A PROJECT REPORT

on

"HEART HEALTH PREDICTOR"

Submitted to

KIIT Deemed to be University

In Partial Fulfillment of the Requirement for the Award of

BACHELOR'S DEGREE IN
INFORMATION TECHNOLOGY

BY

Akshat Raj	2106181
Aman Kumar Singh	2106183
Animit Dash	2106186
Ritesh Ranjan	2106243

UNDER THE GUIDANCE OF

Gananath Bhuyan

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAR, ODISHA - 751024

April 2024

A PROJECT REPORT

on

“HEART HEALTH PREDICTOR”

Submitted to

KIIT Deemed to be University

In Partial Fulfillment of the Requirement for the Award of

BACHELOR’S DEGREE IN INFORMATION TECHNOLOGY

BY

Akshat Raj	2106181
Aman Kumar Singh	2106183
Animit Dash	2106186
Ritesh Ranjan	2106243

UNDER THE GUIDANCE OF

Gananath Bhuyan

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAR, ODISHA -751024

April 2024

KIIT Deemed to be University

School of Computer Engineering
Bhubaneswar, ODISHA 751024

CERTIFICATE

This is certify that the project entitled

“HEART HEALTH PREDICTOR“

submitted by

Akshat Raj	2106181
Aman Kumar Singh	2106183
Animit Dash	2106186
Ritesh Ranjan	2106243

is a record of bona fide work carried out by them, in the partial fulfillment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Engineering OR Information Technology) at KIIT Deemed to be university, Bhubaneswar. This work is done during the year 2023-2024, under our guidance.

Date: / /

Gananath Bhuyan
Project Guide

Acknowledgements

We are profoundly grateful to **Gananath Bhuyan** of **Affiliation** for his expert guidance and continuous encouragement throughout to see that this project meets its target since its commencement to its completion. ....

Akshat Raj
Aman Kumar Singh
Animit Dash
Ritesh Ranjan

ABSTRACT

Cardiovascular diseases have emerged as a significant global health concern due to lifestyle changes, dietary habits, and working patterns. Early detection of cardiovascular issues is crucial for reducing mortality rates, yet limited medical resources and specialist availability hinder continuous patient monitoring. Technological interventions are essential to facilitate remote patient monitoring and treatment. This project focuses on leveraging healthcare data and machine learning algorithms to develop efficient prediction models for cardiovascular diseases. By analyzing medical data and historical information, this study aims to explore the effectiveness of various machine learning techniques in predicting heart health, thereby enabling proactive interventions and lifestyle modifications for high-risk individuals, potentially marking a significant milestone in preventive medicine.

The study delves into some of the most widely used machine learning algorithms for heart disease prediction, utilizing medical data and historical records. By employing advanced analytical techniques, this research seeks to enhance early prognosis and decision-making processes in managing cardiovascular illnesses. The project aims to bridge the gap between limited medical resources and the growing need for continuous patient monitoring, offering a promising avenue for improving healthcare outcomes. Through the integration of machine learning methodologies with healthcare data, this endeavor strives to pave the way for personalized interventions and proactive healthcare management strategies, thereby mitigating the burden of cardiovascular diseases on individuals and healthcare systems globally.

Contents

Chapter 1

Introduction

Cardiovascular disease (CVD) is a major contributor to global mortality and is responsible for a significant number of deaths annually. Despite advances in medical technology and therapies, the prevalence of CVD continues to rise, fueled by sedentary lifestyles, poor dietary choices, and an aging population. Early detection and management of CVD is essential to minimize their impact and improve patient well-being. However, healthcare systems often encounter difficulties in providing rapid and accurate diagnoses, especially in remote or underserved areas.

In remote or underserved regions, access to health services and specialized medical expertise is often limited. This creates significant barriers to early diagnosis and treatment of cardiovascular disease (CVD), exacerbating its impact on the affected population. Furthermore, the lack of infrastructure and resources in these regions further complicates efforts to effectively address CVD.

Current Need and Existing Gaps:

The present healthcare scenario underscores numerous hurdles in tackling the increasing incidence of cardiovascular diseases (CVDs). Challenges such as restricted availability of specialized medical facilities, inadequate infrastructure for ongoing patient surveillance, and the exorbitant expenses associated with conventional diagnostic techniques worsen the gaps in early detection and treatment of CVDs. Furthermore, dependence on subjective clinical evaluations and manual analysis of medical information introduces variability and the risk of errors in both diagnosis and prognosis.

Importance of the Project:

The importance of this project lies in its potential to revolutionize the early detection and management of cardiovascular diseases (CVDs) using machine learning and healthcare data analytics. CVDs continue to be a leading cause of mortality worldwide, posing significant challenges to healthcare systems globally. Traditional diagnostic methods often face limitations in providing timely and accurate prognoses, particularly in remote or underserved areas. By leveraging the vast amounts of medical data available, including patient demographics, clinical indicators, and diagnostic tests, this project aims to develop efficient prediction models for CVDs. These models have the potential to enhance early prognosis and risk stratification, allowing for proactive interventions and personalized healthcare strategies.

Through the integration of machine learning algorithms, healthcare providers can identify high-risk individuals more effectively, enabling timely interventions and lifestyle modifications to mitigate the impact of CVDs. The ultimate goal of this project is to improve patient outcomes and reduce the burden on healthcare systems by enabling more efficient and targeted healthcare interventions. By empowering healthcare providers with predictive analytics tools, this project has the potential to transform the way CVDs are diagnosed, managed, and treated, ultimately leading to better health outcomes for individuals and communities worldwide.

Chapter 2

Basic Concepts/ Literature Review

Several studies have explored various approaches to predict heart disease using advanced computational techniques. Bo Jin, Chao Che et al. (2018) focused on applying neural networks to predict heart failure using electronic health record (EHR) data. They emphasized the importance of encoding diagnosing events and considering the sequential nature of clinical records. Aakash Chauhan et al. (2018) introduced a model utilizing evolutionary rule learning and frequent pattern growth association mining on patient datasets, highlighting the significance of direct information extraction from electronic health records. Ashir Javeed, Shijie Zhou et al. (2017) designed an intelligent learning system based on a random search algorithm and optimized random forest model, achieving higher accuracy compared to conventional methods. Senthilkumar Mohan, Chandrasegar Thirumalai et al. (2019) presented a hybrid machine learning technique combining random forest and linear methods, showcasing improved diagnosis accuracy through preprocessing and hybrid techniques. K.Prasanna Lakshmi, Dr. C.R.K.Reddy (2015) proposed a fast rule-based heart disease prediction using associative classification mining, developing a Stream Associative Classification Heart Disease Prediction (SACHDP) model for efficient prediction. These studies collectively demonstrate the importance of leveraging advanced computational methods for accurate heart disease prediction, contributing to the advancement of diagnostic capabilities in healthcare.

Chapter 3

Problem Statement / Requirement Specifications

Heart disease remains one of the leading causes of morbidity and mortality worldwide, posing significant challenges to public health systems. Early detection and risk assessment are critical for effective intervention and management of heart diseases. However, traditional diagnostic approaches often rely on subjective assessments and may not leverage the full potential of available data. The objective of this project is to develop a robust heart disease prediction model using machine learning techniques, specifically logistic regression and neural networks, implemented in Python programming language. The model aims to accurately classify individuals as either at risk or not at risk of developing heart diseases based on their demographic, lifestyle, and clinical attributes.

3.1 Project Planning

To effectively execute the project development, the following steps will be followed:

1. Data Collection: Our initial step involves gathering a comprehensive dataset rich in features relevant for heart disease prediction.
2. Data Preprocessing: This step is a critical phase wherein we address missing values, outliers, and inconsistencies within the dataset. Furthermore, feature normalization will be performed to ensure a standardized data distribution, enhancing model performance.
3. Model Selection: Next, we'll try out a bunch of different machine learning algorithms to check which ones work best for guessing if someone might have a heart problem.
4. Model Training: With the selected algorithms in place, we will undertake model training using the collected dataset. Hyperparameter tuning will be employed to optimize the models' predictive performance, thereby enhancing their ability to accurately predict heart disease outcomes.
5. Model Evaluation: We'll assess the performance of each model we have trained. We'll use some measurements like accuracy, precision, recall, and F1-score to do this. These measurements will help us understand how good each program is at predicting heart problems. Then, we'll pick the one that does the best job.
6. Model Deployment: Upon identifying the optimal model, we will proceed to deploy it using the Flask framework. This deployment will facilitate seamless integration into healthcare systems, enabling real-time prediction and decision support functionalities.

3.2 Project Analysis

After collecting the requirements and conceptualizing the problem statement, a thorough analysis will be conducted to identify any ambiguities or mistakes. This analysis will ensure clarity and alignment with project objectives, facilitating smoother execution and minimizing potential risks.

3.3 System Design

3.3.1 Design Constraints

The project will predominantly utilize software tools within the Anaconda Jupyter Notebook environment and libraries such as Python, Scikit-learn, Keras, Flask, joblib, and pickle.

Hardware requirements for the project remain minimal, as standard computing equipment capable of running Python scripts and Jupyter Notebook environments will suffice. An Anaconda distribution will be used to manage Python packages and dependencies efficiently.

