NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTION, AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY,

BELGAUM, APPROVED BY AICTE & GOVT.OF KARNATAKA

Introduction to Machine Learning (18CSE751)
LA REPORT
On

COMPARISION OF VARIOUS ML ALGORITHM ON UCI HEART DATASET

Submitted in partial fulfilment of the requirement for the award of Degree of

Bachelor of Engineering

in

Computer Science and Engineering

Submitted by:

KARAN R	1NT18CS067
SAMIKSHA RAJBHUSHAN ULLAL	1NT18CS143

Under the Guidance of
Dr. Vani V.
Professor, Dept. of CS&E, NMIT

Department of Computer Science and Engineering
(Accredited by NBA Tier-1)
2021-22

NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTION, AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM
, APPROVED BY AICTE & GOVT.OF KARNATAKA)

Department of Computer Science and Engineering
(Accredited by NBA Tier-1)

CERTIFICATE

This is to certify that the Report on Nmap is an authentic work carried out by **SAMIKSHA RAJBHUSHAN ULLAL (1NT18CS143) KARAN R (1NT18CS067)** bonafide students of **Nitte Meenakshi Institute of Technology**, Bangalore in partial fulfilment for the award of the degree of ***Bachelor of Engineering*** in COMPUTER SCIENCE AND ENGINEERING of Visvesvaraya Technological University, Belagavi during the academic year ***2021-2022.*** It is certified that all corrections and suggestions indicated during the internal assessment has been incorporated in the report.

Internal Guide	**Signature of HOD**
Dr. Vani V Professor, Dept CSE, NMIT Bangalore	Dr. Saroja Devi.H Professor &Head, Dept CSE, NMIT Bangalore

DECLARATION

We are hereby declare that

(i) The project work is our original work

(i) This Project work has not been submitted for the award of any degree or examination at any other university/College/Institute.

(ii) This Project Work does not contain other persons' data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.

(iv) This Project Work does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:

a) their words have been re-written but the general information attributed to them has been referenced;

b) where their exact words have been used, their writing has been placed inside quotation marks, and referenced.

(v) This Project Work does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the thesis and in the References sections.

NAME	USN	SIGNATURE
SAMIKSHA RAJBHUSHAN ULLAL	1NT18CS143	[illegible]
KARAN R	1NT18CS067	Karan

Date: 16-01-2022

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success. I express my sincere gratitude to our Principal Dr. H. C. Nagaraj, Nitte Meenakshi Institute of Technology for providing facilities.

We wish to thank our HoD, Dr.Sarojadevi H. for the excellent environment created to further educational growth in our college. We also thank him for the invaluable guidance provided which has helped in the creation of a better project.

Thanks to our Departmental Project coordinators. We also thank all our friends, teaching and non-teaching staff at NMIT, Bangalore, for all the direct and indirect help provided in the completion of the project.

NAME	USN	SIGNATURE
SAMIKSHA RAJBHUSHAN ULLAL	1NT18CS143	[illegible]
KARAN R	1NT18CS067	Karan

Date: 16-01-2022

ABSTRACT

Heart disease is one of the most significant causes of mortality in today's world. Prediction of cardiovascular disease is a critical challenge in the area of clinical data analysis, with the current epidemic scenario doctors need a support system for more accurate prediction of heart disease. Diagnosis and prediction of heart related diseases requires more precision, perfection and correctness because a little mistake can cause havoc or death of the person. To deal with this problem there is an essential need of prediction system for awareness about diseases. Machine learning algorithm opens new door opportunities for precise prediction of heart diseases. Machine learning is the branch of Artificial Intelligence, it provides prestigious support in predicting any kind of event which take training from natural events. Our objective is to calculate accuracy of various machine learning algorithms for predicting heart disease, algorithms used are k-nearest neighbor, decision tree, logistic regression, random forest, Naïve bayers and support vector machine (SVM). On basis of accuracy, we conclude the best algorithm for the dataset. We use the existing dataset from the Cleveland database of UCI repository of heart disease patients. The dataset consists of 14 main attributes which is used for performing the analysis. The models are validated using accuracy and confusion matrix.

TABLE OF CONTENT

3. INTRODUCTION

3.1. MOTIVATION

Today, cardiovascular diseases are the leading cause of death worldwide with 17.9 million deaths annually, as per the World Health Organization reports. Heart is one of the most extensive and vital organ of human body so the care of heart is essential. Various unhealthy activities are the reason for the increase in the risk of heart disease like high cholesterol, obesity, increase in triglycerides levels, hypertension, etc. With the rampant increase in the heart stroke rates at juvenile ages, we need to put a system in place to be able to detect the symptoms of a heart stroke at an early stage and thus prevent it. The correct prediction of heart disease can prevent life threats, and incorrect prediction can prove to be fatal at the same time. It is impractical for a common man to frequently undergo costly tests like the ECG and thus there needs to be a system in place which is handy and at the same time reliable, in predicting the chances of a heart disease.

3.2 PROBLEM DOMAIN

Machine learning is one of the most rapidly evolving domains of artificial intelligence. Machine learning algorithms can analyze huge data from various fields, one such important field is the medical field. Its primary focus is to design systems, allow them to learn and make predictions based on the experience. It trains machine learning algorithms using a training dataset to create a model. The model uses the new input data to predict heart disease. Using machine learning, it detects hidden patterns in the input dataset to build models. It makes accurate predictions for new datasets. The dataset is cleaned and missing values are filled. The model uses the new input data to predict heart disease and then tested for accuracy.

The machine learning algorithms that is used to predict the heart disease are Logistic regression, Naïve bayers, Support vector machine (SVM) ,K-NN, decision tree and random forest.

3.3 AIM AND OBJECTIVES

Th aim of the project is to apply various machine learning algorithms -Logistic regression, Navies bayes, SVM, KNN, Decision tree, Radom Forest XGBoost and neural network are on the UCI heart diseases dataset. Preprocessing of the dataset and exploratory data analysis is performed on the dataset. Accuracy is calculated for all the machine learning models and it is validated by confusion matrix. Accuracy sensitivity specificity, positive predictive value and negative predictive value is calculated by all the models . Comparative analysis is performed on all the models to find the best algorithm for the heart dataset.

4.DATA SOURCE AND DATA QUALITY

Source:

Creators:

1. Hungarian Institute of Cardiology. Budapest: Andras Janosi, M.D.
2. University Hospital, Zurich, Switzerland: William Steinbrunn, M.D.
3. University Hospital, Basel, Switzerland: Matthias Pfisterer, M.D.
4. V.A. Medical Center, Long Beach and Cleveland Clinic Foundation: Robert Detrano, M.D., Ph.D.

Further EDA and pre-processing needs to be carried out on the dataset to understand and evaluate it.

