RAYAT SHIKSHAN SANSTHA'S,

SADGURU GADAGE MAHARAJ COLLEGE, KARAD.

(AN AUTONOMOUS COLLEGE)

DEPARTMENT OF STATISTICS

Project report on

"Heart Failure Prediction Using Machine Learning Techniques"

Submitted by,

Miss. Ghorpade Nikita Balasaheb.

M.Sc. II (Statistics)

Under the Guidance of

Miss. Patil R. D.

(2020-2021)

Teacher in-charge P.G. Coordinator Head of department

CERTIFICATE

This is to certify that the Project report entitled **"Heart Failure Prediction using Machine Learning Techniques"** being submitted by "Miss. Ghorpade Nikita Balasaheb" as partial fulfillment for the M.Sc. in Statistics of Sadguru Gadage Maharaj College, Karad record of bonafide work carried out by them under supervision and guidance.

To the best of our knowledge and belief, the matter presented in this project report is original and has not been submitted elsewhere for any other purpose.

Place: Karad

Date: 02-08-2021

(Head, Department of Statistics) (P.G. Co-ordinator)

ACKNOWLEDGEMENT

The satisfaction that accompanies that the successful completion of any task would be incomplete without the mention of people whose ceaseless co-operation made it possible whose constant guidance and encouragement crown all efforts and success.

I am very greatful to my project supervisor **"Prof. Patil R. D."** faculty of "Sadguru Gadage Maharaj College, Karad", **"Smt. Davari S. S."** (Head Department of Statistics) and **"Dr. Mrs. Patil S. P."** (P.G. Coordinator) for the guidance, inspiration and constructive suggestions that helpful me in the preparation of this project.

I won't forget to also mention my all teachers and friends for their wonderful and skillful guidance in assisting me me with the necessary support to ensure that my project is a success.

Yours Sincerely,

Miss. Nikita Balasaheb Ghorpade.

M.Sc. – II

Department of Statistics

Index

Statistical Tools :

- ✓ Exploratory Data Analysis:
  Bar charts, Pie Charts, Count Plot, Cat plot , Scatter Plot, etc.

- ✓ Machine Learning Algorithms (Data Mining Classifier):
  Logistic Regression, Random Forest, Naïve Bayes, KNN, SVM, Decesion Tree

Statistical Software:

- ✓ Python (Jupyter Notebook)

ABSTRACT

In this project, we analyze a dataset of 299 patients with heart failure.

Heart failure (HF) occurs when the heart cannot pump enough blood to meet the needs of the body. Available electronic medical records of patients quantify symptoms, body features, and clinical laboratory test values, which can be used to perform biostatistics analysis aimed at highlighting patterns and correlations otherwise undetectable by medical doctors. Machine learning, in particular, can predict patients' survival from their data and can individuate the most important features among those included in their medical records.

Machine learning applied to medical records, in particular, can be an effective tool both to predict the survival of each patient having heart failure symptoms. The dataset contains 13 features, which report clinical, body. Some features are binary: anaemia, high blood pressure, diabetes, sex, and smoking. Regarding the features, the creatinine phosphokinase (CPK) states the level of the CPK enzyme in blood. When a muscle tissue gets damaged, CPK flows into the blood. Therefore, high levels of CPK in the blood of a patient might indicate a heart failure or injury. The ejection fraction states the percentage of how much blood the left ventricle pumps out with each contraction.

The main objective of this project is to overcome the limitations and to design a robust system which works efficiently and will able to predict the possibility of heart failure accurately. This work is implemented using many algorithms such as SVM, Naive Bayes, Random Forest, Logistic Regression, Decision Tree and KNN. It is found that KNN gave the best result with accuracy up to 87.7%. A comparative statement of all the algorithms also presented in the implementation part of this study.

.

INTRODUCTION

The prevention of disease and death due to heart failure needs to be made a global health priority. Despite the increasingly large numbers of people living with and dying from heart failure, awareness of the disease is low among the public, politician and even some healthcare professionals. Although there is no cure for heart failure, many cases are preventable and most patients can be treated effectively to improve quality of life and survival.

Heart disease is one of the major cause of mortality in the world today. Prediction of cardiovascular disease is a critical challenge in the field of clinical data analysis. With the advanced development in machine learning (ML), artificial intelligence (AI) and data science has been shown to be effective in assisting in decision making and predictions from the large quantity of data produced by the healthcare industry. ML approaches has brought lot of improvements and broadens the study in medical field which recognizes patterns in the human body by using various algorithms and correlation techniques. One such reality is coronary heart disease, various studies gives impression into predicting heart disease with ML techniques. Initially ML was used to find degree of heart failure, but also used to identify significant features that affects the heart disease by using correlation techniques. There are many features/factors that lead to heart disease like age, blood pressure, sodium creatinine, ejection fraction etc. In this paper we propose a method to finding important features by applying machine learning techniques. The work is to design and develop prediction of heart disease by feature ranking machine learning. Hence ML has huge impact in saving lives and helping the doctors, widening the scope of research in actionable insights, drive complex decisions and to create innovative products for businesses to achieve key goals.

OBJECTIVE

- To study the impact of all factors or variable mentioned in the dataset on the target variable 'Death-Event'.
- To build the classification models for predicting the Death-Event that means patient is survived or not.
- To compare the performance of classifiers for selecting more accurate model for prediction.

METHODOLOGY

MACHINE LEARNING IN HEALTH CARE:

Machine learning in health care has achieved huge positive out-comes, which also save many lives. The usage of machine learning in Health care systems has increased in past years The purpose of data mining, is to identify useful and understandable patterns by analysing large sets of data. These data patterns will help to predict information and then determine what to do about them. In the healthcare industry specifically, data mining can decrease cost of Tests, And It also increases efficiency and improves patient quality of life, and perhaps most importantly, save the lives of more patients. Data mining has potential to help to prevent in epidemics, diseases and to identify/ predict. This can also cut-own the costs and makes the out-come more accurate.

PROPOSED WORK:

This work used Python programming for this project, as it is a high level programming language and it has vast libraries and Python automates tasks and makes it efficient. Firstly, we need to install Python then we need to import some libraries, they are:

1 Numpy: Numpy is used for multi-dimensional arrays, It does element to element operations and it also has different methods for processing arrays.

2. Panda: Pandas is one of the highly used python library, it provides high performance. It manipulates data and it makes data analysis fast and easy.

3. Sklearn: It is most useful library, This library contains lot of efficient tools, It is used to build models like statistical modeling including classification, regression, clustering. After loading required packages, we divide dataset as training and testing as follows, here 70 % of dataset is taken as training and remaining 30 % as to perform test.

VARIABLE DISCRIPTION

1. Age: Numerical value

2. Sex: Men=1, Women=0

3. Anaemia: Decrease of red blood cells or hemoglobin. (Yes=1)

4. Creatinine Phosphokinase: An enzyme called creatine phosphokinase (CPK) is important for muscle function. The CPK isoenzymes test is a way to measure the levels of this enzyme in your bloodstream. This enzyme can be broken into three parts:- 1. CPK 1 : mainly found in our brain and lungs. The normal range is 10-120 mcg/L.

5. Diabetes: If the patient has diabetes. (Boolean=1)

6.Ejection Fraction: Ejection fraction is a measurement of the percentage of blood leaving your heart each time it squeezes. A normal ejection fraction is more than 55%. This means that 55% of the total blood in the left ventricle is pumped out with each heartbeat.

7. High Blood Pressure: If the patient has hypertension. (Boolean=1)

8. Platelets: Liver stimulates platelets production. It reduce blood flow the injury site. Platelets become sticky when they come in contact with a damage blood vessel and release serotonin.

Normal range of platelets : 150 to 400* 10^9 per litre & 150,000-450,000 per microliter.

9. Serum creatinine: Creatinine is a waste material which is found in muscle. Creatinine test use to check kidney function properly or not that means kidney filter properly or not.

Normal range is for adult male is 0.6 to 1.2 mg/dL and for adult female is 0.5 to 1.1 mg/dL

10. Serum Sodium: It is important for nerve and muscle function. Too much sodium can raise your BP and lack of sodium can cause symptoms like nausea, vomiting, exhaustion, dizziness.

Normal level is 135-145 mEq/L.

11. Smoking: If patients smoke. (Boolean = 1)

12. Time: follow up period

13. Death Event: If the patient deceased during the follow up period. (Boolean=1)

DISCRIPTIVE STATISTICS

- **Dimensions of dataset:**
  There is 299 rows and 13 columns in this dataset, that means overall size of the dataset is 299.
- **To check missing values and treat using suitable technique.**

```
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   age                       299 non-null    float64
 1   anaemia                   299 non-null    int64
 2   creatinine_phosphokinase  299 non-null    int64
 3   diabetes                  299 non-null    int64
 4   ejection_fraction         299 non-null    int64
 5   high_blood_pressure       299 non-null    int64
 6   platelets                 299 non-null    float64
 7   serum_creatinine          299 non-null    float64
 8   serum_sodium              299 non-null    int64
 9   sex                       299 non-null    int64
 10  smoking                   299 non-null    int64
 11  time                      299 non-null    int64
 12  DEATH_EVENT               299 non-null    int64
```

Interpretation: We see that there is no any missing or null value in our dataset.