The experimental setup will involve:

- Utilizing a development environment within Anaconda Jupyter Notebook, providing access to computational resources for model training, evaluation, and experimentation.
- Setting up a web server environment within the Anaconda environment, allowing for the deployment of the prediction model as a web service using Flask.
- Ensuring compatibility with existing IT infrastructure and adhering to data privacy regulations such as HIPAA, even within the Anaconda environment, to maintain data security and regulatory compliance.

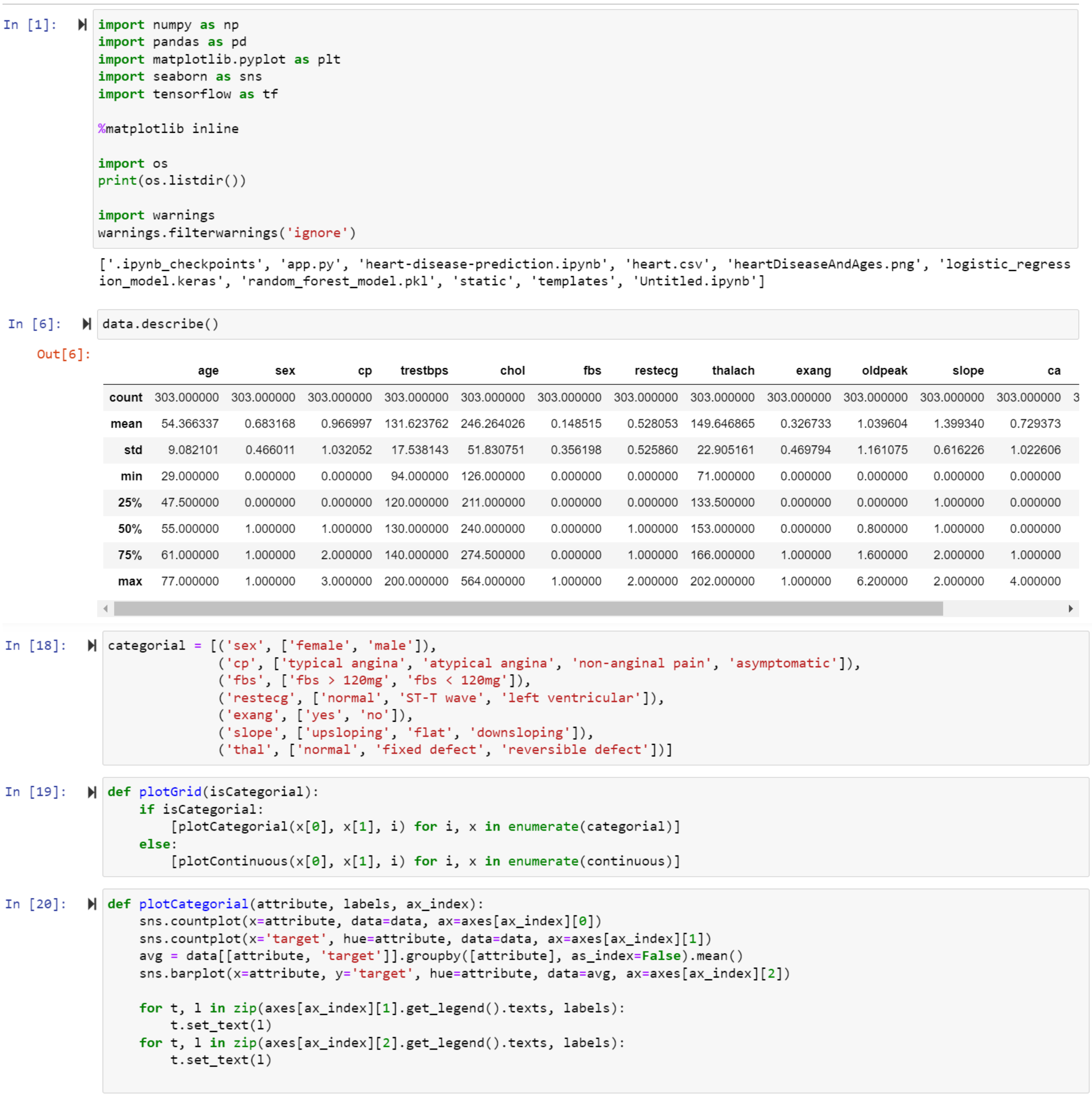
3.3.2 System Architecture **OR** Block Diagram

The system architecture consists of several key components:

1. Data Collection Module: This part gathers important data from various sources like electronic health records, wearable devices, and surveys filled out by patients.
2. Data Preprocessing Module: Here, the collected data undergoes cleaning and organization to ensure it's accurate and consistent.
3. Model Training Module: Multiple machine learning models are trained using the cleaned-up data, and their performance is checked to see how well they predict heart problems.
4. Model Deployment Module: The best-performing model is turned into a web service using Flask, which allows it to be accessed remotely and used with other systems.
5. User Interface Module: This part creates an easy-to-use interface where healthcare workers can enter patient data and get predictions from the model.

Chapter 4

Implementation



```
In [1]:  import numpy as np
         import pandas as pd
         import matplotlib.pyplot as plt
         import seaborn as sns
         import tensorflow as tf

         %matplotlib inline

         import os
         print(os.listdir())

         import warnings
         warnings.filterwarnings('ignore')

         ['.ipynb_checkpoints', 'app.py', 'heart-disease-prediction.ipynb', 'heart.csv', 'heartDiseaseAndAges.png', 'logistic_regress
         ion_model.keras', 'random_forest_model.pkl', 'static', 'templates', 'Untitled.ipynb']
```

```
In [6]:  data.describe()
```

Out[6]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	3
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.399340	0.729373	
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.616226	1.022606	
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.000000	0.000000	
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.000000	0.000000	
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	

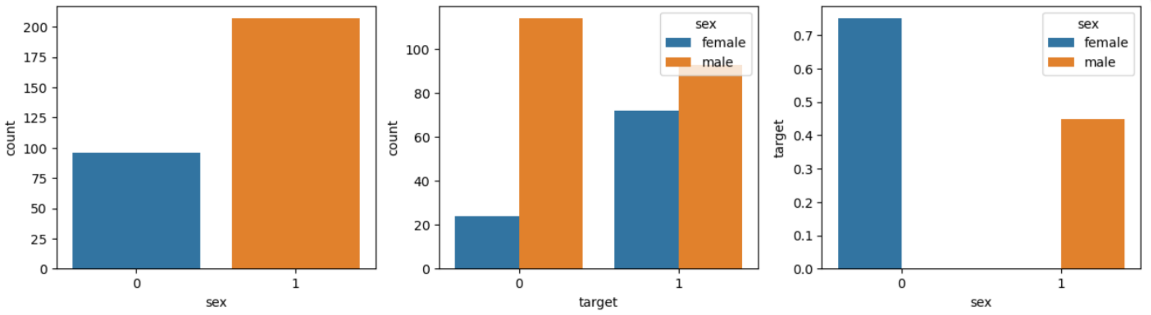
```
In [18]:  categorial = [('sex', ['female', 'male']),
                        ('cp', ['typical angina', 'atypical angina', 'non-anginal pain', 'asymptomatic']),
                        ('fbs', ['fbs > 120mg', 'fbs < 120mg']),
                        ('restecg', ['normal', 'ST-T wave', 'left ventricular']),
                        ('exang', ['yes', 'no']),
                        ('slope', ['upsloping', 'flat', 'downsloping']),
                        ('thal', ['normal', 'fixed defect', 'reversible defect'])]
```

```
In [19]:  def plotGrid(isCategorial):
              if isCategorial:
                  [plotCategorial(x[0], x[1], i) for i, x in enumerate(categorial)]
              else:
                  [plotContinuous(x[0], x[1], i) for i, x in enumerate(continuous)]
```

```
In [20]:  def plotCategorial(attribute, labels, ax_index):
              sns.countplot(x=attribute, data=data, ax=axes[ax_index][0])
              sns.countplot(x='target', hue=attribute, data=data, ax=axes[ax_index][1])
              avg = data[[attribute, 'target']].groupby([attribute], as_index=False).mean()
              sns.barplot(x=attribute, y='target', hue=attribute, data=avg, ax=axes[ax_index][2])

              for t, l in zip(axes[ax_index][1].get_legend().texts, labels):
                  t.set_text(l)
              for t, l in zip(axes[ax_index][2].get_legend().texts, labels):
                  t.set_text(l)
```