Characteristics of the dataset:

Data Set Characteristics	**Multivariate**
Attribute Characteristics	Categorical, Integer, Real
Associated Tasks	Classification
Number of Instances	303
Number of Attributes	75
Missing Values	Yes
Area	Life

This database contains 76 attributes, but all published experiments refer to using a subset of 14 of them. In particular, the Cleveland database is the only one that has been used by ML researchers to this date. The "goal" field refers to the presence of heart disease in the patient. It is integer valued from 0 (no presence) to 4. Experiments with the Cleveland database have concentrated on simply attempting to distinguish presence (values 1,2,3,4) from absence (value 0).
The names and social security numbers of the patients were recently removed from the database, replaced with dummy values.
One file has been "processed", that one containing the Cleveland database. All four unprocessed files also exist in this directory.
To see Test Costs (donated by Peter Turney), please see the folder "Costs".

Attribute Information:

Only 14 attributes used:

S.No	Attribute	Description	Type
1.	Age	Patient’s age (29 to 77)	Numerical
2.	Sex	Gender of patient(male-0 female-1)	Nominal
3.	Cp	Chest pain type	Nominal
4.	Trestbps	Resting blood pressure(in mm Hg on admission to hospital ,values from 94 to 200)	Numerical
5.	Chol	Serum cholesterol (in mg/dl, values from 126 to 564)	Numerical
6.	Fbs	Fasting blood (sugar>120 mg/dl, true-1 false-0)	Nominal
7.	Resting	Resting electrocardiographic result (0 to	Nominal

		1)	
8.	Thali	Maximum heart rate achieved(71 to 202)	Numerical
9.	Exang	Exercise (1-yes 0-no)	Nominal
10.	Oldpeak	ST depression introduced by exercise relative to rest (0 to .2)	Numerical
11.	Slope	The slop of the peak exercise ST segment (0 to 1)	Nominal
12.	Ca	Number of major vessels (0-3)	Numerical
13.	Thal	3-normal	Nominal
14.	Targets	1 or 0	Nominal

5.DATA PRE-PROCESSING AND EXPLORATORY DATA ANALYSIS

First, we check for null values in the dataset, assuring that there are none we then move on to check the correlation between the attributes with respect to the target attribute. The ‘fbs’ - fasting blood sugar is the least correlated and ‘exang’ exercise induced angina is the most correlated.
Next we analyze the categorical and continuous attributes.
Categorical attributes are Sex,cp, fbs, restecg, exang, slope, ca, thal

```
print("Sex -",dataset["sex"].unique())
print("CP -",dataset["cp"].unique())
print("fbs -",dataset["fbs"].unique())
print("restecg -",dataset["restecg"].unique())
print("exang -",dataset["exang"].unique())
print("slope -",dataset["slope"].unique())
print("ca -",dataset["ca"].unique())
print("thal -",dataset["thal"].unique())
```

```
Sex - [1 0]
CP - [3 2 1 0]
fbs - [1 0]
restecg - [0 1 2]
exang - [0 1]
slope - [0 2 1]
ca - [0 2 1 3 4]
thal - [1 2 3 0]
```

```
sns.catplot(x='sex', kind='count',hue = 'target', data = dataset,
palette = 'ch: .28')
sns.catplot(x='cp', kind='count',hue = 'target', data = dataset,
palette = 'ch: .28')
sns.catplot(x='fbs', kind='count',hue = 'target', data = dataset,
palette = 'ch: .28')
sns.catplot(x='restecg', kind='count',hue = 'target', data = dataset,
palette = 'ch: .28')
sns.catplot(x='exang', kind='count',hue = 'target', data = dataset,
palette = 'ch: .28')
sns.catplot(x='slope', kind='count',hue = 'target', data = dataset,
palette = 'ch: .28')
sns.catplot(x='ca', kind='count',hue = 'target', data = dataset,
palette = 'ch: .28')
sns.catplot(x='thal', kind='count',hue = 'target', data = dataset,
palette = 'ch: .28')
```

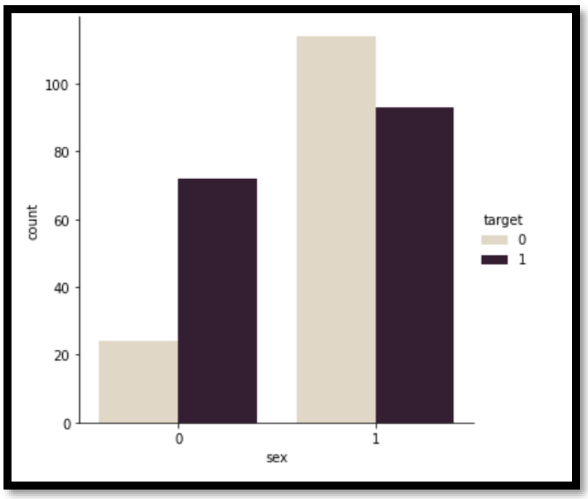


Fig 1: Catplot of sex against the target

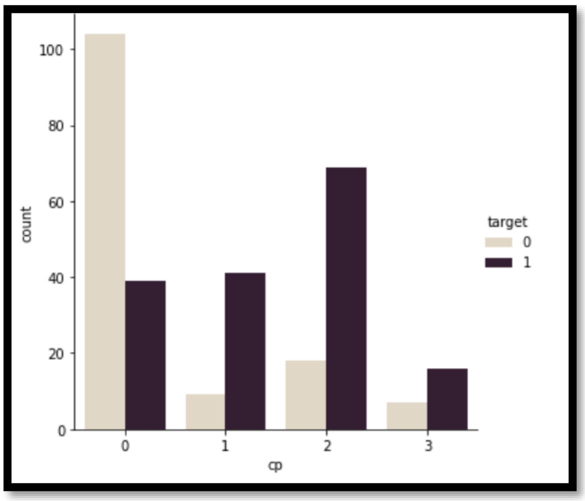


Fig 2: Catplot of cp against the target

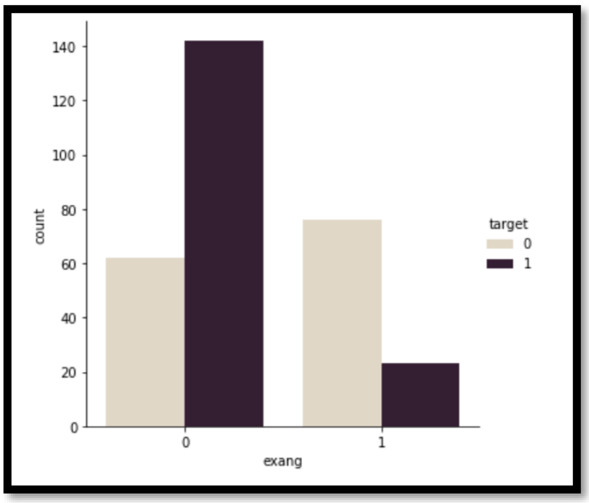


Fig 3 : Catplot of exang against the target

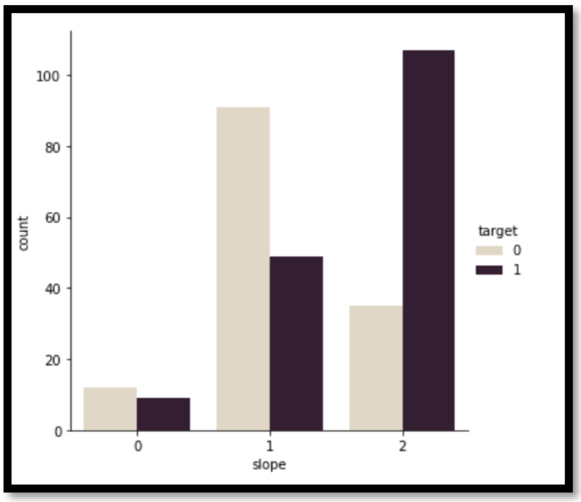


Fig 4 : Catplot of slope against the target

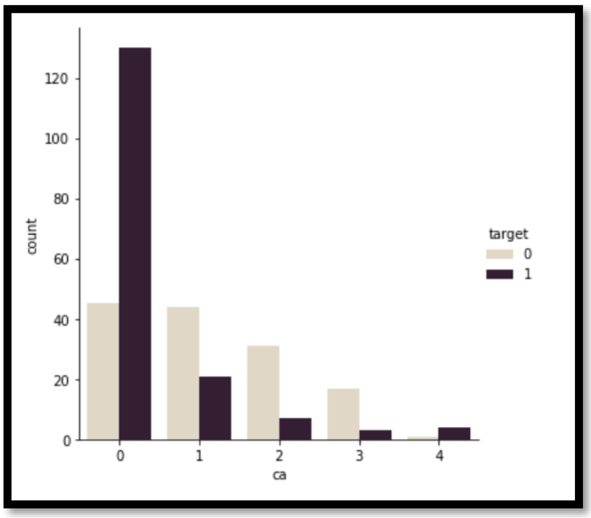


Fig 5 : Catplot of ca against the target

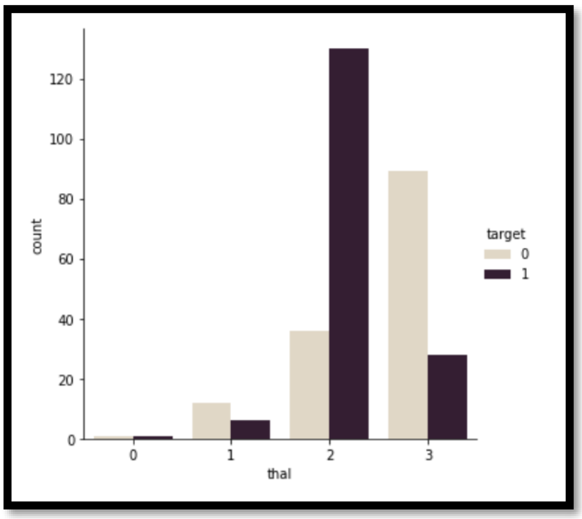
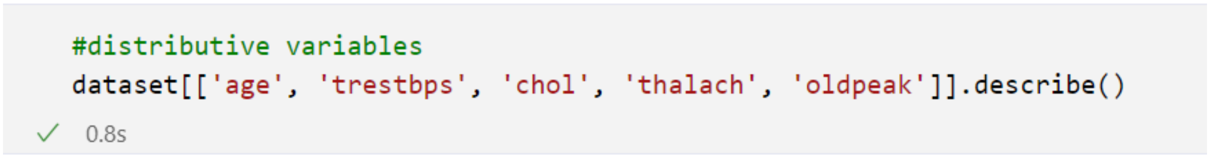


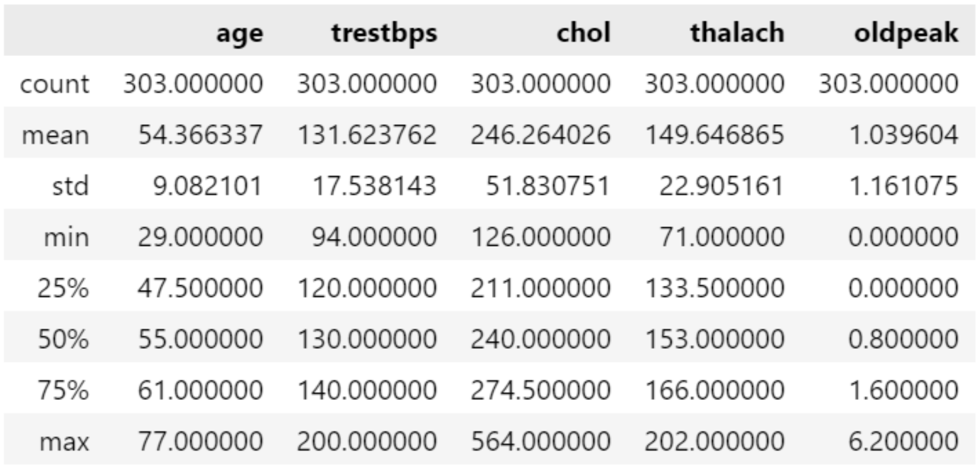
Fig 6: Catplot of thal against the target

Next we analyze the continuous variables like age , trestbps, chol, thalach and oldpeak.



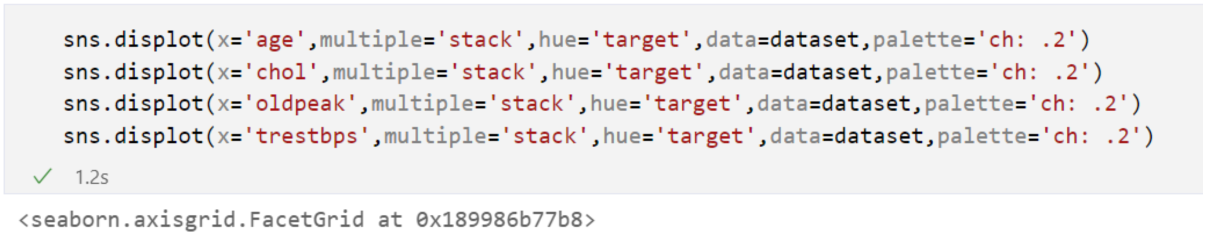
```
#distributive variables
dataset[['age', 'trestbps', 'chol', 'thalach', 'oldpeak']].describe()
```

✓ 0.8s



	age	trestbps	chol	thalach	oldpeak
count	303.000000	303.000000	303.000000	303.000000	303.000000
mean	54.366337	131.623762	246.264026	149.646865	1.039604
std	9.082101	17.538143	51.830751	22.905161	1.161075
min	29.000000	94.000000	126.000000	71.000000	0.000000
25%	47.500000	120.000000	211.000000	133.500000	0.000000
50%	55.000000	130.000000	240.000000	153.000000	0.800000
75%	61.000000	140.000000	274.500000	166.000000	1.600000
max	77.000000	200.000000	564.000000	202.000000	6.200000

Fig7 : description of Continuous attributes



```
sns.displot(x='age',multiple='stack',hue='target',data=dataset,palette='ch: .2')
sns.displot(x='chol',multiple='stack',hue='target',data=dataset,palette='ch: .2')
sns.displot(x='oldpeak',multiple='stack',hue='target',data=dataset,palette='ch: .2')
sns.displot(x='trestbps',multiple='stack',hue='target',data=dataset,palette='ch: .2')
```

✓ 1.2s

```
<seaborn.axisgrid.FacetGrid at 0x189986b77b8>
```

Fig 8: Distplot for categotical attributes

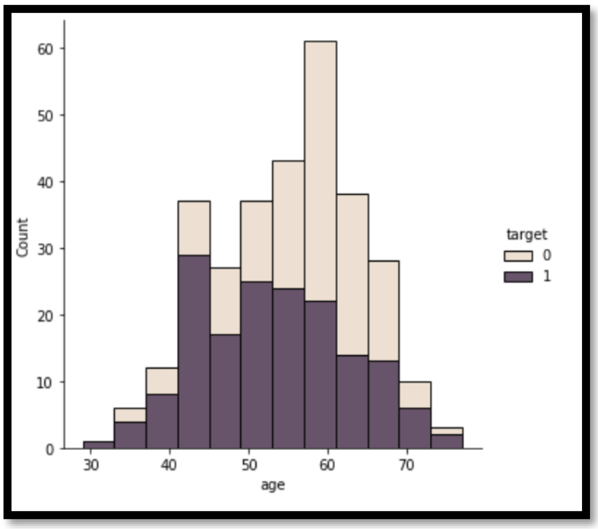


Fig 9: Distplot of age against the target

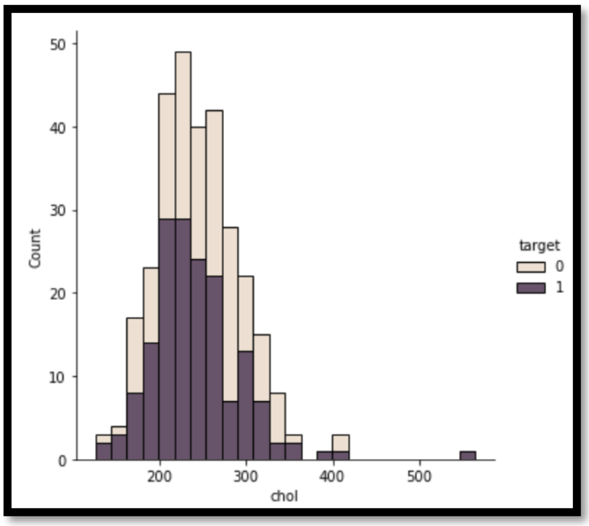


Fig 10: Distplot of chol against the target

6.MACHINE LEARNING METHODS

1.Logistic Regression

Logistic Regression is a Machine Learning algorithm which is used for the classification problems, it is a predictive analysis algorithm and based on the concept of probability.
Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Some of the examples of classification problems are Email spam detection, Online transactions Fraud Detection, Tumor Malignant or Benign. Logistic regression transforms its output using the logistic sigmoid function to return a probability value.

Implementation-

```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
lr.fit(X_train,Y_train)
Y_pred_lr = lr.predict(X_test)
Y_pred_lr.shape
score_lr = round(accuracy_score(Y_pred_lr,Y_test)*100,2)
print("The  accuracy  score  achieved  using  Logistic  Regression  is:
"+str(score_lr)+" %")
```

Output-

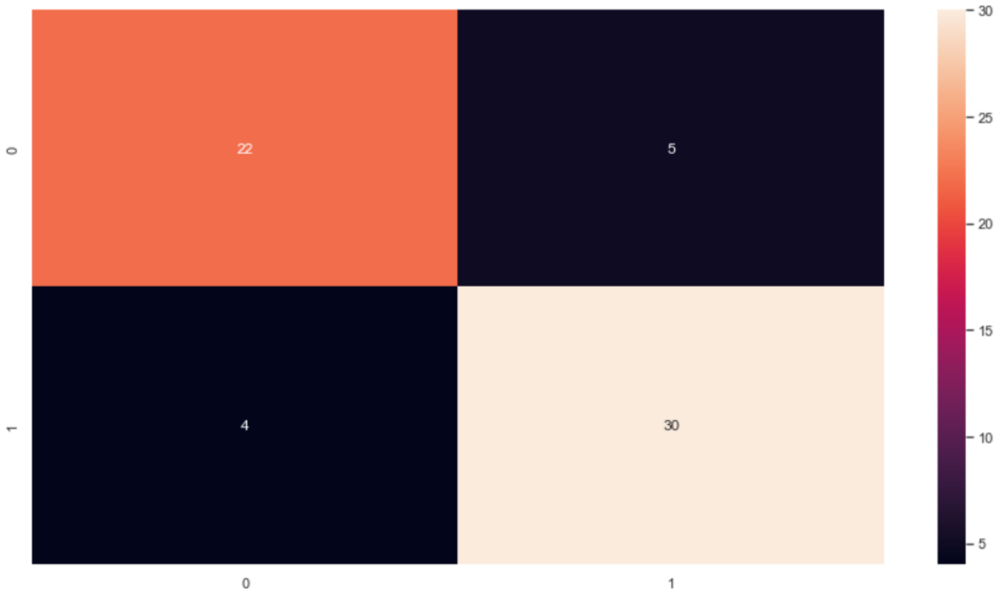


Fig 11.Confusion matrix of Logistic Regression

Accuracy: 85.24590163934425 %
Sensitivity: 81.48148148148148 %
Specificity: 88.23529411764706 %
Positive Predictive Value: 84.61538461538461 %
Negative Predictive Value: 20.0 %

2. Naïve Bayes Classifier Algorithm

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems. It is mainly used in text classification that includes a high-dimensional training dataset. Naïve 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 based on the probability of an object. Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

Implementation-

```
from sklearn.naive_bayes import GaussianNB
nb = GaussianNB()
nb.fit(X_train,Y_train)
Y_pred_nb = nb.predict(X_test)
Y_pred_nb.shape
score_nb = round(accuracy_score(Y_pred_nb,Y_test)*100,2)
print("The   accuracy   score   achieved   using   Naive   Bayes
is:"+str(score_nb)+" %")
```

Output –

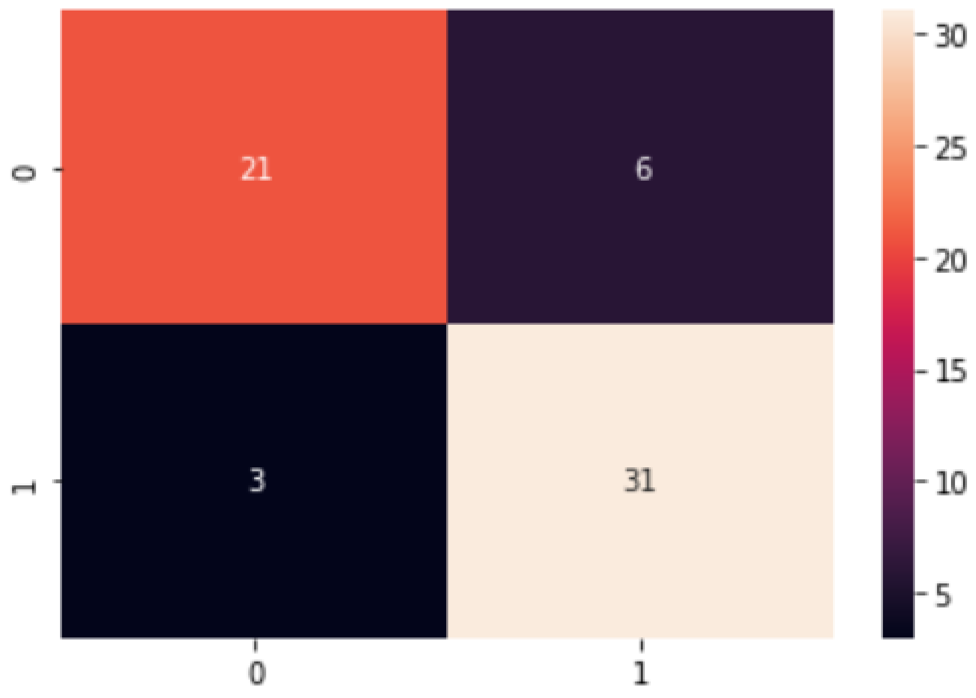


Fig 12.Confusion matrix of Naïve Bayes

```
Accuracy: 85.24590163934425 %
Sensitivity: 77.77777777777779 %
Specificity: 91.17647058823529 %
Positive predictive value: 87.5 %
Negative predictive value: 16.666666666666664 %
```

3. Support Vector Machine

Support Vector Machine (SVM) is a Supervised Learning algorithm, which is used for Classification as well as Regression problems. Primarily, it is used for Classification problems in Machine Learning. 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. SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.

Implementation –

```
from sklearn import svm
sv = svm.SVC(kernel='linear')
sv.fit(X_train, Y_train)
Y_pred_svm = sv.predict(X_test)
Y_pred_svm.shape
score_svm = round(accuracy_score(Y_pred_svm,Y_test)*100,2)
print("The accuracy score achieved using Linear SVM is:
"+str(score_svm)+" %")
```

Output-

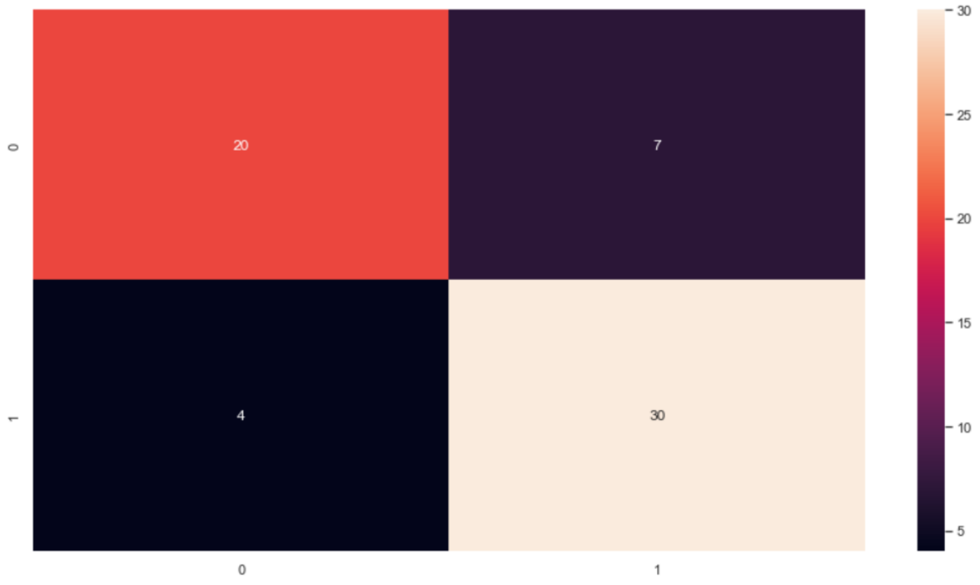


Fig 13.Confusion matrix of SVM

Accuracy: 81.9672131147541 %
Sensitivity: 74.07407407407408 %
Specificity: 88.23529411764706 %
Positive predictive value: 83.33333333333334 %
Negative predictive value: 14.285714285714285 %

4. K-Nearest Neighbour Algorithm

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique. K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories. K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm. K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems. K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data. It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset. KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

Implementation-

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=7)
knn.fit(X_train,Y_train)
Y_pred_knn=knn.predict(X_test)
Y_pred_knn.shape
score_knn = round(accuracy_score(Y_pred_knn,Y_test)*100,2)

print("The accuracy score achieved using KNN is: "+str(score_knn)+"
%")
```

Output –

The accuracy score achieved using KNN is: 67.21 %

To improve the accuracy of KNN algorithm lets standardize the dataset.

```
#standardize data The idea is to reclean original value to have equal
impact
from sklearn.preprocessing import StandardScaler
sc = StandardScaler().fit(X_train)
x_train1 = sc.transform(X_train)
x_test1 = sc.transform(X_test)

print("X TRAIN ", x_train1.shape)
print("X TEST ", x_test1.shape)
```

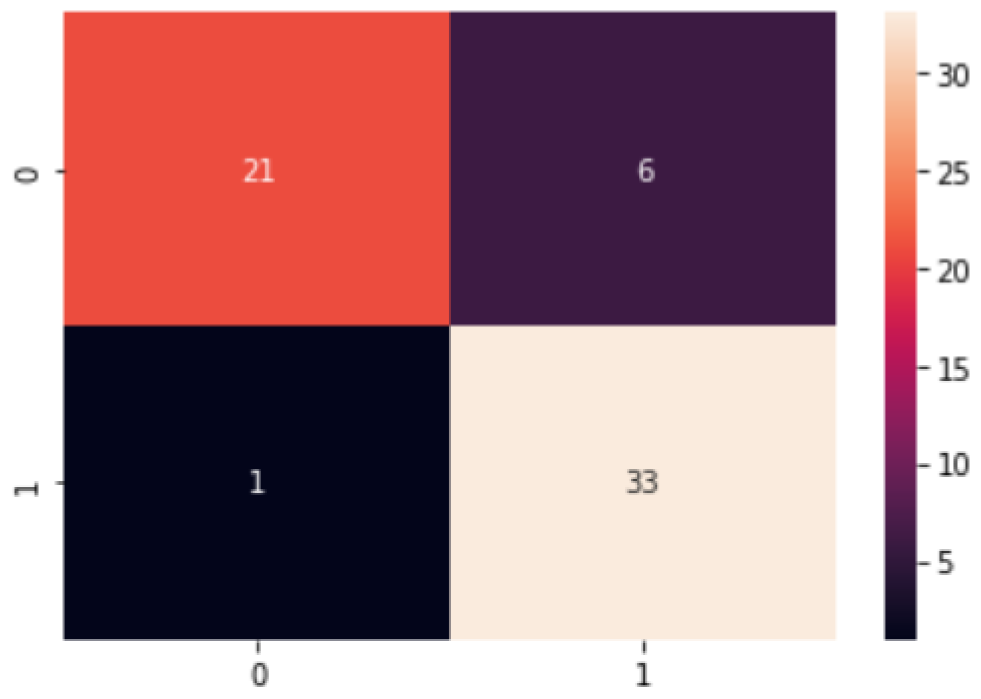


Fig 14.Confusion matrix of KNN

Accuracy: 88.52459016393442 %
Sensitivity: 77.77777777777779 %
Specificity: 97.05882352941177 %
positive predictive value: 95.45454545454545 %
negative predictive value: 16.666666666666664 %

5. Decision Tree

Decision Tree is a Supervised learning technique that can be used for both classification and Regression 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. Tree models where the target variable can take a discrete set of values are called classification trees. Decision trees where the target variable can take continuous values are called regression trees.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.

Implementation-

```
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)

print(Y_pred_dt.shape)

score_dt = round(accuracy_score(Y_pred_dt,Y_test)*100,2)

print("The accuracy score achieved using Decision Tree is: 
"+str(score_dt)+" %")
```

Output-

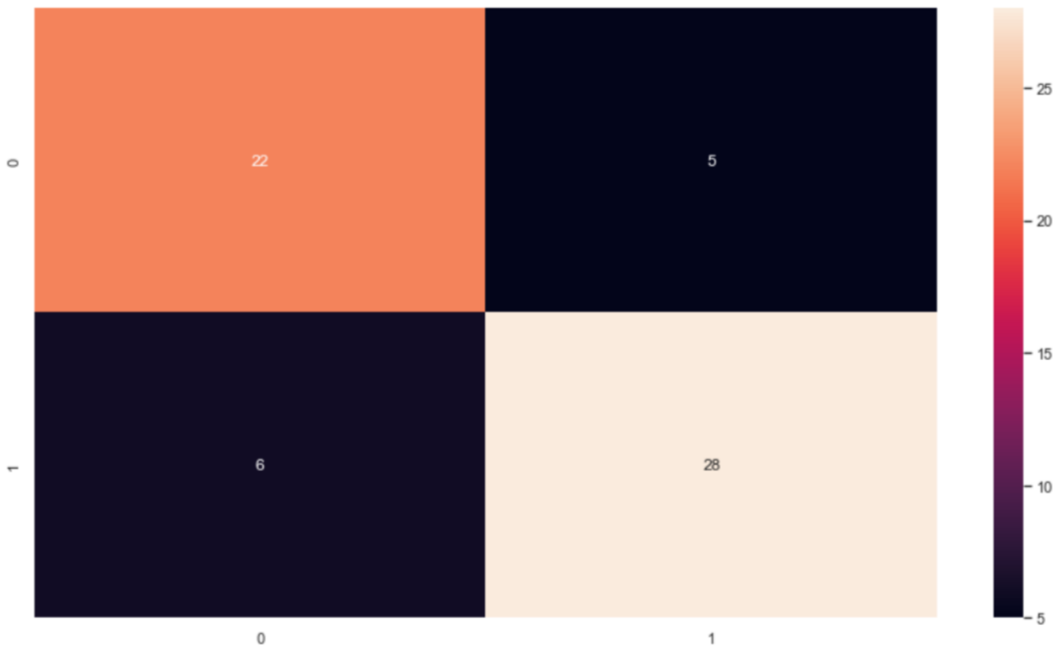


Fig 15.Confusion matrix of Decision tree

Accuracy: 81.9672131147541 %
Sensitivity: 81.48148148148148 %
Specificity: 82.35294117647058 %
positive predictive value: 78.57142857142857 %
negative predictive value: 20.0 %

6.Random Forest Algorithm

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.**

Implementation –

```
from sklearn.ensemble import RandomForestClassifier
max_accuracy = 0


for x in range(2000):
    rf = RandomForestClassifier(random_state=x)
    rf.fit(x_train1,Y_train)
    Y_pred_rf = rf.predict(x_test1)
    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_train1,Y_train)
Y_pred_rf = rf.predict(x_test1)

Y_pred_rf.shape

score_rf = round(accuracy_score(Y_pred_rf,Y_test)*100,2)

print("The accuracy score achieved using Random Forest is: 
"+str(score_rf)+" %")
```

Output –

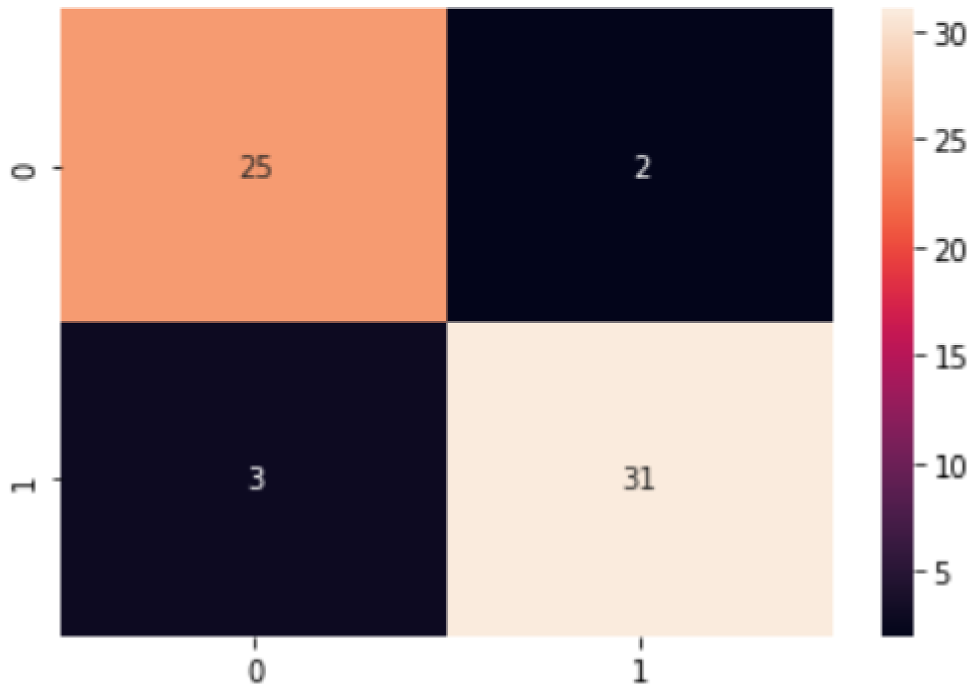


Fig 16.Confusion matrix of Random Forest.

Accuracy: 91.80327868852459 %
Sensitivity: 92.5925925925926 %
Specificity: 91.17647058823529 %
positive predictive value: 89.28571428571429 %
negative predictive value: 50.0 %

7.XGBOOST

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. XGBoost stands for e**X**treme **G**radient **B**oosting. XGBoost and Gradient Boosting Machines (GBMs) are both ensemble tree methods that apply the principle of boosting weak learners (CARTs generally) using the gradient descent architecture. However, XGBoost improves upon the base GBM framework through systems optimization and algorithmic enhancements. XGBoost approaches the process of sequential tree building using parallelized implementation. XGBoost uses '**max_depth'** parameter as specified instead of criterion first, and starts pruning trees backward. This 'depth-first' approach improves computational performance significantly. This algorithm has been designed to make efficient use of hardware resources.

Implementation-

```
import xgboost as xgb

xgb_model = xgb.XGBClassifier(objective="binary:logistic",
random_state=42, eval_metric='mlogloss')
xgb_model.fit(x_train1, Y_train)

Y_pred_xgb = xgb_model.predict(x_test1)

Y_pred_xgb.shape

score_xgb = round(accuracy_score(Y_pred_xgb,Y_test)*100,2)

print("The accuracy score achieved using XGBoost is:
"+str(score_xgb)+" %")
```

Output-

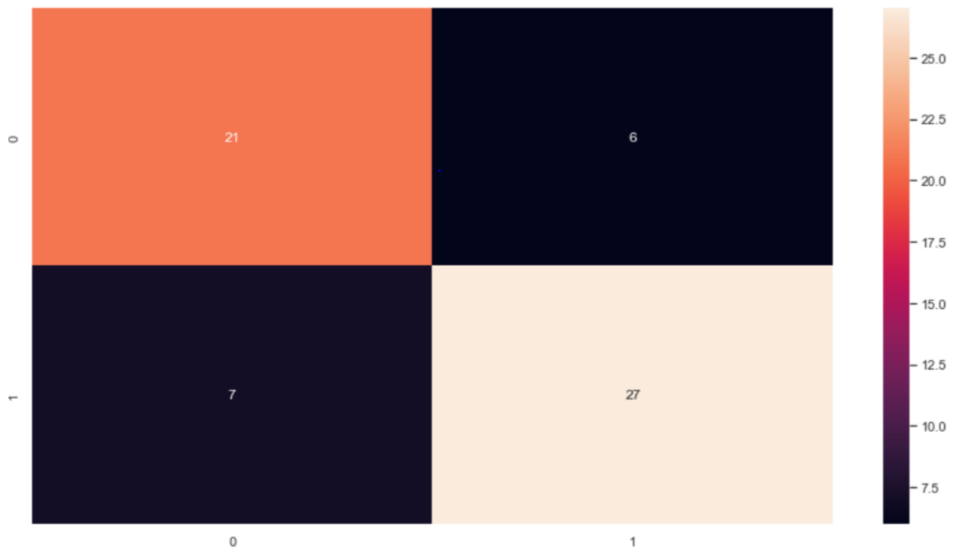


Fig 17.Confusion matrix of XGBoost

```
Accuracy: 78.68852459016394 %
Sensitivity: 77.77777777777779 %
Specificity: 79.41176470588235 %
positive predictive value: 75.0 %
negative predictive value: 16.666666666666664 %
```

8.Neural Network

Implementation –

```
from keras.models import Sequential
from keras.layers import Dense
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'])
model.fit(x_train1,Y_train,epochs=300)
Y_pred_nn = model.predict(x_test1)
Y_pred_nn.shape
rounded = [round(x[0]) for x in Y_pred_nn]
Y_pred_nn = rounded
score_nn = round(accuracy_score(Y_pred_nn,Y_test)*100,2)
print("The accuracy score achieved using Neural Network is: "+str(score_nn)+" %")
#83.61 on standardizing
#Note: Accuracy of 85% can be achieved on the test set, by setting epochs=2000, and
number of nodes = 11.
```

The accuracy score achieved using Neural Network is: 86.89 %

Output –

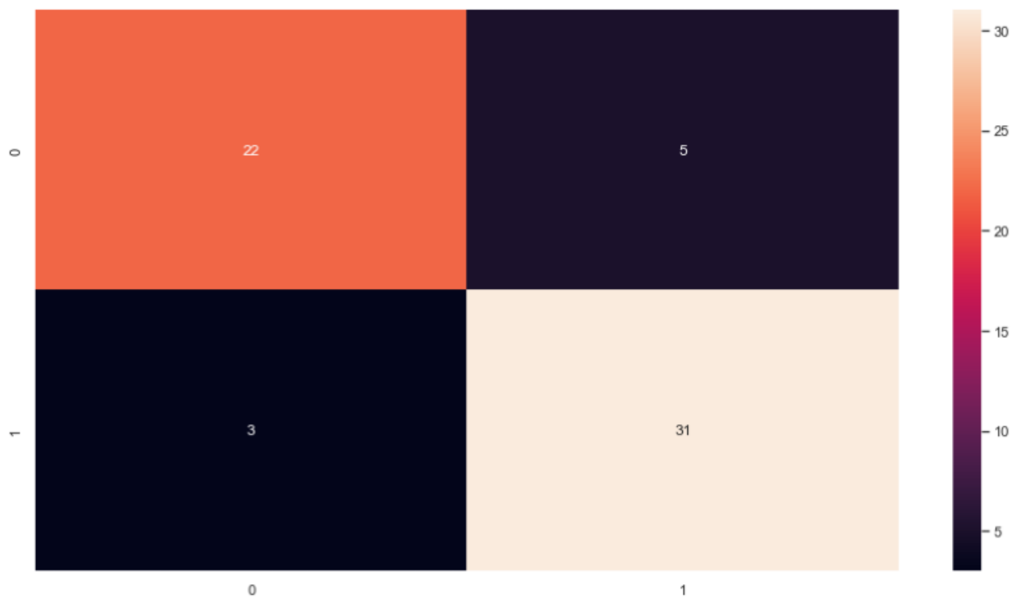


Fig 18.Confusion matrix of Neural Network

Accuracy: 86.88524590163934 %
Sensitivity: 81.48148148148148 %
Specificity: 91.17647058823529 %
positive predictive value: 88.0 %
negative predictive value: 20.0 %

7.RESULT

The accuracy scores of the different machine learning algorithms are computed and compared below

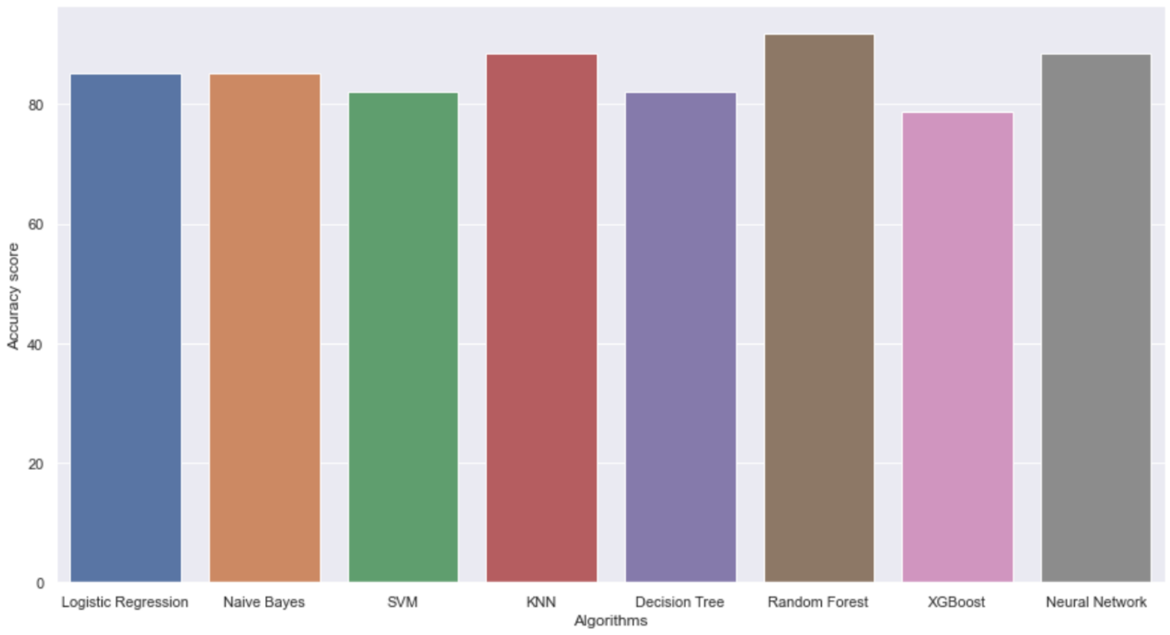


Fig 19 : Comparison of Accuracies

The accuracy score achieved using Logistic Regression is: 85.25 %
The accuracy score achieved using Naive Bayes is: 85.25 %
The accuracy score achieved using Support Vector Machine is: 81.97 %
The accuracy score achieved using K-Nearest Neighbors is: 88.52 %
The accuracy score achieved using Decision Tree is: 81.97 %
The accuracy score achieved using Random Forest is: 91.8 %
The accuracy score achieved using XGBoost is: 78.69 %
The accuracy score achieved using Neural Network is: 88.52 %

8.CONCLUSION & FUTURE SCOPE

The accuracies of Random Forest, k-NN and neural network was seen to be the highest. The accuracy of K-NN was improved after normalizing the dataset. Although the accuracy of Random forest is very high, its training time was seen to be quite significant. 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.

For the Future Scope more machine learning approach will be used for best analysis of the heart diseases and for earlier prediction of diseases so that the rate of the death cases can be minimized by the awareness about the diseases.

9.LESSON LEARNT

- Exploratory data analysis of categorical and continuous variables
- Implementation of various machine learning algorithm – Logistic Regression,Naïve Bayes ,SVM,KNN, Decision Tree, Random Forest Classifier,XGBoost and Neural Network.
- Improving the efficiency of K-NN and Random forest algorithm
- Construction of Confusion matrix to validate the model
- Comparative Analysis of all the machine leaning models

10.REFERENCES

A. Singh and R. Kumar, "Heart Disease Prediction Using Machine Learning Algorithms," *2020 International Conference on Electrical and Electronics Engineering (ICE3)*, 2020, pp. 452-457, doi: 10.1109/ICE348803.2020.9122958.

S. Mohan, C. Thirumalai and G. Srivastava, "Effective Heart Disease Prediction Using Hybrid Machine Learning Techniques," in IEEE Access, vol. 7, pp. 81542-81554, 2019, doi: 10.1109/ACCESS.2019.2923707.

S. K. J. and G. S., "Prediction of Heart Disease Using Machine Learning Algorithms.," 2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT), 2019, pp. 1-5, doi: 10.1109/ICIICT1.2019.8741465.

Sharma, Himanshu, and M. A. Rizvi. "Prediction of heart disease using machine learning algorithms: A survey." *International Journal on Recent and Innovation Trends in Computing and Communication* 5.8 (2017): 99-104.

A. Gavhane, G. Kokkula, I. Pandya and K. Devadkar, "Prediction of Heart Disease Using Machine Learning," 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), 2018, pp. 1275-1278, doi: 10.1109/ICECA.2018.8474922.

M. Nikhil Kumar, K. V. S. Koushik, K. Deepak, “Prediction of Heart Diseases Using Data Mining and Machine Learning Algorithms and Tools” International Journal of Scientific Research in Computer Science, Engineering and Information Technology ,IJSRCSEIT 2019.

Amandeep Kaur and Jyoti Arora,“Heart Diseases Prediction using Data Mining Techniques: A survey” International Journal of Advanced Research in Computer Science , IJARCS 2015-2019.

Pahulpreet Singh Kohli and Shriya Arora, “Application of Machine Learning in Diseases Prediction”, 4th International Conference on Computing Communication And Automation(ICCCA), 2018.

M. Akhil, B. L. Deekshatulu, and P. Chandra, “Classification of Heart Disease Using K- Nearest Neighbor and Genetic Algorithm,” ProcediaTechnol., vol. 10, pp. 85–94, 2013.

Hazra, A., Mandal, S., Gupta, A. and Mukherjee, “ A Heart Disease Diagnosis and Prediction Using Machine Learning and Data Mining Techniques: A Review” Advances in Computational Sciences and Technology , 2017.

11.APPENDIX

Python Code
https://github.com/karanaithal01/Comparison-of-Various-Machine-Learning-Algorithms-on-UCI-Heart-Dataset/blob/main/Comparison_of_ML_models_on_Heart_UCI_dataset.ipynb

Dataset -CSV file
https://github.com/karanaithal01/Comparison-of-Various-Machine-Learning-Algorithms-on-UCI-Heart-Dataset/blob/main/heart.csv