#### **Predict Cardiovascular Diseases**

**This project is made by Badal Gupta**

* **Steps For The Project Work**

1. **Dataset Description**

**This work, we tried to classifying cardiovascular diseases using Different models. Hopefully can be of great help for early detection and management of cardiovascular diseases**.

* **Data set taken from Kaggle**
* **Predictor variable use in classifying Cardiovascular diseases :**
* **Age**
* **Anaemia**
* **Creatinine\_phosphokinase**
* **Diabetes**
* **Ejection\_fraction**
* **High\_blood\_pressure**
* **Platelets**
* **Serum\_creatinine**
* **Serum\_sodium**
* **Sex**
* **Smoking**
* **Time**

1. **Data Preprocessing**
2. **Libraries Imported in Dataset :-**

# Importing Pandas an Numpy Libraries to use on manipulating our Data

import pandas as pd

import numpy as np

# To Split our train data

from sklearn.model\_selection import train\_test\_split

# To Visualize Data

import matplotlib.pyplot as plt

import seaborn as sns

# To Train our data

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.neighbors import KNeighborsClassifier

# To evaluate end result we have

from sklearn.metrics import accuracy\_score

from sklearn.preprocessing import MinMaxScaler

from sklearn.metrics import classification\_report ,confusion\_matrix ,ConfusionMatrixDisplay

#  To use Scaling

from sklearn.preprocessing import StandardScaler   #  used for optimization purposes

from sklearn.decomposition import PCA

1. **Imported Dataset from Kaggle**
2. **Exploring Dataset**

# Print number of rows in data

print(data.shape)

# Size and shape of dataset

print(data.size)

1. **Data pre-processing**

# Prints Summary of Data

data.describe()

1. **Splitting Numerical and Categorical Values**
2. **Investigating Missing Values**

data.isnull().sum()

1. **Data Encoding**
2. **Split Data to Train and Test**

Y = data['DEATH\_EVENT']

X = data.drop(columns=['DEATH\_EVENT'])

1. **Training our models**

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size=0.15, random\_state=1)

1. **Scaling**

#  I have Used Minmax and PCA scalar

1. **Model Selection**

* **Random Forest**
  + **Min Max Scalar**

# We define the model

rfcla = RandomForestClassifier(n\_estimators=100,random\_state=9,n\_jobs=4)

# We train model

rfcla.fit(X\_train\_min, y\_train)

# We predict target values

Y\_predict5 = rfcla.predict(X\_test\_min)

* + **PCA Scalar**

# We define the model

rfcla = RandomForestClassifier(n\_estimators=100,random\_state=9,n\_jobs=4)

# We train model

rfcla.fit(X\_train\_pca, y\_train)

# We predict target values

Y\_predict5 = rfcla.predict(X\_test\_pca)

* **Decision Tree**
  + **Min Max Scalar**

# Define Decision Tree Model

dt = DecisionTreeClassifier()

# We fit our model with our train data

dt.fit(X\_train\_min, y\_train)

# Then predict results from X\_test data

pred\_dt = dt.predict(X\_test\_min)

* + **PCA Scalar**

# Define Decision Tree Model

dt = DecisionTreeClassifier()

# We fit our model with our train data

dt.fit(X\_train\_pca, y\_train)

# Then predict results from X\_test data

pred\_dt = dt.predict(X\_test\_pca)

* **K-Nearest Neighbours**
  + **Min Max Scalar**

# Define KNN Model

knn = KNeighborsClassifier(n\_neighbors=3, metric="minkowski")

# We fit our model with our train data

knn.fit(X\_train\_min, y\_train)

# Then predict results from X\_test data

pred\_knn = knn.predict(X\_test\_min)

* + **PCA Scalar**

# Define KNN Model

knn = KNeighborsClassifier(n\_neighbors=3, metric="minkowski")

# We fit our model with our train data

knn.fit(X\_train\_pca, y\_train)

# Then predict results from X\_test data

pred\_knn = knn.predict(X\_test\_pca)

1. **Result and Discussion**

**We have use Python language to work on this Dataset**

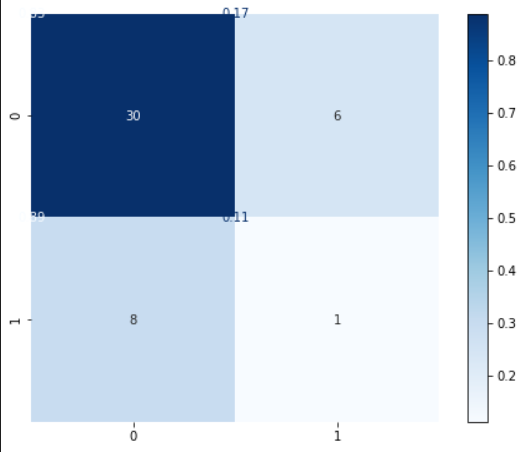
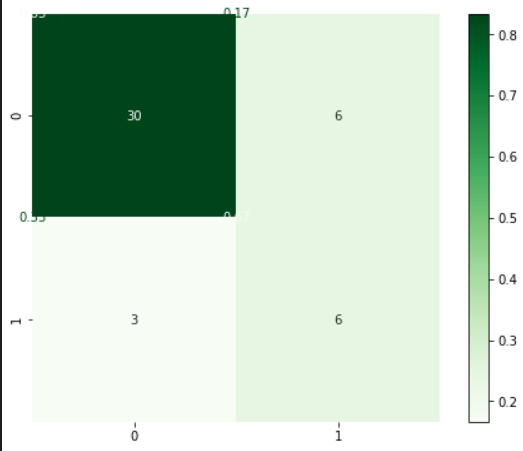
* **Output table**
* **Model comparison for Prediction Cardiovascular Diseases using MinMax**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ML Models | Accuracy | Precision | Recall | f1-score |
| Random Forest | **0.80** | **0.91** | **0.83** | **0.87** |
| Decision Tree | **0.64** | **0.92** | **0.61** | **0.73** |
| KNN | **0.82** | **0.91** | **0.86** | **0.89** |