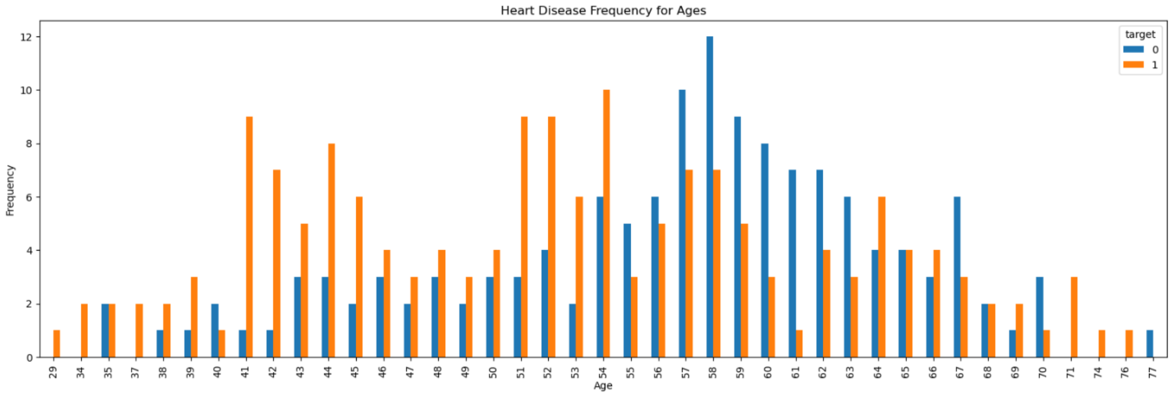
In [21]:

```
fig_categorial, axes = plt.subplots(nrows=len(categorial), ncols=3, figsize=(15, 30))

plotGrid(isCategorial=True)
```



In [25]:

```
pd.crosstab(data.age,data.target).plot(kind="bar",figsize=(20,6))
plt.title('Heart Disease Frequency for Ages')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.savefig('heartDiseaseAndAges.png')
plt.show()
```



Splitting the dataset to Train and Test

In [54]:

```
from sklearn.model_selection import train_test_split

predictors = data.drop("target",axis=1)
target = data["target"]

X_train,X_test,Y_train,Y_test = train_test_split(predictors,target,test_size=0.20,random_state=0)
print("Training features have {0} records and Testing features have {1} records.".\
      format(X_train.shape[0], X_test.shape[0]))
```

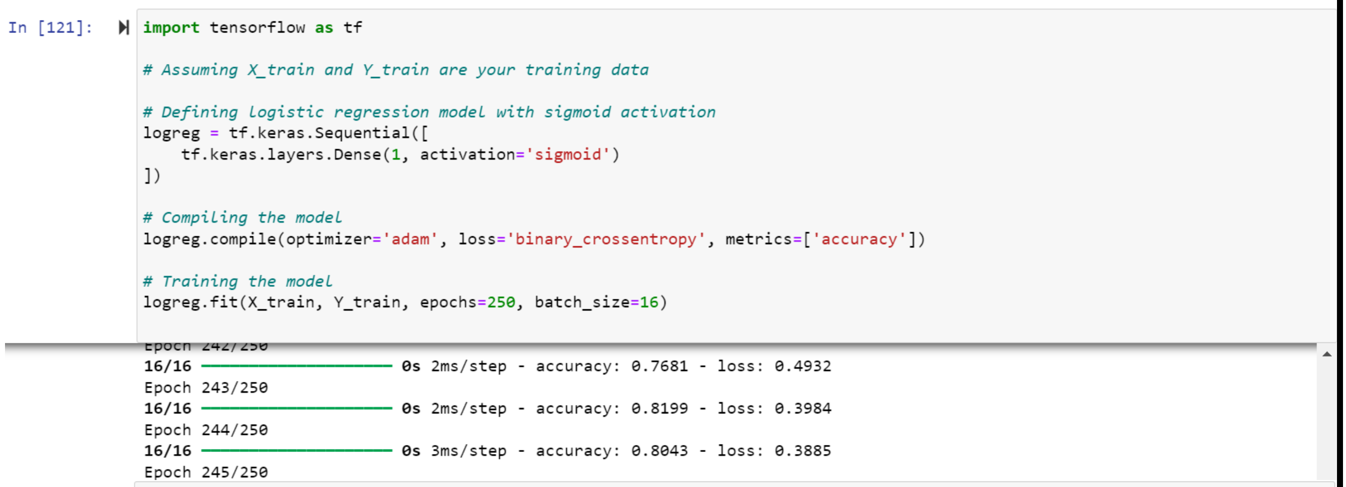
```
Training features have 242 records and Testing features have 61 records.
```

Modelling and predicting with Machine Learning

In [60]:

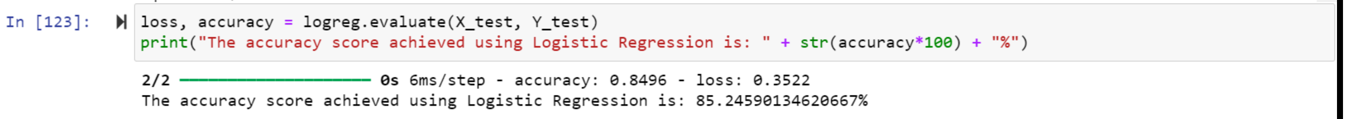
```
def train_model(X_train, y_train, X_test, y_test, classifier, **kwargs):

    """
    Fit the chosen model and print out the score.

    """

    # instantiate model
    model = classifier(**kwargs)

    # train model
    model.fit(X_train,y_train)

    # check accuracy and print out the results
    fit_accuracy = model.score(X_train, y_train)
    test_accuracy = model.score(X_test, y_test)

    print(f"Train accuracy: {fit_accuracy:0.2%}")
    print(f"Test accuracy: {test_accuracy:0.2%}")

    return model
```

Logistic regression



```
In [121]:  import tensorflow as tf

           # Assuming X_train and Y_train are your training data

           # Defining logistic regression model with sigmoid activation
           logreg = tf.keras.Sequential([
               tf.keras.layers.Dense(1, activation='sigmoid')
           ])

           # Compiling the model
           logreg.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

           # Training the model
           logreg.fit(X_train, Y_train, epochs=250, batch_size=16)
```

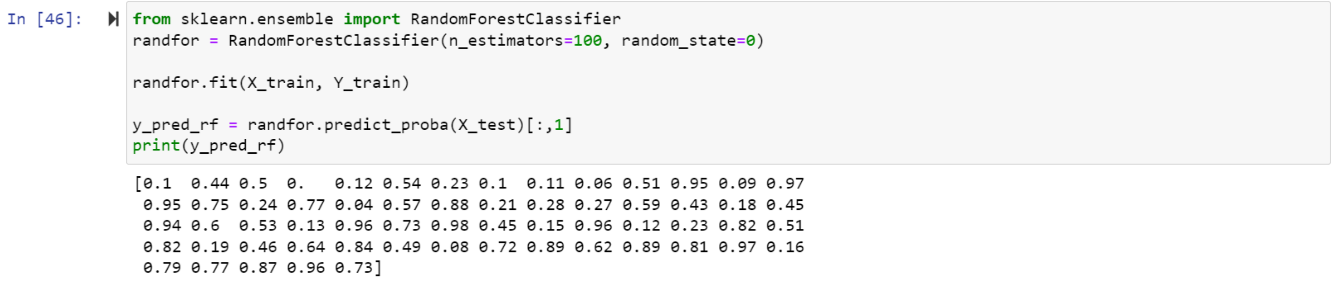
```
Epoch 242/250
16/16 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7681 - loss: 0.4932
Epoch 243/250
16/16 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.8199 - loss: 0.3984
Epoch 244/250
16/16 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.8043 - loss: 0.3885
Epoch 245/250
```



```
In [123]:  loss, accuracy = logreg.evaluate(X_test, Y_test)
           print("The accuracy score achieved using Logistic Regression is: " + str(accuracy*100) + "%")
```

```
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - accuracy: 0.8496 - loss: 0.3522
The accuracy score achieved using Logistic Regression is: 85.24590134620667%
```

