Comparison of Naive Bayes, Random Forest, and Decision Tree for Predicting Heart Disease

Mochamad Taufiqul Hafizh[1)], Muammar Qois Al Qorni[2)], Ferry Triwantono[3)]

Abstract

Background: Heart disease is a leading cause of death worldwide, and early detection of the disease can help prevent its occurrence. Machine learning algorithms can aid in predicting the occurrence of heart disease by analyzing patient data. In this study, we compare the performance of three popular classification algorithms, Naive Bayes, Random Forest, and Decision Tree, in predicting the occurrence of heart disease.
Objective: The objective of this study is to compare the performance of Naive Bayes, Random Forest, and Decision Tree algorithms in predicting heart disease using patient data.
Methods: The dataset used in this study was obtained from the Kaggle Website. The dataset would be preprocessed to handle missing value and filling outliers. The dataset would then be transformed and feature extraction would be done. Afterwards, the dataset would be separated into training and testing data. Classification methods of Naive Bayes, Random Forest, and Decision Tree would then be applied to the training data. The models would be tested with the testing data to obtain their performances which would then be compared to each other.
Results: After the models have been tested and evaluated, we discovered that, for this scenario, Naive Bayes has an accuracy of 82.08%, Random Forest has an accuracy of 97.72%, and Decision Tree has an accuracy of 93.49%.
Conclusion: The Random Forest model has a better performance in predicting heart disease compared to the other two.

Keywords: Machine learning, heart disease, performance comparison, naive bayes, decision tree, random forest, prediction

I. INTRODUCTION

Heart disease is a major health concern and a leading cause of death worldwide. Early detection and accurate prediction of heart disease can significantly reduce the mortality rate associated with it. Machine learning algorithms have shown promising results in predicting heart disease by analyzing patient data. Three popular machine learning algorithms for heart disease prediction are Naive Bayes, Random Forest, and Decision Tree.

Naive Bayes is a probabilistic algorithm that assumes the independence of the features. It is a simple and efficient algorithm that is often used as a baseline for comparison with other algorithms. Random Forest is an ensemble algorithm that constructs multiple decision trees and combines their predictions to improve accuracy. Decision Tree is a tree-based algorithm that constructs a decision tree based on the features and their relationships to the target variable.

II. METHODS

A. Data Collection

The dataset used in this work is a secondary data that is obtained from Kaggle and originated from the UCI Machine Learning Repository. The dataset dates from 1988 and consists of four databases: Cleveland, Hungary, Switzerland, and Long Beach V patient data. Here is a snippet of the dataset:

TABLE 1
UNPROCESSED HEART DISEASE DATASET

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
52	1	0	125	212	0	1	168	0	1	2	2	3	0
53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
61	1	0	148	203	0	1	161	0	0	2	1	3	0
62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
58	0	0	100	248	0	0	122	0	1	1	0	2	1
58	1	0	114	318	0	2	140	0	4.4	0	3	1	0
55	1	0	160	289	0	0	145	1	0.8	1	1	3	0

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
46	1	0	120	249	0	0	144	0	0.8	2	0	3	0
54	1	0	122	286	0	0	116	1	3.2	1	2	2	0
71	0	0	112	149	0	1	125	0	1.6	1	0	2	1

The dataset originally contains 76 attributes, including the predicted attribute, but all published experiments refer to using a subset of 14 of them. The "target" field refers to the presence of heart disease in the patient. It is integer valued 0 = “no disease” and 1 = “disease”. The variables are listed as below:

TABLE 2
HEART DISEASE DATASET LIST OF VARIABLES

Attribute	Description	Range
age	age in years	29-77
sex	sex (1 = male, 0 = female)	0,1
cp	chest pain type (0 = typical angina, 1 = atypical angina, 2 = non-anginal pain, 3 = asymptomatic)	0,1,2,3
trestbps	resting blood pressure (in mm Hg on admission to the hospital)	94-200
chol	serum cholesterol in mg/dl	126-564
fbs	(fasting blood sugar > 120 mg/dl) (1 = true, 0 = false)	0,1
restecg	resting electrocardiographic results (0 = normal, 1 = having ST-T wave abnormality, 2 = showing probable or definite left ventricular hypertrophy by Estes' criteria)	0,1,2
thalach	maximum heart rate achieved	71-202
exang	exercise induced angina (1 = yes, 0 = no)	0,1
oldpeak	ST depression induced by exercise relative to rest	0-6.2
slope	the slope of the peak exercise ST segment (0 = upsloping, 1 = flat, 2 = downsloping)	0,1,2
ca	number of major vessels colored by fluoroscopy	0-4
thal	0 = normal, 1 = fixed defect, 2 = reversible defect	0,1,2
condition	0 = no disease, 1 = disease	0,1

