with a resting blood pressure of 120 mm Hg, 118 individuals with 130 mm Hg, and 107 individuals with 140 mm Hg, and so on.

5) **Cholesterol:** The serum cholesterol level of the patient is measured in milligrams per deciliter (mm/dl), reflecting the concentration of cholesterol in the blood. The dataset contains various unique cholesterol readings, ranging from 0 mm/dl to higher values.There are 172 individuals with a cholesterol level of 0 mm/dl, followed by 11 individuals with 254 mm/dl, and 10 individuals each with 223 mm/dl and 220 mm/dl, respectively, and so on. Analyzing the distribution of serum cholesterol levels can provide insights into the prevalence of different cholesterol concentrations within the patient population.

6) **FastingBS:** The "FastingBS" column in the dataset represents the fasting blood sugar level of the patients, categorized into two unique values: 0 and 1. A value of 1 indicates elevated blood sugar, as the fasting blood sugar is greater than 120 mg/dl, while a value of 0 represents a fasting blood sugar level that is not greater than 120 mg/dl.The Counts displays the frequency of each unique fasting blood sugar value in the dataset. There are 704 instances where the fasting blood sugar level is 0, indicating a normal or non-elevated fasting blood sugar level. Conversely, there are 214 instances where the fasting blood sugar level is 1, representing elevated fasting blood

sugar.This information plays a crucial role in medical research and clinical practice, as elevated fasting blood sugar levels can serve as an important indicator of diabetes or other metabolic disorders.

7) **RestingECG:**The "RestingECG" column in the dataset contains information about the results of the resting electrocardiogram, which provides insights into the heart's electrical activity. It is a categorical variable with three unique values:
   - "Normal": This value indicates a normal result in the resting electrocardiogram, suggesting that there are no significant abnormalities in the heart's electrical signals.
   - "ST": This value represents the presence of ST-T wave abnormalities, which may include T wave inversions and/or ST elevation or depression of greater than 0.05 mV. Such abnormalities could be indicative of certain heart conditions or issues.
   - "LVH": This value indicates probable or definite left ventricular hypertrophy (LVH) based on Estes' criteria. LVH is a condition where the muscle of the heart's left ventricle becomes abnormally thickened.

   Specifically, there are 552 instances of "Normal" results, 188 instances of "LVH" results, and 178 instances of "ST" results.

8) **MaxHR:** MaxHR (Maximum Heart Rate): The "MaxHR" column in the dataset represents the maximum heart rate achieved by the patient during exercise. It is expressed as a numeric value ranging between 60 and 202 beats per minute.The distinct maximum heart rate readings recorded in the dataset vary from a minimum of 60 beats per minute to a maximum of 202 beats per minute.There are 43 instances where the maximum heart rate reached during exercise is 150 beats per minute, followed by 41 occurrences of a maximum heart rate of 140 beats per minute, and 36 occurrences of 120 beats per minute, and so on.

9) **ExerciseAngina:** The "ExerciseAngina" column in the dataset provides information about whether the patient experiences exercise-induced angina. This categorical variable has two distinct values: "Y" indicating that the patient does experience exercise-induced angina, and "N" representing that the patient does not experience exercise-induced angina.
   - "Y" (Yes): This value denotes that the patient experiences exercise-induced angina.
   - "N" (No): This value represents that the patient does not experience exercise-induced angina.

   There are 547 instances where patients do not experience exercise-induced angina (denoted by "N"), and there are 371 instances where patients do experience exercise-induced angina (denoted by "Y").

10) **Oldpeak:**The "Oldpeak" column in the dataset represents the depression of the ST segment measured in numeric values. It reflects the extent of ST segment depression during exercise relative to rest.All the distinct values of ST segment depression recorded in the dataset, ranging from -2.6 to 6.2.The "Value Counts" section shows the frequency of each unique ST segment depression value in the dataset.There are 368 instances where the ST segment depression is 0.0, followed by 86 occurrences of 1.0, 76 occurrences of 2.0, and so on.

11) **ST_Slope:**
    The "Slope" column in the dataset represents the slope of the peak exercise ST segment. It is a categorical variable with three unique values:
    - **Flat**: This value indicates a flat slope of the peak exercise ST segment.
    - **Up**: This value represents an upsloping slope of the peak exercise ST segment.
    - **Down**: This value denotes a downsloping slope of the peak exercise ST segment.