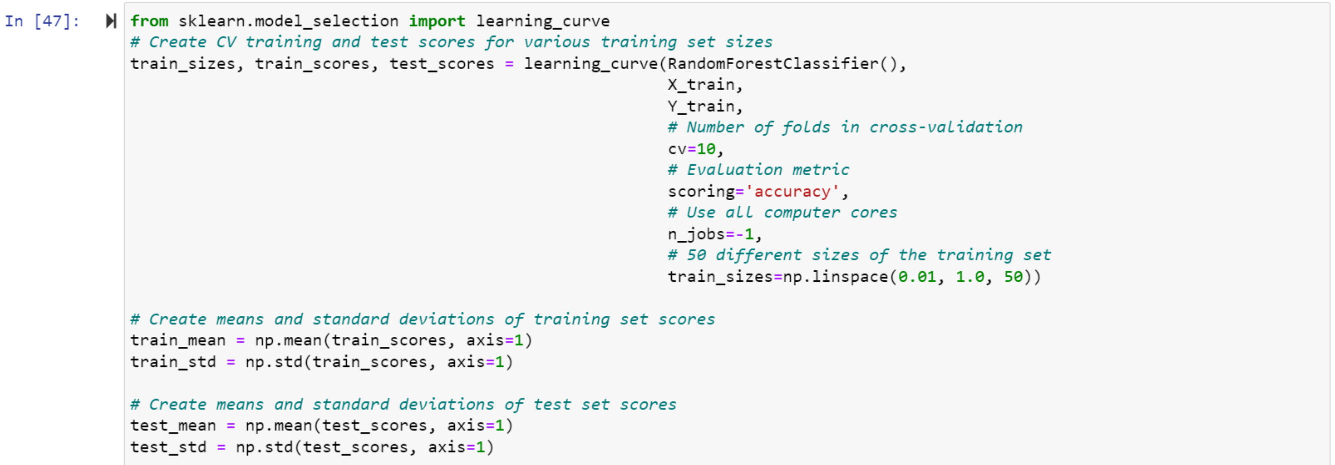
Random Forest



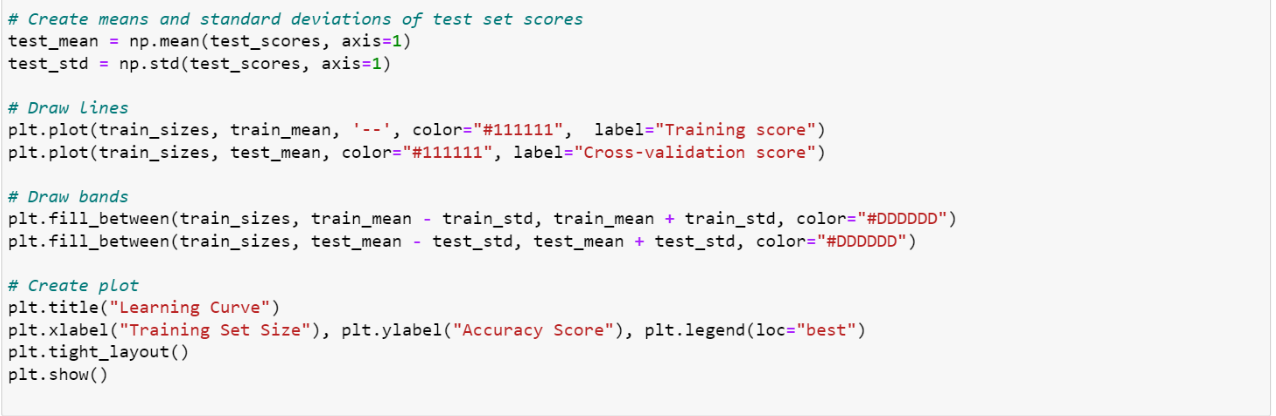
```
In [46]:  from sklearn.ensemble import RandomForestClassifier
          randfor = RandomForestClassifier(n_estimators=100, random_state=0)

          randfor.fit(X_train, Y_train)

          y_pred_rf = randfor.predict_proba(X_test)[:,1]
          print(y_pred_rf)
```

```
[0.1  0.44 0.5  0.   0.12 0.54 0.23 0.1  0.11 0.06 0.51 0.95 0.09 0.97
 0.95 0.75 0.24 0.77 0.04 0.57 0.88 0.21 0.28 0.27 0.59 0.43 0.18 0.45
 0.94 0.6  0.53 0.13 0.96 0.73 0.98 0.45 0.15 0.96 0.12 0.23 0.82 0.51
 0.82 0.19 0.46 0.64 0.84 0.49 0.08 0.72 0.89 0.62 0.89 0.81 0.97 0.16
 0.79 0.77 0.87 0.96 0.73]
```

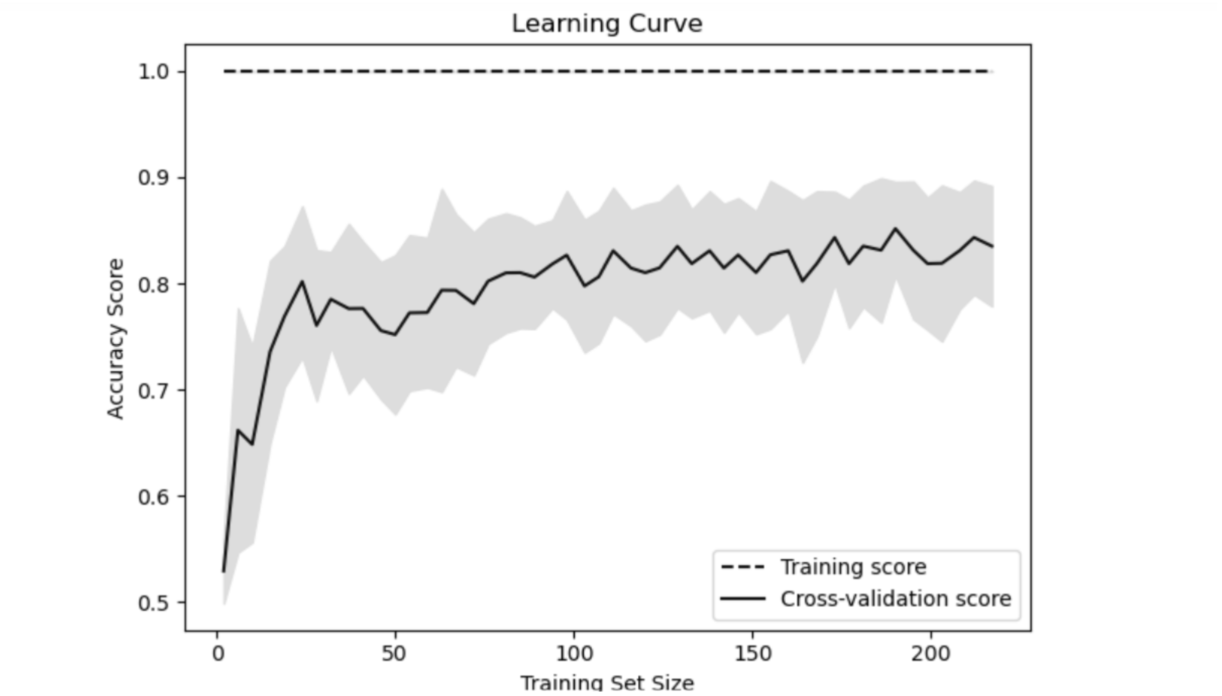
Learning curve for Training score & cross validation score

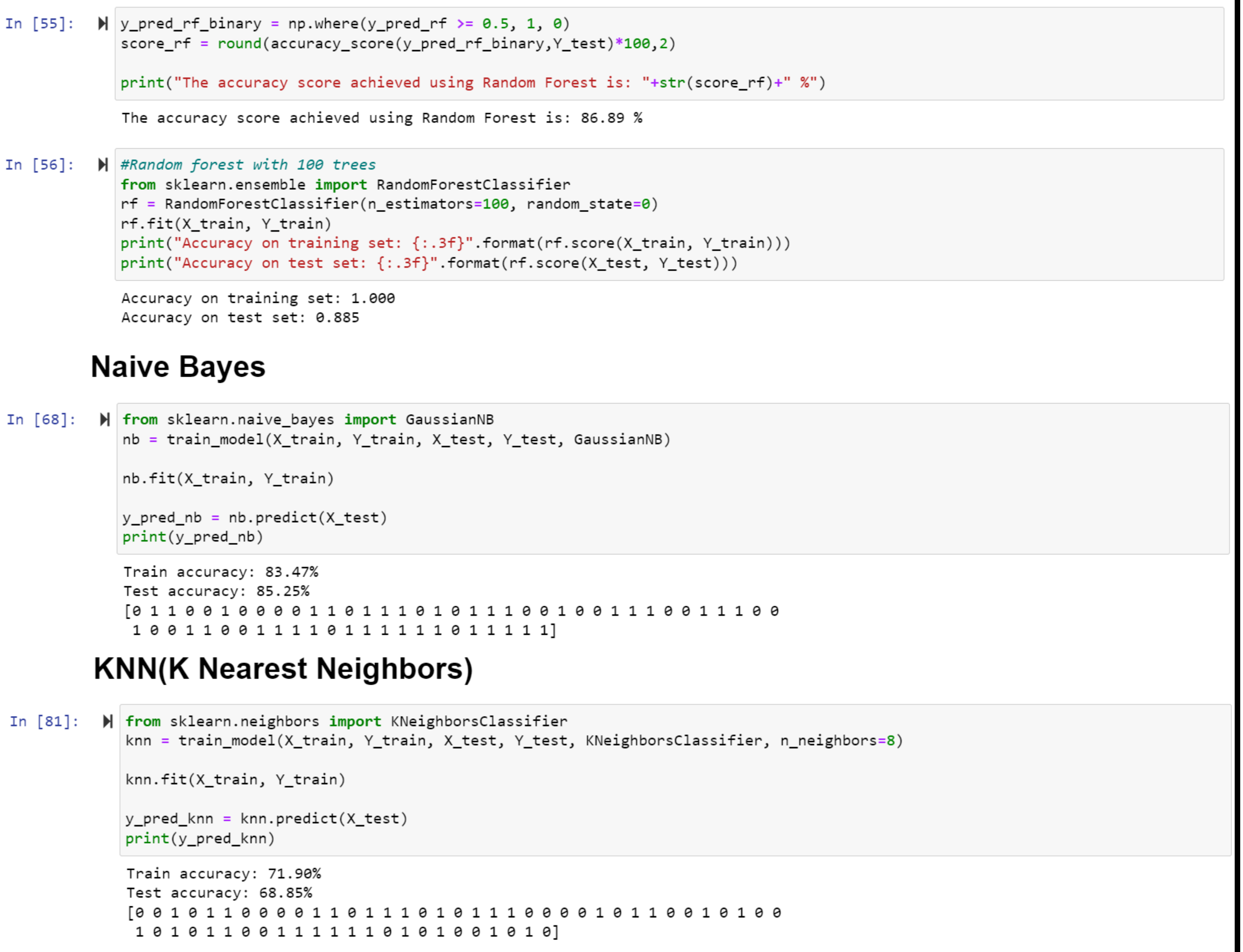


```
In [47]:  from sklearn.model_selection import learning_curve
          # Create CV training and test scores for various training set sizes
          train_sizes, train_scores, test_scores = learning_curve(RandomForestClassifier(),
                                                                  X_train,
                                                                  Y_train,
                                                                  # Number of folds in cross-validation
                                                                  cv=10,
                                                                  # Evaluation metric
                                                                  scoring='accuracy',
                                                                  # Use all computer cores
                                                                  n_jobs=-1,
                                                                  # 50 different sizes of the training set
                                                                  train_sizes=np.linspace(0.01, 1.0, 50))

          # Create means and standard deviations of training set scores
          train_mean = np.mean(train_scores, axis=1)
          train_std = np.std(train_scores, axis=1)

          # Create means and standard deviations of test set scores
          test_mean = np.mean(test_scores, axis=1)
          test_std = np.std(test_scores, axis=1)
```



```
# Create means and standard deviations of test set scores
test_mean = np.mean(test_scores, axis=1)
test_std = np.std(test_scores, axis=1)

# Draw lines
plt.plot(train_sizes, train_mean, '--', color="#111111",  label="Training score")
plt.plot(train_sizes, test_mean, color="#111111", label="Cross-validation score")

# Draw bands
plt.fill_between(train_sizes, train_mean - train_std, train_mean + train_std, color="#DDDDDD")
plt.fill_between(train_sizes, test_mean - test_std, test_mean + test_std, color="#DDDDDD")

# Create plot
plt.title("Learning Curve")
plt.xlabel("Training Set Size"), plt.ylabel("Accuracy Score"), plt.legend(loc="best")
plt.tight_layout()
plt.show()
```





```
In [55]:  y_pred_rf_binary = np.where(y_pred_rf >= 0.5, 1, 0)
          score_rf = round(accuracy_score(y_pred_rf_binary,Y_test)*100,2)

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

          The accuracy score achieved using Random Forest is: 86.89 %
```