B. Data Pre-processing

In this work, data preprocessing is done by data cleaning. Data cleaning is done in order to identify and correct errors, inconsistencies, and missing values in a dataset to avoid biased or inaccurate results in data analysis. This step includes the processes of:

1. Missing Value Removal
2. Outlier Replacement

C. Data Transformation

In this work, z-score normalization will be used to standardize the dataset's values, which have different units and scales. Z-score normalization works by transforming every value in a dataset such that the mean of all of the values is 0 and the standard deviation is 1. This technique will prevent one variable from dominating the analysis due to its larger scale and enable meaningful comparisons between variables.

$$Z = \frac{X-\mu}{\sigma} \qquad (1)$$

D. Feature Selection

In this study, PCA will be used in feature extraction. Principal component analysis (PCA) is a technique for dimensionality reduction and feature extraction. It transforms the original variables into a smaller set of uncorrelated variables. This technique is especially useful in cases of large data sets or when there are many features, as it allows to select the features with the most information while avoiding overfitting.

E. Classification

The classification methods used in this research are Naive Bayes, Decision Tree, and Random Forest. In this section, we will explain each method in detail.

1. Naive Bayes

Naive Bayes is one of the simplest and most frequently used classification methods in machine learning. It is based on Bayes' theorem, which calculates the probability of an event occurring based on the associated probabilities of previous events. In the context of classification, Naive Bayes is used to calculate the probability of an instance belonging to a certain class (e.g. whether a patient has heart disease or not) based on the values of the associated features (variables).

Naive Bayes works by assuming independence between features in the data. This means that the presence or value of a particular feature does not affect the possible values of other features. There are several types of Naive Bayes, such as Bernoulli Naive Bayes, Multinomial Naive Bayes, and Gaussian Naive Bayes. In this research, we use Gaussian Naive Bayes because the data used is continuous.

The Naive Bayes training process is done by calculating the conditional probability for each feature in each class. In this case, we use the training data to calculate the mean and standard deviation for each feature in each class. Next, to classify a new instance, Naive Bayes calculates the probability for each class based on the feature values, and selects the class with the highest probability as the prediction.

2. Decision Tree

Decision tree is a classification algorithm that works on categorical and numerical data. Decision trees are used for creating tree-like structures. It is easy to implement and analyze the data in tree-shaped graphs.

This algorithm splits the data into two or more sets based on the most important indicators. The entropy of each attribute is calculated. The data would then be divided, with predictors having maximum information gain or minimum entropy:

$$E(S) = -(p^{+} \log_2 p^{+}) - (p^{-} \log_2 p^{-}) \quad (1)$$

$$IG(S,A) = E(Y) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} E(S_v) \quad (2)$$

The results obtained are easier to read and interpret. This algorithm has more accuracy in comparison to other algorithms as it analyzes the dataset in the tree-like graph. However, the data may be over classified and only one attribute is tested at a time for decision-making.

3. Random Forest

Random Forest is a classification method based on decision trees. A decision tree is a tree-shaped data structure consisting of nodes that represent decisions based on features in the data, as well as branches that describe possible values for those features.

Random Forest works by building many decision trees and combining the results from all the trees to make the final prediction. The decision trees built in Random Forest will be randomly selected from a sample of data and the features selected are also randomly selected. The advantage of Random Forest is that it is able to overcome overfitting on a single decision tree, thus improving model performance.

The Random Forest training process is done by creating a number of decision trees, with each decision tree built from a randomly selected subset of the data. For each decision tree, each node in the decision tree is based on features randomly selected from the data. Then, the best threshold value for each node is chosen based on information gain. After a number of decision trees are built, the results of all trees are combined to make the final prediction.

F. Model Evaluation

Cross-validation will be used as a technique for evaluating the performance of the models in this work. Cross-validation is a statistical method used to assess the generalization performance of a model by splitting the data into subsets for training and testing. Cross-validation is used to assess the performance of the model on a new, unseen dataset by training the model on one subset of the data and testing it on another. This technique is particularly useful in situations where the dataset is small or where there is a risk of overfitting, as it allows for the estimation of the model's accuracy on new data beyond the training set. Confusion Matrix will also be used in measuring the *accuracy*.