- **Discription of data:**

	age	creatinine_phosphokinase	ejection_fraction	platelets	serum_creatinine	serum_sodium
count	299	299	299	299	299	299
mean	60.833893	581.839465	38.083612	263358.0293	1.39388	136.625418
std	11.894809	970.287881	11.834841	97804.23687	1.03451	4.412477
min	40	23	14	25100	0.5	113
25%	51	116.5	30	212500	0.9	134
50%	60	250	38	262000	1.1	137
75%	70	582	45	303500	1.4	140
max	95	7861	80	850000	9.4	148

Interpretation:

Age: The age range of patients in data is between 40-70 years. Average age is 60 that means the data Containing more patient is of age 60. And 25% of data having age patients less than 51 years, 50% of data having age of patients less than 60 and 75% of data having age less than 70.

Creatinine phosphokinase (CPK): Here we observe that the average level of CPK is 581.83 which is not normal. The range of level in the dataset is 23 to 7861. That means very few patients having CPK level is normal.

Ejection fraction: We observe that the average value of ejection fraction is 38.08% which is very low. We see that the 75% of data is having an ejection fraction is very low and only 25% of data having ejection fraction is normal. The max value of ejection fraction is 80.

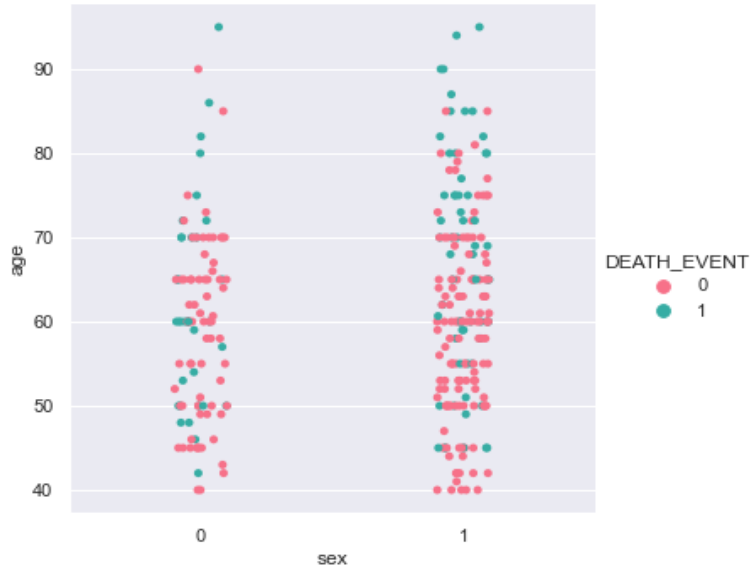
Platelets: The average count of platelets is 263358.02 which is normal. We observe that 75% patients having normal range of platelets.

Serum Creatinine: The average value of this factor is 1.39 which is not normal for men as well as women. Approximate 50% patients having normal range of platelets. 25% patients having more than 1.4 serum creatinine level which is high.

Serum Sodium: Average level of serum sodium is 136.62 which is a normal. We observe that the approximate very few patient's serum sodium level abnormal low and abnormal high.

EXPLORATORY DATA ANALYSIS

1) Cat plot of evaluating death event in age and sex



Interpretation:

From above cat plot we see that the data is distributed that means death event is distributed with sex and age.

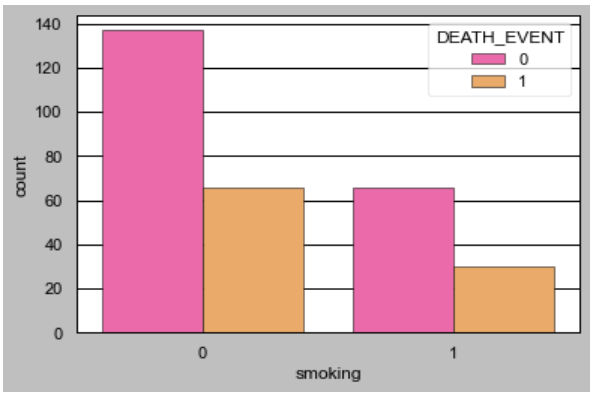
We know that 0 is for patient is survived and 1 is for patient is not survived.

We observe that in both male and female less number of patients are not survived. and also we conclude that, the patients having age 80 or above 80 are not survive that means the have less chances to survive.

2) Count plot for analysing the given data with smoker and death event

Cross table:

DEATH_EVENT	0	1
smoking		
0	137	66
1	66	30



Interpretation:

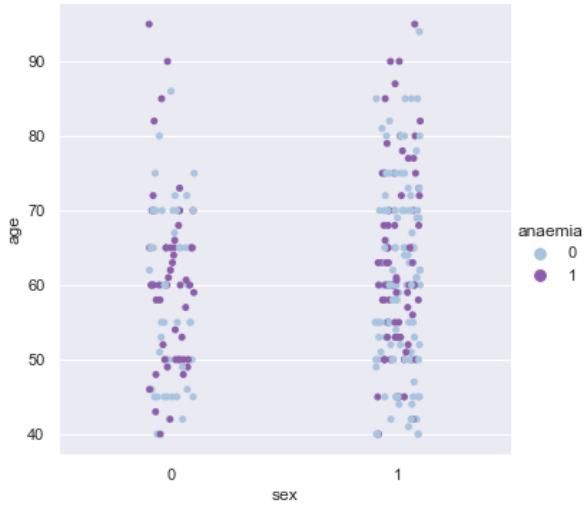
From above plot and cross table, we notice that out of 137 non smoker 66 patients are not survive and out of 66 smoker patient there are 30 patients are not survived. Hence we conclude that the there is no relation between smoker and heart failure.

3) Distribution of smoker patients with gender and age:

Interpretation:

From above plot we see that the very few females are smoker but there are many more male patients are smoker. And we notice that the there is a no age limit for having habit of smoking.

4) Analysis of age and sex wise anaemia



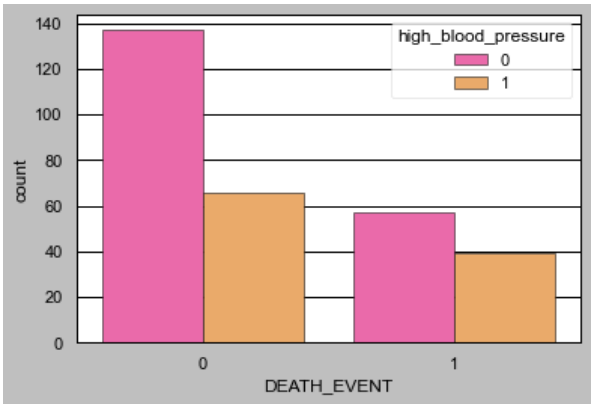
Interpretation:

From above plot we conclude that in female the age between 50 to 70yrs patients having anaemia and in male, the age group 50 to 78yrs patients having anaemia, that means that years age patients having lack of haemoglobin. And we also conclude that the having anaemia is does not depend on sex. Either male or female.

5) The analysis of survival data with high blood pressure (BP)

Cross table:

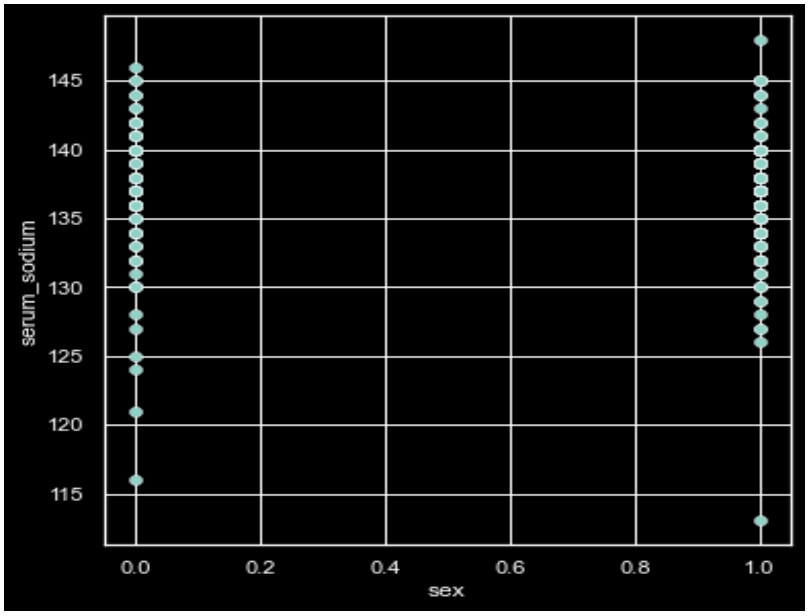
DEATH_EVENT	0	1
high_blood_pressure		
0	137	57
1	66	39



Interpretation:

From above cross table and count plot we notice that the out of 137 no BP patients there are only 57 patients are not survived and out of 66 patients having BP 39 patients are not survived. And hence we conclude that there is no effect of high blood pressure on death event.