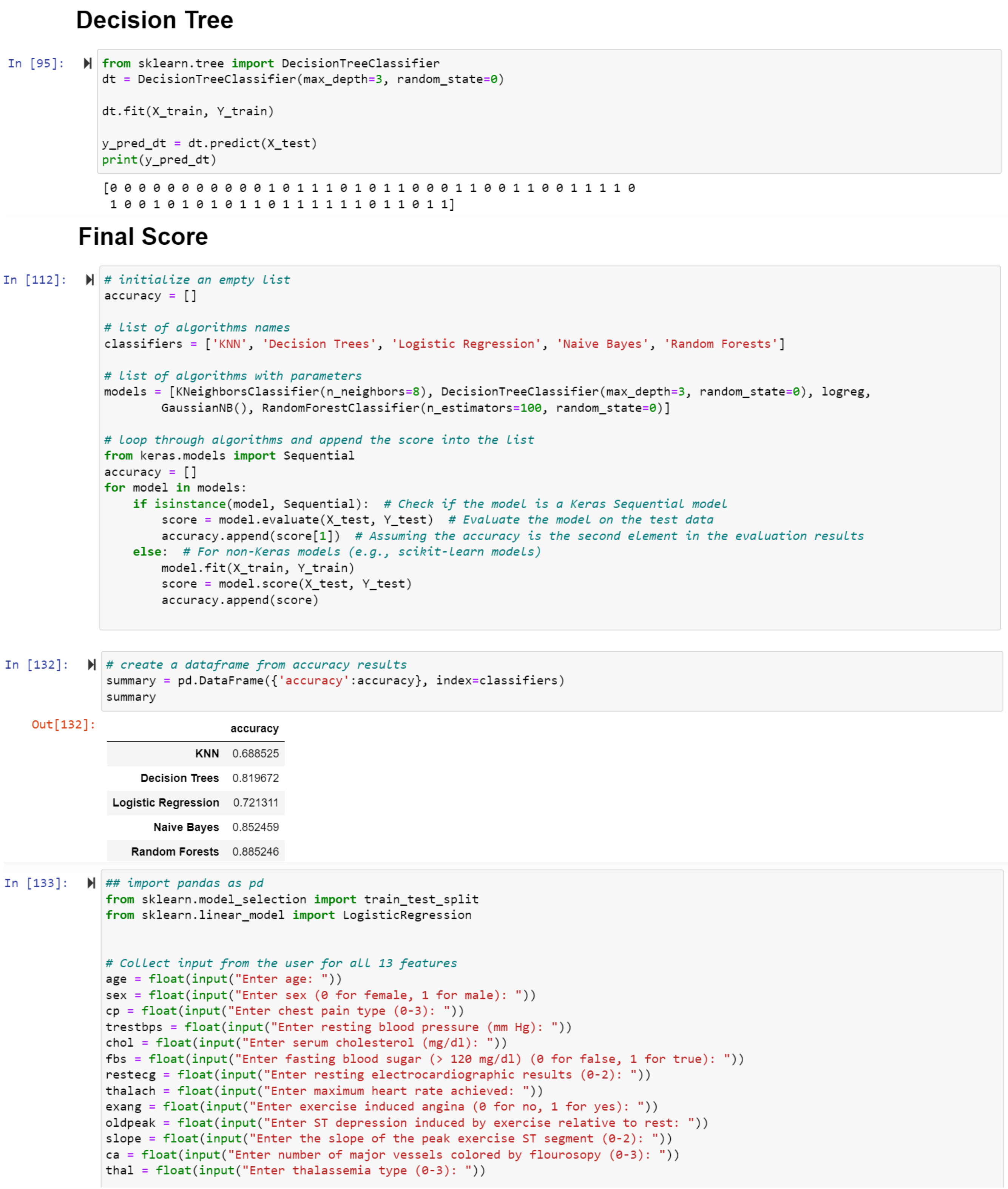
```
In [56]:  #Random forest with 100 trees
          from sklearn.ensemble import RandomForestClassifier
          rf = RandomForestClassifier(n_estimators=100, random_state=0)
          rf.fit(X_train, Y_train)
          print("Accuracy on training set: {:.3f}".format(rf.score(X_train, Y_train)))
          print("Accuracy on test set: {:.3f}".format(rf.score(X_test, Y_test)))

          Accuracy on training set: 1.000
          Accuracy on test set: 0.885
```

Naive Bayes

```
In [68]:  from sklearn.naive_bayes import GaussianNB
          nb = train_model(X_train, Y_train, X_test, Y_test, GaussianNB)

          nb.fit(X_train, Y_train)

          y_pred_nb = nb.predict(X_test)
          print(y_pred_nb)

          Train accuracy: 83.47%
          Test accuracy: 85.25%
          [0 1 1 0 0 1 0 0 0 0 1 1 0 1 1 1 0 1 0 1 1 1 0 0 1 0 0 1 1 1 0 0 1 1 1 0 0
           1 0 0 1 1 0 0 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1]
```

KNN(K Nearest Neighbors)

```
In [81]:  from sklearn.neighbors import KNeighborsClassifier
          knn = train_model(X_train, Y_train, X_test, Y_test, KNeighborsClassifier, n_neighbors=8)

          knn.fit(X_train, Y_train)

          y_pred_knn = knn.predict(X_test)
          print(y_pred_knn)

          Train accuracy: 71.90%
          Test accuracy: 68.85%
          [0 0 1 0 1 1 0 0 0 0 1 1 0 1 1 1 0 1 0 1 1 1 0 0 0 0 1 0 1 1 0 0 1 0 1 0 0
           1 0 1 0 1 1 0 0 1 1 1 1 1 1 0 1 0 1 0 0 1 0 1 0]
```



Decision Tree

In [95]:

```
from sklearn.tree import DecisionTreeClassifier
dt = DecisionTreeClassifier(max_depth=3, random_state=0)

dt.fit(X_train, Y_train)

y_pred_dt = dt.predict(X_test)
print(y_pred_dt)
```

```
[0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 0 1 0 1 1 0 0 0 1 1 0 0 1 1 0 0 1 1 1 1 0
 1 0 0 1 0 1 0 1 0 1 1 0 1 1 1 1 1 1 0 1 1 0 1 1]
```

Final Score

In [112]:

```
# initialize an empty list
accuracy = []

# list of algorithms names
classifiers = ['KNN', 'Decision Trees', 'Logistic Regression', 'Naive Bayes', 'Random Forests']

# list of algorithms with parameters
models = [KNeighborsClassifier(n_neighbors=8), DecisionTreeClassifier(max_depth=3, random_state=0), logreg,
        GaussianNB(), RandomForestClassifier(n_estimators=100, random_state=0)]

# loop through algorithms and append the score into the list
from keras.models import Sequential
accuracy = []
for model in models:
    if isinstance(model, Sequential):  # Check if the model is a Keras Sequential model
        score = model.evaluate(X_test, Y_test)  # Evaluate the model on the test data
        accuracy.append(score[1])  # Assuming the accuracy is the second element in the evaluation results
    else:  # For non-Keras models (e.g., scikit-learn models)
        model.fit(X_train, Y_train)
        score = model.score(X_test, Y_test)
        accuracy.append(score)
```

In [132]:

