HEART FAILURE PREDICTION

Presented by

1.Miss S.Gomathi

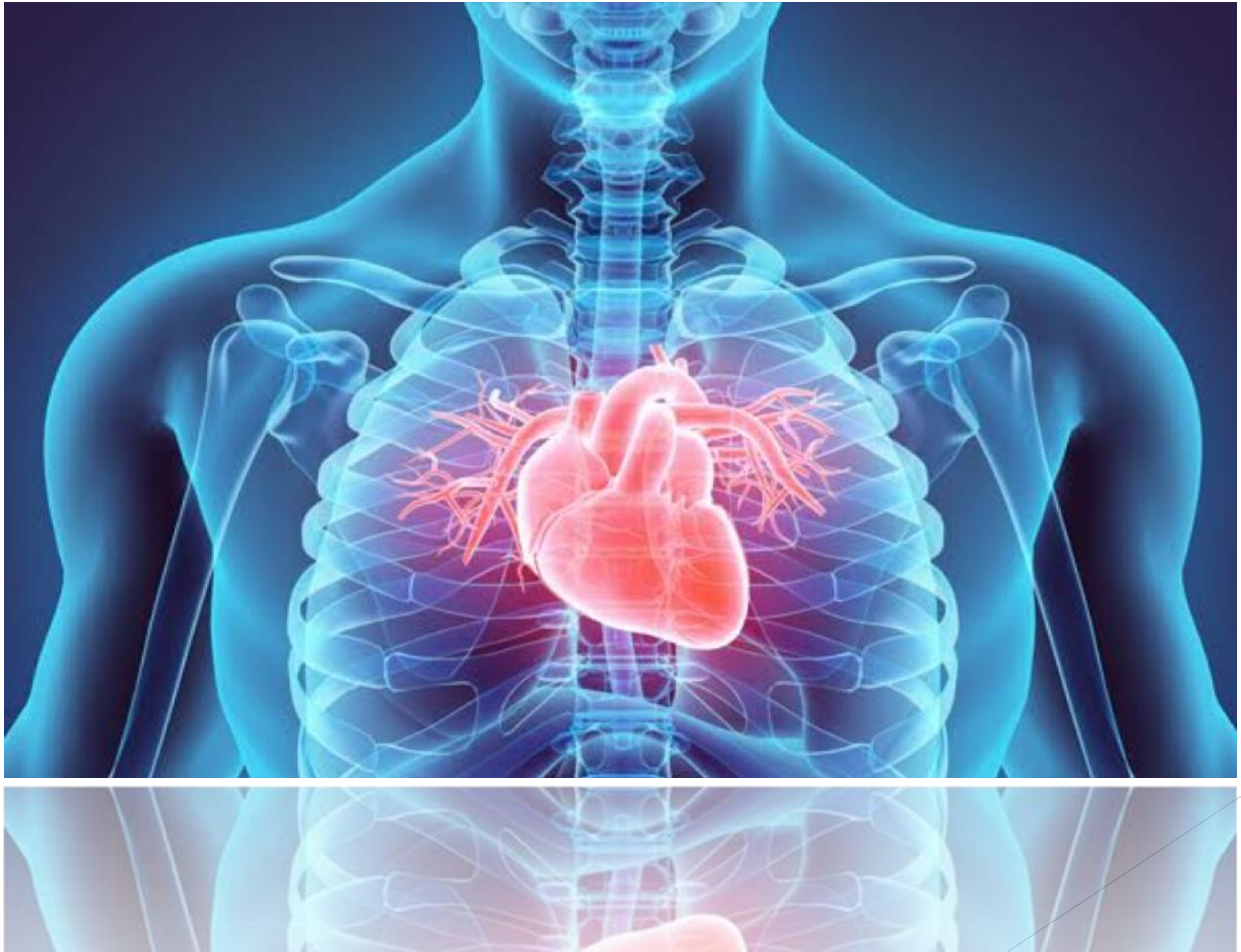
2.Miss S.Preethika

3.Miss N.Haritha

4.Miss C.Shubikshaa

Guided by

Miss S.Gomathi



OUTLINE

=>Introduction

=>Understanding Heart Failure

=>Data Collection and Preprocessing

=>Machine Learning Models for Heart Failure Prediction

=>Challenges and Limitations

=>Coding

=>Conclusion

INTRODUCTION

- Heart failure is a serious medical condition that affects millions of people worldwide. It occurs when the heart is unable to pump enough blood to meet the body's needs. Predicting heart failure can be challenging but with the help of artificial intelligence and python programming, it becomes possible.In this presentation, we will discuss how AI and machine learning algorithms can be used to predict heart failure in patients using data analysis techniques.

UNDERSTANDING HEART FAILURE

- Understanding Heart FailureBefore we dive into the details of predicting heart failure using AI, let's first understand what heart failure is and its causes. Heart failure occurs when the heart muscle becomes weak or damaged, leading to reduced blood flow to the body's organs and tissues.There are several causes of heart failure, including high blood pressure, coronary artery disease, heart attack, diabetes, and obesity. Early detection and treatment of heart failure can improve outcomes and prevent complications such as stroke and kidney damage.

DATA COLLECTION AND PREPROCESSING

- Data Collection and PreprocessingTo predict heart failure using AI, we need to collect and preprocess patient data. This includes demographic information, medical history, lab results, and imaging studies. The data is then cleaned, transformed, and standardized to ensure consistency and accuracy.Python libraries such as Pandas and NumPy are commonly used for data preprocessing tasks. Once the data is ready, we can move on to training our machine learning models.

