Artificial Intelligence

Lab Project Submission :

Project Title:

HEART DISEASE PREDICTION USING MACHINE LEARNING ALGORITHMS

Under Guidance of : Dr Jyoti Maggu

Team Members:

Name:	*Roll Number:*
Mrityunjay Pandey	102117182
Santosh Kumar Rathi	102117186
Kushagra Chandra Agarwal	102117176
Hargun Singh	102297016

INTRODUCTION

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.

Machine learning proves to be effective in assisting in making decisions and predictions from the large quantity of data produced by the health care industry. This project aims to predict future Heart Disease by analysing data of patients which classifies whether they have heart disease or not using machine learning algorithm. Machine Learning techniques can be a boon in this regard.

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 four classification algorithms namely Support Vector Machine, KNN, Logistic Regression, 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

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 everyday in all cases accurately and consultation of a patient for 24 h ours by a doctor is not available since it requires more sapience, time and expertise. Since we have a good amount of data in today's world, we can use various machine learning algorithms to analyse the data for hidden patterns. The hidden patterns can be used for health diagnosis in medicinal data.

LITERATURE SURVEY

[1] **Purushottam , proposed a paper "Efficient Heart Disease Prediction System" using hill climbing and decision tree algorithms** .They used Cleveland dataset and pre-processing of data is performed before using classification algorithms. The Knowledge Extraction is done based on Evolutionary Learning (KEEL), an open-source data mining tool that fills the missing values in the data set. A decision tree follows top-down order. For each actual node selected by hill-climbing algorithm a node is selected by a test at each level. The accuracy of the system is about 86.7%.

2] **Sonam Nikhar, proposed paper " Prediction of Heart Disease Using Machine Learning Algorithms"** their research gives point to point explanation of Naïve Bayes and decision tree classifier that are used especially in the prediction of Heart Disease. Some analysis has been led to think about the execution of prescient data mining strategy on the same dataset, and the result decided that Decision Tree
has highest accuracy than Bayesian classifier.

[3] **Santhana Krishnan. J , proposed a paper "Prediction of Heart Disease Using Machine Learning Algorithms"** using decision tree and Naive Bayes algorithm for prediction of heart disease. In decision tree algorithm the tree is built using certain conditions which gives True or False decisions. The algorithms like SVM, KNN are results based on vertical or horizontal split conditions depends on dependent variables.. They have also used Cleveland data set. Dataset splits in 70% training and 30% testing by using some methods. This algorithm gives 91% accuracy. The second algorithm is Naive Bayes, which is used for classification. It can handle complicated, nonlinear, dependent data so it is found suitable for heart disease dataset as this dataset is also complicated, dependent and nonlinear in nature. This algorithm gives an 87% accuracy.

[4] **Senthil Kumar Mohan, proposed "Effective Heart Disease Prediction Using Hybrid Machine Learning Techniques"** in which their main objective is to improve exactness in cardiovascular problems. The algorithms used are **KNN, LR, SVM, NN** to produce an improved exhibition level with a precision level of 88.7% through the prediction model for heart disease with hybrid random forest with linear model**(HRFLM)**.

METHODOLOGY

PROPOSED SYSTEM

The working of the system starts with the collection of data and selecting the important attributes. Then the required data is preprocessed into the required format. The data is then divided into two parts training and testing data. The algorithms are applied and the model is trained using the training data. The accuracy of the system is obtained by testing the system using the testing data. This system is implemented using the following modules.

1.) Collection of Dataset
2.) Selection of attributes
3.) Data Pre-Processing
4.) Disease Prediction

Collection of dataset :

Initially, we collect a dataset for our heart disease prediction system. After the collection of the dataset, we split the dataset into training data and testing data. The training dataset is used for prediction model learning and testing data is used for evaluating the prediction model. For this project, 80% of training data is used and 20% of data is used for testing. The dataset used for this project is Heart Disease UCI. The dataset has 14 attributes that are used for the system.

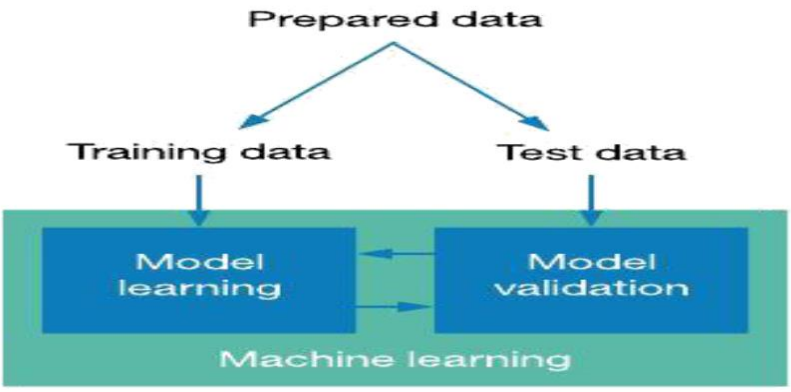


Figure: Collection of Data

Selection of attributes :

Attribute or Feature selection includes the selection of appropriate attributes for the prediction system. This is used to increase the efficiency of the system. Various attributes of the patient like gender, chest pain type, fasting blood pressure, serum cholesterol, exang , etc are selected for the prediction. The Correlation matrix is used for attribute selection for this model.

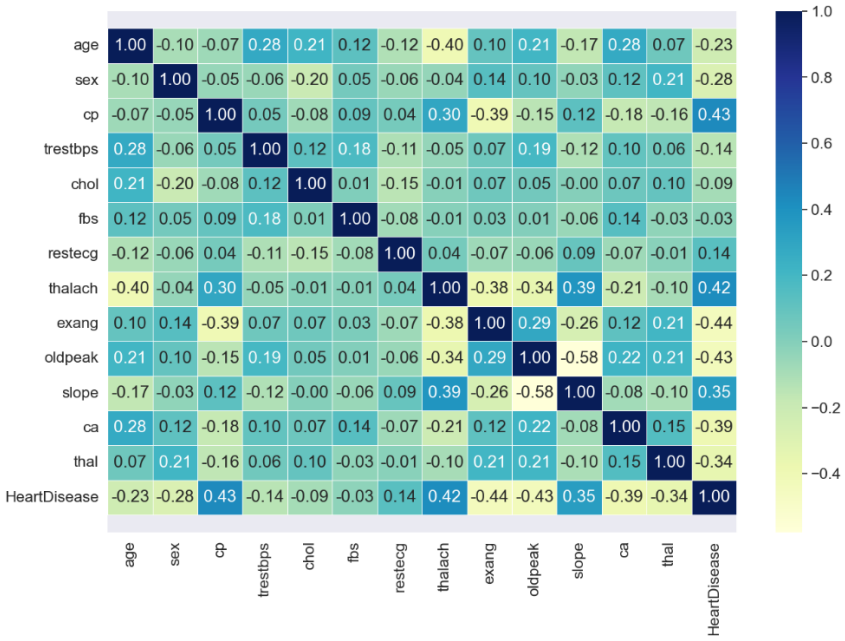


Figure: Correlation matrix

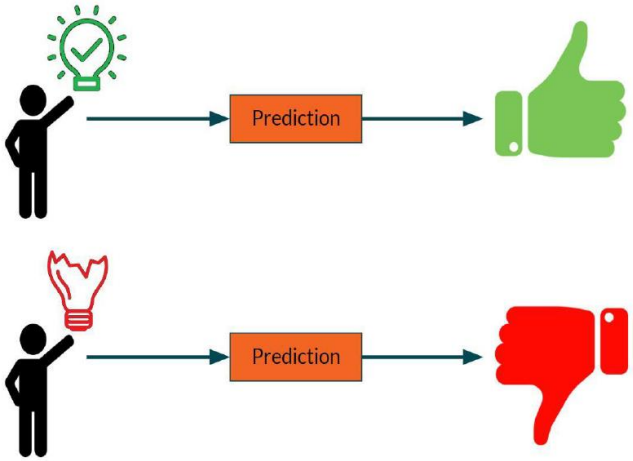
Pre-processing of Data

Data pre-processing is an important step for the creation of a machine learning model. Initially, data may not be clean or in the required format for the model which can cause misleading outcomes. In pre-processing of data, we transform data into our required format. It is used to deal with noises, duplicates, and missing values of the dataset. Data pre-processing has the activities like importing datasets, splitting datasets, attribute scaling, etc. Pre-processing of data is required for improving the accuracy of the model.

Figure: Data Pre-processing

Prediction of Disease:

Various machine learning algorithms like SVM, KNN, Random Tree and Logistic Regression are used for classification. Comparative analysis is performed among algorithms and the algorithm that gives the highest accuracy is used for heart disease prediction.



ALGORITHMS:

[1] 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 its weight, 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.

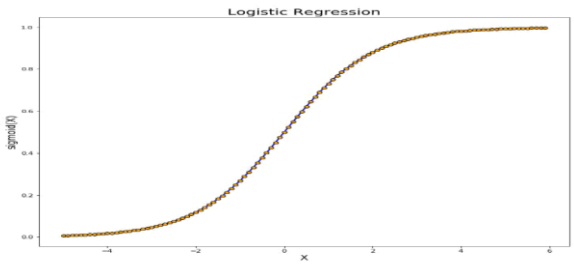


Figure: Logistic Regression

[2] 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 can 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. 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 N-dimensional space (N - the number of features) that distinctly classifies the data points.