* **Model comparison for Prediction Cardiovascular Diseases using PCA**

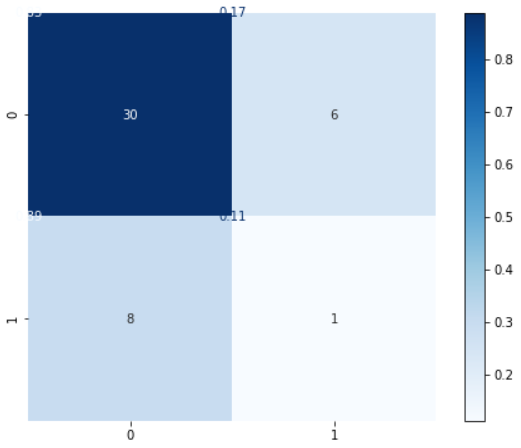
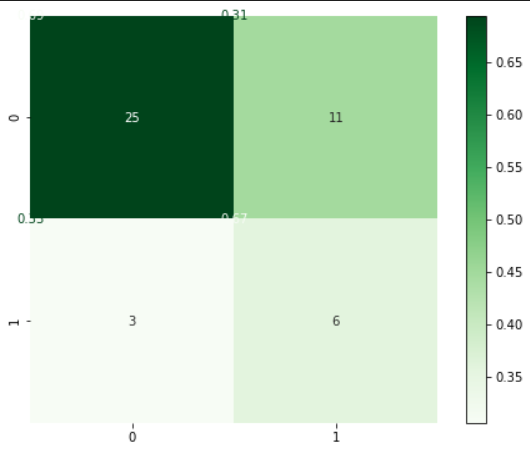
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ML Models | Accuracy | Precision | Recall | f1-score |
| Random Forest | **0.69** | **0.79** | **0.83** | **0.81** |
| Decision Tree | **0.67** | **0.78** | **0.81** | **0.79** |
| KNN | **0.64** | **0.79** | **0.75** | **0.77** |

* **Confusion Matrix**
* **Random Forest**

****

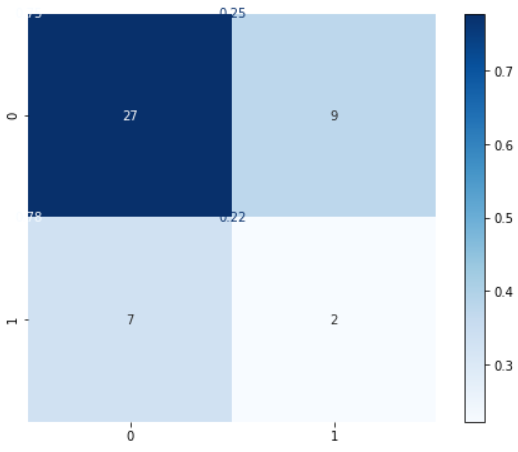
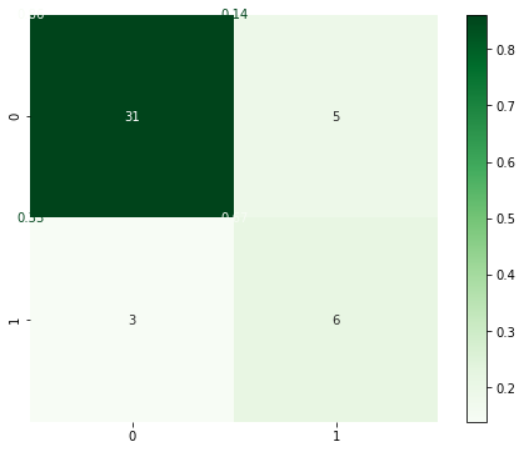
**MinMax Scalar PCA Scalar**

* **Decision Tree**

****

**MinMax Scalar PCA Scalar**

* **K-Nearest Neighbours**

****

**MinMax Scalar PCA Scalar**

* **So by the observation of evaluation of all the Models the most precise model is *Random Forest* for my dataset .**
* **Accuracy of 80%**
* **Precision of 91%**
* **Recall of 83%**
* **f1 score 87%**