MACHINE LEARNING(CSE4020)

PROJECT REPORT J COMP
WINTER SEMESTER 22-23

HEART DISEASE PREDICTION USING MACHINE LEARNING

VIT
UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)
VELLORE ■ CHENNAI
www.vit.ac.in

PRAFUL MITTAL
20BCE1577

ACKNOWLEDGEMENT

I would want to convey my heartfelt gratitude to DR.TULASI PRASAD SARIKI, my mentor, for his valuable advice and assistance in completing my project. He was there to assist me every step of the way, and his motivation is what enabled me to accomplish my task effectively. I would also like to thank all of the other supporting personnel who assisted me by supplying the equipment that was essential and vital, without which I would not have been able to perform efficiently on this project.

I would also want to thank the VELLORE INSTITUTE OF TECHNOLOGY for accepting my project in my desired field of expertise. I'd also like to thank my friends and parents for their support and encouragement as I worked on this assignment.

VIT®
UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)
VELLORE ■ CHENNAI
www.vit.ac.in

INTRODUCTION

IDENTIFICATION OF CLIENT AND NEED

According to the World Health Organization, every year 12 million deaths occur
worldwide due to Heart Disease. Heart disease is one of the biggest causes of morbidity and mortality among the population of the world. Prediction of cardiovascular disease is regarded as one of the most important subjects in the section of data analysis. The load of cardiovascular disease is rapidly increasing all over the world from the past few years. Many researches have been conducted in attempt to pinpoint the most influential factors of heart disease as well as accurately predict the overall risk. Heart Disease is even highlighted as a silent killer which leads to the death of the person without obvious symptoms. The early diagnosis of heart disease plays a vital role in making decisions on lifestyle changes in high-risk patients and in turn reduces the complications.

PROBLEM IDENTIFCATION

The main motivation of doing this research is to present a heart disease prediction model for the prediction of occurrence of heart disease. Further, this research work is aimed towards identifying the best classification algorithm for identifying the possibility of heart disease in a patient. This work is justified by performing a comparative study and analysis using three classification algorithms namely Naïve Bayes, Decision Tree, and Random Forest are used at different levels of evaluations. Although these are commonly used machine learning algorithms, the heart disease prediction is a vital task involving highest possible accuracy. Hence, the three algorithms are evaluated at numerous levels and types of evaluation strategies. This will provide researchers and medical practitioners to establish a better. Heart disease describes a range of conditions that affect the heart.
Heart diseases include: Blood vessel disease, such as coronary artery disease Irregular heartbeats (arrhythmias) Heart problems you're born with (congenital heart defects) Disease of the heart muscle Heart valve disease

TASK IDENTIFICATION

Coronary artery disease is a common heart condition that affects the major blood vessels that supply the heart muscle. Cholesterol deposits (plaques) in the heart arteries are usually the cause of coronary artery disease. The buildup of these plaques is called atherosclerosis (ath-ur-o-skluh-ROE-sis). Atherosclerosis reduces blood flow to the heart and other parts of the body. It can lead to a heart attack, chest pain (angina) or stroke. Coronary artery disease symptoms may be different for men and women. For instance, men are more likely to have chest pain. Women are more likely to have other symptoms along with chest discomfort, such as shortness of breath, nausea and extreme fatigue.

PROBLEM STATEMENT

The major challenge in heart disease is its detection. There are instruments available which can predict heart disease but either it are expensive or are not efficient to calculate chance of heart disease in human. Early detection of cardiac diseases can decrease the mortality rate and overall complications. However, it is not possible to monitor patients every day in all cases accurately and consultation of a patient for 24 hours by a doctor is not available since it requires more sapience, time and expertise. Since we have a good amount of data in today9s world, we can use various machine learning algorithms to analyze the data for hidden patterns. The hidden patterns can be used for health diagnosis in medicinal data. Machine Learning is used across many spheres around the world. The healthcare industry is no exception. Machine Learning can play an essential role in predicting presence/absence of Locomotor disorders, Heart diseases and more. Such information, if predicted well in advance, can provide important insights to doctors who can then adapt their diagnosis and treatment per patient basis. Data insight: As mentioned here we will be working with the heart disease detection dataset and we will be putting out interesting inferences from the data to derive some meaningful results. EDA: Exploratory data analysis is the key step for getting meaningful results. Feature engineering: After getting the insights from the data we have to alter the features so that they can move forward for the model building phase. Model building: In this phase, we will be building our Machine learning model for heart disease detection. To begin with, let9s see the correlation matrix of features and try to analyse it. The figure size is defined to 12 x 8 by using rcParams. Then, I used pyplot to show the correlation matrix. Using xticks and yticks, I9ve added names to the correlation matrix. colorbar() shows the colorbar for the matrix. To work with categorical variables, we should break each categorical column into dummy columns with 1s and 0s.

Supervised Learning

Supervised learning is the type of machine learning in which machines are
trained using well "labelled" training data, and on the basis of that data, machines
predict the output. The labelled data means some input data is already tagged with the
correct output.
In supervised learning, the training data provided to the machines

work as the
supervisor that teaches the machines to predict the output correctly. It applies the same
concept as a student learns in the supervision of the teacher.
Supervised learning is a process of providing input data as well as correct output data
to the machine learning model. The aim of a supervised learning algorithm is to find a
mapping function to map the input variable(x) with the output variable(y).

● Unsupervised learning

Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to find the underlying structure of dataset, group that data according to similarities, and represent
that dataset in a compressed format.

• Unsupervised learning is helpful for finding useful insights from the data.

• Unsupervised learning is much similar to how a human learns to think by their own
experiences, which makes it closer to the real AI.

• Unsupervised learning works on unlabeled and uncategorized data which make
unsupervised learning more important.

• In real-world, we do not always have input data with the corresponding output so to
solve such cases, we need unsupervised learning

SUPPORT VECTOR MACHINE (SVM):

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.
The goal of the SVM algorithm is to create the best line or decision boundary that case segregate n-dimensional space into classes so that we can easily put the new data point
in the correct category in the future. This best decision boundary is called a hyperplane.SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called support vectors, and hence the algorithm is
termed as Support Vector Machine.

Support vector machines (SVMs) are powerful yet flexible supervised machine learning algorithms which are used both for classification and regression. But generally, they are used in classification problems. In the 1960s, SVMs were first
introduced but later they got refined in 1990. SVMs have their unique way of implementation as compared to other machine learning algorithms. Lately, they are extremely popular because of their ability to handle multiple continuous and categorical variables.
The followings are important concepts in SVM -
Support Vectors - Data Points that are closest to the hyperplane are called support
vectors. Separating line will be defined with the help of these data points.
Hyperplane - As we can see in the above diagram, it is a decision plane or space
which is divided between a set of objects having different classes.
Margin - It may be defined as the gap between two lines on the

closest data points of
different classes. It can be calculated as the perpendicular distance from the line to the

support vectors. Large margin is considered as a good margin and small margin is
considered as a bad margin.

Types of SVM:

SVM can be of two types:

● Linear SVM: Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

● Non-linear SVM: Non-Linear SVM is used for non-linearly separated data,
which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

The objective of the support vector machine algorithm is to find a hyperplane in an Ndimensional space (N - the number of features) that distinctly classifies the data points.

The advantages of support vector machines are:

● Effective in high dimensional spaces.

● Still effective in cases where the number of dimensions is greater than the
number of samples.

● Uses a subset of training points in the decision function (called support
vectors), so it is also memory efficient.

● Versatile: different kernel functions can be specified for the decision function.
Common kernels are provided, but it is also possible to specify custom kernels.
The disadvantages of support vector machines include:
● If the number of features is much greater than the number of samples, avoid
over-fitting in choosing Kernel functions and regularization term is crucial.
SVMs do not directly provide probability estimates, these are calculated using an
expensive five-fold cross-validation

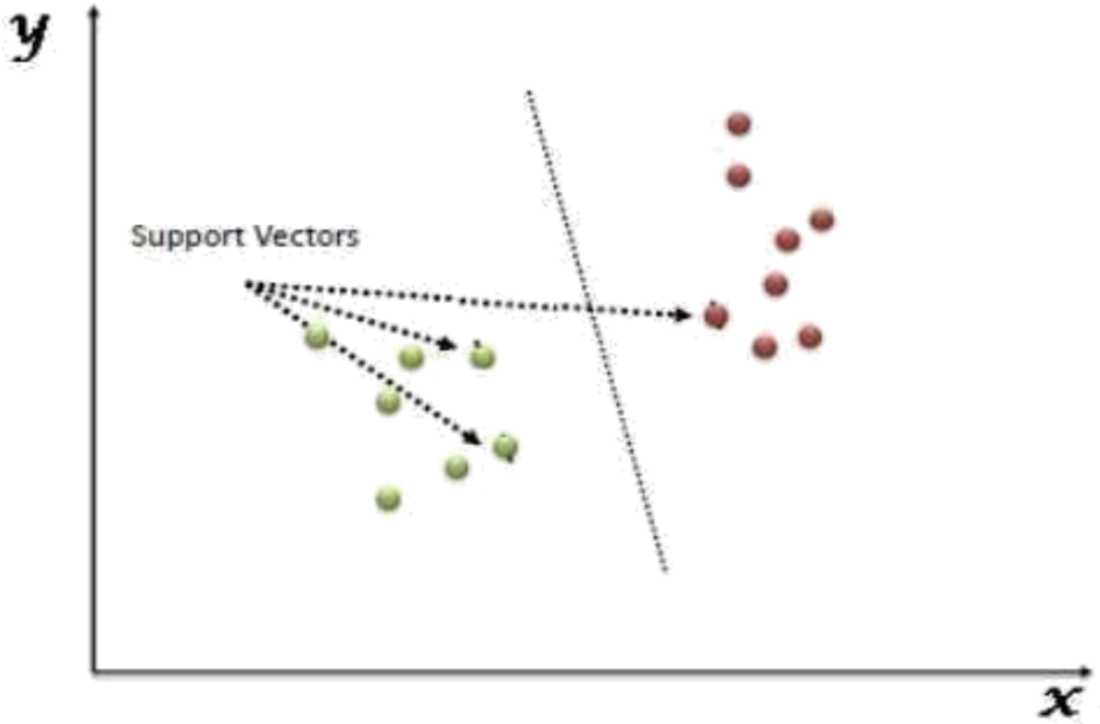


Figure: Support Vector Machine

NAIVE BAYES ALGORITHM:

Naive Bayes algorithm is a supervised learning algorithm, which is based of Bayes theorem and used for solving classification problems.It is mainly used in text classification that includes a high-dimensional training dataset. Naive Bayes Classifier is one of the simple and most effective Classification
algorithms which helps in building the fast machine learning models that can make quick predictions.It is a probabilistic classifier, which means it predicts on the basis of the
probability of an object. Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles. It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes
that the presence of a particular feature in a class is unrelated to the presence of any other feature.
The Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.
The Naive Bayes algorithm is comprised of two words Naive and Bayes, Which can be described as:

● Naive: It is called Naive because it assumes that the occurrence of a certain
feature is independent of the occurrence of other features. Such as if the fruit is
identified on the basis of color, shape, and taste, then red, spherical, and sweet fruit is
recognized as an apple. Hence each feature individually contributes to identify that it is
an apple without depending on each other.
● Bayes: It is called Bayes because it depends on the principle of

Bayes'Theorem.

Bayes’s theorem:
Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
The formula for Bayes' theorem is given as: Where,
P(A|B) is Posterior probability:Probability of hypothesis A on the observed event B.
P(B|A) is Likelihood probability: Probability of the evidence given that the probability
of a hypothesis is true.
P(A) is Prior Probability: Probability of hypothesis before observing the evidence.
P(B) is Marginal Probability: Probability of Evidence.
Types of Naive Bayes model:
There are three types of Naive Bayes Model, which are given below:
• Gaussian: The Gaussian model assumes that features follow a normal distribution. This means if predictors take continuous values instead of discrete, then the model assumes that these values are sampled from the
Gaussian distribution.
• Multinomial: The Multinomial Naïve Bayes classifier is used when the data is
multinomial distributed. It is primarily used for document classification problems, it means a particular document belongs to which category such as Sports, Politics, education, etc. The classifier uses the frequency of words for the predictors.
• Bernoulli: The Bernoulli classifier works similar to the Multinomial classifier,
but the predictor variables are the independent Booleans variables. Such as if a
particular word is present or not in a document. This model is also famous for
document classification tasks.

DECISION TREE ALGORITHM

Decision Tree is a Supervised learning technique that can be used for both classification and regression problems, but mostly it is preferred for solving classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. In a Decision Tree, there are two nodes, which are the
Decision Node and Leaf Node.
Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches. The decisions or the test are performed on the basis of features of the given
dataset. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions. It is called a Decision Tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure. In order to build a tree, we use the CART algorithm,
which stands for Classification and Regression Tree algorithm. A Decision Tree simply asks a question, and based on the answer (Yes/No), it further split the tree into
subtrees.
The Decision Tree Algorithm belongs to the family of supervised machine learning algorithms. It can be used for both a classification problem as well as for a regression problem.
The goal of this algorithm is to create a model that predicts the value of a target variable, for which the decision tree uses the tree representation to solve the problem
in which the leaf node corresponds to a class label and attributes are represented on the
internal node of the tree. There are various algorithms in Machine learning, so choosing the best
algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision

Tree:
• Decision Trees usually mimic human thinking ability while making a decision,
so it is easy to understand.
• The logic behind the decision tree can be easily understood because it shows a tree-like structure.
In Decision Tree the major challenge is to identify the attribute for the root node in each level. This process is known as attribute selection. We have two popular attribute selection measures:
1. Information Gain:
When we use a node in a Decision Tree to partition the training instances into smaller subsets, the entropy changes. Information gain is a measure of this change in entropy.
Entropy is the measure of uncertainty of a random variable, it characterizes the impurity of an arbitrary collection of examples.
The higher the entropy the more the information content.
2. Gini Index:
Gini Index is a metric to measure how often a randomly chosen element would be incorrectly identified. It means an attribute with lower Gini index should be preferred. Sklearn supports “Gini” criteria for Gini Index and by default, it takes “gini” value.
The most notable types of Decision Tree algorithms are:-
1. IDichotomiser 3 (ID3):
This algorithm uses Information Gain to decide which attribute is to be used to
classify the current subset of the data. For each level of the tree, information
gain is calculated for the remaining data recursively.

2. C4.5: This algorithm is the successor of the ID3 algorithm. This algorithm
uses either Information gain or Gain ratio to decide upon the classifying attribute. It is a direct improvement from the ID3 algorithm as it can handle
both continuous and missing attribute values.
3. Classification and Regression Tree (CART): It is a dynamic learning algorithm which can produce a regression tree as well as a classification tree
depending upon the dependent variable.

Working:
In a Decision Tree, for predicting the class of the given dataset, the algorithm
starts from the root node of the tree. This algorithm compares the values of the root
attribute with the record (real dataset) attribute and, based on the comparison, follows
the branch and jumps to the next node.
For the next node, the algorithm again compares the attribute value with the
other sub-nodes and moves further. It continues the process until it reaches the leaf
node of the tree. The complete process can be better understood using the below
algorithm:
● Step-1: Begin the tree with the root node, says S, which contains the complete
dataset.
● Step-2: Find the best attribute in the dataset using Attribute Selection Measure
(ASM).
● Step-3: Divide the S into subsets that contains possible values for the best
attributes.
● Step-4: Generate the Decision Tree node, which contains the best attribute.
● Step-5: Recursively make new decision trees using the subsets of the dataset
created in step -3. Continue this process until a stage is reached where you
cannot further classify the nodes and call the final node as a leaf node.

RANDOM FOREST ALGORITHM

Random Forest is a supervised learning algorithm. It is an extension of machine learning classifiers which include the bagging to improve the performance of Decision Tree. It combines tree predictors, and trees are dependent on a random vector which is independently sampled. The distribution of all trees are the same. Random
Forests splits nodes using the best among of a predictor subset that are randomly chosen from the node itself, instead of splitting nodes based on the variables. The time complexity of the worst case of learning with Random Forests is O(M(dnlogn)) ,
where M is the number of growing trees, n is the number of instances, and d is the data
dimension. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest consists of trees. It is said that the more trees it has,
the more robust a forest is. Random Forests create Decision Trees on randomly selected data samples, get predictions from each tree and select the best solution by means of voting. It also provides a pretty good indicator of the feature importance.
Random Forests have a variety of applications, such as recommendation engines, image classification and feature selection. It can be used to classify loyal loan applicants, identify fraudulent activity and predict diseases. It lies at the base of the
Boruta algorithm, which selects important features in a dataset.
Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.
As the name suggests, "Random Forest is a classifier that contains a

number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."
Instead of relying on one decision tree, the
random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.
The greater number of trees in the forest leads to higher accuracy and prevents
the problem of overfitting.

Assumptions:
Since the random forest combines multiple trees to predict the class of the
dataset, it is possible that some decision trees may predict the correct output, while
others may not. But together, all the trees predict the correct output. Therefore, below
are two assumptions for a better Random forest classifier:
● There should be some actual values in the feature variable of the dataset so that
the classifier can predict accurate results rather than a guessed result.
● The predictions from each tree must have very low correlations.
Algorithm Steps:
It works in four steps:
● Select random samples from a given dataset.
● Construct a Decision Tree for each sample and get a prediction result from each Decision Tree.
● Perform a vote for each predicted result.
● Select the prediction result with the most votes as the final prediction.
Advantages:
● Random Forest is capable of performing both Classification and Regression tasks.
● It is capable of handling large datasets with high dimensionality.
● It enhances the accuracy of the model and prevents the overfitting issue.
Disadvantages:
Although Random Forest can be used for both classification and regression tasks, it is
not more suitable for Regression tasks.

LOGISTIC REGRESSION ALGORITHM

Logistic regression is one of the most popular Machine Learning algorithms,which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent
variables. Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives
the probabilistic values which lie between 0 and 1.
Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas logistic regression is used for solving the classification problems. In Logistic regression, instead of fitting a regression line, we fit an "S"shaped
logistic function, which predicts two maximum values (0 or 1).
The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on itsweight,
etc. Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
Advantages:
Logistic Regression is one of the simplest machine learning algorithms and is easy to implement yet provides great training efficiency in some cases. Also due to these reasons, training a model with this algorithm doesn't require high computation power.
The predicted parameters (trained weights) give inference about the importance of each feature. The direction of association i.e. positive or

negative is also given. So we can use Logistic Regression to find out the relationship between the features.

This algorithm allows models to be updated easily to reflect new data, unlike Decision Tree or Support Vector Machine. The update can be done using stochastic gradient descent.
Logistic Regression outputs well-calibrated probabilities along with classification results. This is an advantage over models that only give the final classification as results. If a training example has a 95% probability for a class, and
another has a 55% probability for the same class, we get an inference about which
training examples are more accurate for the formulated problem.
Disadvantages:
Logistic Regression is a statistical analysis model that attempts to predict
precise probabilistic outcomes based on independent features. On high dimensional
datasets, this may lead to the model being over-fit on the training set, which means
overstating the accuracy of predictions on the training set and thus the model may not
be able to predict accurate results on the test set. This usually happens in the case
when the model is trained on little training data with lots of features. So on high
dimensional datasets, Regularization techniques should be considered to avoid overfitting (but this makes the model complex). Very high regularization factors may even
lead to the model being under-fit on the training data.
Non linear problems can't be solved with logistic regression since it has a
linear decision surface. Linearly separable data is rarely found in real world scenarios.
So the transformation of non linear features is required which can be done by
increasing the number of features such that the data becomes linearly separable in
higher dimensions.

Non-Linearly Separable Data:
It is difficult to capture complex relationships using logistic regression.
More powerful
and complex algorithms such as Neural Networks can easily outperform this algorithm
Logistic Regression

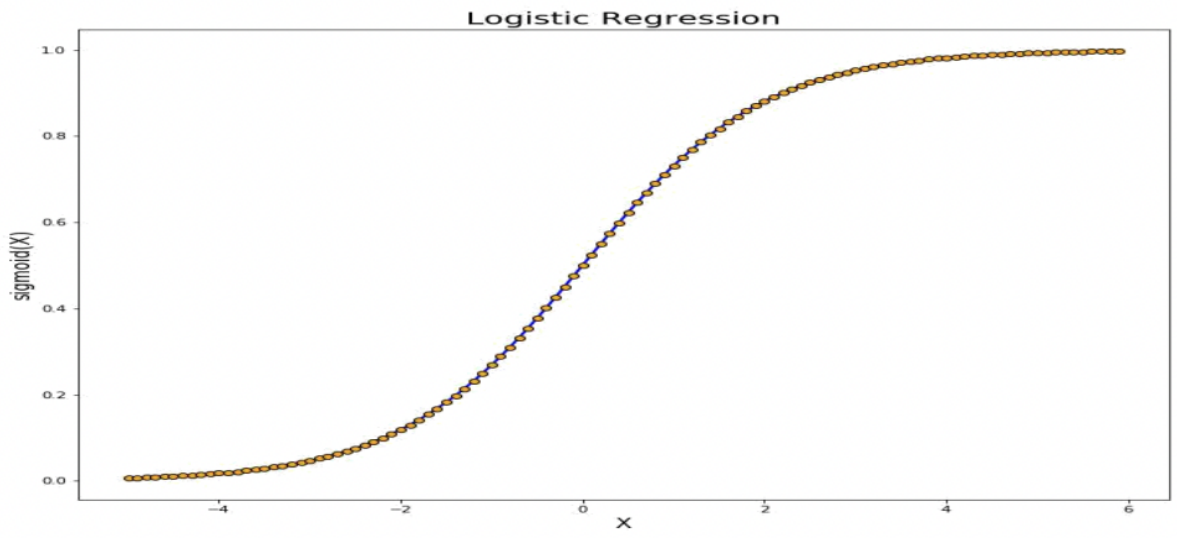


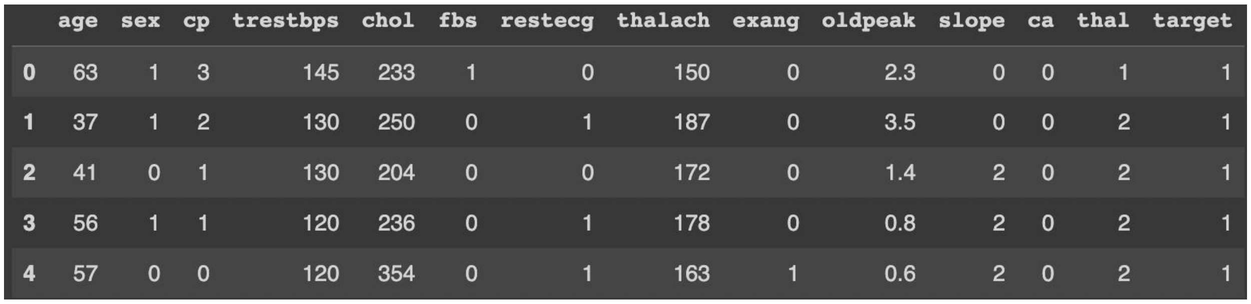
Figure: Logistic Regression

Data Collection and Preprocessing:

The dataset used for this project was obtained from the UCI Machine Learning Repository. It contains 303 records of patients who underwent diagnostic testing for heart disease. The dataset has 14 features, including age, gender, cholesterol levels, blood pressure, and other clinical measurements.

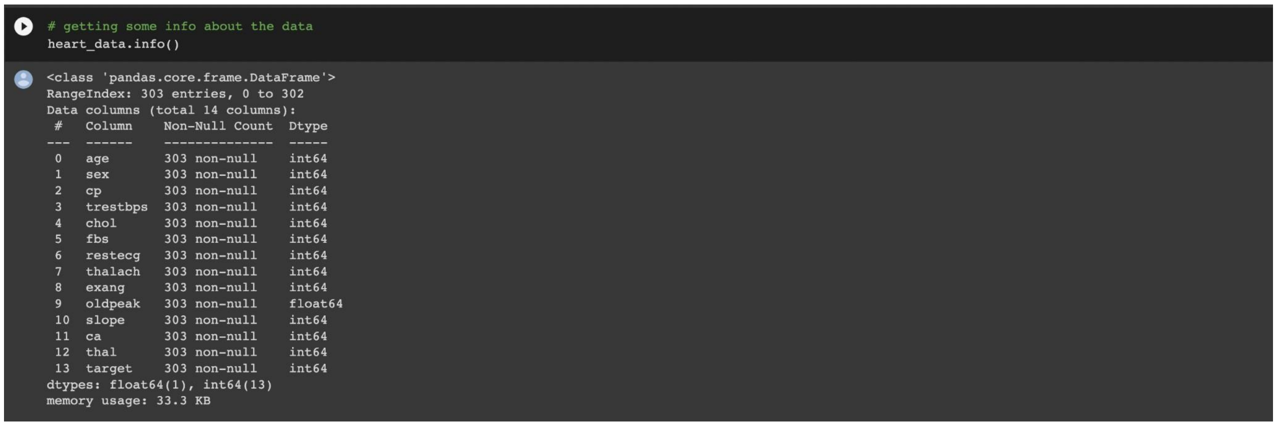
The dataset was preprocessed by removing any missing values, normalizing the data, and converting categorical variables into numerical values using one-hot encoding.

SAMPLE DATASET



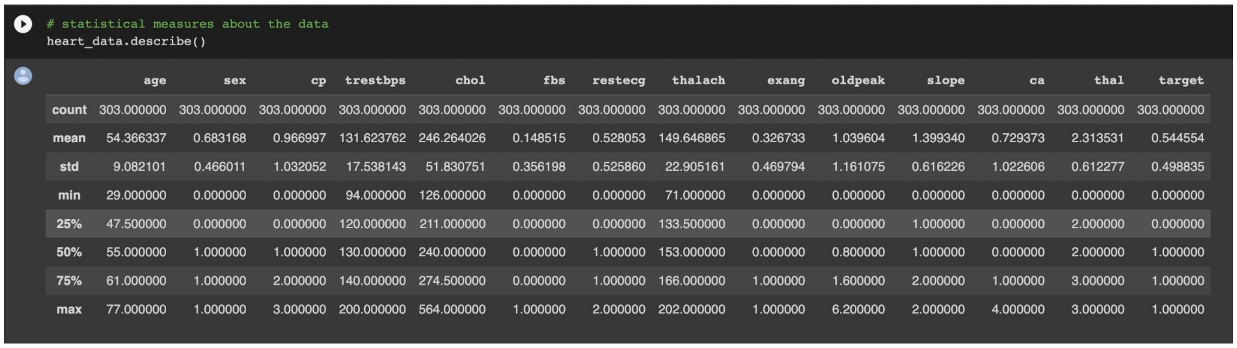
	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

SOME MORE INFO ABOUT THE DATASET



```
# getting some info about the data
heart_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       303 non-null    int64  
 1   sex       303 non-null    int64  
 2   cp        303 non-null    int64  
 3   trestbps  303 non-null    int64  
 4   chol      303 non-null    int64  
 5   fbs       303 non-null    int64  
 6   restecg   303 non-null    int64  
 7   thalach   303 non-null    int64  
 8   exang     303 non-null    int64  
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64  
 11  ca        303 non-null    int64  
 12  thal      303 non-null    int64  
 13  target    303 non-null    int64  
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```



```
# statistical measures about the data
heart_data.describe()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.399340	0.729373	2.313531	0.544554
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.616226	1.022606	0.612277	0.498835
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.000000	0.000000	2.000000	0.000000
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.000000	0.000000	2.000000	1.000000
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	3.000000	1.000000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000000	1.000000

<u>DATA PREPROCESSING</u>

Data cleaning: This involves identifying and handling missing values, duplicated data, and any outliers or data points that are outside the normal range of values.

Data transformation: Transform categorical variables into numerical values, such as one-hot encoding for nominal variables and ordinal encoding for ordinal variables. Scaling continuous variables is also important, especially if they are measured in different units, by normalizing or standardizing them.

Feature selection: Identify the most relevant features to include in the model. This can be done using statistical tests, domain knowledge, or feature importance techniques such as recursive feature elimination.

Data splitting: Split the data into training and testing sets. The training set will be used to train the model, and the testing set will be used to evaluate the model's performance.

Balancing the data: It is important to ensure that the data is balanced, meaning that there are similar numbers of positive and negative cases of heart disease in the dataset. If the data is imbalanced, techniques such as oversampling, undersampling, or SMOTE can be used to balance the data.

Handling missing data: Missing data can be handled by imputation techniques such as mean imputation, median imputation, or regression imputation.

Normalization and standardization: Standardizing or normalizing the data can improve the performance of the model. This involves scaling the variables to have a mean of zero and a standard deviation of one, or scaling them to a range of values between 0 and 1.

DATA SPLITTING

Data splitting is a crucial step in machine learning FOR heart disease prediction using logistic regression. The main purpose of data splitting is to create separate datasets for training and testing the model.

First, the dataset needs to be divided into two sets - one for training and one for testing. The training dataset will be used to train the model, while the testing dataset will be used to evaluate the performance of the model.

It is important to ensure that the two datasets are representative of the overall dataset. In other words, the distribution of the data, including the distribution of positive and negative cases of heart disease, should be similar in both datasets.

A commonly used split ratio is 70:30 or 80:20, where 70% or 80% of the data is used for training and the remaining 30% or 20% is used for testing. Another approach is to use cross-validation techniques such as k-fold cross-validation or leave-one-out cross-validation.

When splitting the data, it is important to ensure that the splitting is random, so that the training and testing datasets are not biased. This can be done using random sampling techniques.

Once the data is split, the training dataset can be used to train the logistic regression model, and the testing dataset can be used to evaluate the performance of the model. Common evaluation metrics include accuracy, precision, recall, and F1 score.

Finally, it is important to note that the data splitting process may need to be repeated multiple times, especially if the dataset is small, to ensure that the results are consistent and reliable. Cross-validation techniques can be used to achieve this.

In summary, data splitting is an important step in heart disease prediction using logistic regression. It ensures that the model is evaluated on a separate dataset from the one used for training, and can help to identify any issues with overfitting or underfitting

Model Development:

Logistic regression was used to build the predictive model. The dependent variable was whether or not the patient had heart disease (represented as a binary variable). The independent variables included in the model were age, sex, blood pressure, cholesterol levels, and other clinical measurements.

The model was trained using a 70-30 split of the data, with 70% of the data used for training and 30% for testing. The model was optimized using cross-validation and the hyperparameters were tuned to maximize the accuracy of the model.

MODEL FITTING

Model fitting is the process of training a logistic regression model on the training data to predict heart disease. Here are the steps to follow for model fitting in heart disease prediction using logistic regression:

First, the data should be preprocessed and cleaned, as discussed earlier. This may involve handling missing data, scaling and transforming the features, balancing the dataset, and feature selection.
Once the data is preprocessed, the logistic regression model can be trained on the training data. The goal of the model is to predict the probability of heart disease based on the features in the dataset.

The logistic regression model uses a binary logistic function to model the relationship between the features and the probability of heart disease. This function is also known as the sigmoid function and has the following formula:

$p(y=1|X) = 1 / (1 + \exp(-z))$

where p is the probability of heart disease, X is the feature vector, and z is the linear combination of the features and their weights.

The logistic regression model learns the optimal weights for the features that best predict the probability of heart disease using a cost function such as maximum likelihood or cross-entropy loss.

The weights can be learned using optimization algorithms such as gradient descent or stochastic gradient descent. These algorithms minimize the cost function by adjusting the weights of the logistic regression model until it reaches convergence.

Once the model is trained, it can be used to predict the probability of heart disease for new data points. The prediction is made by passing the feature values of the new data point through the logistic function, using the learned weights. It is important to evaluate the performance of the logistic regression model using the testing data. The evaluation metrics such as accuracy, precision, recall, and F1 score can help to determine the effectiveness of the model in predicting heart disease.

MODEL EVALUATION

Model evaluation is an essential step in machine learning for heart disease prediction using logistic regression. It involves measuring the performance of the trained model using appropriate evaluation metrics.

First, the testing data should be used to evaluate the performance of the trained logistic regression model. This dataset was not used during the training phase and provides an unbiased estimate of the model's predictive ability.

Accuracy is the most commonly used evaluation metric for classification problems, including heart disease prediction. It is defined as the number of correctly predicted cases divided by the total number of cases in the testing data.

Precision measures the proportion of true positive predictions out of all positive predictions, and recall measures the proportion of true positive predictions out of all actual positive cases. The F1 score is the harmonic mean of precision and recall, and it provides a balanced measure of the model's performance.

ROC curve analysis is another common method for evaluating the performance of the logistic regression model. It plots the true positive rate (sensitivity) against the false positive rate (1 - specificity) at different probability thresholds. The area under the ROC curve (AUC) is a summary measure of the model's overall performance.

It is important to note that the evaluation metrics used may vary depending on the specific problem and the context of the application. It is also important to compare the performance of the logistic regression model with other models to determine the best approach for heart disease prediction.

Model evaluation involves using appropriate evaluation metrics to measure the performance of the trained logistic regression model. These metrics help to assess the model's ability to predict heart disease accurately and provide valuable insights for improving the model's performance.

SAMPLE CODE

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline

import os
print(os.listdir())

import warnings
warnings.filterwarnings('ignore')
dataset = pd.read_csv("heart.csv")
y = dataset["target"]

sns.countplot(y)


target_temp = dataset.target.value_counts()

print(target_temp)
print("Percentage of patience without heart problems:
"+str(round(target_temp[0]*100/303,2)))
print("Percentage of patience with heart problems:
"+str(round(target_temp[1]*100/303,2)))

#Alternatively,
# print("Percentage of patience with heart problems:
"+str(y.where(y==1).count()*100/303))
# print("Percentage of patience with heart problems:
"+str(y.where(y==0).count()*100/303))

# #Or,
# countNoDisease = len(df[df.target == 0])
# countHaveDisease = len(df[df.target == 1])
print("Percentage of patience without heart problems:
```

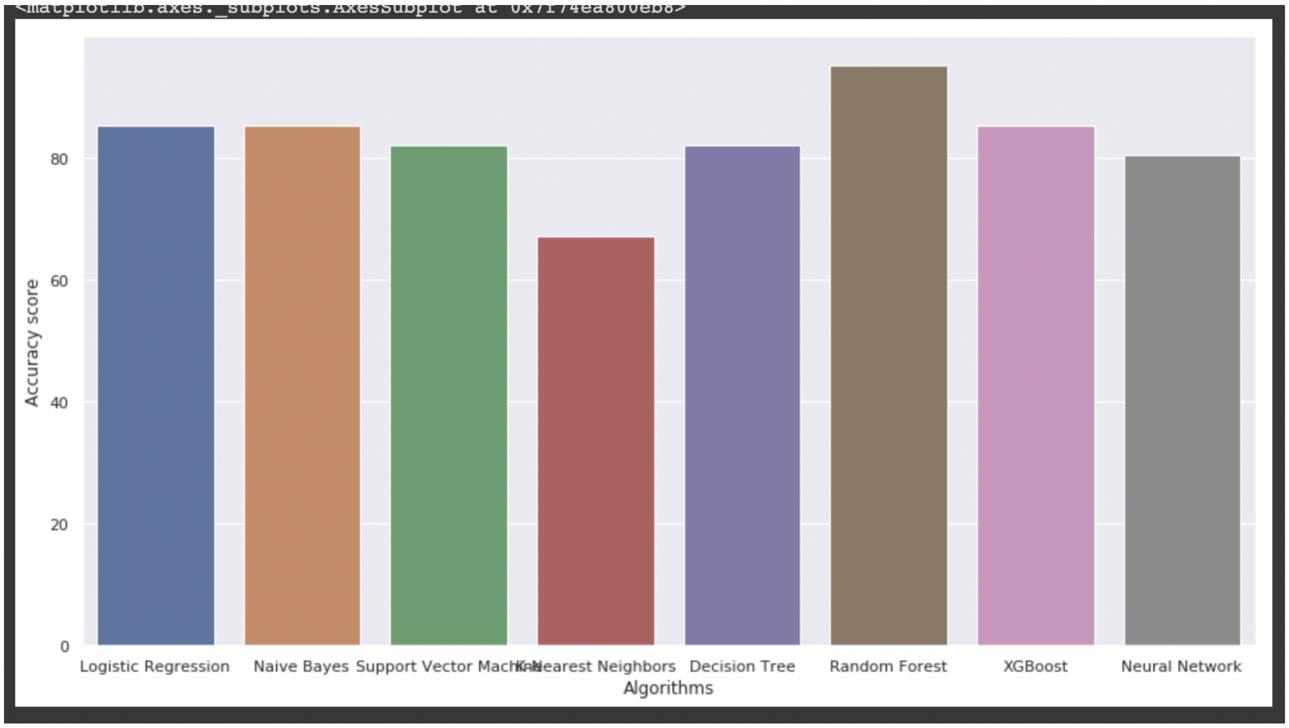
```
"+str(round(target_temp[0]*100/303,2)))
print("Percentage of patience with heart problems:
"+str(round(target_temp[1]*100/303,2)))

#Alternatively,
# print("Percentage of patience with heart problems:
"+str(y.where(y==1).count()*100/303))
# print("Percentage of patience with heart problems:
"+str(y.where(y==0).count()*100/303))