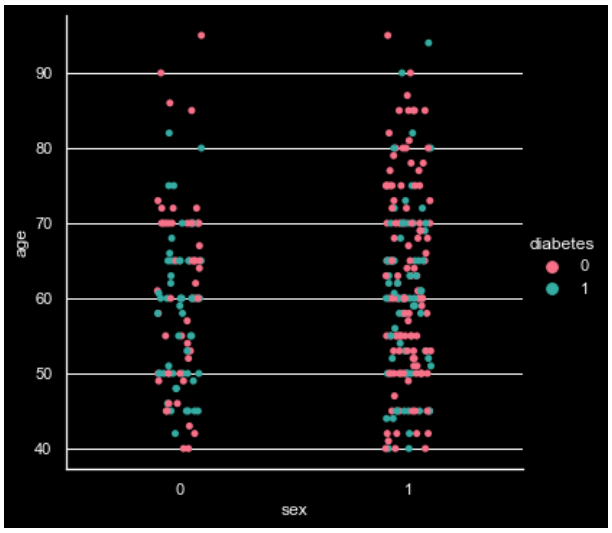
6) Scatter plot of Serum Sodium level



Interpretation:

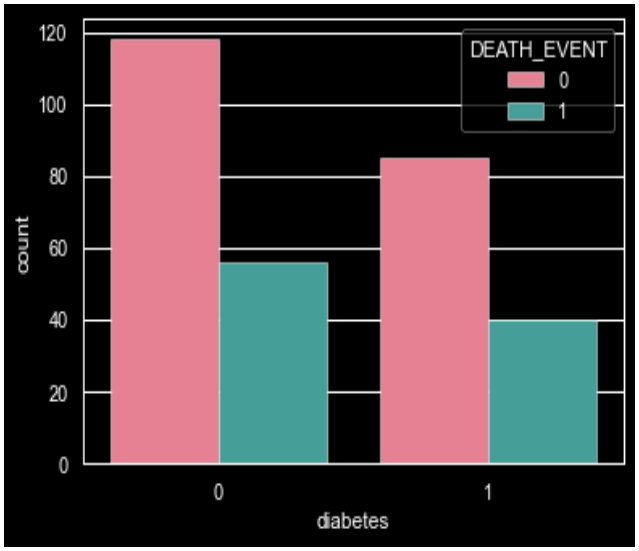
From above we observe that there is a lack of serum sodium level in male as well as female patients. But as compared to male patients female patients having lack of serum sodium level because the lower level of serum sodium in male patient is 125- 135mEq/L and only few patients having low level serum sodium.

7) count plot of diabetes and death event and the distribution of diabetes patients by using cat plot:



diabetes	0	1
sex		
0	50	55
1	124	70

diabetes	0	1
DEATH_EVENT		
0	118	85
1	56	4



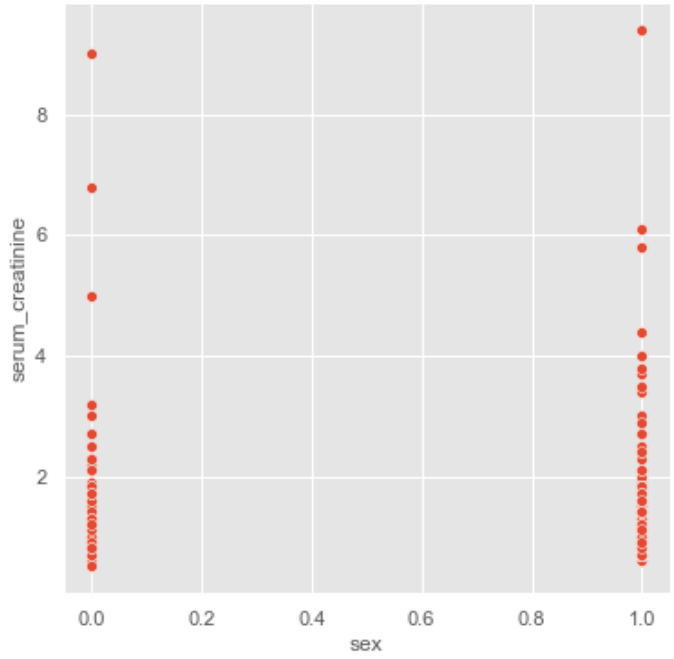
Interpretation:

From cat plot we observe that as compared to male patients, more female patients having diabetes.

From count plot and cross-table we observe that out of 118 non diabetes patients there are 85 are not survived and out of 56 diabetes patient 4 are not survived.

Hence we conclude that the factor diabetes does not affect on the death event.

8) Scatterplot of Serum Creatinine level in male and female patients



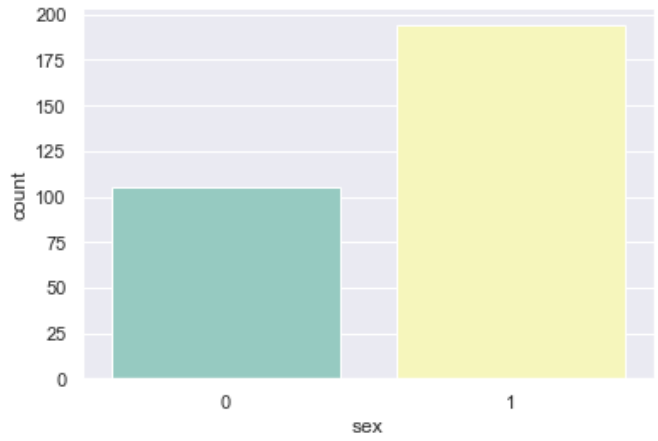
Interpretation:

From above Scatterplot we notice that the level of Serum Creatinine level in both male and female patients are very large as compare to normal level.

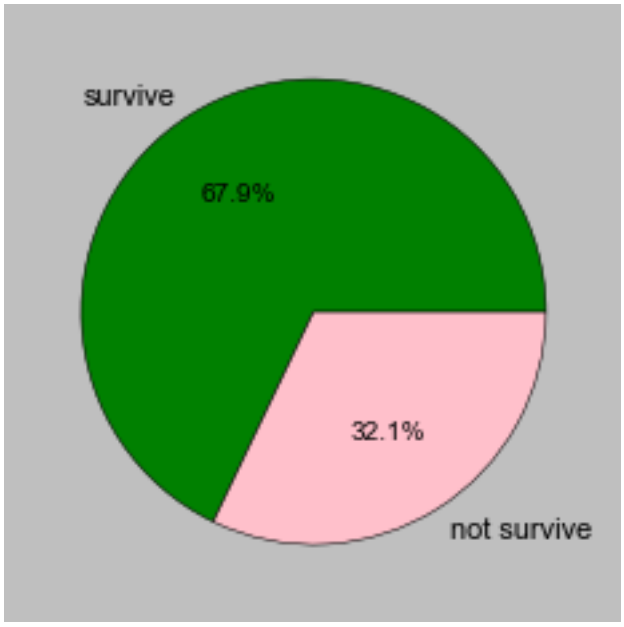
Serum Creatinine is the very important factor for kidney and abnormally high levels of creatinine thus warn of possible malfunction or the failure of the Kidneys.

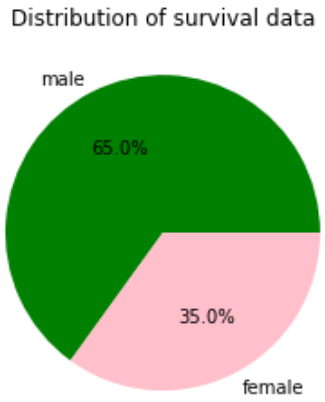
And this chances are approximate equal in both male and female patients.