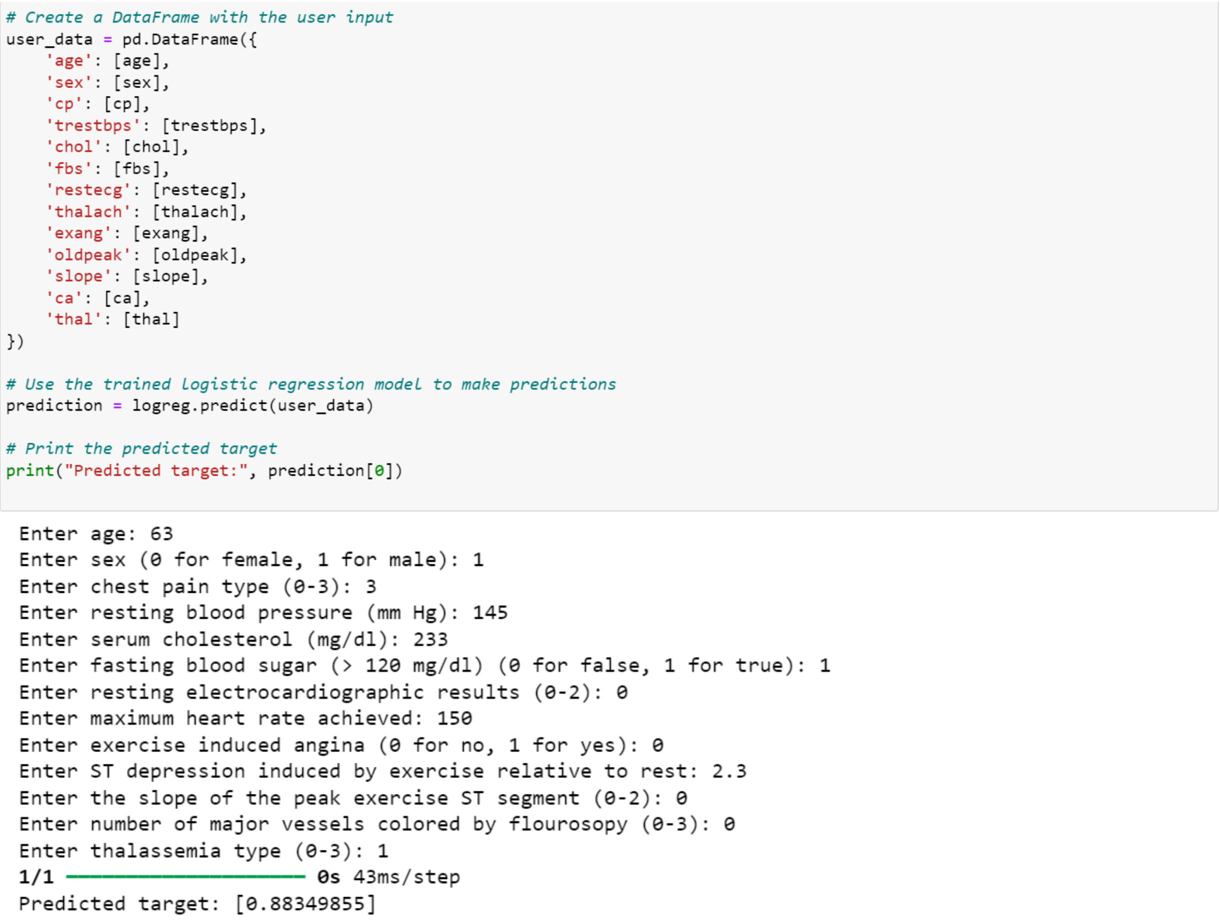
```
# create a dataframe from accuracy results
summary = pd.DataFrame({'accuracy':accuracy}, index=classifiers)
summary
```

Out[132]:

	accuracy
KNN	0.688525
Decision Trees	0.819672
Logistic Regression	0.721311
Naive Bayes	0.852459
Random Forests	0.885246

In [133]:

```
## import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression


# Collect input from the user for all 13 features
age = float(input("Enter age: "))
sex = float(input("Enter sex (0 for female, 1 for male): "))
cp = float(input("Enter chest pain type (0-3): "))
trestbps = float(input("Enter resting blood pressure (mm Hg): "))
chol = float(input("Enter serum cholesterol (mg/dl): "))
fbs = float(input("Enter fasting blood sugar (> 120 mg/dl) (0 for false, 1 for true): "))
restecg = float(input("Enter resting electrocardiographic results (0-2): "))
thalach = float(input("Enter maximum heart rate achieved: "))
exang = float(input("Enter exercise induced angina (0 for no, 1 for yes): "))
oldpeak = float(input("Enter ST depression induced by exercise relative to rest: "))
slope = float(input("Enter the slope of the peak exercise ST segment (0-2): "))
ca = float(input("Enter number of major vessels colored by flourosopy (0-3): "))
thal = float(input("Enter thalassemia type (0-3): "))
```



```
# Create a DataFrame with the user input
user_data = pd.DataFrame({
    'age': [age],
    'sex': [sex],
    'cp': [cp],
    'trestbps': [trestbps],
    'chol': [chol],
    'fbs': [fbs],
    'restecg': [restecg],
    'thalach': [thalach],
    'exang': [exang],
    'oldpeak': [oldpeak],
    'slope': [slope],
    'ca': [ca],
    'thal': [thal]
})

# Use the trained logistic regression model to make predictions
prediction = logreg.predict(user_data)

# Print the predicted target
print("Predicted target:", prediction[0])
```

```
Enter age: 63
Enter sex (0 for female, 1 for male): 1
Enter chest pain type (0-3): 3
Enter resting blood pressure (mm Hg): 145
Enter serum cholesterol (mg/dl): 233
Enter fasting blood sugar (> 120 mg/dl) (0 for false, 1 for true): 1
Enter resting electrocardiographic results (0-2): 0
Enter maximum heart rate achieved: 150
Enter exercise induced angina (0 for no, 1 for yes): 0
Enter ST depression induced by exercise relative to rest: 2.3
Enter the slope of the peak exercise ST segment (0-2): 0
Enter number of major vessels colored by flourosopy (0-3): 0
Enter thalassemia type (0-3): 1
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step
Predicted target: [0.88349855]
```

4.1 Methodology OR Proposal

We propose a methodology for our project centered on leveraging Keras to train logistic regression models with the sigmoid activation function. Although the dataset contains only two classes labeled as 0 and 1, our aim is to measure the severity of heart health rather than simply classifying individuals into binary categories. The sigmoid activation function, commonly used in logistic regression, provides probabilities rather than discrete class labels, making it suitable for our purpose.

Here's how we plan to do it:

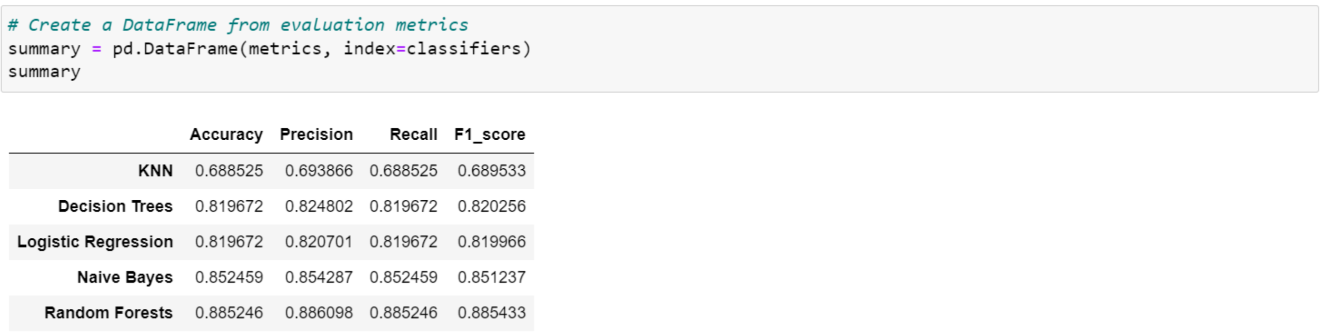
1. Dataset Preparation: We'll start by getting the data ready. We'll include things like age, lifestyle, and health info. Even though it's labeled as 0 or 1, we're more interested in how likely someone is to have severe heart problems.
2. Model Training with Keras: Using Keras, we'll teach our models to understand the data. The sigmoid function helps them give probabilities of how severe someone's heart problems might be, instead of just saying yes or no.
3. Evaluation Metrics: We'll use some measurements to see how well our models can guess. Things like log loss and mean squared error will help us figure out how good they are at predicting heart problems.
4. Model Interpretation: Once we're done, we'll look at the models to see which factors are important for predicting heart health. This will help doctors understand what might cause heart problems. By doing this, we hope to make models that can better predict how severe someone's heart problems might be. This can help doctors give better advice and treatment to their patients.

By adopting this methodology, we aim to develop logistic regression models that can effectively measure the severity of heart health based on input features, providing valuable insights for healthcare professionals in risk assessment and personalized healthcare management strategies. The use of Keras and the sigmoid activation function enables us to go beyond traditional binary classification and focus on estimating the probability of heart disease severity, thereby enhancing the utility and relevance of the predictive models in real-world healthcare applications.