    There are 460 instances with a "Flat" slope, followed by 395 occurrences of an "Up" slope, and 63 occurrences of a "Down" slope.

12) **HeartDisease:** The "HeartDisease" column in the dataset serves as the output class, representing the target variable for the classification task. It is a binary variable with two unique values:
    - "1": This value signifies the presence of heart disease in the patient.
    - "0": This value indicates a normal condition without heart disease.

The "Value Counts" section shows that there are 508 instances where heart disease is present (denoted by "1"), and 410 instances where there is no heart disease (denoted by "0"). The goal of the classification task would be to build a predictive model that can accurately classify individuals as either having heart

C. WORKING PRINCIPLE

The methodology performed on the heart disease dataset involves the following steps:

1. Data Preprocessing: The dataset is loaded from a CSV file, creating a pandas DataFrame named 'data'. Categorical variables are converted into dummy variables using one-hot encoding.

2. Data Exploration and Visualization: The first few rows of the DataFrame are displayed to get a glimpse of the data. The unique values of the "ChestPainType" column are printed to understand the different types of chest pain in the dataset. A statistical summary of the dataset is provided to obtain descriptive statistics. A histogram with KDE is plotted for the "Age" column to visualize the age distribution of patients, colored by the presence or absence of heart disease ("HeartDisease"). A simple histogram is plotted for the "Cholesterol" column to understand its distribution.

3. Correlation Analysis: The correlation matrix is calculated to identify the relationships between different features. A heatmap is plotted to visualize the correlations between features, with higher correlations shown using brighter colors. A bar plot is created to display the correlation of features with the target variable ("HeartDisease").

4. Feature Selection: Features with low correlation values (less than 0.2) are identified and dropped from the dataset. Features with negative correlation values are also identified and dropped from the dataset.

5. Model Building and Evaluation: The dataset is split into training and testing sets. A Gaussian Naïve Bayes model is trained on the training data. The evaluation function (`eval`) is used to assess the model's performance on the test data. The confusion matrix is plotted to visualize the true positive, true negative, false positive, and false negative predictions. The classification report is printed, which includes metrics like precision, recall, and F1-score for both classes ("No Disease" and "Disease").

6. Standardization: The numerical features are standardized using the StandardScaler from scikit-learn.

Hence, the methodology involves data preprocessing, correlation analysis, feature selection, and model evaluation using Naïve Bayes for both numerical and categorical data. However, some sections may require fixing or clarification, such as the undefined variable mentioned in the last step.

D. INSTRUMENTATION

In this lab, the following major libraries and functions were used for implementing the Navie Bayes algorithm, performing data manipulation, and evaluating the results:

- `pandas`: Data manipulation and analysis, data frames creation, dataset loading, and preprocessing.
- `numpy`: Numerical computations, array operations, mathematical functions for data manipulation and preprocessing.
- `seaborn`: Data visualization, informative statistical graphics, high-level interface for visually appealing plots.
- `plotly.express`: Interactive and dynamic data visualization with customizable charts and plots.
- `sklearn.naive_bayes.GaussianNB`: Provides access to Gaussian Naïve Bayes algorithm, allowing us to build and train models using continuous numerical features that follow a Gaussian distribution.
- `sklearn.naive_bayes.CategoricalNB`: Provides access to Categorical Naïve Bayes algorithm, enabling us to build and train models using discrete or categorical features.
- `train_test_split`: Used for dataset splitting into training and test sets for model evaluation.
- `classification_report`: Generates a comprehensive report of classification results with metrics like precision, recall, F1-score, and support for each class.
- `confusion_matrix`: Computation of the confusion matrix for evaluating model performance and misclassifications.
- `matplotlib.pyplot`: Data visualization, creation of plots, graphs, and figures, including decision tree visualization.

By utilizing these libraries and functions, we were able to effectively implement the decision tree algorithm, preprocess the data, split the dataset, train the models, evaluate their performance, and visualize the results in this lab.

E. SYSTEM METHODOLOGY

The system block diagram is represented in figure 1.

III. RESULTS

A. DATASET ANALYSIS

When conducting Exploratory Data Analysis (EDA) on the heart disease dataset, we observed an imbalance in the gender distribution, with a larger number of male samples compared to female samples. However, this slight bias in gender representation does not appear to be statistically significant, and we can still draw meaningful insights from the data.