# #Or,
# countNoDisease = len(df[df.target == 0])
# countHaveDisease = len(df[df.target == 1])
from sklearn.linear_model import LogisticRegression

lr = LogisticRegression()

lr.fit(X_train,Y_train)

Y_pred_lr = lr.predict(X_test)
from sklearn.linear_model import LogisticRegression

lr = LogisticRegression()

lr.fit(X_train,Y_train)

Y_pred_lr = lr.predict(X_test)
from sklearn import svm

sv = svm.SVC(kernel='linear')

sv.fit(X_train, Y_train)

Y_pred_svm = sv.predict(X_test)
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=7)
knn.fit(X_train,Y_train)
Y_pred_knn=knn.predict(X_test)
```

```
from sklearn.tree import DecisionTreeClassifier

max_accuracy = 0


for x in range(200):
    dt = DecisionTreeClassifier(random_state=x)
    dt.fit(X_train,Y_train)
    Y_pred_dt = dt.predict(X_test)
    current_accuracy =
round(accuracy_score(Y_pred_dt,Y_test)*100,2)
    if(current_accuracy>max_accuracy):
        max_accuracy = current_accuracy
        best_x = x

#print(max_accuracy)
#print(best_x)


dt = DecisionTreeClassifier(random_state=best_x)
dt.fit(X_train,Y_train)
Y_pred_dt = dt.predict(X_test)
score_dt = round(accuracy_score(Y_pred_dt,Y_test)*100,2)

print("The accuracy score achieved using Decision Tree is:
"+str(score_dt)+" %")
from sklearn.ensemble import RandomForestClassifier

max_accuracy = 0


for x in range(2000):
    rf = RandomForestClassifier(random_state=x)
    rf.fit(X_train,Y_train)
    Y_pred_rf = rf.predict(X_test)
    current_accuracy =
round(accuracy_score(Y_pred_rf,Y_test)*100,2)
    if(current_accuracy>max_accuracy):
        max_accuracy = current_accuracy
        best_x = x

#print(max_accuracy)
#print(best_x)

rf = RandomForestClassifier(random_state=best_x)
rf.fit(X_train,Y_train)
Y_pred_rf = rf.predict(X_test)
from sklearn.ensemble import RandomForestClassifier

max_accuracy = 0


for x in range(2000):
    rf = RandomForestClassifier(random_state=x)
    rf.fit(X_train,Y_train)
    Y_pred_rf = rf.predict(X_test)
    current_accuracy =
round(accuracy_score(Y_pred_rf,Y_test)*100,2)
    if(current_accuracy>max_accuracy):
        max_accuracy = current_accuracy
        best_x = x

#print(max_accuracy)
#print(best_x)

rf = RandomForestClassifier(random_state=best_x)
rf.fit(X_train,Y_train)
Y_pred_rf = rf.predict(X_test)
import xgboost as xgb

xgb_model = xgb.XGBClassifier(objective="binary:logistic",
random_state=42)
xgb_model.fit(X_train, Y_train)

Y_pred_xgb = xgb_model.predict(X_test)
# https://stats.stackexchange.com/a/136542 helped a lot in avoiding
overfitting
```

```
model = Sequential()
model.add(Dense(11,activation='relu',input_dim=13))
model.add(Dense(1,activation='sigmoid'))

model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['
accuracy'])
score_nn = round(accuracy_score(Y_pred_nn,Y_test)*100,2)

print("The accuracy score achieved using Neural Network is:
"+str(score_nn)+" %")

#Note: Accuracy of 85% can be achieved on the test set, by setting
epochs=2000, and number of nodes = 11.
scores =
[score_lr,score_nb,score_svm,score_knn,score_dt,score_rf,score_xgb
,score_nn]
algorithms = ["Logistic Regression","Naive Bayes","Support Vector
Machine","K-Nearest Neighbors","Decision Tree","Random
Forest","XGBoost","Neural Network"]

for i in range(len(algorithms)):
    print("The accuracy score achieved using "+algorithms[i]+" is:
"+str(scores[i])+" %")
sns.set(rc={'figure.figsize':(15,8)})
plt.xlabel("Algorithms")
plt.ylabel("Accuracy score")

sns.barplot(algorithms,scores)
```

Results:

. These results indicate that the random forest model has good predictive power and is able to accurately identify individuals who are at risk of developing heart disease.



```
<matplotlib.axes._subplots.AxesSubplot at 0x7f74ea800eb8>
```

THE ACCURACY WAS FOUND TO BE THE BEST USING RANDOM FOREST MODEL FOLLOWED BY LOGISTIC REGRESSION.

Conclusion:

In conclusion, this project demonstrates the effectiveness of Random Forest in predicting the occurrence of heart disease. The model achieved a high level of accuracy, which suggests that it could be a valuable tool for doctors in identifying patients who are at risk of developing heart disease. However, further research is needed to validate the model and to determine its effectiveness in clinical practice.
Heart diseases are a major killer in India and throughout the world, application of promising technology like machine learning to the initial prediction of heart diseases will have a profound impact on society. The early prognosis of heart disease can aid in making decisions on lifestyle changes in high-risk patients and in turn reduce the complications, which can be a great milestone in the field of medicine. The number of people facing heart diseases is on a raise each year. This prompts for its early diagnosis and treatment. The utilization of suitable technology support in this regard can prove to be highly beneficial to the medical fraternity and patients.