9) Gender distribution by using count-plot and Death-event distribution by using pie chart:



DEATH_EVENT	0	1	Total
sex			
0	71	34	105
1	132	62	194
Total	203	96	299





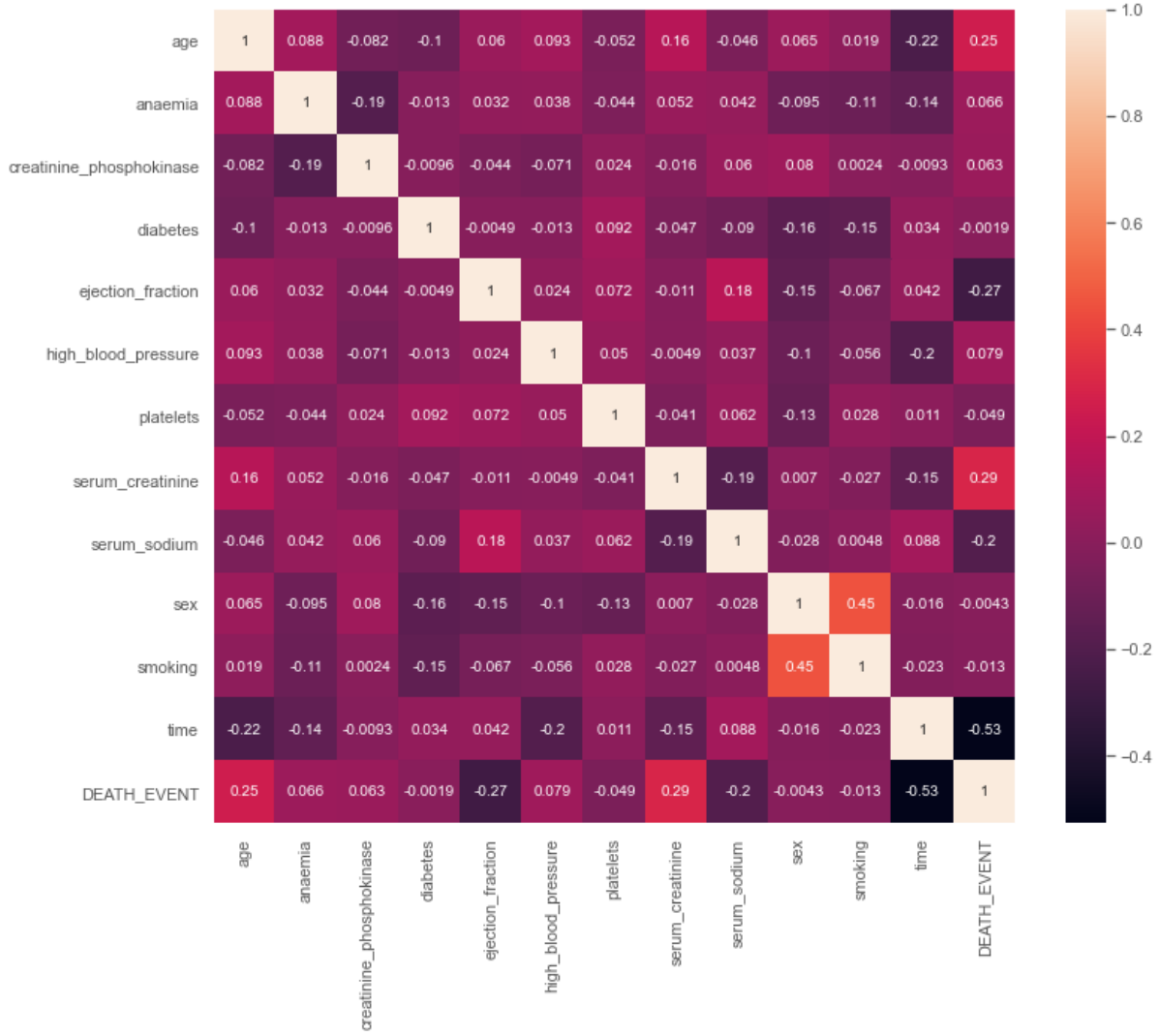
Interpretation:

From count plot we observe that there are 105 female patients and 194 male patients. From pie chart we conclude that out data 67.9% patients are survived and 32.1% has loss his life.

And out of survival data 65% are male and 35% are female.

Heatmap

Analysis of Correlation between factors of dataset by using Heatmap:



Interpretation:

From above correation map we observe that the positive correleation between sex and smoking. And the negative correlation between time and death event that means when time is increasing then probability or number of not survived patient are decreases. And other in other factor having very low correlation near to the no correlation.

HYPOTHESIS TESTING

1)Shapiro-Wilk test for the assessment of normality:

Hypothesis:-

H_0 : The data is normal.

H_1 : The data is not normal.

Result
statistic=0.29646962881088257, p-value=0.0

Interpretation:

Since P-value < level of significant (0.05)
Therefore we reject null hypothesis.
We conclude that the our data is not normal.

2) Chi-Square Test:

Hypothesis:

H_0 : Death-event and smoking both factors are independent.

H_1 : Death-event and smoking both factors are dependent.

Contingency table:

Observed Values		
Smoking	No	Yes
Death-Event		
No	137	66
Yes	66	30

Expected Values		
Smoking	No	Yes
Death-Event		
No	137.82827	65.1772
Yes	65.1772	30.8227

Chi-square statistics: 0.047643851312819833
P-value: 0.8272150738132376
Significance level: 0.05
Degree of freedom: 1

Conclusion:

Since P-value > significance value

Hence we accept H_0 and hence we conclude that the smoking and death-event factor are independent, that means smoking are not affected on heart disease and hence it not affected on death-event factor of this dataset.

DATA MINING CLASSIFIER

In this section I have started Modelling – Prediciting the attrition dataset by using python software. I have split the dataset into training and testing data so that, the model is trained on training data and predicts the result on test data. Here, the target variable is 'DEATH_EVENT'. Different classifiers are used below and their performance is measured.

- **Logistic Regression:-**

  Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is dichotomous, which means there would be only two possible classes.

  In simple words, the dependent variable is binary in nature having data coded as either 1 (stands for success/yes) or 0 (stands for failure/no).

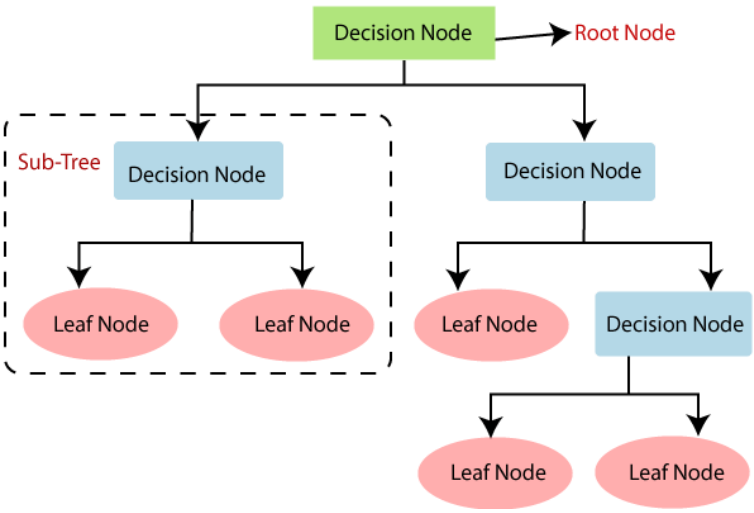
  Mathematically, a logistic regression model predicts P(Y=1) as a function of X. It is one of the simplest ML algorithms that can be used for various classification problems such as spam detection, Diabetes prediction, cancer detection, heart failure prediction etc.

- **Decision Tree:-**

  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 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. Below diagram explains the general structure of a decision tree:

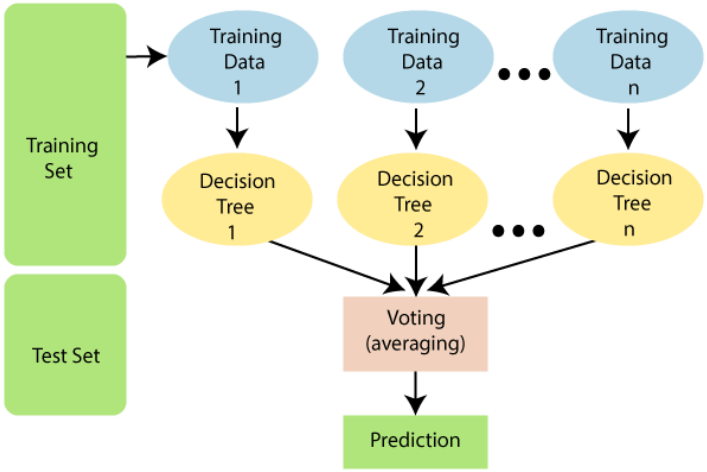


- **Random Forest:**

  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.



- **Naïve Bayes:**

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 on the basis of the probability of an object. Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

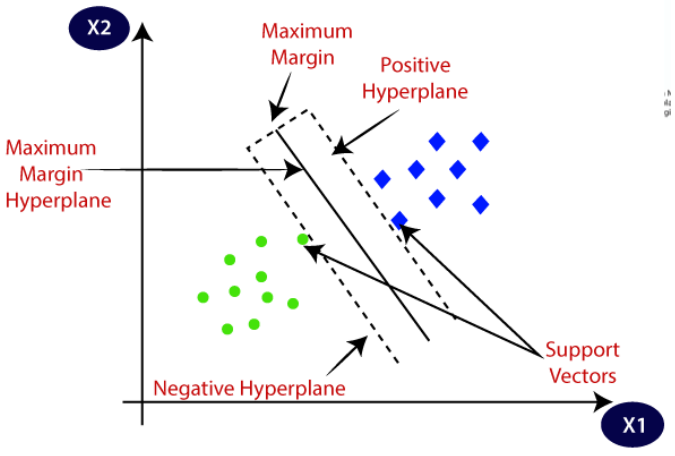
It is called Bayes because it depends on the principle of "Bayes' Theorem".

- **Support Vector Machine Algorithm:**

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 as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

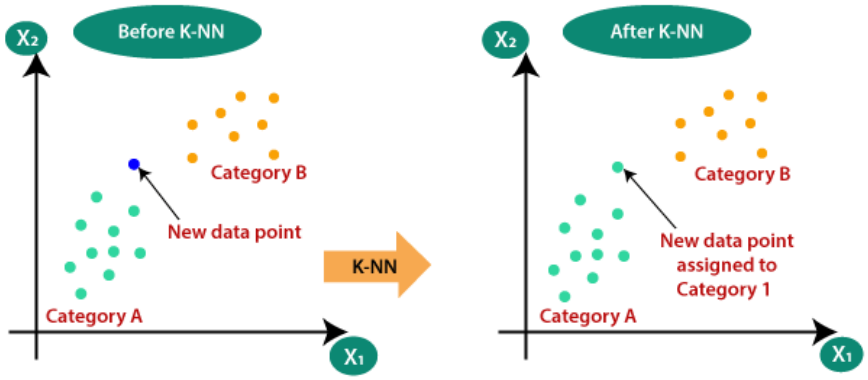


- **K-Nearest Neighbor(KNN) Algorithm:**

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique. It 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. It 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. It can be used for Regression as well as for Classification but mostly it is used for the Classification problems. It is a non-parametric algorithm, which means it does not make any assumption on underlying data.

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.



Confusion Matrix in Machine Learning:

he confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data. It can only be determined if the true values for test data are known. The matrix itself can be easily understood, but the related terminologies may be confusing. Since it shows the errors in the model performance in the form of a matrix, hence also known as an error matrix**.**

It looks like the below table:

n = total predictions	Actual: No	Actual: Yes
Predicted: No	True Negative	False Positive
Predicted: Yes	False Negative	True Positive

The above table has the following cases:

- True Negative:

  Model has given prediction No, and the real or actual value was also No.

- True Positive:

  The model has predicted yes, and the actual value was also true.

- False Negative:

  The model has predicted no, but the actual value was Yes, it is also called as Type-II error**.**

- False Positive:

  The model has predicted Yes, but the actual value was No. It is also called a Type-I error.

ACCURACY:

Accuracy is used to find the correct values; it is the sum of all true values divided by total values

$$\textbf{Accuracy}=\frac{TP+TN}{TP+FP+FN+TN}$$

PRECISION:

How often a model predicts a positive value is correct? It is all the true positives divided by the total number of predicted positive values.

$$\text{Precision}=\frac{TP}{TP+FP}$$

RECALL:

It used to calculate the models ability to predict positive values. How often does the model actually predict the correct positive values? It is true positives divided by the total number of actual positive values.

$$\text{Recall}=\frac{TP}{TP+FN}$$

F-1 SCORE:

If two models have low precision and high recall or vice versa, it is difficult to compare these models. So, for this purpose, we can use F-score. This score helps us to evaluate the recall and precision at the same time. The F-score is maximum if the recall is equal to the precision. It can be calculated using the below

$$\textbf{F-measure}=\frac{2*Recall*Precision}{Recall+Precision}$$

Feature Selection:

Univariate Selection:

Statistical test can be used to select those feature that have the strongest relationship with the output variable.

The sciki-learn library provides the SelectKBest class that can be used with a suite of different statistical tests to select a specific number of features.

In this project uses the Chi-squared statistical test for non-negative features to select the best features from the Heart Failure Prediction dataset.

Feature-Score Table:

	factor	Score
0	age	44.619455
1	anaemia	0.746593
2	creatinine_phosphokinase	1897.314839
3	diabetes	0.000657
4	ejection_fraction	79.072541
5	high_blood_pressure	1.221539
6	platelets	26135.77199
7	serum_creatinine	19.814118
8	serum_sodium	1.618175
9	sex	0.001956
10	smoking	0.032347
11	time	3826.892661

Selected Best-Feature Table:

Factor	Score
platelets	26135.77199
time	3826.892661
creatinine_phosphokinase	1897.314839
ejection_fraction	79.072541
age	44.619455
serum_creatinine	19.814118
serum_sodium	1.61817
high_blood_pressure	1.221539

Interpretation:

From the feature score table we select the best feature having more score which is highly affected on our target variable (Death-Event). These features we used for the machine learning algorithm as 'X'.

We observe that the highest score is of platelets factor and then time which is highly affected on our target variable.

Data Preprocessing:

- ✓ **Scaling the data:**

	age	creatinine_phosphokinase	ejection_fraction	platelets	serum_creatinine	serum_sodium
0	1.192945	0.000166	-1.53056	1.68E-02	0.490057	-1.504036
1	-0.491279	7.51464	-0.007077	7.54E-09	-0.284552	-0.141976
2	0.350833	-0.449939	-1.53056	-1.04E+00	-0.0909	-1.731046
3	-0.912335	-0.486071	-1.53056	-5.46E-01	0.490057	0.085034
4	0.350833	-0.435486	-1.53056	6.52E-01	1.264666	-4.682176

Splitting the data:

Here our target variable is DEATH_EVENT that means y variable and other variable are in x. Dimensions of Train and Test datasets are as follows:

x_train: (209, 8)
y_train: (209,)
x_test: (90, 8)
y_test: (90,)

Logistic Regression:

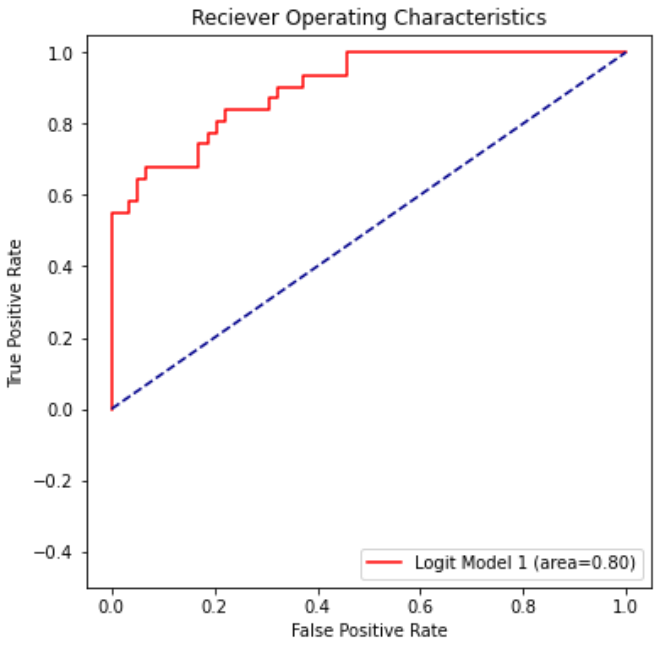
Confusion Matrix:

n=90	Predicted No	Predicted Yes
Actual No	54	5
Actual Yes	10	21

Classification Report:

	precision	recall	f1-score	support
0	0.84	0.92	0.88	59
1	0.81	0.68	0.74	31
accuracy	0.83			90
macro avg.	0.83	0.8	0.81	90
weighted avg.	0.83	0.83	0.83	90

Receiver Operating Characteristic Curve:



Interpretation:

From the confusion matrix, the classifier has made a total of 90 predictions. Out of 90 predictions, 75 are true predictions, and 15 are incorrect predictions according to test data. The model has predicted 10 are survived, but the actual value was not survived, it is also Type-II error**.** The model has predicted 5 are not survived and actual as survived which is Type-I error.

For this model the error type is Type-II error.

The accuracy of this model is 83% with 17% misclassification. The area under the curve is 80% hence the model represent good discrimination

Decesion Tree:

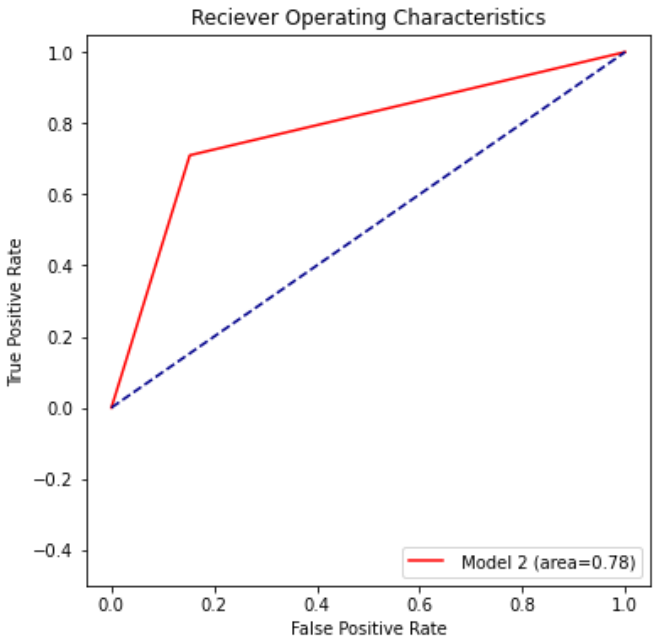
Confusion Matrix:

n=90	Predicted No	Predicted Yes
Actual No	50	9
Actual Yes	9	22

Classification Report:

	precision	recall	f1-score	support
0	0.85	0.85	0.85	59
1	0.71	0.71	0.71	31
accuracy		0.8		90
macro avg.	0.78	0.78	0.78	90
weighted avg.	0.8	0.8	0.8	90

Receiver Operating Characteristic Curve:



Interpretation:

From the confusion matrix, the classifier has made a total of 90 predictions. Out of 90 predictions, 72 are true predictions, and 18 are incorrect predictions according to test data. The model has predicted 9 are survived, but the actual value was not survived, it is also Type-II error**.** The model has predicted 9 are not survived and actual as survived which is Type-I error.

This model having both type of error.

The accuracy of this model is 80% with 20% misclassification. The area under the curve is 78% hence the model represent fair discrimination.

Random Forest:

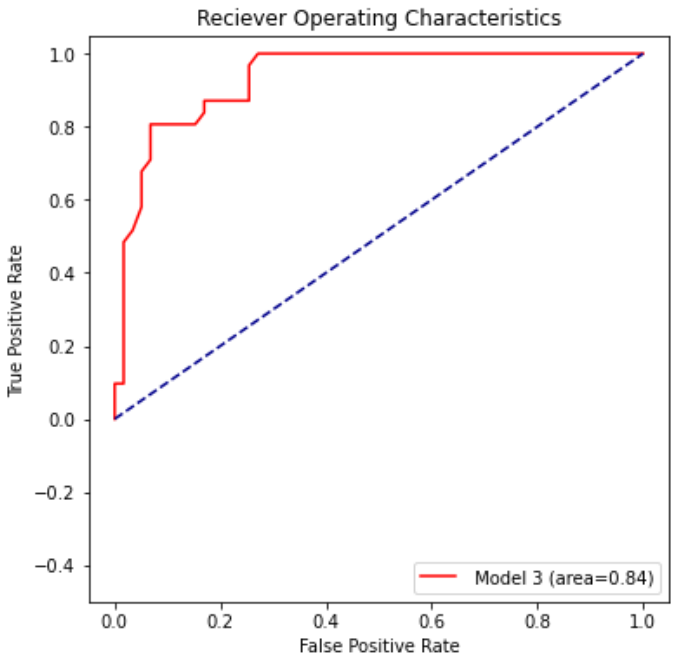
Confusion Matrix:

n=90	Predicted No	Predicted Yes
Actual No	51	8
Actual Yes	6	25

Classification Report:

	precision	recall	f1-score	support
0	0.89	0.86	0.88	59
1	0.76	0.81	0.78	31
accuracy	0.84			90
macro avg	0.83	0.84	0.83	90
weighted avg	0.85	0.84	0.85	90

Receiver Operating Characeristic Curve:



Interpretation:

From the confusion matrix, the classifier has made a total of 90 predictions. Out of 90 predictions, 76 are true predictions, and 14 are incorrect predictions according to test data. The model has predicted 6 are survied, but the actual value was not survived, it is also Type-II error**.** The model has predicted 8 are not survived and actual as survived which is Type-I error.

This model having Type-I error.

The accuracy of this model is 84% with 16% misclassification. The area under the curve is 84% hence the model represent good discrimination.

Support Vector Machine:

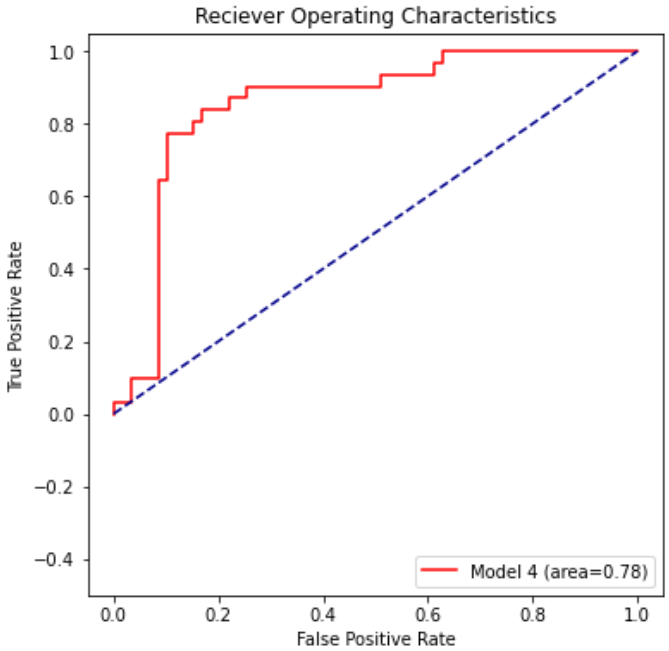
Confusion Matrix:

n=90	Predicted No	Predicted Yes
Actual No	56	3
Actual Yes	12	19

Classification Report:

	precision	recall	f1-score	support
0	0.82	0.95	0.88	59
1	0.86	0.61	0.72	31
accuracy	0.83			90
macro avg	0.84	0.78	0.8	90
weighted avg	0.84	0.83	0.83	90

Receiver Operating Characteristic curve:



Interpretation:

From the confusion matrix, the classifier has made a total of 90 predictions. Out of 90 predictions, 75 are true predictions, and 15 are incorrect predictions according to test data. The model has predicted 12 are survived, but the actual value was not survived, it is also Type-II error**.** The model has predicted 3 are not survived and actual as survived which is Type-I error.

This model having Type-II error.

The accuracy of this model is 83% with 17% misclassification. The area under the curve is 78% hence the model represent fair discrimination.

Naïve Bayes:

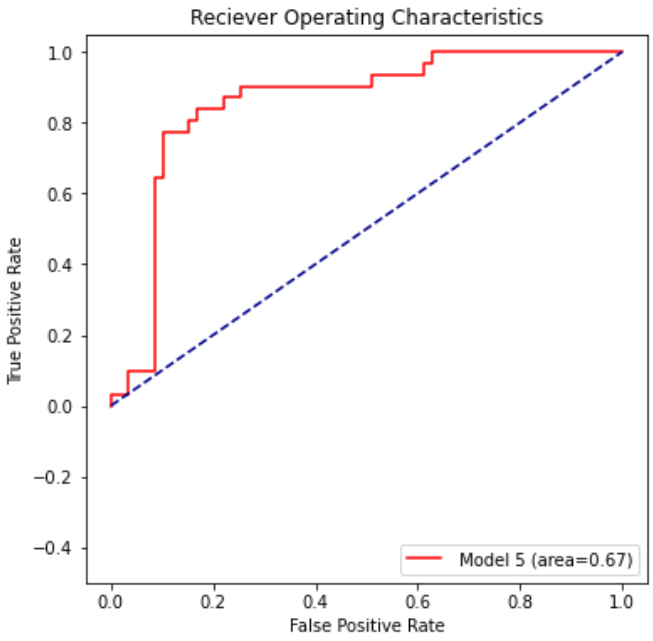
Confusion Matrix:

n=90	Predicted No	Predicted Yes
Actual No	54	5
Actual Yes	18	13

Classification Report:

	precision	recall	f1-score	support
0	0.75	0.92	0.82	59
1	0.72	0.42	0.53	31
accuracy	0.74			90
macro avg	0.74	0.67	0.68	90
weighted avg	0.74	0.74	0.72	90

Receiver Operating Characteristics Curve:



Interpretation:

From the confusion matrix, the classifier has made a total of 90 predictions. Out of 90 predictions, 67 are true predictions, and 21 are incorrect predictions according to test data. The model has predicted 18 are survived, but the actual value was not survived, it is also Type-II error**.** The model has predicted 5 are not survived and actual as survived which is Type-I error.

This model having Type-II error.

The accuracy of this model is 74% with 26% misclassification. The area under the curve is 67% hence the model represent fair discrimination.