G. Model Comparison

The performance of the three models used in this work is then compared to each other to determine which one is the best for predicting heart disease. Metrics obtained from the previous evaluation will be used to compare the models.

III. RESULTS

1. Preprocessing

cleanData =

1025×14 table

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
52	1	0	125	212	0	1	168	0	1	2	2	3	0
53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
61	1	0	148	203	0	1	161	0	0	2	1	3	0
62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
58	0	0	100	248	0	0	122	0	1	1	0	2	1
:	:	:	:	:	:	:	:	:	:	:	:	:	:
47	1	0	112	204	0	1	143	0	0.1	2	0	2	1
59	1	1	140	221	0	1	164	1	0	2	0	2	1
60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
47	1	0	110	275	0	0	118	1	1	1	1	2	0
50	0	0	110	254	0	0	159	0	0	2	0	2	1
54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

Display all 1025 rows.

2. Normalization

normalizedData =

1025×14 table

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
-0.26831	0.66118	-0.91531	-0.37745	-0.65901	-0.41867	0.89082	0.82092	-0.71194	-0.060859	0.99495	1.2086	1.0893	-1.0262
-0.15808	0.66118	-0.91531	0.47887	-0.83345	2.3862	-1.0036	0.25584	1.4032	1.7263	-2.2426	-0.73161	1.0893	-1.0262
1.7158	0.66118	-0.91531	0.76432	-1.3956	-0.41867	0.89082	-1.0482	1.4032	1.3008	-2.2426	-0.73161	1.0893	-1.0262
0.72373	0.66118	-0.91531	0.93558	-0.83345	-0.41867	0.89082	0.51665	-0.71194	-0.91188	0.99495	0.23851	1.0893	-1.0262
0.83395	-1.511	-0.91531	0.3647	0.93037	2.3862	0.89082	-1.8741	-0.71194	0.70506	-0.62382	2.1788	-0.52187	-1.0262
0.39305	-1.511	-0.91531	-1.8047	0.038765	-0.41867	-1.0036	-1.1786	-0.71194	-0.060859	-0.62382	-0.73161	-0.52187	0.97352
:	:	:	:	:	:	:	:	:	:	:	:	:	:
-0.81943	0.66118	-0.91531	-1.1196	-0.81407	-0.41867	0.89082	-0.26577	-0.71194	-0.82678	0.99495	-0.73161	-0.52187	0.97352
0.50327	0.66118	0.055904	0.47887	-0.48457	-0.41867	0.89082	0.64705	1.4032	-0.91188	0.99495	-0.73161	-0.52187	0.97352
0.6135	0.66118	-0.91531	-0.37745	0.23259	-0.41867	-1.0036	-0.3527	1.4032	1.471	-0.62382	0.23851	1.0893	-1.0262
-0.81943	0.66118	-0.91531	-1.2338	0.5621	-0.41867	-1.0036	-1.3525	1.4032	-0.060859	-0.62382	0.23851	-0.52187	-1.0262
-0.48876	-1.511	-0.91531	-1.2338	0.15506	-0.41867	-1.0036	0.42971	-0.71194	-0.91188	0.99495	-0.73161	-0.52187	0.97352
-0.047854	0.66118	-0.91531	-0.66289	-1.1242	-0.41867	0.89082	-1.5698	-0.71194	0.27955	-0.62382	0.23851	1.0893	-1.0262

Display all 1025 rows.