MACHINE LEARNING MODELS FOR HEART FAILURE PREDICTION

- There are several machine learning models that can be used for heart failure prediction, including logistic regression, decision trees, random forests, and support vector machines. These models use statistical algorithms to analyze patient data and identify patterns and relationships that can predict heart failure.The performance of these models can be evaluated using metrics such as accuracy, precision, and recall. The best- performing model can then be deployed in a clinical setting to assist healthcare providers in predicting heart failure in their patients.

CHALLENGES AND LIMITATIONS

- While AI and machine learning have great potential for predicting heart failure, there are also challenges and limitations to consider. One of the biggest challenges is the quality and quantity of data available. Without sufficient data, machine learning algorithms may not be accurate or reliable.Another limitation is the interpretability of machine learning models. Healthcare providers need to understand how the model arrived at its predictions to make informed decisions about patient care. Finally, ethical considerations such as privacy and bias must be addressed when implementing AI in healthcare settings.

CODINGS

```
import pandas as pd
heart_data=pd.read_csv("heart_failure_clinical_records_dataset (1).csv")
heart_data
heart_data.head()
Heart_data.describe()
```

Visualization

```
Heart_data.hist(figsize=(15,15),edgecolor=‘black’);
heart_data.isnull().sum()
```

pie charts

```
Import plotly.graph_objs as go
labels = [‘No Diabetes’,’Diabetes’]
diabetes_yes = heart_data[heart_data[‘diabetes’]==1]
diabetes_no = heart_data[heart_data[‘diabetes’]==0]
values = [len(diabetes_no), len(diabetes_yes)]
fig = go.Figure(data=[go.Pie(labels=labels, values=values, hole=.2)])
fig.update_layout(
    title_text=“Analysis on Diabetes”)
fig.show()
```

Continued….

Import plotly.express as px

fig=px.pie(heart_data,values=‘diabetes’,names=‘DEATH_EVENT’,title=‘Death Analysis’)

fig.show()

Heat Map

Import matplotlib.pyplot as plt

import seaborn as sns

plt.figure(figsize=(10,10))

sns.heatmap(heart_data.corr(),vmin=-1,cmap=‘coolwarm’,annot=True);

Data Modeling

Logistic Regressio

from sklearn.model_selection import train_test_split

from sklearn.metrics import confusion_matrix,accuracy_score

Feature=[‘time’,’ejection_fraction’,’serum_creatinine’]

x=heart_data[Feature]

y=heart_data[“DEATH_EVENT”]

Xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2,random_state=2)

Continued…..

```
From sklearn.linear_model import LogisticRegression
Log_re=LogisticRegression()
Log_re.fit(xtrain,ytrain)
log_re_pred=log_re.predict(xtest)
Log_acc=accuracy_score(ytest,log_re_pred)
print("Logistic Accuracy Score: ","{:.2f}%".format(100*log_acc))
%pip install mlxtend
from mlxtend.plotting import plot_confusion_matrix
cm = confusion_matrix(ytest, log_re_pred)
plt.figure()
plot_confusion_matrix(cm, figsize=(12,8), hide_ticks=True, cmap=plt.cm.Blues)
plt.title("Logistic Regerssion  - Confusion Matrix")
plt.xticks(range(2), ["Heart Not Failed","Heart Fail"], fontsize=16)
plt.yticks(range(2), ["Heart Not Failed","Heart Fail"], fontsize=16)
plt.show()
```

Continued….

Age VS Death_count

```
Import plotly.express as px
fig = px.histogram(heart_data, x="age", color="DEATH_EVENT", marginal="violin", hover_data=heart_data.columns,
                   title ="Distribution of AGE Vs DEATH_EVENT",
                   labels={"age": "AGE"},
                   template="plotly_dark",
                   color_discrete_map={"0": "RebeccaPurple", "1": "MediumPurple"}
                  )
fig.show()
```

ANN

```
X=heart_data.drop(["DEATH_EVENT"],axis=1)
y=heart_data["DEATH_EVENT"]
From sklearn import preprocessing
from sklearn.preprocessing import StandardScaler
col_names = list(X.columns)
s_scaler = preprocessing.StandardScaler()
X_df= s_scaler.fit_transform(X)
X_df = pd.DataFrame(X_df, columns=col_names)
X_df.describe().T
```

Continued….

import seaborn as snscolours =["#774571","#b398af","#f1f1f1" ,"#afcdc7", "#6daa9f"]plt.figure(figsize=(20,10))sns.boxenplot(data = X_df,palette = colours)plt.xticks(rotation=90)plt.show()

import plotly.graph_objects as go

from plotly.subplots import make_subplots

d1 = heart_data[(heart_data[“DEATH_EVENT”]==0) & (heart_data[“sex”]==1)]

d2 = heart_data[(heart_data[“DEATH_EVENT”]==1) & (heart_data[“sex”]==1)]

d3 = heart_data[(heart_data[“DEATH_EVENT”]==0) & (heart_data[“sex”]==0)]

d4 = heart_data[(heart_data[“DEATH_EVENT”]==1) & (heart_data[“sex”]==0)]

label1 = [“Male”,”Female”]

label2 = [‘Male – Survived’,’Male – Died’, “Female - Survived”, “Female – Died”]

Values1 = [(len(d1)+len(d2)), (len(d3)+len(d4))]

values2 = [len(d1),len(d2),len(d3),len(d4)]

Create subplots: use ‘domain’ type for Pie subplot

fig = make_subplots(rows=1, cols=2, specs=[[{‘type’:’domain’}, {‘type’:’domain’}]])

fig.add_trace(go.Pie(labels=label1, values=values1, name=“GENDER”),1, 1)

Continued….

fig.add_trace(go.Pie(labels=label2, values=values2, name="GENDER VS DEATH_EVENT"),1, 2)

Use `hole` to create a donut-like pie chartfig.update_traces(hole=.4, hoverinfo="label+percent")fig.update_layout(title_text="GENDER DISTRIBUTION IN THE DATASET \ GENDER VS DEATH_EVENT",