K-Nearest Neighbor Classifier:

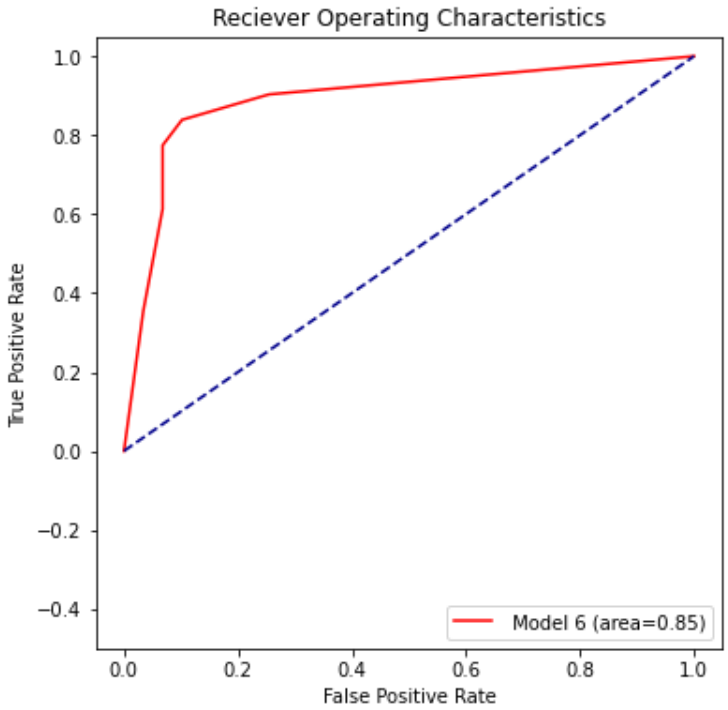
Confusion Matrix:

n=90	Predicted No	Predicted Yes
Actual No	55	4
Actual Yes	7	24

Classification Report:

	precision	recall	f1-score	support
0	0.89	0.93	0.91	59
1	0.86	0.77	0.81	31
accuracy	0.88			90
macro avg	0.87	0.85	0.86	90
weighted avg	0.88	0.88	0.88	90

Receiver Operating Characteristic Curve:



Interpretation:

From the confusion matrix, the classifier has made a total of 90 predictions. Out of 90 predictions, 79 are true predictions, and 11 are incorrect predictions according to test data. The model has predicted 7 are survived, but the actual value was not survived, it is also Type-II error**.** The model has predicted 4 are not survived and actual as survived which is Type-I error.

This model having Type-II error.

The accuracy of this model is 88% with 12% misclassification. The area under the curve is 85% hence the model represent good discrimination.

PREDICTION

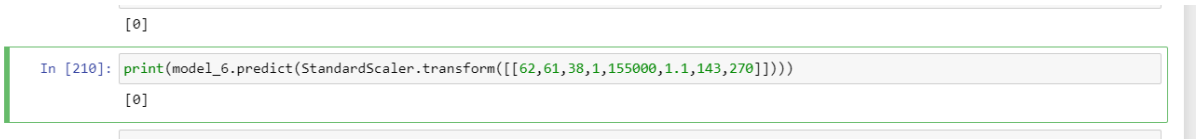
The performance of all Classifiers are as follows:

Model	Accuracy
K-Nearest Neighbor	0.88
Random Forest	0.84
Logistic Regression	0.83
Support Vector Machine	0.83
Decision Tree	0.8
Naïve Bayes	0.74

The best fitted model is K-Nearest Neighbor which gives greater accuracy 88%.

The type of error for this model is Type II error.

Machine Learning gives the prediction algorithm as given below:



Interpretation:

We use best fitted model for prediction of heart failure to the patient of 294 index number and we getting the prediction as the particular patient is survive and actually that patient is survived. Hence we can say that our model is doing work better.

Scope and Limitations

Scope:

Here the scope of the study is to apply machine learning algorithm to medical records in our dataset, in particular can be an effective too both to predict the survival of each patient having heart failure symptoms and to detect the most important clinical feature symptoms that may lead to heart failure. Predicting a heart disease in early stage will save many people's life.

Limitations:

- Model applied to the data can change their performance if we change the data.
- We develop model only with the available variables but if add other important variable eg. weight, Body Mass Index then we expect that our model gives better result

CONCLUSION

- We conclude that the K-Nearest Neighbor model is best fitted. It gives more accuracy than the all other models. Type error for this model is Type II error that means actually patient
- was not survive but the model and the model getting output as patient has survived.
- As compared to all Classifiers the greatest accuracy is of the K-Nearest model and the poor accuracy getting from the Naïve Bayes Classifier. The better performance after the K-Nearest Neighbor is getting from Random Forest. Both Logistic Regression and Support Vector Machine having accuracy same that means 83%.
  And the accuracy of Decision Tree is 8%.
- By using Chi-square test we conclude that the smoking does not affect on our target variable Death-Event.

REFERENCE

- ✓ Great Learning Online Application.
- ✓ Jiawei Han, Michelin Kamber, Jain Pei (2011): Data Mining Concepts and Techniques
- ✓ https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm
- ✓ https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm

APPENDIX

In [ ]:
```
import pandas as pd
from scipy import stats
import scipy as scipy
```

In [ ]:
```
data= pd.read_csv(r"E:\Data\Heart_failure.csv")
data
```

In [ ]:
```
data.isnull().sum()
```

In [ ]:
```
print(data.info())
```

In [ ]:
```
data.describe()
```

In [ ]:
```
data.shape
```

In [ ]:
```
stats.shapiro(data)
```

In [ ]:
```
data.corr()
```

Exploratory Data Analysis

In [ ]:
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(color_codes=True)
from matplotlib import style
```

In [ ]:
```
sns.catplot(data=data,x='sex',y='age',hue='DEATH_EVENT',palette='husl')
```

In [ ]:
```
sns.catplot(data=data,x='sex',y='age',hue='anaemia',palette='BuPu')
```

In [ ]:
```
data['sex'].value_counts()
```

In [ ]:
```
d.crosstab(data['sex'],data['DEATH_EVENT'])
```

In [ ]:
```
ns.countplot(x='smoking',data=data,palette='spring',hue='DEATH_EVENT')
```

In [ ]:
```
sns.catplot(data=data,x='sex',y='age',hue='high_blood_pressure',palette='husl')
```

In [ ]:
```
sns.catplot(data=data,x='sex',y='time',hue='DEATH_EVENT',palette='husl')
```

In [ ]:
```
pd.crosstab(data['high_blood_pressure'],data['DEATH_EVENT'])
```

In [ ]:
```
sns.countplot(x='DEATH_EVENT',data=data,hue='high_blood_pressure',palette='spring')
```

In [ ]:
```
style.use('dark_background')
sns.catplot(data=data,x='sex',y='age',hue='diabetes',palette='husl')
```

In [ ]:
```
pd.crosstab(data['DEATH_EVENT'],data['diabetes'])
```

In [ ]:
```
sns.countplot(data=data,x='diabetes',hue='DEATH_EVENT',palette='husl')
```

In [ ]:
```
style.use('dark_background')
plt.figure(figsize=(6,6))
sns.scatterplot(data=data,x='sex',y='serum_sodium',palette='terrain');
```

In [ ]:
```
style.use('ggplot')
plt.figure(figsize=(6,6))
sns.scatterplot(data=data,x='sex',y='serum_creatinine',palette='terrain');
```

In [ ]:
```
plt.figure(figsize=(12,10))
sns.heatmap(data.corr(),annot=True)
plt.show()
```

In [ ]:
```
import matplotlib.pyplot as plt
```

In [ ]:
```
gen_count=data['sex'].value_counts()
gen_count
```

In [ ]:
```
gender=['male','female']
to_gen={"gender":gender,"Gen_count":gen_count}
to_gen_count_df=pd.DataFrame(to_gen)
```

```
to_gen_count_df
```

In [ ]:
```
plt.pie(gen_count,labels=gender,autopct="%0.1f%%",colors=['yellow','pink'])
plt.show()
```

In [ ]:
```
a=data['sex']
b=data['DEATH_EVENT']
c={'gender':a,'death_event':b}
d=pd.DataFrame(c)
df_2=d[d['death_event']==0]
df_2
```

In [ ]:
```
surv_=df_2['gender'].value_counts()
surv_
```

In [ ]:
```
surv_gender=['male','female']
surv={"surv_gender":surv_gender,"count":surv_}
df_surv=pd.DataFrame(surv)
df_surv
```

In [ ]:
```
plt.pie(surv_,labels=surv_gender,autopct="%0.1f%%",colors=['green','pink'])
plt.title('Distribution of survival data')
plt.show()
```

Hypothesis Testing

In [ ]:
```
import scipy.stats as stats
import seaborn as sns
```

In [ ]:
```
data_table.values
```

In [ ]:
```
obs_val = data_table.values
print("observed values :-\n",obs_val)
```

In [ ]:
```
val = stats.chi2_contingency(data_table)
val
```

In [ ]:
```
exp_val = val[3]
```

In [ ]:
```
print("Expected Value :-\n",exp_val)
```

In [ ]:
```
no_of_rows = len(data_table.iloc[0:2,0])
no_of_columns = len(data_table.iloc[0,0:2])
ddof = (no_of_rows-1)*(no_of_columns-1)
print("Degree of freedom : ",ddof)
alpha=0.05
```

In [ ]:
```
from scipy.stats import chi2
chi_square = sum([(o-e)**2./e for o,e in zip(obs_val,exp_val)])
chi_square_statistics=chi_square[0]+chi_square[1]
```

In [ ]:
```
print("Chi-square statistics : ", chi_square_statistics)
```

In [ ]:
```
critical_value= chi2.ppf(q=1-alpha,df=ddof)
print('critical value :',critical_value)
```

In [ ]:
```
p_value = 1-chi2.cdf(x=chi_square_statistics,df=ddof)
```

In [ ]:
```
print("Chi-square statistics : ", chi_square_statistics)
print("P-value: ",p_value)
print("Significance level: ",alpha)
print("Degree of freedom: ",ddof)
```

Feature Score

In [ ]:
```
x= data.drop(["DEATH_EVENT"],axis=1)
y= data['DEATH_EVENT']
```

In [ ]:
```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
```

In [ ]:
```
bestfeature = SelectKBest(score_func=chi2,k=11)
fit = bestfeature.fit(x,y)
```

In [ ]:
```
dataScore = pd.DataFrame(fit.scores_)
datacolumns = pd.DataFrame(x.columns)
```

In [ ]:
```
featureScores = pd.concat([datacolumns, dataScore],axis=1)
```

```
featureScores.columns = ['factor','Score']
featurScores
```

In [ ]:
```
pd.DataFrame(featureScores.nlargest(8,'Score'))
```

Data Preprocessing

In [ ]:
```
from sklearn.preprocessing import StandardScaler
StandardScaler = StandardScaler()
columns_to_scale=['age', 'creatinine_phosphokinase',
       'ejection_fraction', 'platelets',
       'serum_creatinine', 'serum_sodium','high_blood_pressure','time']
data[columns_to_scale]=StandardScaler.fit_transform(data[columns_to_scale])
data.head()
```

In [ ]:
```
from sklearn.model_selection import train_test_split
data.drop(['anaemia'],axis=1,inplace=True)
data.drop(['diabetes'],axis=1,inplace=True)
data.drop(['sex'],axis=1,inplace=True)
data.drop(['smoking'],axis=1,inplace=True)
```

In [ ]:
```
x= data.drop(['DEATH_EVENT'],axis=1)
y= data['DEATH_EVENT']
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3,random_state=40)
print('x_train :',x_train.shape)
print('y_train :',y_train.shape)
print('x_test :',x_test.shape)
print('y_test :',y_test.shape)
```

Logistic Regression

In [ ]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
lr = LogisticRegression()
model_1 = lr.fit(x_train,y_train)
prediction_1= model_1.predict(x_test)
cm = confusion_matrix(y_test,prediction_1)
cm
TP = cm[0][0]
TN = cm[1][1]
FN = cm[1][0]
FP = cm[0][1]
print('Testing accuracy: ',(TP+TN)/(TP+TN+FN+FP))
accuracy_score(y_test,prediction_1)
print(classification_report(y_test,prediction_1))
print(classification_report(y_test,prediction_1))
log_roc_auc = roc_auc_score(y_test,model_1.predict(x_test))
fpr, tpr, threshold = roc_curve(y_test,model_1.predict_proba(x_test)[:,1])
#style.use('dark_background')
plt.figure(figsize=(6,6))
plt.plot(fpr,tpr,color='red',label="Logit Model 1 (area=%0.2f)"%log_roc_auc)
plt.plot([0,1],[0,1],color='darkblue',linestyle='--')
plt.xlim([-0.05,1.05])
plt.ylim([-0.5,1.05])
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("Reciever Operating Characteristics")
plt.legend(loc="lower right")
plt.savefig("log_ROC")

plt.show()
```

Decision Tree

In [ ]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [ ]:
```
DTC = DecisionTreeClassifier()
model_2 = DTC.fit(x_train,y_train)
prediction_2 = model_2.predict(x_test)
cm_2 = confusion_matrix(y_test,prediction_2)
cm_2
print('Accuracy : ', accuracy_score(y_test,prediction_2))
print(classification_report(y_test,prediction_2))
log_roc_auc = roc_auc_score(y_test,model_2.predict(x_test))
fpr, tpr, threshold = roc_curve(y_test,model_2.predict_proba(x_test)[:,1])
plt.figure(figsize=(6,6))
plt.plot(fpr,tpr,color='red',label=" Model 2 (area=%0.2f)"%log_roc_auc)
plt.plot([0,1],[0,1],color='darkblue',linestyle='--')
plt.xlim([-0.05,1.05])
plt.ylim([-0.5,1.05])
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
```

```
plt.title("Reciever Operating Characteristics")
plt.legend(loc="lower right")
plt.savefig("log_ROC")

plt.show()
```

Random Forest

In [ ]:
```
from sklearn.ensemble import RandomForestClassifier
RFC = RandomForestClassifier()
model_3 = RFC.fit(x_train,y_train)
prediction_3 = model_3.predict(x_test)
print('Accuracy : ', accuracy_score(y_test,prediction_3))
print(classification_report(y_test,prediction_3))
log_roc_auc = roc_auc_score(y_test,model_3.predict(x_test))
fpr, tpr, threshold = roc_curve(y_test,model_3.predict_proba(x_test)[:,1])
plt.figure(figsize=(6,6))
plt.plot(fpr,tpr,color='red',label=" Model 3 (area=%0.2f)"%log_roc_auc)
plt.plot([0,1],[0,1],color='darkblue',linestyle='--')
plt.xlim([-0.05,1.05])
plt.ylim([-0.5,1.05])
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("Reciever Operating Characteristics")
plt.legend(loc="lower right")
plt.savefig("log_ROC")

plt.show()
```

Support Vector Machine

In [ ]:
```
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.svm import SVC
svm = SVC()
model_4 = svm.fit(x_train,y_train)
prediction_4 = model_4.predict(x_test)
cm_4 = confusion_matrix(y_test,prediction_4)
cm_4
print('Accuracy : ', accuracy_score(y_test,prediction_4))
print(classification_report(y_test,prediction_4))
```

In [ ]:
```
log_roc_auc = roc_auc_score(y_test,model_4.predict(x_test))
fpr, tpr, threshold = roc_curve(y_test,model_4.predict_proba(x_test)[:,1])
plt.figure(figsize=(6,6))
plt.plot(fpr,tpr,color='red',label="Model 4 (area=%0.2f)"%log_roc_auc)
plt.plot([0,1],[0,1],color='darkblue',linestyle='--')
plt.xlim([-0.05,1.05])
plt.ylim([-0.5,1.05])
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("Reciever Operating Characteristics")
plt.legend(loc="lower right")
plt.savefig("log_ROC")

plt.show()
```

Naive Bayes

In [ ]:
```
from sklearn.naive_bayes import GaussianNB
NB = GaussianNB()
model_5 = NB.fit(x_train,y_train)
prediction_5 = model_5.predict(x_test)
cm_5=confusion_matrix(y_test,prediction_5)
cm_5
print('Accuracy : ', accuracy_score(y_test,prediction_5))
print(classification_report(y_test,prediction_5))
log_roc_auc = roc_auc_score(y_test,model_5.predict(x_test))
fpr, tpr, threshold = roc_curve(y_test,model_5.predict_proba(x_test)[:,1])
plt.figure(figsize=(6,6))
plt.plot(fpr,tpr,color='red',label=" Model 5 (area=%0.2f)"%log_roc_auc)
plt.plot([0,1],[0,1],color='darkblue',linestyle='--')
plt.xlim([-0.05,1.05])
plt.ylim([-0.5,1.05])
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("Reciever Operating Characteristics")
plt.legend(loc="lower right")
plt.savefig("log_ROC")

plt.show()
```

KNN

In [ ]:
```
from sklearn.neighbors import KNeighborsClassifier
```

```
KNN = KNeighborsClassifier()
model_6 = KNN.fit(x_train,y_train)
prediction_6 = model_6.predict(x_test)
cm_6 = confusion_matrix(y_test,prediction_6)
cm_6
print('Accuracy : ', accuracy_score(y_test,prediction_6))
print(classification_report(y_test,prediction_6))
log_roc_auc = roc_auc_score(y_test,model_6.predict(x_test))
fpr, tpr, threshold = roc_curve(y_test,model_6.predict_proba(x_test)[:,1])
plt.figure(figsize=(6,6))
plt.plot(fpr,tpr,color='red',label=" Model 6 (area=%0.2f)"%log_roc_auc)
plt.plot([0,1],[0,1],color='darkblue',linestyle='--')
plt.xlim([-0.05,1.05])
plt.ylim([-0.5,1.05])
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("Reciever Operating Characteristics")
plt.legend(loc="lower right")
plt.savefig("log_ROC")

plt.show()
```

Prediction

In [ ]:
```
print(model_6.predict(StandardScaler.transform([[62,61,38,1,155000,1.1,143,270]])))
```