matrixData =

-0.2683	0.6612	-0.9153	-0.3775	-0.6590	-0.4187	0.8908	0.8209	-0.7119	-0.0609	0.9949	1.2086	1.0893	-1.0262
-0.1581	0.6612	-0.9153	0.4789	-0.8335	2.3862	-1.0036	0.2558	1.4032	1.7263	-2.2426	-0.7316	1.0893	-1.0262
1.7158	0.6612	-0.9153	0.7643	-1.3956	-0.4187	0.8908	-1.0482	1.4032	1.3008	-2.2426	-0.7316	1.0893	-1.0262
0.7237	0.6612	-0.9153	0.9356	-0.8335	-0.4187	0.8908	0.5166	-0.7119	-0.9119	0.9949	0.2385	1.0893	-1.0262
0.8340	-1.5110	-0.9153	0.3647	0.9304	2.3862	0.8908	-1.8741	-0.7119	0.7051	-0.6238	2.1788	-0.5219	-1.0262
0.3930	-1.5110	-0.9153	-1.8047	0.0388	-0.4187	-1.0036	-1.1786	-0.7119	-0.0609	-0.6238	-0.7316	-0.5219	0.9735
0.3930	0.6612	-0.9153	-1.0054	1.3956	-0.4187	2.7852	-0.3962	-0.7119	2.8326	-2.2426	2.1788	-2.1331	-1.0262
0.0624	0.6612	-0.9153	1.6206	0.8335	-0.4187	-1.0036	-0.1788	1.4032	-0.2311	-0.6238	0.2385	1.0893	-1.0262
-0.9297	0.6612	-0.9153	-0.6629	0.0581	-0.4187	-1.0036	-0.2223	-0.7119	-0.2311	0.9949	-0.7316	1.0893	-1.0262
-0.0479	0.6612	-0.9153	-0.5487	0.7753	-0.4187	-1.0036	-1.4394	1.4032	1.8114	-0.6238	1.2086	-0.5219	-1.0262
1.8260	-1.5110	-0.9153	-1.1196	-1.8801	-0.4187	0.8908	-1.0482	-0.7119	0.4498	-0.6238	-0.7316	-0.5219	0.9735
-1.2603	-1.5110	-0.9153	0.0222	1.8414	2.3862	-1.0036	-0.5700	1.4032	1.6412	-0.6238	-0.7316	1.0893	-1.0262
-2.2524	-1.5110	0.0559	-0.7771	-0.6978	-0.4187	0.8908	1.8641	-0.7119	-0.3162	0.9949	-0.7316	-0.5219	0.9735
-0.3785	0.6612	-0.9153	0.4789	1.0079	-0.4187	0.8908	-1.1786	1.4032	2.6624	-0.6238	2.1788	1.0893	-1.0262
-0.2683	0.6612	-0.9153	-0.2062	-0.8141	2.3862	0.8908	0.2993	1.4032	-0.0609	-0.6238	-0.7316	-3.7442	-1.0262
-2.2524	-1.5110	0.0559	-0.7771	-0.6978	-0.4187	0.8908	1.8641	-0.7119	-0.3162	0.9949	-0.7316	-0.5219	0.9735
-0.3785	-1.5110	1.0271	0.4789	1.2017	-0.4187	-1.0036	-0.3092	-0.7119	0.3647	0.9949	0.2385	-0.5219	0.9735
-0.0479	0.6612	-0.9153	-0.4345	0.3877	-0.4187	-1.0036	-1.7437	1.4032	0.9604	-0.6238	0.2385	1.0893	-1.0262
-0.4888	-1.5110	0.0559	-0.6629	-0.0388	-0.4187	0.8908	0.5601	-0.7119	0.0242	0.9949	-0.7316	-0.5219	0.9735
0.3930	0.6612	1.0271	0.4789	-0.6784	2.3862	-1.0036	0.6905	-0.7119	-0.9119	0.9949	-0.7316	-0.5219	0.9735
0.6135	0.6612	1.0271	0.4789	-1.1823	-0.4187	-1.0036	0.2558	-0.7119	1.6412	-0.6238	-0.7316	-0.5219	-1.0262
1.3851	-1.5110	-0.9153	-1.4621	-0.4458	-0.4187	0.8908	-0.3092	-0.7119	-0.6566	0.9949	1.2086	-0.5219	0.9735
-1.0399	0.6612	-0.9153	-1.5763	-0.7365	-0.4187	-1.0036	-0.0484	1.4032	1.6412	-0.6238	-0.7316	-0.5219	0.9735
0.9442	-1.5110	1.0271	0.1934	0.1163	-0.4187	-1.0036	0.9948	-0.7119	-0.9119	0.9949	-0.7316	-0.5219	0.9735
-1.3706	-1.5110	1.0271	-0.6629	-0.7172	-0.4187	0.8908	1.0383	-0.7119	-0.9119	-0.6238	-0.7316	-0.5219	0.9735

```
Y =

    0.9949
   -2.2426
   -2.2426
    0.9949
   -0.6238
   -0.6238
   -2.2426
   -0.6238
    0.9949
   -0.6238
   -0.6238
   -0.6238
    0.9949
   -0.6238
   -0.6238
    0.9949
    0.9949
   -0.6238
    0.9949
    0.9949
   -0.6238
    0.9949
   -0.6238
    0.9949
```