The dataset includes individuals within the age range of 27 to 76 years, which is a reasonable choice for analyzing the possibility of heart disease. This age range avoids potential biases that might arise from including very young children or extremely elderly individuals. By focusing on this age range, we can better understand the prevalence of heart disease in the adult population, which is more relevant for our analysis.

The visual analysis of the age distribution in the heart disease dataset, represented by Figure 2, reveals a notable characteristic: the data follows a roughly normal distribution. This aspect is advantageous for our analysis because it allows us to leverage statistical techniques that assume normality, simplifying our modeling process.

Moreover, the mean age of approximately 54 years provides valuable insight into the age distribution of individuals in the dataset. The fact that a significant portion of the dataset consists of individuals in their mid-50s suggests that this age group experiences a relatively higher incidence of heart diseases compared to other age groups.

For a more comprehensive understanding of the relationship between age and heart disease, we further examine the probability density plot in Figure 2. This visualization offers breakthrough insights into how age impacts the likelihood of heart disease. From the plot, it becomes evident that younger individuals, specifically those below the age of 50, have a lower chance of developing heart disease. As age increases beyond 50, the probability of having heart disease begins to rise, peaking between the ages of 50 to 65.

The kernel density estimation in Figure 2 clarifies the trends in the data more effectively. Specifically, it highlights that individuals aged between 30 to 50 have a relatively lower probability of having heart disease, whereas the probability increases as age surpasses 50. These findings are significant for our analysis, as they underscore the association between age and heart disease risk.

Also, exploring the relationship between gender and heart disease through a histogram plot (Figure 4), we observe a notable difference in the likelihood of heart disease between males and females. Males seem to have a substantially higher chance of suffering from heart disease compared to females. Additionally, when analyzing the cross-tabulation of gender and heart disease status (Figure 5), we can see that around 74% of the individuals in the dataset are not suffering from heart disease. Moreover, the percentage of males who are disease-free is higher than the percentage of females without heart disease, which further highlights the disparity in heart disease prevalence between the two genders.

The correlation analysis of the heart disease dataset reveals interesting insights into the relationships between various attributes. By examining the confusion matrix plot (Figure 3), we find that most attribute pairs demonstrate correlation coefficients below 0.6. This indicates a lack of strong linear associations between any single attribute and the likelihood of an individual suffering from heart disease. Consequently, it suggests that a comprehensive approach, considering multiple features together, will be essential for building an effective predictive model for heart disease.

Nonetheless, there are notable exceptions to this observation. Among the attributes, *ChestPainType* with type ASY exhibits a relatively strong positive correlation with heart disease, suggesting that individuals experiencing atypical chest pain may be more likely to have heart issues. Additionally, the *ST_Slope* attribute, particularly with a Flat or Up slope, also shows meaningful correlations with heart disease. This implies that certain patterns in the ST segment of the electrocardiogram during exercise could be indicative of heart disease risk. Figure 6 makes this more clear

On the other hand, the majority of attributes display correlations below 0.6, indicating a lack of strong linear connections with heart disease. While this might pose challenges when designing a classifier, it also emphasizes the significance of considering a combination of attributes to achieve better predictive accuracy. Furthermore, certain attributes demonstrate low correlation with heart disease, suggesting that they may not significantly contribute to the predictive task. These attributes could potentially be dropped from the dataset to reduce noise and enhance the classifier's performance.

Turning our attention to the *Oldpeak* attribute, it exhibits a positive correlation with heart disease. By examining the plots (Figure 9 and Figure 10), we observe that individuals experiencing exercise-induced angina are more likely to suffer from heart disease, particularly if they have higher *Oldpeak* values. The *Oldpeak* values represent the extent of ST segment depression during exercise relative to rest, and elevated values can be indicative of stress on the heart during physical activity, potentially associated with heart disease.

Moreover, the confusion matrix of correlation reinforces the direct positive correlation between *Oldpeak* and *exerciseAngina_Y*. This finding further supports the link between exercise-induced angina, higher *Oldpeak* values, and an increased likelihood of having heart disease.

The analysis of the possibility of having heart disease based on different chest pain types, as depicted in Figure 7, reveals a surprising finding. Individuals with Asymptomatic chest pain, which represents no chest pain, exhibit the highest occurrence of heart disease compared to other types of chest pain. This unexpected result contradicts the common notion that chest pain is a significant symptom associated with heart disease.