Add annotations in the center of the donut pies. annotations=[dict(text='GENDER', x=0.19, y=0.5, font_size=10, showarrow=False), dict(text='GENDER VS DEATH_EVENT', x=0.84, y=0.5, font_size=9, showarrow=False)], autosize=False,width=1200, height=500, paper_bgcolor="pink")fig.show()

Train/Test Split & Normalization

X = heart_data.drop(“DEATH_EVENT”, axis = 1)

y = heart_data[‘DEATH_EVENT’]

Heart_data

X_train, x_test, y_train, y_test = train_test_split(x, y, random_state =100 ,stratify=y, test_size = 0.3)

print(y_train.value_counts())

from sklearn.preprocessing import MinMaxScalerscale = MinMaxScaler()col = ["anaemia","creatinine_phosphokinase","diabetes","ejection_fraction","high_blood_pressure","platelets","serum_creatinine","serum_sodium","sex","smoking","time"]

X_train[col] = scale.fit_transform(x_train[col])

x_test[col] = scale.transform(x_test[col])

Continued…..

Decision Tree Classifier¶

```
from sklearn.model_selection import tr
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, plot_confusion_matrix
from sklearn.tree import DecisionTreeClassifier
list1 = []
for leaves in range(2,10):
    classifier = DecisionTreeClassifier(max_leaf_nodes = leaves, random_state=0, criterion='entropy')
    classifier.fit(x_train, y_train)
    y_pred = classifier.predict(x_test)
    list1.append(accuracy_score(y_test,y_pred)*100)
print("Decision Tree Classifier Top 5 Success Rates:")
print([round(I, 2) for I in sorted(list1, reverse=True)[:5]])
plot_confusion_matrix(classifier, x_test, y_test)
plt.show()
```

K Nearest Neighbors

Continued…..

```
from sklearn.neighbors import KNeighborsClassifierfrom sklearn.metrics import accuracy_score, f1_score,confusion_matrix, recall_score, precision_score, classification_reportfrom sklearn.model_selection import cross_val_score, cross_val_predictKNN = KNeighborsClassifier(n_neighbors=8)KNN.fit(x_train, y_train)y_test_pred_KNN = KNN.predict(x_test)y_train_pred_KNN = KNN.predict(x_train)test_acc_KNN = accuracy_score(y_test, y_test_pred_KNN)train_acc_KNN = accuracy_score(y_train, y_train_pred_KNN)scores_KNN = cross_val_score(KNN, x_train , y_train , cv = 10, scoring = 'accuracy' )precision_score_KNN = precision_score(y_test, y_test_pred_KNN)recall_score_KNN = recall_score(y_test, y_test_pred_KNN)f1_score_KNN = f1_score(y_test, y_test_pred_KNN)conf_KNN = confusion_matrix(y_test, y_test_pred_KNN)accuracy_score_KNN = accuracy_score(y_test, y_test_pred_KNN)print("accuracy score:", accuracy_score_KNN)print("Train set Accuracy: ", train_acc_KNN)print("Test set Accuracy: ", test_acc_KNN)print("cv:  %s\n"% scores_KNN.mean())print("**************************************************")print ("precision_score: ", precision_score_KNN)print("recall_score: ", recall_score_KNN)print("f1_score: ", f1_score_KNN)print("**************************************************")print("\nReport:\n%s\n"%classification_report(y_test, y_test_pred_KNN))

Print(f'Decision Tree Classifier: {round(sorted(list1, reverse=True)[0])}%')

Print(f'Logistic Regression: {round(100*log_acc, 2)} %')

Print(f'K Nearest Neighbors: {(accuracy_score_KNN)} %')
```

CONCLUSION

- Try TomeIn conclusion, AI and machine learning have great potential for predicting heart failure in patients. By collecting and preprocessing patient data and using machine learning models, healthcare providers can identify patients at risk of heart failure and provide early intervention.However, there are also challenges and limitations to consider, and ethical considerations must be addressed when implementing AI in healthcare settings. With careful consideration and implementation, AI can be a powerful tool for improving patient outcomes and reducing healthcare costs.

THANK YOU