3. Principal Component Analysis

```
coeff =

   -0.2520    0.4347   -0.0372    0.0785    0.2911    0.2001   -0.2496    0.2289   -0.3938   -0.0098   -0.1032    0.5361   -0.0664   -0.2069
   -0.1126   -0.3975    0.4984   -0.2794   -0.0535   -0.0274   -0.1844    0.0998   -0.2139   -0.5347   -0.2149    0.0760    0.2674   -0.0118
    0.2819    0.2446    0.0681   -0.4292   -0.1852    0.2358   -0.2162   -0.1243   -0.2908   -0.1668    0.5294   -0.1978   -0.2946   -0.0346
   -0.1431    0.4454    0.1225   -0.1706   -0.2402    0.1441    0.3103    0.6273    0.2846   -0.1283   -0.0473   -0.2230    0.1389    0.0102
   -0.1022    0.3684   -0.0035    0.5369   -0.3067    0.0048    0.0618   -0.4127   -0.1279   -0.4897   -0.0824   -0.1509    0.1041    0.0284
   -0.0604    0.3123    0.3407   -0.3490    0.2308   -0.2853    0.5181   -0.3650   -0.2227    0.1703   -0.1743    0.0203   -0.0285    0.1204
    0.1127   -0.2376   -0.2858   -0.0557    0.2791    0.6364    0.5112   -0.0766   -0.0886   -0.2664   -0.1033   -0.0066   -0.0170   -0.0782
    0.3678    0.0183    0.2787    0.0232   -0.3329    0.0608    0.1652   -0.1325    0.3715   -0.0202   -0.0288    0.5692   -0.1549   -0.3735
   -0.3377   -0.2037   -0.0752    0.1097   -0.0214   -0.3192    0.3862    0.1304   -0.1307   -0.1550    0.6645    0.2061    0.0433   -0.1770
   -0.3734    0.0031   -0.2240   -0.2881   -0.2592    0.1671   -0.0547   -0.2108    0.2321   -0.0590    0.0474    0.4014   -0.0119    0.6033
    0.3263   -0.0186    0.3711    0.3901    0.2344    0.0571    0.0858    0.2550   -0.0400   -0.0774    0.1815    0.1328   -0.2091    0.6089
   -0.2545    0.1035    0.3649    0.0559    0.4309    0.2929   -0.1741   -0.2735    0.4610    0.0358    0.3269   -0.1090    0.2518   -0.1252
   -0.2159   -0.1783    0.3157    0.1861   -0.4196    0.4165    0.1003   -0.0177   -0.3469    0.5354    0.0392   -0.0563    0.1255    0.0033
    0.4364    0.1431   -0.1698   -0.0617   -0.0060   -0.0204    0.0005   -0.0102   -0.1279    0.0898    0.1641    0.1856    0.8106    0.1139

score =

   -0.1187   -1.4312    1.4602    0.6744    0.5616    1.4935    0.0713   -0.0363    0.8763    0.4190   -0.5955    0.3015   -0.4161    0.0605
   -2.7862   -0.3164    0.4325   -2.0975   -1.3552   -1.5317    1.4689   -0.6017   -0.1947    1.1349   -0.6415    0.8785   -0.0240   -0.2861
   -3.1798   -0.9329   -1.3655   -1.3351   -0.2799    0.7371    0.3783    1.3795   -0.9059    0.4242   -0.4915    0.9215    0.0875   -0.9423
   -0.0860   -0.5881    1.3365    0.6185    0.4924    1.4356    0.3856    1.5715    0.1238    0.3479   -1.0942    0.1577   -0.5048   -0.4147
   -2.2943    2.0744   -0.3002   -0.0445    2.3937    0.5730    1.0219   -1.6689    0.5376    0.4804   -0.8233   -1.0777   -0.0925    0.6046
    0.3293    0.1573   -1.9968    0.8470    0.5847   -1.2165   -1.3535   -0.7490   -0.5165    1.3332   -0.5967    0.0365    0.4443    0.3295
   -2.3278   -0.3946   -2.0417   -0.8392    1.4701    1.9176   -0.3225   -2.7544    1.9205   -2.6039   -1.2237    0.5878    0.3054   -0.0758
   -2.1538    0.0824    0.8747    0.8384   -1.2742   -0.4917    0.4235    1.1186    0.2217   -0.2739    0.1640   -0.4902    0.3698   -0.7647
   -0.0242   -1.3520    1.0281    1.0780   -0.7557   -0.5422   -0.6016    0.1871   -0.1900    0.5773   -0.9894   -0.5358   -0.6299    0.8930
   -2.9350   -0.5821   -0.3501    0.3054    0.2169   -0.9489   -0.8762   -0.7439    0.6671   -0.8891    0.6697    0.0305    0.2829    0.8107
    0.1370   -0.0677   -2.5790   -0.4371    1.7788    0.4586   -0.6551    0.5308   -0.6413    1.6335   -0.7942    1.2080    0.1860    0.0965
   -2.2154    0.8045   -0.2847    0.5737   -1.5939   -1.7175    2.1753   -1.9210   -0.1593    0.9525    0.0537   -0.4685   -0.5262    1.2818
    3.1543   -0.9942   -0.7394    0.3219   -0.4249    0.0680    0.9942   -0.6082    1.4680    0.7499    0.2009    0.1402   -0.4392    0.3796
   -3.6255   -0.8119   -0.0095    0.2386   -0.2342    1.4529    0.4972   -0.9237    1.0749   -0.7944    0.8535   -0.1299    0.8355    1.0479
   -0.1938   -0.2856   -0.7016   -1.8365    2.1565   -2.6633    2.0127   -0.3405    0.6970   -1.8987   -0.7768    0.7512   -1.0727   -0.5428
    3.1543   -0.9942   -0.7394    0.3219   -0.4249    0.0680    0.9942   -0.6082    1.4680    0.7499    0.2009    0.1402   -0.4392    0.3796
    1.0681    1.8300   -0.4584    0.7711   -0.5071   -0.0776   -0.7099   -0.2718    0.5278    0.0211    0.9134   -0.6710    0.1275    1.2744
   -2.8068   -1.0699   -0.0744    0.5615   -0.4638   -0.7081   -0.5380   -0.0557   -0.5676    0.1703    0.4108   -0.4363    0.2735    0.5279
    2.0194    0.0433   -1.2330    0.6664    0.2049    0.4182    0.3961   -0.3038    0.3163    0.4014    0.0128    0.3551   -0.2739    0.7271
    1.7430    1.3990    1.7681   -1.4268    0.4051   -1.2288    0.4378    0.2755   -0.9969    0.2709   -0.3734    0.4108   -0.0025    0.3512
   -0.6058    0.1773   -0.1481   -1.9549   -0.9079   -0.0383   -1.4529    0.6847    0.3568   -0.1584   -0.3542    0.7393   -1.2352    0.4429
    0.8699    0.2973   -0.8471    1.4327    2.5490    0.8487   -0.5790   -0.3594    0.0990    0.9739    0.0037    0.8118    0.3658    0.0242
   -0.4418   -1.9178   -1.0556   -0.5597   -0.6285   -1.8558   -0.4631   -0.6298    0.2856    0.0850    0.5910    1.2619    0.9676    0.8111
    2.0896    1.7974   -0.2434    0.6845   -0.2419   -0.2775   -0.7416    0.6615   -0.1945    0.5906    0.4647    0.6011   -0.5441   -0.1688

explained =

   23.6696
   11.3443
    8.7889
    8.4238
    7.1372
    6.9481
    6.2602
    5.4859
    5.2417
    4.5244
    3.7724
    3.1059
    2.6616
    2.6360
```