Upon examining the confusion matrix plot of correlation, we find that the correlation value between chest pain type ASY and heart disease is positive, with a value of 0.52. This positive correlation indicates that a higher incidence of Asymptomatic chest pain is associated with an increased likelihood of heart disease. This indicates that the possibility of having heart disease is not directly dependent on the type of chest pain rather it depends on multiple attributes.

On analyze figure 8 , if an individual has peak exercise ST segment, he is more likely to have heart diseases rather than if peak exercise ST segment is up or down. This finding suggests that the behavior of the peak exercise ST segment during physical activity is relevant in assessing heart disease risk. The elevated peak exercise ST segment might indicate stress or abnormalities in the heart during exercise, potentially linked to heart disease.

B. MODEL TRAINING

A Naive Bayes classifier was employed to analyze a heart disease dataset, aiming to predict the presence of heart disease in individuals. The model achieved an accuracy of 88%, indicating its overall ability to correctly classify instances. However, a closer examination of the results reveals some interesting observations.

The Naive Bayes classifier demonstrated higher accuracy in predicting cases where individuals had heart disease, while it struggled with misclassifying normal instances. This outcome could be attributed to an imbalance in the dataset, where there were more instances of individuals with heart disease than those without it. The model likely became biased towards the majority class due to this imbalance.

To delve further into the classifier's performance, the F1 score was analyzed for each class. The F1 score is a metric that considers both precision and recall, providing a balanced measure of the classifier's effectiveness for a specific class. For the "disease" class, the F1 score was found to be 0.91, indicating a good balance between precision and recall, and suggesting the classifier performed well for identifying cases of heart disease. In contrast, for the "normal" class, the F1 score was only 0.86, suggesting some room for improvement in correctly classifying individuals without heart disease.

1) Classification Report

The classification report in Table **??** presents a comprehensive overview of the classifier's performance for each class, along with other relevant metrics.

TABLE 1: Classification Report

Class	Precision	Recall	F1-Score	Support
0	0.84	0.87	0.86	75
1	0.91	0.89	0.90	109
Accuracy			0.88	184
Macro Avg	0.88	0.88	0.88	184
Weighted Avg	0.88	0.88	0.88	184

From the classification report, it is evident that the model's precision and recall for the "heart disease" class are 0.84 and 0.87, respectively. Precision indicates the proportion of true positive predictions among all positive predictions, while recall represents the proportion of true positive predictions among all actual positive instances. Similarly, for the "normal" class, the precision and recall are 0.91 and 0.89, respectively.

The accuracy of the classifier, which measures the proportion of correctly classified instances out of the total, is 88%. The macro-averaged F1 score (0.88) and the weighted average F1 score (0.88) are also included in the report. The macro-averaged F1 score calculates the unweighted mean of the F1 scores for each class, while the weighted average F1 score considers the number of instances in each class.

2) Confusion Matrix

The information described in the classification report about predictions can be visualized in the confusion matrix in figure 11 where a nearly equal number of instances in both of the classes are being misclassified.

3) Improving accuracy

To improve the accuracy of the Naive Bayes classifier for heart disease prediction, we performed feature selection by dropping columns that showed a low correlation with the target variable (heart disease). Specifically, we removed all columns that had a correlation magnitude of 0.2 or lower. The dropped columns were: 'RestingBP', 'ChestPainType_TA', 'RestingECG_LVH', 'RestingECG_Normal', 'RestingECG_ST', and 'ST_Slope$_{Down}'$.

The results of this feature selection were promising, as the accuracy of the model increased by 2%, reaching an impressive total accuracy of 90%. This improvement suggests that the removed features were less relevant in contributing to the heart disease prediction task, and their exclusion allowed the classifier to focus on more informative features, leading to better performance.

TABLE 2: Classification Report after improvement

Class	Precision	Recall	F1-Score	Support
0	0.88	0.87	0.87	75
1	0.91	0.92	0.91	109
Accuracy			0.90	184
Macro Avg	0.89	0.89	0.89	184
Weighted Avg	0.90	0.90	0.90	184

As shown in Table **??**, the classification report after feature selection demonstrates improved performance. The precision and recall for the "heart disease" class increased to 0.87 and 0.91, respectively, while for the "normal" class, they improved to 0.92 and 0.88, respectively. Consequently, both classes achieved higher F1 scores, with the "heart disease" class reaching 0.89 and the "normal" class maintaining a score of 0.90.