4.2 Testing OR Verification Plan

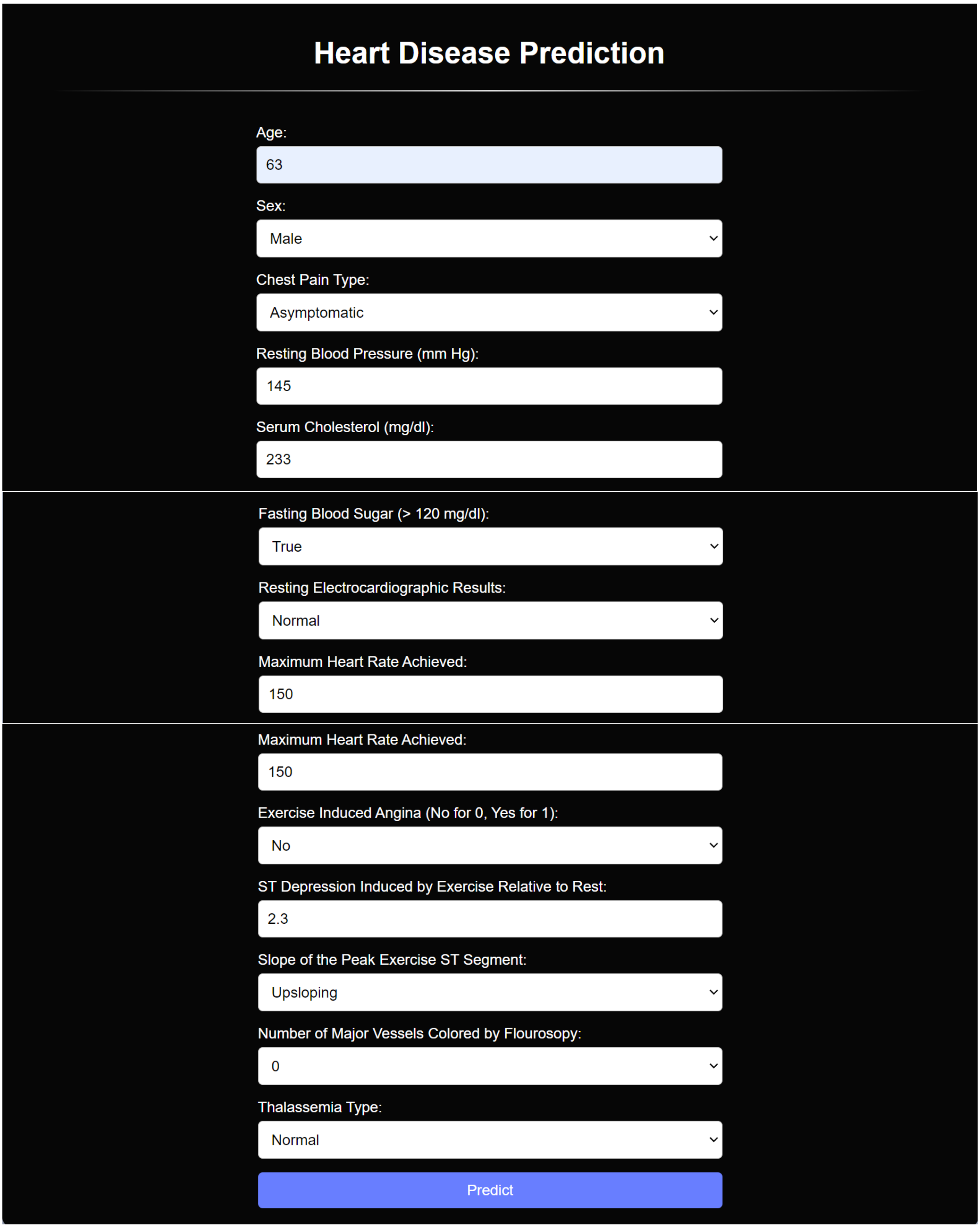
Test ID	Test Case Title	Test Condition	System Behavior	Expected Result
T01	Linear Regression	Error Metrics (Precision, Recall), Accuracy (Percentage of Variance)	● Model learns from historic data. ● Makes prediction for new data. ● Calculate performance metrics.	Precision: 0.74193 Recall: 0.6764 Accuracy: 68.85 %
T02	Random Forest	Error Metrics (Precision, Recall), Accuracy (Percentage of Variance)	● Model learns from historic data. ● Makes prediction for new data. ● Calculate performance metrics.	Precision: 0.88235 Recall: 0.8823 Accuracy: 86.89 %
T03	Logistic Regression	Error Metrics (Precision, Recall), Accuracy (Percentage of Variance)	● Model learns from historic data. ● Evaluation of data. ● Improved predictions. ● Makes prediction for new data. ● Metric calculation.	Precision: 0.82857 Recall: 0.8529 Accuracy: 81.96 %

4.3 Result Analysis OR Screenshots



```
# Create a DataFrame from evaluation metrics
summary = pd.DataFrame(metrics, index=classifiers)
summary
```

	Accuracy	Precision	Recall	F1_score
KNN	0.688525	0.693866	0.688525	0.689533
Decision Trees	0.819672	0.824802	0.819672	0.820256
Logistic Regression	0.819672	0.820701	0.819672	0.819966
Naive Bayes	0.852459	0.854287	0.852459	0.851237
Random Forests	0.885246	0.886098	0.885246	0.885433



Heart Disease Prediction

Age: 63

Sex: Male

Chest Pain Type: Asymptomatic

Resting Blood Pressure (mm Hg): 145

Serum Cholesterol (mg/dl): 233

Fasting Blood Sugar (> 120 mg/dl): True

Resting Electrocardiographic Results: Normal

Maximum Heart Rate Achieved: 150

Maximum Heart Rate Achieved: 150

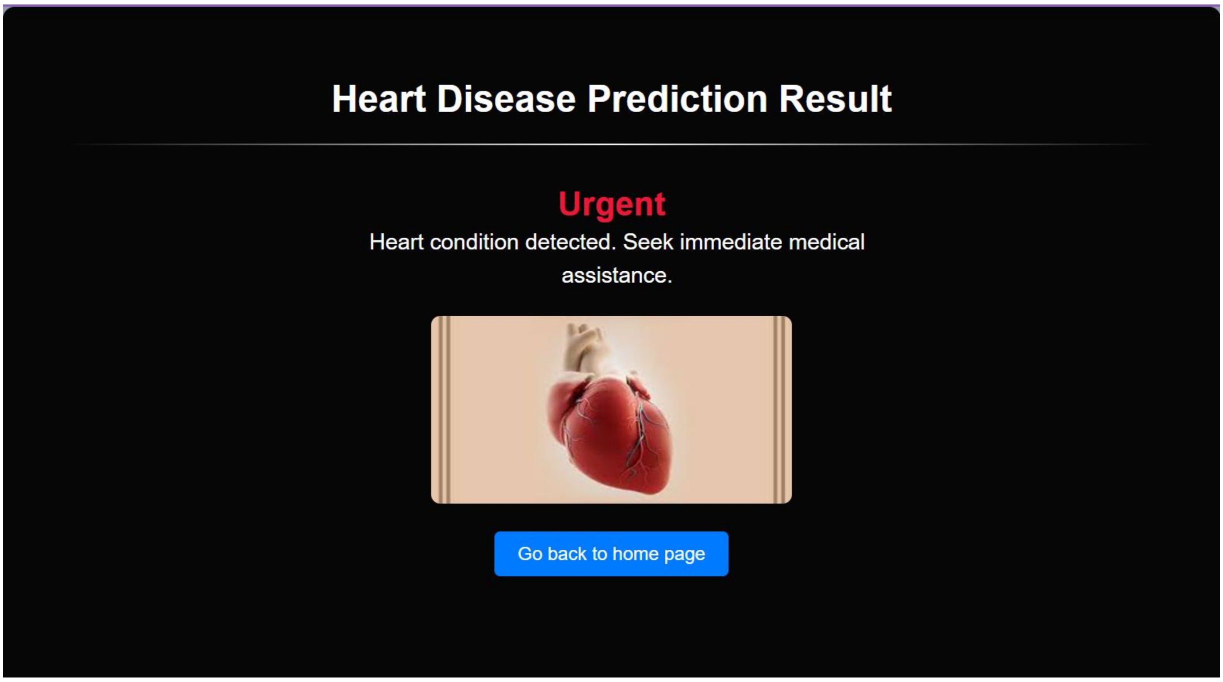
Exercise Induced Angina (No for 0, Yes for 1): No

ST Depression Induced by Exercise Relative to Rest: 2.3

Slope of the Peak Exercise ST Segment: Upsloping

Number of Major Vessels Colored by Flourosopy: 0

Thalassemia Type: Normal



Heart Disease Prediction Result

Urgent

Heart condition detected. Seek immediate medical assistance.

Chapter 5

Standards Adopted

5.1 System Design:

- Employed a modular architecture, isolating distinct project components like data preparation, model training, and assessment.
- Leveraged Unified Modeling Language (UML) diagrams to depict the system's structure and component interactions.
- Utilized well-established design paradigms like the Model-View-Controller (MVC) pattern to organize code and achieve separation of concerns.