4. Check Outlier

```
checkOutlier =

     4   312     0    24    16   153   512     4   345    11     0   447   481   499
```

5. Check Missing Value

```
checkMissing =

     0     0     0     0     0     0     0     0     0     0     0     0     0     0
```

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
52	1	0	125	212	0	1	168	0	1	2	2	3	0
53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
61	1	0	148	203	0	1	161	0	0	2	1	3	0
62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
58	0	0	100	248	0	0	122	0	1	1	0	2	1
58	1	0	114	318	0	2	140	0	4.4	0	3	1	0
55	1	0	160	289	0	0	145	1	0.8	1	1	3	0
46	1	0	120	249	0	0	144	0	0.8	2	0	3	0
54	1	0	122	286	0	0	116	1	3.2	1	2	2	0
71	0	0	112	149	0	1	125	0	1.6	1	0	2	1
43	0	0	132	341	1	0	136	1	3	1	0	3	0
34	0	1	118	210	0	1	192	0	0.7	2	0	2	1
51	1	0	140	298	0	1	122	1	4.2	1	3	3	0
52	1	0	128	204	1	1	156	1	1	1	0	2	0
34	0	1	118	210	0	1	192	0	0.7	2	0	2	1
51	0	2	140	308	0	0	142	0	1.5	2	1	2	1
54	1	0	124	266	0	0	109	1	2.2	1	1	3	0
50	0	1	120	244	0	1	162	0	1.1	2	0	2	1
58	1	2	140	211	1	0	165	0	0	2	0	2	1
60	1	2	140	185	0	0	155	0	3	1	0	2	0