IV. DISCUSSION & ANALYSIS

After applying Gaussian Naive Bayes as the classifier for our heart disease prediction task, we observed decent but not outstanding results. While the model provided satisfactory outputs, we sought to enhance its accuracy by leveraging insights gained from the dataset.

To improve the classifier's performance, we experimented with different approaches. One such attempt involved removing attributes that exhibited low correlation with heart disease. This step yielded a moderate improvement in accuracy, but it did not surpass our desired results. Additionally, we tried replacing zero values in the "cholesterol" attribute with the mean value to investigate whether this adjustment could enhance accuracy. However, this modification had little effect on the classifier's performance.

Our analysis revealed that all attributes collectively played a significant role in determining the likelihood of an individual having heart disease. This insight was evident from the confusing plot of correlation, emphasizing the importance of considering multiple features when making predictions.

One factor that may have influenced the accuracy of our model is the relatively small size of the dataset. With only around 900 samples, the limited amount of data may have hindered the classifier's ability to learn and differentiate the relevant characteristics effectively. Also, it is essential to recognize that the prediction of heart disease may not be solely determined by the attributes available in our dataset. Other factors, not captured in the data, could also play a crucial role in assessing an individual's risk of heart disease. This realization underscores the complexity of predicting heart disease accurately and suggests that further analysis and research in the medical field may be necessary to uncover additional predictive factors.

V. CONCLUSION

We performed comprehensive analysis on the heart disease dataset which provided valuable insights into the relationships between various attributes and the likelihood of heart disease. Despite a slight imbalance in gender distribution, our analysis demonstrates that meaningful insights can still be drawn from the data. Focusing on the age range of 27 to 76 years, we identified that individuals in their mid-50s are more prone to heart disease, and the age distribution follows a roughly normal pattern, simplifying our modeling process. We also discovered that multiple attributes, such as chest pain type, ST slope, and Oldpeak, show meaningful correlations with heart disease, indicating their potential significance for predictive modeling.

Our analysis also led us to build a Naive Bayes classifier to predict the presence of heart disease in individuals. Although the model achieved an overall accuracy of 88%, we identified areas for improvement. By performing feature selection and excluding less relevant attributes, we successfully increased the accuracy to 90%. The improved precision, recall, and F1 scores for both classes demonstrate the efficacy of this approach in enhancing the classifier's performance.

ref

References

[1] Feng-jiun Yang, *An Implementation of Naive Bayes Classifier*, 2018 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2018, pp. 301-306, doi: 10.1109/CSCI46756.2018.00065.

[2] Wei Zhang, Feng Gao, *An Improvement to Naive Bayes for Text Classification*, *Procedia Engineering*, Volume 15, 2011, Pages 2160-2164, ISSN 1877-7058.

APPENDIX A FIGURES

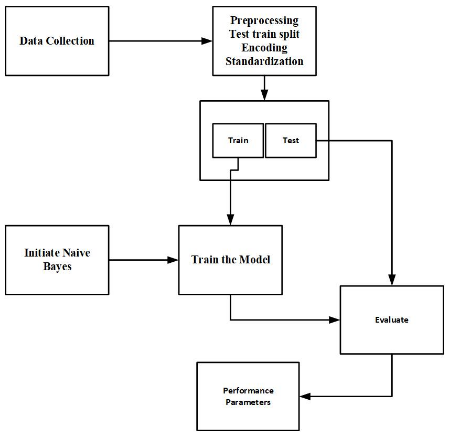

FIGURE 1: System block diagram

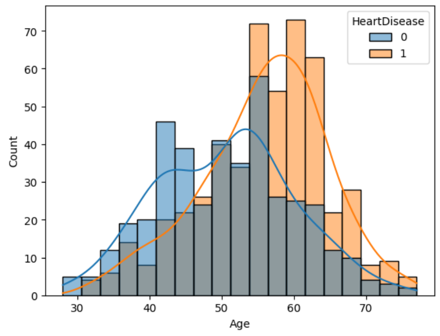

FIGURE 2: KDE plot of Age

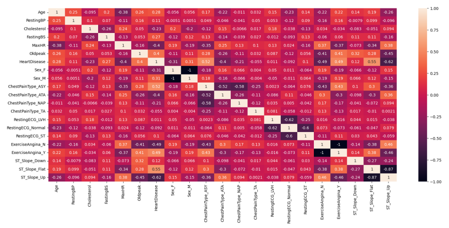

FIGURE 3: Visualizing the correlation Matrix

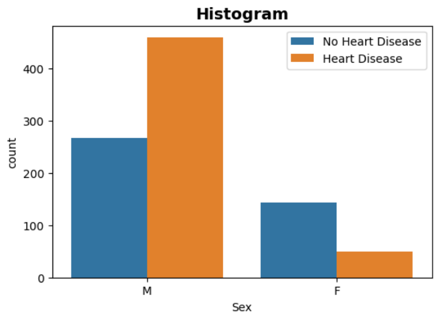

FIGURE 4: Disease based on gender type

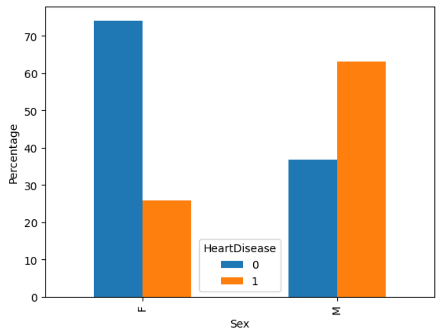

FIGURE 5: Percentage possibility of having heart disease based on gender

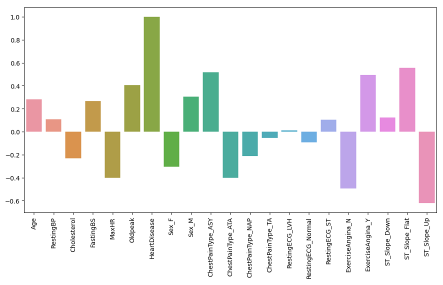

FIGURE 6: Bar plot of Correlation of different attributes with heart disease

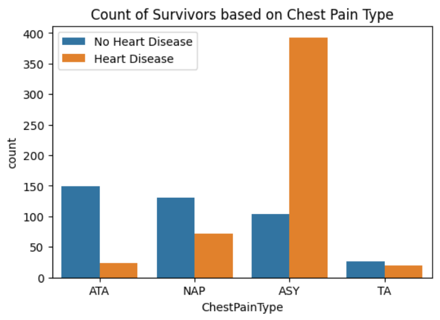

FIGURE 7: Disease with respect to Chest Pain

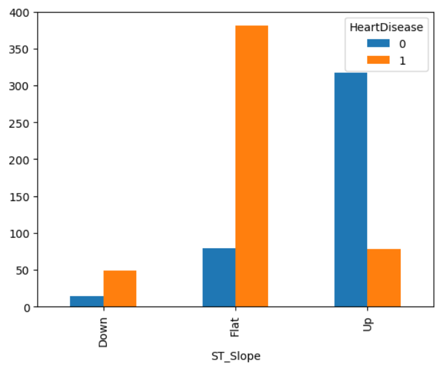

FIGURE 8: Relationship between ST_Slope and disease

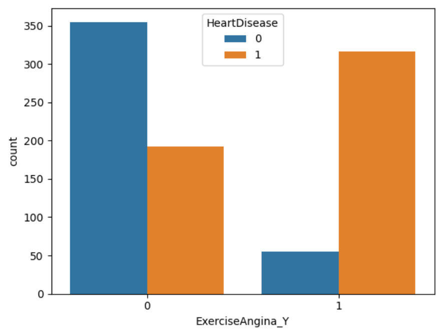

FIGURE 9: Relationship between Exercise Angina Y and disease

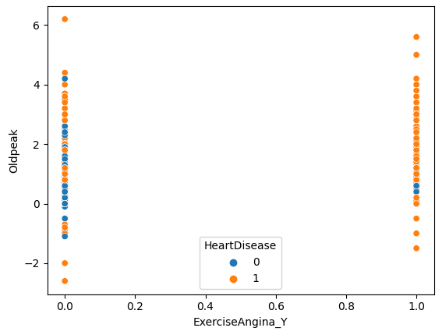

FIGURE 10: Oldpeak and heart disease

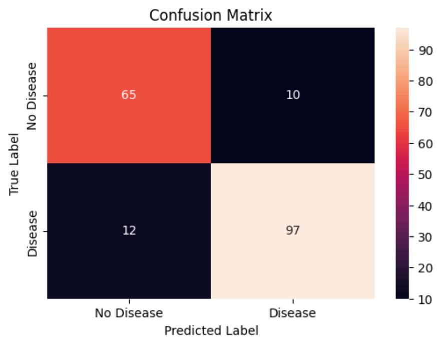

FIGURE 11: Visual plot of Confusion matrix of classifier

APPENDIX B CODE

```python

# Importing libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report, confusion_matrix
import numpy as np
from sklearn.preprocessing import StandardScaler

# Loading the dataset
data = pd.read_csv('heart.csv')
df = pd.get_dummies(data)

# Displaying the first few rows of the dataset
data.head()

# Checking the unique values in the "ChestPainType" column
data["ChestPainType"].unique()

# Displaying descriptive statistics of the dataset
data.describe()

# Visualizing the distribution of "Age" with respect to "HeartDisease"
sns.histplot(data=data, x='Age', kde=True, hue='HeartDisease')

# Visualizing the distribution of "Cholesterol"
sns.histplot(data['Cholesterol'])

# Calculating and visualizing the correlation matrix
cr = df.corr()
plt.figure(figsize=(20, 8))
sns.heatmap(cr, annot=True)

# Finding columns with low correlation to "HeartDisease" for potential dropping later
idx = np.where(abs(cr['HeartDisease'].values) < 0.2)
drop_col_names = cr.columns[idx]
print(drop_col_names)

# Finding columns with high correlation to "HeartDisease"
idx = np.where(abs(cr['HeartDisease'].values) > 0.5)
drop_col_names = cr.columns[idx]
print(drop_col_names)

# Displaying the count of each class in the "HeartDisease" column
print(data['HeartDisease'].value_counts())

# Visualizing the relationship between "Oldpeak" and "ExerciseAngina_Y" with respect to "HeartDisease"
sns.scatterplot(y='Oldpeak', x='ExerciseAngina_Y', data=df, hue='HeartDisease')

# Count plot for "ExerciseAngina_Y" with respect to "HeartDisease"
sns.countplot(x='ExerciseAngina_Y', data=df, hue='HeartDisease')

# Count plot for "HeartDisease"
plt.figure(figsize=(6, 4))
sns.countplot(x='HeartDisease', data=data, hue='HeartDisease')
plt.legend(labels=["No Heart Disease", "Heart Disease"])
plt.show()

# Count plot for "Sex" with respect to "HeartDisease"
plt.figure(figsize=(6, 4))
plt.title('Histogram', fontweight='bold', fontsize=14, fontfamily='sans-serif', color='#000')
sns.countplot(x='Sex', data=data, hue='HeartDisease')
plt.legend(labels=["No Heart Disease", "Heart Disease"])
plt.show()

# Analyzing the percentage of "HeartDisease" for each sex
d = pd.crosstab(data['Sex'], data['HeartDisease']).apply(lambda r: round((r / r.sum()) * 100, 1), axis=1)
print(d)

# Plotting the percentage of "HeartDisease" for each sex
d.plot(kind='bar')
plt.ylabel('Percentage')

# Count plot for "ChestPainType" with respect to "HeartDisease"
plt.figure(figsize=(6, 4))
sns.countplot(x='ChestPainType', hue='HeartDisease', data=data)
plt.legend(labels=["No Heart Disease", "Heart Disease"])
plt.title("Count of Survivors based on Chest Pain Type")
plt.show()

# Histogram of "Cholesterol"
sns.histplot(data, x=data['Cholesterol'])

# Checking the relationship between "ST_Slope" and "HeartDisease"
crs = pd.crosstab(data['ST_Slope'], data['HeartDisease'])
crs.plot(kind='bar')

"""# Training the Naive Bayes classifier"""

# Installing pandas-profiling
!pip install pandas-profiling

from pandas_profiling import ProfileReport

# Creating a profile report for the data
prof = ProfileReport(data)
prof.to_file(output_file='out.html')

# Selecting the features for the pair plot
selected_features = ['Age', 'RestingBP', 'Cholesterol', 'MaxHR', 'Oldpeak']

# Add 'HeartDisease' column to the DataFrame data if it's available
if 'HeartDisease' in data.columns:
    selected_features.append('HeartDisease')

# Pair plot to visualize the distribution of features
plt.figure(figsize=(10, 8))
sns.pairplot(data[selected_features], hue='HeartDisease', palette='husl')
plt.suptitle("Distribution of Features", y=1.02)
plt.show()

# Preparing the data for training and
    testing
X = pd.get_dummies(data)
y = X["HeartDisease"]
X = X.drop('HeartDisease', axis=1)
X.head()

# Splitting the data into training and
    testing sets
X_train, X_test, y_train, y_test =
    train_test_split(X, y, test_size=0.2,
    random_state=40)

# Training the Gaussian Naive Bayes model
gnb_model = GaussianNB()
gnb_model.fit(X_train, y_train)

def eval(gnb_model):
    # Evaluating the model using a
        confusion matrix
    conf_matrix = confusion_matrix(y_test,
        gnb_model.predict(X_test))
    labels = ['No Disease', 'Disease']
    plt.figure(figsize=(6, 4))
    sns.heatmap(conf_matrix, annot=True,
        fmt='d', xticklabels=labels,
        yticklabels=labels)
    plt.title("Confusion Matrix")
    plt.xlabel("Predicted Label")
    plt.ylabel("True Label")
    plt.show()

    # Printing the classification report
    classification_rpt =
        classification_report(y_test,
        gnb_model.predict(X_test))
    print("\nClassification Report:")
    print(classification_rpt)

eval(gnb_model)

# Dropping columns with low correlation to
    "HeartDisease"
idx =
    np.where(abs(cr['HeartDisease'].values)
    < 0.2)
drop_col_names = cr.columns[idx]
print(drop_col_names)
new_X = X.drop(drop_col_names, axis=1)

# Training and evaluating the model with
    the reduced feature set
X_train, X_test, y_train, y_test =
    train_test_split(new_X, y,
    test_size=0.2, random_state=40)
gnb_model = GaussianNB()
gnb_model.fit(X_train, y_train)
eval(gnb_model)

# Dropping all the columns with negative
    correlation
idx = np.where((cr['HeartDisease'].values)
    < 0)
print(idx)
#I will drop these later
drop_neg = cr.columns[idx]

# Dropping columns with negative correlation
new_X = X.drop(drop_neg, axis=1)

# Training and evaluating the model with
    the remaining features
X_train, X_test, y_train, y_test =
    train_test_split(new_X, y,
    test_size=0.2, random_state=40)
gnb_model = GaussianNB()
gnb_model.fit(X_train, y_train)
eval(gnb_model)

"""## Standardizing the Data and testing"""

# Standardizing the data
scaler = StandardScaler()
X_s = scaler.fit_transform(X)

# Training and evaluating the model with
    standardized data
X_train, X_test, y_train, y_test =
    train_test_split(X_s, y,
    test_size=0.2, random_state=40)
gnb_model = GaussianNB()
gnb_model.fit(X_train, y_train)
eval(gnb_model)

# Repeating the process with the reduced
    feature set and standardized data
drop_col_names
new_X = X.drop(drop_col_names, axis=1)
new_X = scaler.fit_transform(new_X)
X_train, X_test, y_train, y_test =
    train_test_split(new_X, y,
    test_size=0.2, random_state=40)
gnb_model = GaussianNB()
gnb_model.fit(X_train, y_train)
eval(gnb_model)
```

APPENDIX C AUTHORS

ARAHANTA POKHAREL was born in 1999 in Biratnagar, Nepal. He is a dedicated individual with a strong passion for learning and research. Currently pursuing a Bachelor's degree in Computer Technology at the Institute of Engineering, Thapathali Campus, he is in the final year of his studies. Throughout his academic journey, he has developed a keen interest in machine learning and data science. Additionally, he has a curious mind and actively engages in quizzes and current affairs to stay updated with the latest information and is committed to acquiring new skills.

AMIT RAJ PANT Amit Raj Pant is a dedicated student pursuing his studies in the Department of Electronics and Computers at Thapathali Engineering Campus, Tribhuvan University, located in Kathmandu, Nepal. With a strong passion for technology, his interests include computer vision and machine learning on resource-constrained edge devices, which involves performing computational tasks on local devices rather than relying solely on remote servers.