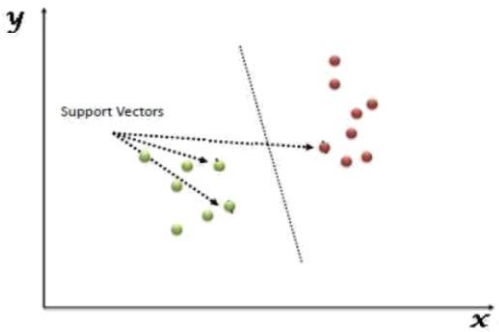


Figure: Support Vector Machine

[3] 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 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:

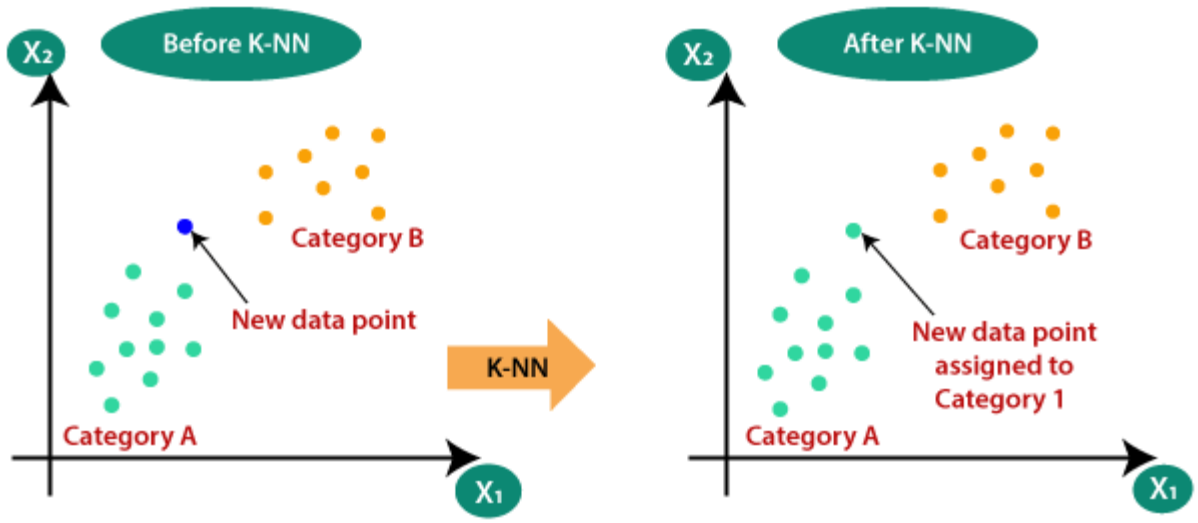
It works in four steps:

- 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.

[4] K nearest neighbour classifier:

K nearest neighbour(KNN) is a simple algorithm, which stores all cases and classify new cases based on similarity measure. KNN algorithm also called as 1) case based reasoning 2) k nearest neighbour 3)example based reasoning 4) instance based learning 5) memory based reasoning 6) lazy learning [4].KNN algorithms have been used since 1970 in many applications like statistical estimation and pattern recognition etc. KNN is a non parametric classification method which is broadly classified into two types1) structure less NN techniques 2) structure based NN techniques. In structure less NN techniques whole data is classified into training and test sample data. From training point to sample point distance is evaluated, and the point with lowest distance is called nearest neighbour. Structure based NN techniques are based on structures of data like orthogonal structure tree (OST), ball tree, k-d tree, axis tree, nearest future line and central line [5].Nearest neighbour classification is used mainly when all the attributes are continuous

Algorithm Steps:
Steps 1) find the K training instances which are closest to unknown instance
Step2) pick the most commonly occurring classification for these K instances



DATASET DETAILS: (https://drive.google.com/drive/folders/1Nivo0Z0Rwl8k9_dOslcQ0SC-3U1uZ0Ze)

- They are 14 attributes in our dataset.
- They are total 303 entries from which we are using 242 for training purposes for our AI Model and 61 for testing purposes.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target	
	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1	
	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1	
	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1	
	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1	
	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1	
	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1	
	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1	
	44	1	1	120	263	0	1	173	0	0	2	0	3	1	
)	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1	
1	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1	
2	54	1	0	140	239	0	1	160	0	1.2	2	0	2	1	
3	48	0	2	130	275	0	1	139	0	0.2	2	0	2	1	
4	49	1	1	130	266	0	1	171	0	0.6	2	0	2	1	
5	64	1	3	110	211	0	0	144	1	1.8	1	0	2	1	
6	58	0	3	150	283	1	0	162	0	1	2	0	2	1	
7	50	0	2	120	219	0	1	158	0	1.6	1	0	2	1	
8	58	0	2	120	340	0	1	172	0	0	2	0	2	1	
9	66	0	3	150	226	0	1	114	0	2.6	0	0	2	1	
0	43	1	0	150	247	0	1	171	0	1.5	2	0	2	1	
1	69	0	3	140	239	0	1	151	0	1.8	2	2	2	1	
2	59	1	0	135	234	0	1	161	0	0.5	1	0	3	1	
3	44	1	2	130	233	0	1	179	1	0.4	2	0	2	1	
4	42	1	0	140	226	0	1	178	0	0	2	0	2	1	
5	61	1	2	150	243	1	1	137	1	1	1	0	2	1	
6	40	1	3	140	199	0	1	178	1	1.4	2	0	3	1	
7	71	0	1	160	302	0	1	162	0	0.4	2	2	2	1	
8	59	1	2	150	212	1	1	157	0	1.6	2	0	2	1	
9	51	1	2	110	175	0	1	123	0	0.6	2	0	2	1	

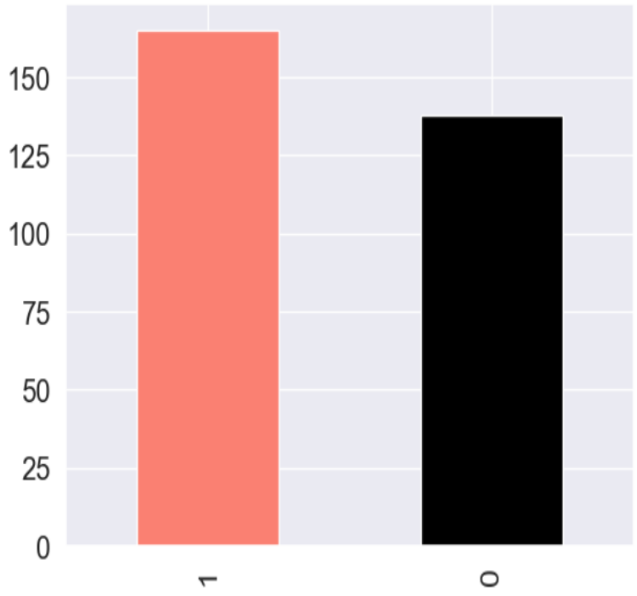
Figure: Dataset Attributes

```
<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  HeartDisease  303 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

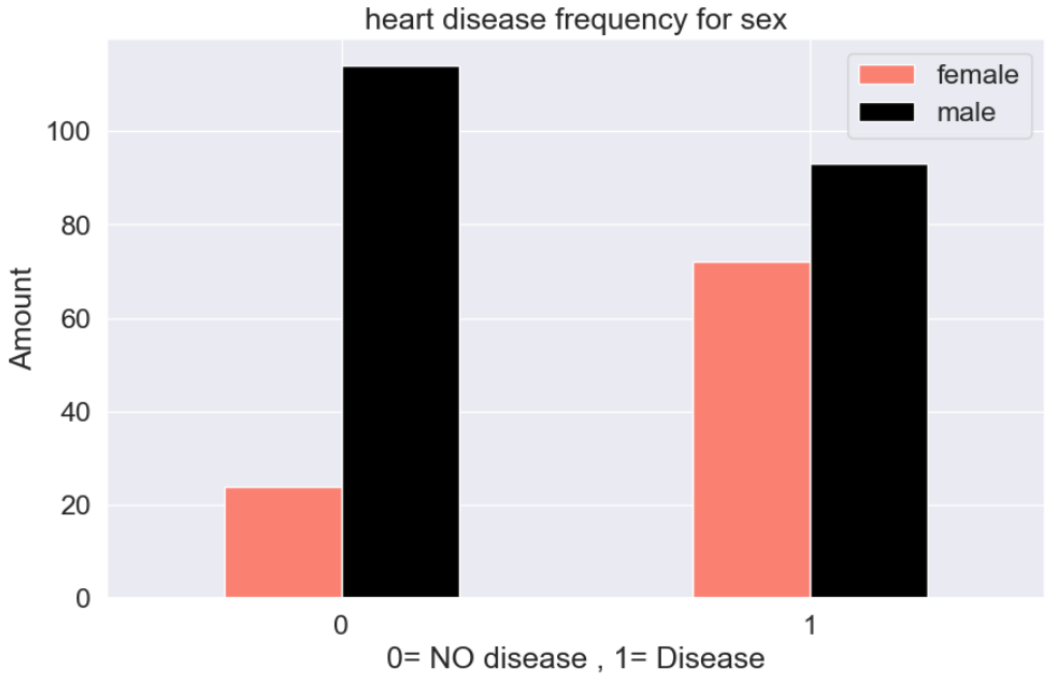
DATA REPRESENTATION:USING MATPLOTLIB and SEABORN.

Graph showing the number of people with heart disease in our dataset.(1-Heart Disease, 2-No Heart Disease

In [105]:
```
df["HeartDisease"].value_counts().plot(kind="bar", color=["salmon", "black"]);
```

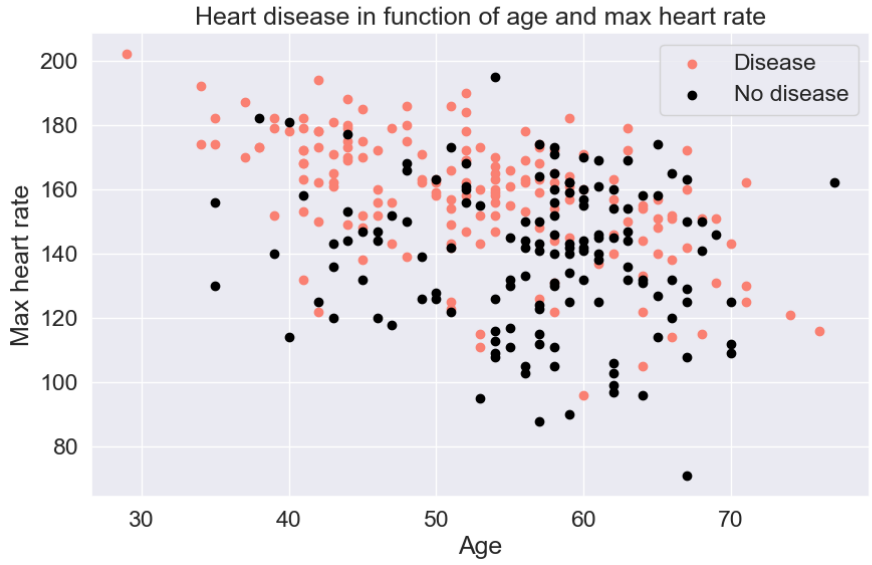



```
# create a plot for cross tab
pd.crosstab(df.HeartDisease, df.sex).plot(kind="bar",
                                          figsize=(10,6),
                                          color=["salmon", "black"])
plt.title("heart disease frequency for sex")
plt.xlabel("0= NO disease , 1= Disease")
plt.ylabel("Amount")
plt.legend(["female", "male"])
plt.xticks(rotation=0);
```

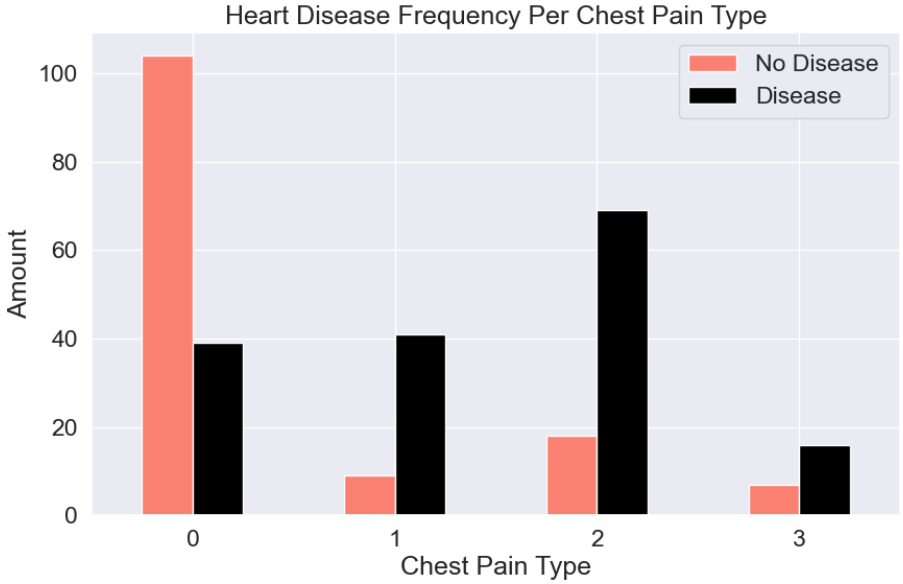


Age vs Max Heart rate for heart disease

```
# create another figure
plt.figure(figsize=(10,6))


plt.scatter(df.age[df.HeartDisease==1],
            df.thalach[df.HeartDisease==1],
            c="salmon")


plt.scatter(df.age[df.HeartDisease==0],
            df.thalach[df.HeartDisease==0],
            c="black");


plt.title("Heart disease in function of age and max heart rate")
plt.xlabel("Age")
plt.ylabel("Max heart rate")
plt.legend(["Disease", "No disease"]);
```

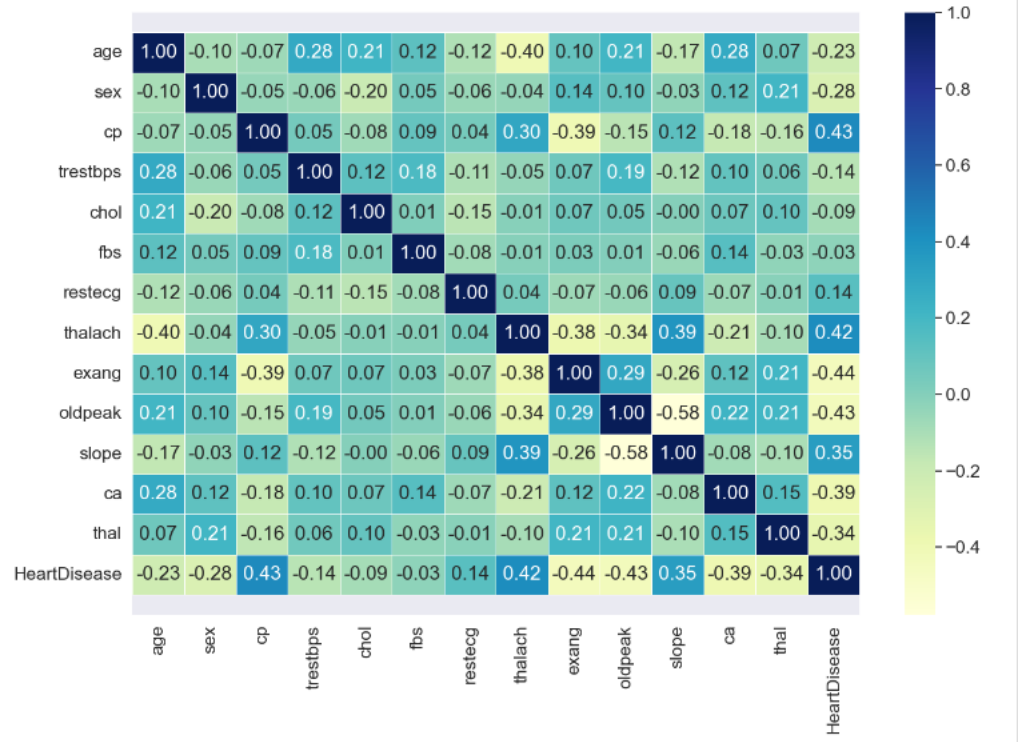



```
pd.crosstab(df.cp, df.HeartDisease).plot(kind="bar",
                                         figsize=(10, 6),
                                         color=["salmon", "black"])


plt.title("Heart Disease Frequency Per Chest Pain Type")
plt.xlabel("Chest Pain Type")
plt.ylabel("Amount")
plt.legend(["No Disease", "Disease"])
plt.xticks(rotation=0);
```



```
# correlation matrix prettier
corr_matrix = df.corr()
fig, ax = plt.subplots(figsize=(15, 10))
ax = sns.heatmap(corr_matrix,
                 annot=True,
                 linewidths=0.5,
                 fmt=".2f",
                 cmap="YlGnBu");
bottom, top = ax.get_ylim()
ax.set_ylim(bottom + 0.5, top - 0.5)
```

```
(14.5, -0.5)
```



CODE EXPLANATION(of our finalised model) :

[1]:

```
import pandas as pd #reading of the data set
import numpy as np #matrix multiplication and data manipulation
import matplotlib.pyplot as plt
import seaborn as sns


##iterate dataset into two parts one for training and other for testing
from sklearn.model_selection import train_test_split,cross_val_score

##standardscaler function to standardize the numerical_cols in unit variance

from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import mean_squared_error,accuracy_score, confusion_matrix
from sklearn.metrics import precision_score, recall_score, f1_score
```

Here we are importing different important libraries that are being used in the model:

Brief description of the imported modules:

1. **Pandas:** Pandas is a powerful open-source data manipulation and analysis library for Python. It provides easy-to-use data structures such as DataFrame and Series, which allow for efficient data handling and analysis tasks
2. **Numpy:** NumPy contains a multi-dimensional array and matrix data structures. It can be utilised to perform a number of mathematical operations on arrays
3. **Seaborn:** Seaborn is a Python data visualization library based on Matplotlib that provides a high-level interface for creating attractive and informative statistical graphics. Seaborn is designed to work well with Pandas, another popular data manipulation library in Python, making it a powerful tool for data visualization and analysis.
4. **matplotlib:** Matplotlib is a multi-platform data visualization library built on NumPy arrays. It allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

[2]:

```
df =pd.read_csv(r"D:\AI project\heart-disease.csv.xls")
```

```
df.head()
```

	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

```
df=df.rename(columns={"target":"HeartDisease"})
```

Now we load our given dataset, using panda library and further change the target column name into HeartDisease for our convenience.

[3]:

```
df=pd.get_dummies(df,columns=["cp","restecg"])
```

```
df.head()
```

	age	sex	trestbps	chol	fbs	thalach	exang	oldpeak	slope	ca	thal	HeartDisease	cp_0	cp_1	cp_2	cp_3	restecg_0	restecg_1	restecg_2
0	63	1	145	233	1	150	0	2.3	0	0	1	1	0	0	0	1	1	0	0
1	37	1	130	250	0	187	0	3.5	0	0	2	1	0	0	1	0	0	1	0
2	41	0	130	204	0	172	0	1.4	2	0	2	1	0	1	0	0	1	0	0
3	56	1	120	236	0	178	0	0.8	2	0	2	1	0	1	0	0	0	1	0
4	57	0	120	354	0	163	1	0.6	2	0	2	1	1	0	0	0	0	1	0

```
numerical_cols=["age","trestbps","chol","thalach","oldpeak"]
cat_cols=list(set(df.columns)-set(numerical_cols)-{"HeartDisease"})
```

```
cat_cols
```

```
['cp_1',
 'restecg_1',
 'cp_3',
 'cp_2',
 'fbs',
 'cp_0',
 'restecg_0',
 'restecg_2',
 'sex',
 'slope',
 'ca',
 'thal',
 'exang']
```

```
numerical_cols
```

```
['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
```

Here we are converting our dataset into a formatted form in order for our algorithms to work better , one such technique we are applying here which is one-hot encoding. One-hot encoding transforms categorical features to a format that works better with classification and regression algorithms. It's very useful in methods where multiple types of data representation is necessary. Dummies using pandas to convert categorical value to one-hot encoding.

[4]:

```
df_train , df_test= train_test_split(df,test_size=0.2,random_state=42)
```

```
len(df_train),len(df_test)
```

```
(242, 61)
```

```
##creating object of standard scaler
scaler=StandardScaler()

#for custom input
def get_features_and_target_arrays(df,numericals_cols,cat_cols,scaler):
    x_numeric_scaled=scaler.fit_transform(df[numerical_cols])   ##normalization (0,1) scale
    x_categorical=df[cat_cols].to_numpy()   ## categorical cols
    x=np.hstack((x_categorical,x_numeric_scaled))

    return x

def get_features_and_target_arrays(df,numericals_cols,cat_cols,scaler):
    x_numeric_scaled=scaler.fit_transform(df[numerical_cols])   ##normalization (0,1) scale
    x_categorical=df[cat_cols].to_numpy()   ## categorical cols
    x=np.hstack((x_categorical,x_numeric_scaled))
    y=df["HeartDisease"]

    return x,y

x_train, y_train=get_features_and_target_arrays(df_train,cat_cols,numerical_cols,scaler)
X = df.drop("HeartDisease", axis=1)

y = df["HeartDisease"]
```

- Here we are splitting our dataset into two set, one for training purposes and another one for testing purposes. We used 80% percent of dataset for training and 20% for testing.
- In the next part we are using Standard Scaler Library to convert numerical data which has wide range of value into range of 0-1.
- After doing the last part, we are training our data for further evaluations.

[5]:

Logistic Regression Model

```
## Training of the model

clf=LogisticRegression()
clf.fit(x_train,y_train)
x_test,y_test=get_features_and_target_arrays(df_test,cat_cols,numerical_cols,scaler)
test_pred=clf.predict(x_test)
print("Error:",mean_squared_error(y_test,test_pred))
print("Accuracy:",accuracy_score(y_test,test_pred))
```

```
Error: 0.21311475409836064
Accuracy: 0.7868852459016393
```

Now we are testing our model using logistic regression modelling by using the inbuild function for it, further we are printing the error and accuracy percentage obtained by it.

[6]:

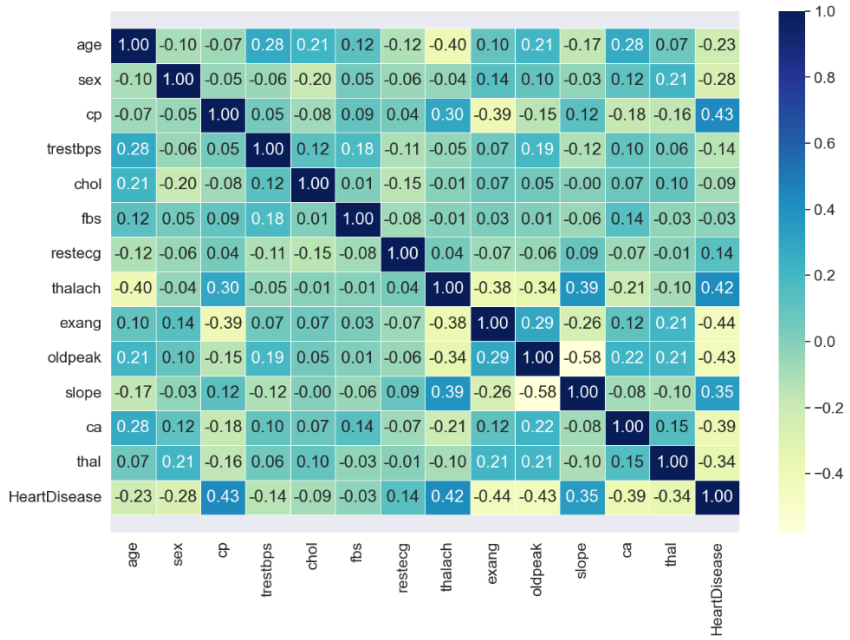
```
# Prompt the user for input
age = int(input("Enter age: "))
sex = int(input("Enter sex (0 for female, 1 for male): "))
cp = int(input("Enter cp (chest pain type): "))
trestbps = int(input("Enter trestbps (resting blood pressure): "))
chol = int(input("Enter chol (serum cholesterol level): "))
fbs = int(input("Enter fbs (fasting blood sugar > 120 mg/dl): "))
restecg = int(input("Enter restecg (resting electrocardiographic results): "))
thalach = int(input("Enter thalach (maximum heart rate achieved): "))
exang = int(input("Enter exang (exercise induced angina): "))
oldpeak = float(input("Enter oldpeak (ST depression induced by exercise relative to rest): "))
slope = int(input("Enter slope (the slope of the peak exercise ST segment): "))
ca = int(input("Enter ca (number of major vessels (0-3) colored by fluoroscopy): "))
thal = int(input("Enter thal (thalassemia): "))

# Create a dictionary to store the input values
input_dict = {
    "age": age,
    "sex": sex,
    'cp_0': [int(cp == 0)],
    'cp_1': [int(cp == 1)],
    'cp_2': [int(cp == 2)],
    'cp_3': [int(cp == 3)],
    'restecg_0': [int(restecg == 0)],
    'restecg_1': [int(restecg == 1)],
    'restecg_2': [int(restecg == 2)],
    "trestbps": trestbps,
    "chol": chol,
    "fbs": fbs,
    "thalach": thalach,
    "exang": exang,
    "oldpeak": oldpeak,
    "slope": slope,
    "ca": ca,
    "thal": thal
}

# Convert the dictionary to a DataFrame
input_df = pd.DataFrame([input_dict])

# Use the get_features_and_target_arrays function to preprocess the input data
x_input= get_features_and_target_array(input_df, cat_cols, numerical_cols, scaler)

# Make predictions using the trained model
y_pred = clf.predict(x_input)

# Display the predicted heart disease status
if y_pred[0] == 0:
    print("Predicted Heart Disease Status: No")
else:
    print("Predicted Heart Disease Status: Yes")
```

```
Enter age: 48
Enter sex (0 for female, 1 for male): 0
Enter cp (chest pain type): 2
Enter trestbps (resting blood pressure): 130
Enter chol (serum cholesterol level): 275
Enter fbs (fasting blood sugar > 120 mg/dl): 0
Enter restecg (resting electrocardiographic results): 1
Enter thalach (maximum heart rate achieved): 139
Enter exang (exercise induced angina): 0
Enter oldpeak (ST depression induced by exercise relative to rest): 0.2
Enter slope (the slope of the peak exercise ST segment): 2
Enter ca (number of major vessels (0-3) colored by fluoroscopy): 0
Enter thal (thalassemia): 2
Predicted Heart Disease Status: Yes
```

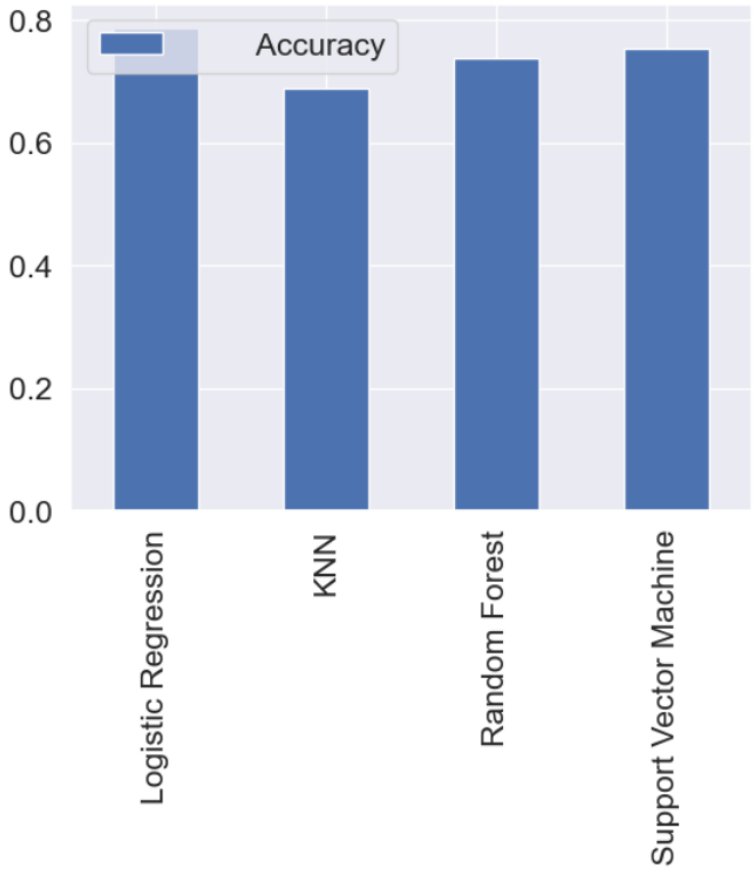
Now we have created a function we takes a user input of attributes and then return whether a person is prone to heart disease or not.(YES-if the person is prone, NO-if the person is not prone).

PERFORMANCE ANALYSIS:

Correlation Matrix: The correlation matrix in machine learning is used for feature selection. It represents dependency between various attributes.



ACCURACY:

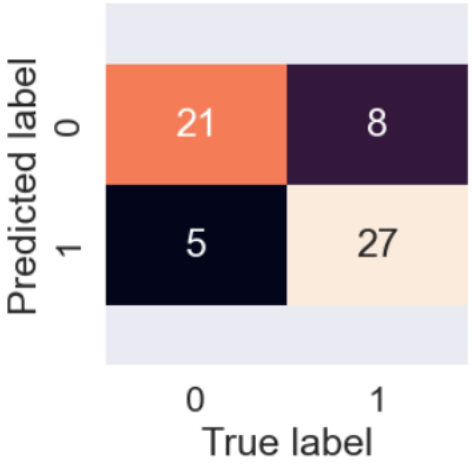


ACCURACY OBTAINED BY DIFFERENT MACHINE LEARNING ALGORITHMS.

THE FOLLOWING ARE THE EXACT VALUES OF THE ACCURACY:

```
{'Logistic Regression': 0.7868852459016393,
 'KNN': 0.6885245901639344,
 'Random Forest': 0.7377049180327869,
 ' Support Vector Machine': 0.7540983606557377}
```

The best accuracy is being obtained using Logistic Regression Model and thus it is used in the final machine learning model.



CONFUSION MATRIX FOR LOGISTIC REGRESSION MODEL.

RESULT:

- The best accuracy is obtained using **logistic regression model**.
  (Accuracy: 78.68885245%)
- The model even returns Yes or No for custom input to detect a disease or not.

CONCLUSION AND FUTURE WORK:

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.

All the four machine learning methods accuracies are compared based on which one prediction model is generated. The best results to be logistic regression model. Further, we can use Deep Learning Algorithms in order to further increase accuracy in future.