6. Filling Outlier

```
fixData =

  1025×14 table
```

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
52	1	0	125	212	0	1	168	0	1	2	2	3	0
53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
61	1	0	148	203	0	1	161	0	0	2	1	3	0
62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
58	0	0	100	248	0	0	122	0	1	1	0	2	1
:	:	:	:	:	:	:	:	:	:	:	:	:	:
47	1	0	112	204	0	1	143	0	0.1	2	0	2	1
59	1	1	140	221	0	1	164	1	0	2	0	2	1
60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
47	1	0	110	275	0	0	118	1	1	1	1	2	0
50	0	0	110	254	0	0	159	0	0	2	0	2	1
54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

7. Split Data: Cross-validation partition

```
index =

  1025×1 logical array

   1
   0
   0
   1
   0
   0
   0
   1
   1
   0
   1
   0
   1
   0
   0
   1
   0
   0
   1
   0
   0
   0
   0
```

8. Decision Tree

8.1 Train classifier on training data

```
dt =

  ClassificationTree
           PredictorNames: {'age'  'sex'  'cp'  'trestbps'  'chol'  'fbs'  'restecg'  'thalach'  'exang'  'oldpeak'  'slope'  'ca'  'thal'}
             ResponseName: 'target'
    CategoricalPredictors: []
               ClassNames: [0 1]
           ScoreTransform: 'none'
          NumObservations: 718


  Properties, Methods
```

8.2 Test classifier on test data

```
predictYdt =

     0
     0
     0
     0
     1
     1
     1
     1
     1
     0
     0
     1
     1
     1
     0
     1
     0
     0
     1
     1
     0
     0
     0
     1
     1
```

8.3 Evaluate Performance (Decision Tree)

8.3.1 Confusion Matrix

```
cmdt =

   137    14
     6   150
```

8.3.2 Accuracy

```
accuracydt =

    0.9349
```

9. Naive Bayes

9.1 Train Classifier on Training Data

```
nb =

  ClassificationNaiveBayes
           PredictorNames: {'age'  'sex'  'cp'  'trestbps'  'chol'  'fbs'  'restecg'  'thalach'  'exang'  'oldpeak'  'slope'  'ca'  'thal'}
             ResponseName: 'target'
    CategoricalPredictors: []
               ClassNames: [0 1]
           ScoreTransform: 'none'
          NumObservations: 718
        DistributionNames: {'normal'  'normal'  'normal'  'normal'  'normal'  'normal'  'normal'  'normal'  'normal'  'normal'  'normal'  'normal'  'normal'}
    DistributionParameters: {2×13 cell}


  Properties, Methods
```

9.2 Test Classifier on Testing Data

```
predictYnb =

     0
     0
     0
     1
     1
     1
     1
     0
     1
     1
     0
     0
     0
     1
     1
     0
     1
     1
     1
     1
     1
     1
     1
     0
     0
```

9.3 Evaluate Performance (Naive Bayes)

9.3.1 Confusion Matrix

```
cmnb =

   121    34
    21   131
```

9.3.2 Accuracy

```
accuracynb =

    0.8208
```

10. Random Forest

10.1 Train Classifier on Training Data

```
rf =

  TreeBagger
Ensemble with 100 bagged decision trees:
                    Training X:              [718x13]
                    Training Y:              [718x1]
                        Method:       classification
                NumPredictors:                    13
       NumPredictorsToSample:                     4
                   MinLeafSize:                    1
                InBagFraction:                     1
       SampleWithReplacement:                      1
        ComputeOOBPrediction:                      0
 ComputeOOBPredictorImportance:                    0
                    Proximity:                    []
                   ClassNames:             '0'            '1'

  Properties, Methods
```

10.2 Test Classifier on Testing Data

```
predictYrf =

     0
     0
     0
     1
     1
     1
     1
     0
     1
     1
     0
     0
     0
     1
     1
     0
     1
     1
     1
     1
     1
     0
     1
     0
     0
     0
```

10.3 Evaluate Performance (Random Forest)

10.3.1 Confusion Matrix

```
cmrf =

   151     4
     3   149
```

10.3.2 Accuracy

```
accuracyrf =

    0.9772
```

IV. Discussion

The aim of this research is to analyze the performance of various classification algorithms and in doing so find the most accurate algorithm for predicting whether a patient would develop a heart disease or not. This research was done using techniques of Naive Bayes, Decision Tree, and Random Forest on the UCI Heart Disease dataset. Dataset was split into training and test data and models were trained and the accuracy was noted using Matlab. Each algorithm has its unique strengths and weaknesses, the choice of the most appropriate algorithm depends on various factors. A comparison of the performance of the algorithms are depicted below and their accuracy scores are presented in the table.

TABLE 3
Accuracy Table

Algorithm	Accuracy
Naive Bayes	82.08%
Random Forest	97.72%
Decision Tree	93.49%

V. Conclusions

The overall aim of this research is to define various techniques useful in effective heart disease prediction. Efficient and accurate prediction with a lesser number of attributes and tests is the goal of this research. The data were pre-processed and then used in the model. Random Forest with 97.72% accuracy has the best performance in this scenario. Meanwhile, Naive Bayes has the least accurate performance with 82.08%, but still a relatively high score. Our analysis reveals that while each algorithm has its unique strengths and weaknesses, the choice of the most appropriate algorithm depends on various factors.

Our Project findings underscore the importance of careful evaluation of algorithm performance before selecting the most appropriate one for predicting heart disease. Further research can be done to compare these algorithms' performance on different datasets or to incorporate other classification algorithms into the comparison.

Overall, the results of this study are expected to be able to help healthcare professionals and data scientists in choosing the best algorithm for predicting heart disease and other related health conditions, leading to more accurate and efficient medical diagnoses and treatments.

REFERENCES

[1] Kotsiantis, S. B. (2013). Decision trees: a recent overview. Artificial Intelligence Review, 39(4), 261-283.

[2] Zhai, J., Zhang, S., & Wang, C. (2016). The classification of imbalanced large data sets based on MapReduce and ensemble of ELM classifiers. International Journal of Machine Learning and Cybernetics, 7(3), 497-503.

[3] Rokach, L., & Maimon, O. (2014). Data mining with decision trees: theory and applications. World Scientific.

[4] Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.

[5] Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician, 46(3), 175-185.

[6] Domingos, P., & Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. Machine learning, 29(2-3), 103-130.

[7] Liaw, A., & Wiener, M. (2002). Classification and regression by randomForest. R news, 2(3), 18-22.

[8] Biau, G. (2012). Analysis of a random forests model. The Journal of Machine Learning Research, 13(1), 1063-1095.

[9] Quinlan, J. R. (2014). C4. 5: programs for machine learning. Elsevier.

[10] Han, J., Pei, J., & Kamber, M. (2011). Data mining: concepts and techniques. Elsevier.

[11] Chen, X., & Ishwaran, H. (2012). Random forests for genomic data analysis. Genomics, 99(6), 323-329.

[12] Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences, 55(1), 119-139.

[13] McCallum, A., & Nigam, K. (1998). A comparison of event models for Naive Bayes text classification. AAAI 1, 41-48.

[14] T.Nagamani, S.Logeswari, B.Gomathy, Heart Disease Prediction using Data Mining with Mapreduce Algorithm, International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN:2278-3075, Volume-8 Issue-3, January 2019.

[15] Bouali H, Akaichi J. Comparative study of different classification techniques: heart disease use case. In: 2014 13th international conference on machine learning and applications. IEEE. p. 482–86.

[16] Pouriyeh S, Vahid S, Sannino G, De Pietro G, Arabnia H, Gutierrez J. A comprehensive investigation and comparison of machine learning techniques in the domain of heart disease. In: 2017 IEEE symposium on computers and communications (ISCC). IEEE. p. 204–207.

[17] Wahono, H., & Riana, D. (2020). Prediksi Calon Pendonor Darah Potensial Dengan Algoritma Naïve Bayes, K-Nearest Neighbors dan Decision Tree C4.5. JURIKOM (Jurnal Riset Komputer), 7(1), 7. https://doi.org/10.30865/jurikom.v7i1.1953

[18] Rohman, A., Suhartono, V., & Supriyanto, C. (2017). Penerapan Algoritma C4.5 Berbasis Adaboost Untuk Prediksi Penyakit Jantung. Jurnal Teknologi Informasi, 13, 13–19.

[19] Amelia, Y., Eosina, P., & Setiawan, F. A. (2018). Perbandingan Metode Deep Learning Dan Machine Learning Untuk Klasifikasi (Uji Coba Pada Data Penyakit Kanker Payudara). Seminar Nasional Teknologi Informasi, 1, 789–796.

[20] Bahri, S., Marisa Midyanti, D., Hidayati, R., Sistem Komputer, J., & Mipa, F. (2018). Perbandingan Algoritma Naive Bayes dan C4.5 Untuk Klasifikasi Penyakit Anak. Seminar Nasional Aplikasi Teknologi Informasi (SNATi), 24–31.