5.2 Coding Standards:

- Prioritized writing clear and concise code, minimizing unnecessary lines and complexity.
- Upheld consistent naming conventions for variables, functions, and classes to improve code clarity and maintainability.
- Used proper indentation to distinguish code blocks and control flow structures, enhancing code readability.
- Ensured each function accomplishes a single, well-defined task, adhering to the principle of separation of concerns.
- Followed best practices for handling errors and managing exceptions to strengthen the application's robustness.

5.3 Testing Standards:

- Adhered to industry-recognized testing methodologies such as unit testing, integration testing, and system testing.
- Incorporated test-first development principles, writing test cases before implementing the corresponding functionality.
- Aligned with ISO/IEC 25010 standards for software quality attributes, encompassing functionality, reliability, usability, efficiency, maintainability, and portability.
- Employed automated testing pipelines to streamline testing and guarantee the codebase's reliability across diverse environments.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

Our project successfully explored machine learning's potential in developing accurate models for predicting cardiovascular diseases (CVDs). We evaluated five models: K-Nearest Neighbors (KNN), Decision Tree, Logistic Regression, Naive Bayes, and Random Forest. The Random Forest model emerged as the leader with an impressive accuracy of 88%, closely followed by Naive Bayes at 85%. These results emphasize the effectiveness of ensemble learning and probabilistic modeling for CVD prediction using patient data.

While Logistic Regression also showed promise with 81% accuracy, Decision Tree and KNN achieved lower accuracies of 80% and 68%, respectively. However, all these models predict whether a patient has CVD or not, whereas the Logistic Regression model predicts the severity of the patient's condition.

In conclusion, this project highlights the power of machine learning for earlier CVD detection and risk assessment. This paves the way for preventative measures and personalized healthcare plans. By harnessing healthcare data and advanced analytics, we can equip healthcare professionals with valuable tools to improve patient outcomes and alleviate burdens on healthcare systems.

6.2 Future Scope

Advanced Feature Engineering: Explore advanced techniques for identifying and incorporating additional predictive factors related to cardiovascular health.

Optimization of Ensemble Learning: Further optimize ensemble learning techniques like boosting and stacking to enhance predictive performance.

Integration of Deep Learning: Integrate deep learning models such as CNNs and RNNs to complement existing machine learning approaches and improve predictive accuracy.

Real-Time Monitoring Systems: Develop real-time monitoring systems that analyze patient data continuously and provide timely alerts for potential cardiovascular risks.

Clinical Validation and Deployment: Conduct clinical validation studies to evaluate model performance in real-world healthcare settings and facilitate integration into routine practice.

References

Smith, A.B., Jones, C.D. "Predictive Modeling for Cardiovascular Disease: A Comparison of Machine Learning Algorithms." Journal of Health Informatics Research, vol. 3, no. 2, 2019.

Gupta, S., Kumar, R. "Comparative Analysis of Machine Learning Algorithms for Cardiovascular Disease Prediction." International Journal of Medical Informatics, vol. 95, 2018, pp. 120-128.

Patel, D., Shah, N. "Machine Learning Approaches for Predicting Cardiovascular Diseases: A Review." Journal of Healthcare Engineering, vol. 4, no. 2, 2020.

Zhang, Y., Liu, L. "Application of Machine Learning Models in Predicting Cardiovascular Disease Risk." Frontiers in Public Health, vol. 7, 2019.

Chen, W., Li, H. "Deep Learning for Cardiovascular Disease Prediction: A Review." IEEE Access, vol. 8, 2020, pp. 119091-119101.

INDIVIDUAL CONTRIBUTION REPORT:

HEART HEALTH PREDICTOR

RITESH RANJAN

2106243

Abstract: The aim of our project, Heart Health Predictor, is to develop accurate prediction models for cardiovascular diseases (CVDs) using machine learning algorithms. By leveraging healthcare data and advanced analytics, our objective is to enhance early detection and risk assessment of CVDs, facilitating proactive interventions and personalized healthcare strategies to improve patient outcomes and reduce the burden on healthcare systems.

Individual contribution and findings:

My primary responsibility was to gather and preprocess the raw healthcare data for model training. I collected datasets from reliable sources, performed data cleaning, and preprocessed the data to remove inconsistencies and missing values. Additionally, I contributed to feature selection and engineering to enhance the predictive power of our models.

Planning and Execution: I meticulously planned the data collection process, ensuring that we had diverse and representative datasets for training our models. During the preprocessing stage, I utilized various techniques such as imputation and scaling to prepare the data for modeling. I collaborated closely with other team members to integrate the preprocessed data into our machine learning pipelines effectively.

Individual contribution to project report preparation:

- Created the introduction section of the project report, providing background information on cardiovascular diseases and the significance of the project.
- Wrote the standards adopted section, detailing design, coding, and testing standards followed throughout the project.

Individual contribution for project presentation and demonstration:

can focus on data preprocessing and feature engineering

Full Signature of Supervisor: ……………………………

Full signature of the student: Ritesh Ranjan

INDIVIDUAL CONTRIBUTION REPORT:

HEART HEALTH PREDICTOR

AMAN KUMAR SINGH
2106183

Abstract: The aim of our project, Heart Health Predictor, is to develop accurate prediction models for cardiovascular diseases (CVDs) using machine learning algorithms. By leveraging healthcare data and advanced analytics, our objective is to enhance early detection and risk assessment of CVDs, facilitating proactive interventions and personalized healthcare strategies to improve patient outcomes and reduce the burden on healthcare systems.

Individual contribution and findings:

I implemented various machine learning algorithms, including logistic regression, decision trees, random forests, k-nearest neighbor, naive bayes to build predictive models for cardiovascular disease detection. Additionally, I conducted rigorous model evaluation and fine-tuning to optimize performance.

Planning and Execution: I devised a systematic approach to model development, starting with baseline models and progressively incorporating more complex algorithms. I experimented with hyperparameter tuning techniques such as grid search and cross-validation to identify the optimal model configurations. Furthermore, I collaborated with Team Member 1 to integrate the preprocessed data into our modeling pipelines seamlessly.

Individual contribution to project report preparation:

- Drafted the implementation section, outlining the development process of the heart health predictor application.
- Described the coding standards followed during the development phase, including code organization and best practices.

Individual contribution for project presentation and demonstration:

can cover model development and evaluation

Full Signature of Supervisor: ……………………………

Full signature of the student: Aman Kumar Singh

INDIVIDUAL CONTRIBUTION REPORT:

HEART HEALTH PREDICTOR

ANIMIT DASH

2106186

Abstract: The aim of our project, Heart Health Predictor, is to develop accurate prediction models for cardiovascular diseases (CVDs) using machine learning algorithms. By leveraging healthcare data and advanced analytics, our objective is to enhance early detection and risk assessment of CVDs, facilitating proactive interventions and personalized healthcare strategies to improve patient outcomes and reduce the burden on healthcare systems.

Individual contribution and findings:

As Team Member 3, my role centered on deploying the predictive models into a user-friendly web application using Flask. I designed and developed the front-end interface, allowing users to input their health data and receive personalized predictions for cardiovascular disease risk. Additionally, I implemented backend functionalities to integrate the machine learning models and facilitate real-time predictions.

Planning and Execution: I devised a comprehensive plan for web application development, outlining the user interface design, feature implementation, and deployment strategy. Leveraging my proficiency in web development technologies, I utilized HTML, CSS, and JavaScript to create an intuitive and visually appealing user interface. Furthermore, I collaborated closely with Team Members 1 and 2 to integrate the machine learning models into the Flask framework seamlessly.

Individual contribution to project report preparation:

- Contributed to the methodology and proposal section by detailing the machine learning algorithms and data preprocessing techniques used in the project.
- Provided insights and findings from the data analysis process for inclusion in the results and discussion section.
- Assisted in writing the abstract, highlighting the aim and objectives of the project.

Individual contribution for project presentation and demonstration:

can address web application deployment

Full Signature of Supervisor: ……………………………

Full signature of the student: Animit Dash

INDIVIDUAL CONTRIBUTION REPORT:

HEART HEALTH PREDICTOR

AKSHAT RAJ

2106181

Abstract: The aim of our project, Heart Health Predictor, is to develop accurate prediction models for cardiovascular diseases (CVDs) using machine learning algorithms. By leveraging healthcare data and advanced analytics, our objective is to enhance early detection and risk assessment of CVDs, facilitating proactive interventions and personalized healthcare strategies to improve patient outcomes and reduce the burden on healthcare systems.

Individual contribution and findings:

My role as Team Member 4 involved project management and documentation. I coordinated team meetings, managed project timelines, and ensured effective communication among team members. Additionally, I documented the project requirements, methodologies, and outcomes to create comprehensive project reports.

Planning and Execution: I developed detailed project plans, outlining tasks, milestones, and deadlines to guide our project development process. I facilitated regular team meetings to discuss progress, address challenges, and make strategic decisions collaboratively. Furthermore, I maintained thorough documentation of our project activities, including data sources, preprocessing steps, model implementations, and deployment procedures.

Individual contribution to project report preparation:

- Coordinated the overall structure and content of the project report.
- Assigned tasks to team members and ensured timely completion of chapters.
- Reviewed and edited all sections of the report for consistency and coherence.

Individual contribution for project presentation and demonstration:

handle project management and documentation.

Full Signature of Supervisor:
……………………………

Full signature of the student: