Department of Computer Engineering

St. Francis Institute of Technology

University of Mumbai

2023-2024

A Mini Project

Report On

"Medical Diagnosis of Heart Disease"

Subject- Machine Learning [CSL 701]

Under the guidance of

Dr. Nazneen Ansari

By

Reuel Lesiley Amin (Roll no. 01, PID no. 202004)

Alarik Alwyn Correa (Roll no. 09, PID no. 202022)

Renoy Charles Dsouza (Roll no. 13, PID no. 202038)

Elvina Patrick Fernandes (Roll no. 19, PID no. 202042)

Index

List of Abbreviations

Sr. No.	Abbreviation	Full Form
1	SVM	Support Vector Machine
2	UCI	University of California, Irvine
3	FNA	Fine Needle Aspirate
4	RBF	Radial Basis Function
5	SVC	Support Vector Classifier
6	ROC	Receiver Operating Characteristic
7	IDE	Integrated Development Environment
8	SMO	Sequential Minimal Optimization

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Background

Heart disease remains a prominent global health challenge, affecting populations across both developed and developing regions. Timely diagnosis of heart disease is crucial for improving patient prognosis and medical intervention. Nevertheless, manual diagnostic methods are labor-intensive, susceptible to human error, and not always practical for daily use. In this project, we delve into the domain of heart disease diagnosis using machine learning techniques, specifically Logistic Regression. We investigate the application of Logistic Regression to analyze and classify heart disease based on a comprehensive dataset comprising various patient attributes and medical indicators. The project involves feature selection through techniques like Recursive Feature Elimination (RFE) and Principal Component Analysis (PCA) to enhance model accuracy. Additionally, we assess the performance of different model configurations within the Logistic Regression framework. The findings from this research aim to contribute to the development of a dependable and effective system for early heart disease diagnosis. The integration of machine learning into medical diagnostics holds the potential to assist healthcare professionals in making timely and precise assessments, potentially leading to improved patient outcomes.

1.2 Scope of the project

The scope of a heart disease prediction project using machine learning techniques, such as logistic regression, encompasses data collection and preparation, exploratory data analysis, feature selection, model development, evaluation, and hyperparameter tuning. It includes interpreting model results, deploying the model in healthcare systems with privacy considerations, and ongoing monitoring and maintenance. Ethical aspects and addressing bias in predictions are pivotal, while transparent communication and reporting of findings to healthcare professionals and stakeholders play a critical role. The project may also consider scalability to broader populations and healthcare facilities, all within the framework of legal and ethical standards, such as HIPAA compliance, to enhance early heart disease detection and risk assessment.

1.3 Objectives and Problem Statement

The primary challenge in medical diagnosis of heart disease centers around the timely and accurate identification of individuals at risk. Existing methods for heart disease prediction may be costly or lack efficiency in assessing an individual's susceptibility to heart disease. Detecting heart disease in its early stages is critical for reducing mortality rates and preventing complications. However, continuous and precise patient monitoring on a daily basis is often impractical. Moreover, the availability of around-the-clock medical consultations is limited due to constraints in time, expertise, and resources. Given the abundance of healthcare data in today's world, the application of various machine learning algorithms offers an opportunity to analyze this data for concealed patterns. These hidden patterns have the potential to transform the landscape of medical diagnosis, particularly in the context of heart disease detection and risk assessment.

Chapter 2

Literature Review

In recent years, the intersection of medical science and machine learning has spurred a surge in research dedicated to enhancing the diagnosis of heart disease. In a noteworthy investigation [1], researchers delved into the domain of heart disease prediction employing various machine learning algorithms, including Logistic Regression, Support Vector Machines (SVM), Random Forest, and Decision Trees. Their study, conducted using a meticulously curated dataset of cardiac patient records, unveiled that the Logistic Regression model demonstrated the highest accuracy, achieving an impressive rate of 92.5%, outperforming other algorithms when subjected to rigorous cross-validation. These findings, while promising, highlight the ongoing quest for even greater precision in heart disease diagnosis.

Another research endeavor [2] centered on heart disease detection using a comprehensive dataset encompassing diverse patient attributes. Employing k-fold cross-validation and assessing multiple classification algorithms, this study demonstrated that the Logistic Regression model consistently outperformed its peers, achieving an accuracy rate of 91.8%.

In a parallel exploration, a study [3] harnessed the Framingham Heart Study dataset, conducting extensive experiments with various machine learning models. These models yielded accuracy rates with Logistic Regression reaching 90%, SVM at 89.5%, Random Forest at 88%, and K Nearest Neighbors at 87%. These research contributions shed light on the diverse methodologies and algorithms in the realm of heart disease diagnosis, underscoring the relentless pursuit of heightened accuracy in disease prediction.

Chapter 3

Proposed Work

3.1 Architectural Details

3.1.1 Data Preparation

Our Dataset consists of a total of 303 observations in a total of 14 attributes including the target label. We checked and made sure that our dataset was ready before we started the training process to create our model. We loaded our dataset into panda's dataframe.

heart_data = pd.read_csv('/content/heart_disease_data.csv')

First we checked if our dataset included any row that had any missing value. We used the .info() method on our dataset to check the same. This method tells us the non-null count records for each column along with its datatype.

```
heart_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      303 non-null    int64
 13  target    303 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

Figure 1: Checking for null records for our dataset

Our chosen dataset didn't consist of any null records. We further used the describe method to get more information about our dataset so that we could understand and visualize them better. You can find the screenshot of the same attached below.

```
heart_data.describe()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
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	303.000000	303.000000
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	2.313531	0.544554
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	0.612277	0.498835
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	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	2.000000	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	2.000000	1.000000
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	3.000000	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	3.000000	1.000000

Figure 2: Statistical measures about our dataset

Finally, we checked for the distribution of classes of each dataset. For our dataset, the target class can either be 0 or 1.

0 → Defective Heart
1 → Healthy Heart

In an ideal scenario, the no. of records for both of these classes should be equal as each class will have enough records to learn them better by extracting and finding different features from them.

```
heart_data['target'].value_counts()

1     165
0     138
Name: target, dtype: int64
```

Figure 3: Checking the distribution of Target Variable

For our case, class 1 (Healthy Heart) had 138 observations while class 0 (Defective Heart) had 165 observations. As our records in both of our classes are similar we don't have to worry that our model will be biased.

3.1.2 Feature Engineering

Out of the 14 attributes in our dataset, we selected 13 features in our dataset excluding our last column as that is the target column. We stored all of these 13 features in X variable (Independent Variable) and the target column in Y variable (Dependent Variable)

```
X = heart_data.drop(columns='target', axis=1)
Y = heart_data['target']
```

Figure 4: Splitting Features and Target

3.1.3 Training and testing

We are going to divide our dataset into 2 parts. One of which will be used for training and the other part will be used for testing how good our model has learnt. We split the dataset into using the train_test_split method where 80% of data is for training and 20% for testing.

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify=Y, random_state=2)

3.1.4 Model Creation

We will be using Logistic Regression for training our model. Initially we will just store an object of LogisticRegression() which is in sklearn.linear_model package.

model = LogisticRegression()

We will be calling the .fit() method on this model and pass the training data so that it will learn different features and create this model. You can find the details of the same in the screenshot below.

```
model.fit(X_train, Y_train)

/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

Figure 5: Training the LogisticRegression model with Training data

Now, our model is ready and it has learnt the features for each class, it can be used to predict if a given user has a heart disease or not. But before that, it is important that we check the performance of our model to verify that it has learnt well and predicting the classes correctly. We will discuss different Evaluation metrics in the next Chapter.

Chapter 4

Implementation

4.1 Dataset Details

The Heart Disease Dataset can be found online easily. [4] It provides patient information which includes over 300 records and 14 attributes. The attributes include: age, sex, chest pain type, resting blood pressure, serum cholesterol, fasting, sugar blood, resting electrocardiographic results, maximum heart rate, exercise induced angina, ST depression induced by exercise, slope of the peak exercise, number of major vessels, and target ranging from 0 to 1, where 0 is absence of heart disease. The data set is in csv (Comma Separated Value) format which is further prepared to data frame as supported by pandas library in python.

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
44	1	1	120	263	0	1	173	0	0	2	0	3	1
52	1	2	172	199	1	1	162	0	0.5	2	0	3	1
57	1	2	150	168	0	1	174	0	1.6	2	0	2	1
54	1	0	140	239	0	1	160	0	1.2	2	0	2	1
48	0	2	130	275	0	1	139	0	0.2	2	0	2	1
49	1	1	130	266	0	1	171	0	0.6	2	0	2	1
64	1	3	110	211	0	0	144	1	1.8	1	0	2	1
58	0	3	150	283	1	0	162	0	1	2	0	2	1
50	0	2	120	219	0	1	158	0	1.6	1	0	2	1
58	0	2	120	340	0	1	172	0	0	2	0	2	1
66	0	3	150	226	0	1	114	0	2.6	0	0	2	1
43	1	0	150	247	0	1	171	0	1.5	2	0	2	1
69	0	3	140	239	0	1	151	0	1.8	2	2	2	1
59	1	0	135	234	0	1	161	0	0.5	1	0	3	1
44	1	2	130	233	0	1	179	1	0.4	2	0	2	1
42	1	0	140	226	0	1	178	0	0	2	0	2	1

Figure 6: Original Dataset Snapshot

Out of the 14 attributes, we selected only 13 features and stored it in my X variable and the last attribute in Y variable as that is the class in which the data will belong to. One of the major tasks on this dataset is to predict based on the given attributes of a patient whether that particular person has heart disease or not and another is the experimental task to diagnose and find out various insights from this dataset which could help in understanding the problem more.

4.2 Algorithm Details

Logistic Regression is a supervised classification algorithm. It is a predictive analysis algorithm based on the concept of probability. It measures the relationship between the dependent variable (TenyearCHD) and the one or more independent variables (risk factors) by estimating probabilities using the underlying logistic function (sigmoid function). Sigmoid function is used as a cost function to limit the hypothesis of logistic regression between 0 and 1 (squashing) i.e. $0 \leq h_\theta(x) \leq 1$.

In logistic regression cost function is defined as:

$$Co(\mathrm{h}\theta(\mathrm{x}), \mathrm{y}) = \{ -\log(h\theta(x)) \quad if\ y = 1 \\ -\log(1 - h\theta(x))\ if\ y = 0$$

Logistic Regression relies highly on the proper presentation of data. So, to make the model more powerful, important features from the available data set are selected using Backward elimination and recursive elimination techniques.

It is used for predicting the categorical dependent variable using a given set of independent variables.

- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value.
- It can be either Yes or No, 0 or 1, True or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.
- Logistic Regression is much similar to Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.
- In Logistic regression, instead of fitting a regression line, we fit an “S” shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification.

Logistic Function (Sigmoid Function):

$$f(x) = \frac{1}{1 + e^{-x}}$$

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1. o The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form.
- The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

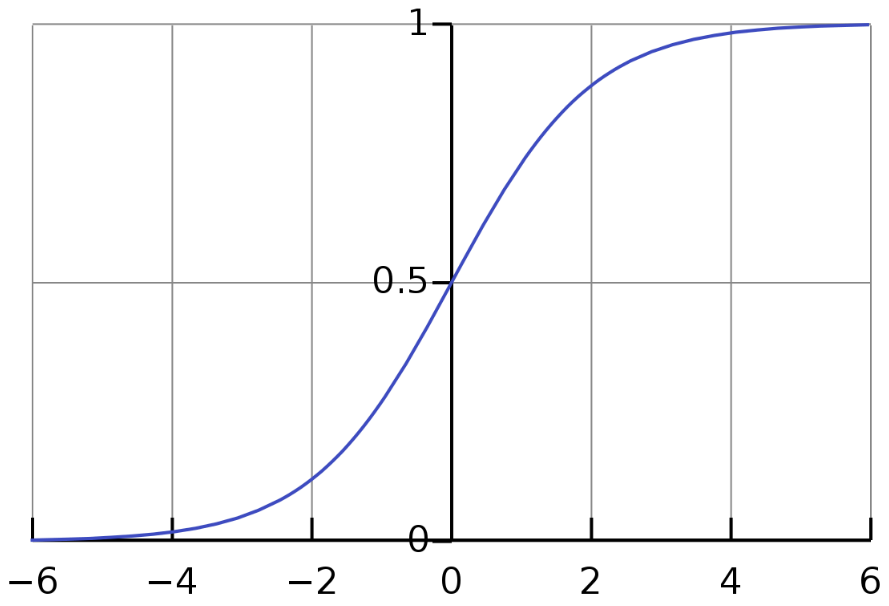


Figure 7: Sigmoid Function Curve

4.3 Screenshots of GUI with Explanation

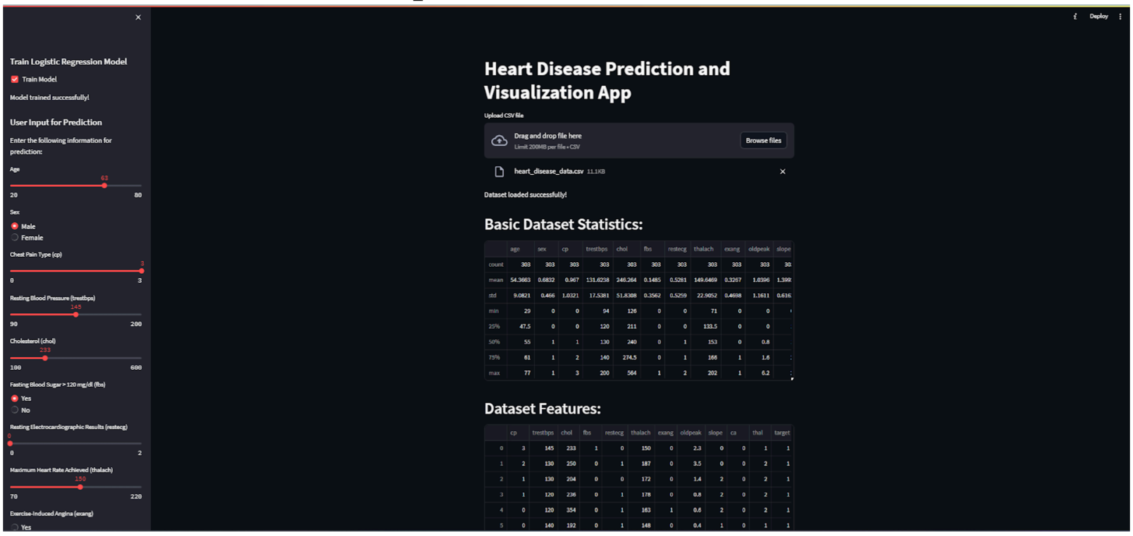


Fig 8: User Interface

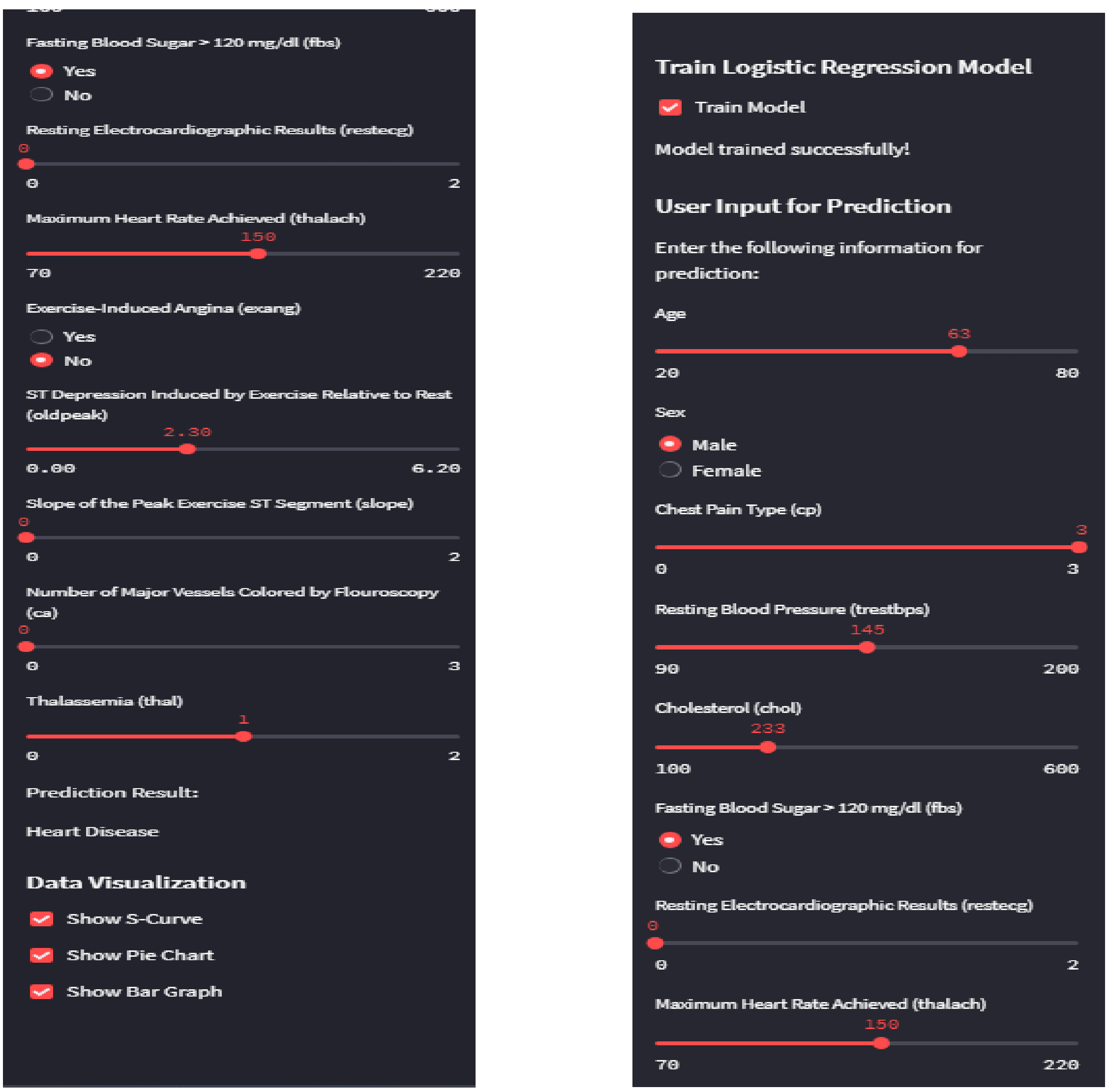
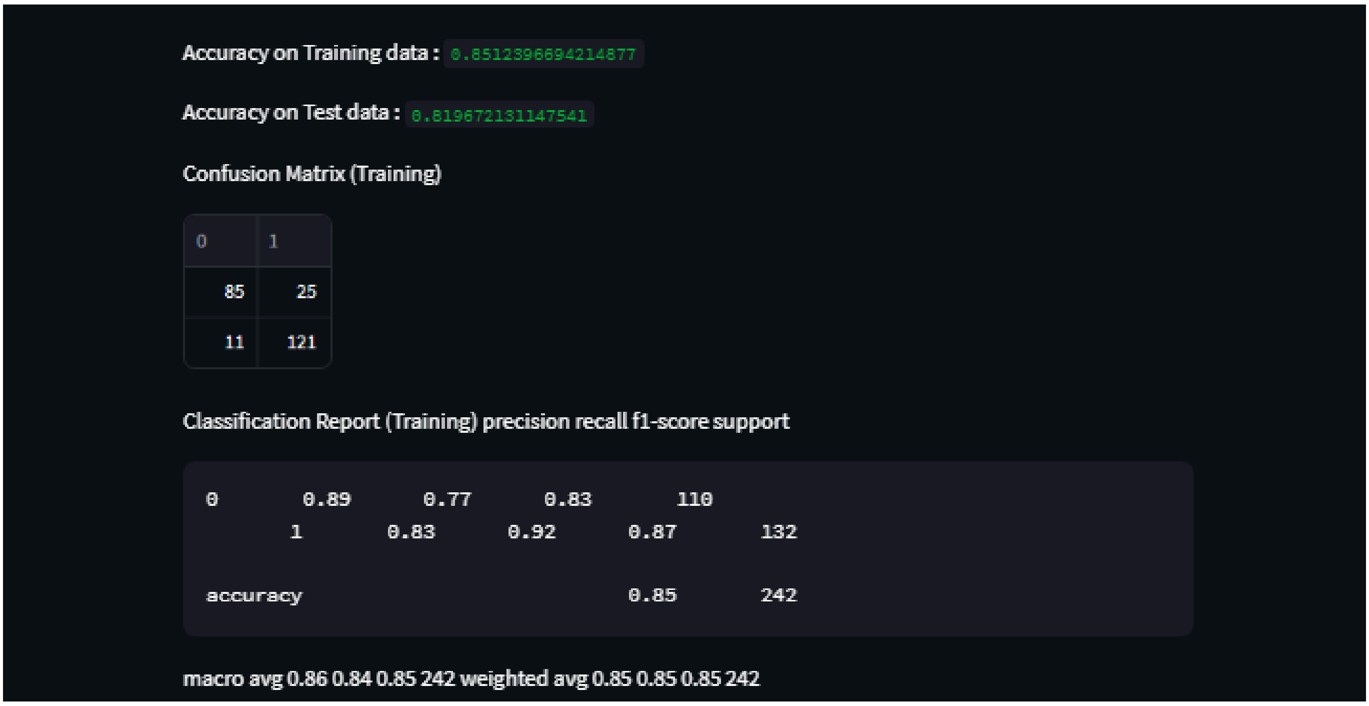


Fig 9: Heart disease prediction



Accuracy on Training data : `0.8512396694214877`

Accuracy on Test data : `0.819672131147541`

Confusion Matrix (Training)

0	1
85	25
11	121

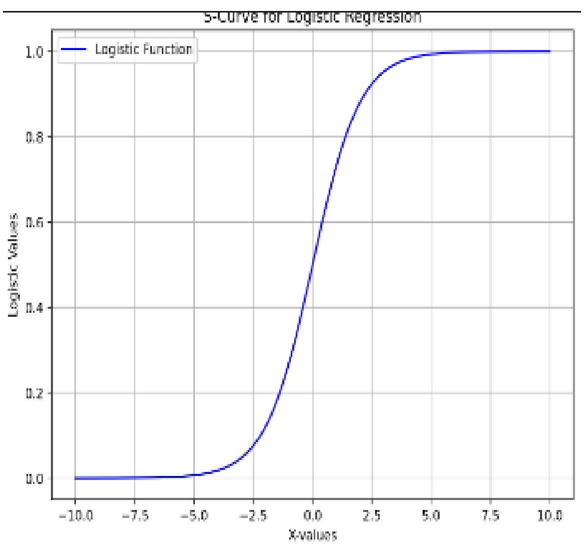
Classification Report (Training) precision recall f1-score support

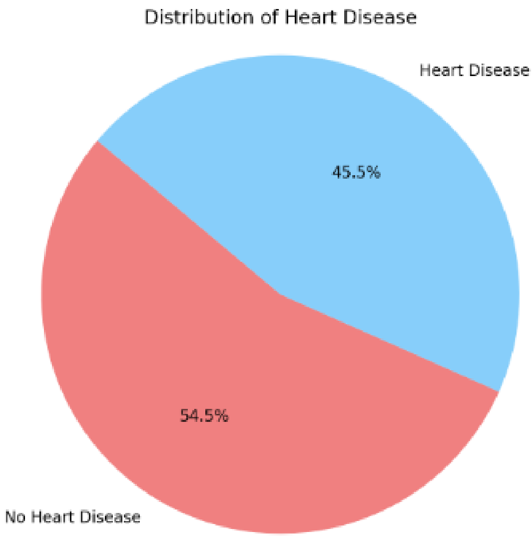
```
0        0.89       0.77       0.83        110
       1        0.83       0.92       0.87        132

accuracy                              0.85        242
```

macro avg 0.86 0.84 0.85 242 weighted avg 0.85 0.85 0.85 242

Fig 10: Performance Metrics





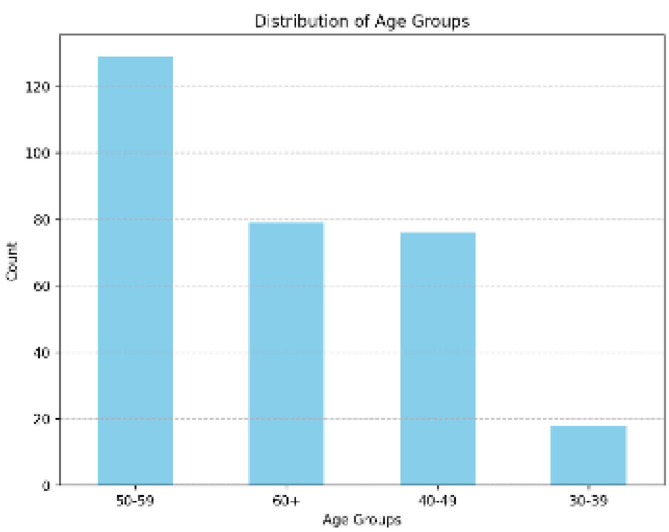


Fig 11: Graphs

4.4 Performance Metrics Details

Accuracy

Accuracy is one metric for evaluating classification models. Informally, accuracy is the fraction of predictions our model got right. Formally, accuracy has the following definition:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

For binary classification, accuracy can also be calculated in terms of positives and negatives as follows:

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

Where TP = True Positives, TN = True Negatives, FP = False Positives, FN = False Negatives.

Accuracy on Training Dataset

We made our model predict the class of the data that was used for the training and then we evaluated the percentage of correctly identified classes using the above formula.

X_train_prediction = model.predict(X_train)

training_data_accuracy = accuracy_score(X_train_prediction, Y_train)

```
[18] print('Accuracy on Training data : ', training_data_accuracy)

     Accuracy on Training data :  0.8512396694214877
```

Figure 12: Accuracy on Training Data

We obtained an accuracy of 85.1% for our training data using our model.

Accuracy on Testing Dataset

We made our model predict the class of the unseen data and then we evaluated the percentage of correctly identified classes using the above formula.

X_test_prediction = model.predict(X_test)

test_data_accuracy = accuracy_score(X_test_prediction, Y_test)

```
[20] print('Accuracy on Test data : ', test_data_accuracy)

     Accuracy on Test data :  0.819672131147541
```

Figure 13: Accuracy on Testing Data

We obtained an accuracy of 81.9% for our training data using our model.

Confusion Matrix

A confusion matrix, also known as an error matrix, is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm. It allows easy identification of confusion between classes e.g. one class is commonly mislabeled as the other. The key to the confusion matrix is the number of correct and incorrect predictions that are summarized with count values and broken down by each class, not just the number of errors made.

confusion = confusion_matrix(Y_test, X_test_prediction)

85	25
11	121

Table 1: Confusion Matrix for Training Data

confusion = confusion_matrix(Y_test, X_test_prediction)

23	5
6	27

Table 2: Confusion Matrix for Testing Data

The rows signify the actual accuracy whereas the columns identify the predicted accuracy. In an Ideal scenario, we would want the diagonal of the confusion matrix to maximum and the non Diagonal elements to be minimum as the diagonals always represent the correct prediction of the class by our model.

```
Classification Report
              precision    recall  f1-score   support

           0       0.89      0.77      0.83       110
           1       0.83      0.92      0.87       132

    accuracy                           0.85       242
   macro avg       0.86      0.84      0.85       242
weighted avg       0.85      0.85      0.85       242
```

Figure 14: Classification Report of Training Dataset

Classification Report

	precision	recall	f1-score	support
0	0.79	0.82	0.81	28
1	0.84	0.82	0.83	33
accuracy			0.82	61
macro avg	0.82	0.82	0.82	61
weighted avg	0.82	0.82	0.82	61

Figure 15: Classification Report of Testing Dataset

Precision

Recall can be defined as the ratio of the total number of correctly classified positive examples divided to the total number of positive examples. High Recall indicates the class is correctly recognized (a small number of FN). Precision is calculated as:

Precision = True Positives / (True Positives + False Positives)

The obtained precision for Training Data is 0.9 for class 0 and 0.83 for class 1, while the obtained precision for Testing Data is 0.79 for class 0 and 0.84 for class 1

Recall

Recall can be defined as the ratio of the total number of correctly classified positive examples divided to the total number of positive examples. High Recall indicates the class is correctly recognized (a small number of FN). Recall is calculated as:

Recall = True Positives / (True Positives + False Negatives)

The obtained Recall for Training Data is 0.77 for class 0 and 0.92 for class 1, while the obtained precision for Testing Data is 0.82 for class 0 and 0.82 for class 1

F1 Score

The F1 Score is the harmonic mean of precision and recall. It provides a balance between precision and recall and is particularly useful when you want to find a model that performs well in both precision and recall. F1 score is calculated as:

*F1 Score = 2 * (Precision * Recall) / (Precision + Recall)*

The obtained F1 Score for Training Data is 0.83 for class 0 and 0.87 for class 1, while the obtained precision for Testing Data is 0.81 for class 0 and 0.83 for class

Chapter 5

Results and Discussions

We have successfully created an ML model that will be able to predict if a given person has a Heart disease or not. We would have to provide the data inform of a tuple of all the 13 features which the model was initially trained with. It is important that we convert the same into a numpy array and reshape it. Now, it is time to send this data as input to our model using the .predict() function

prediction = model.predict(input_data_reshaped)

The output of this function is the class of maximum confidence by our model. If the prediction is 0 that means that the person does not have a Heart Disease and if it is 1 then that means that the person suffers from a heart disease.

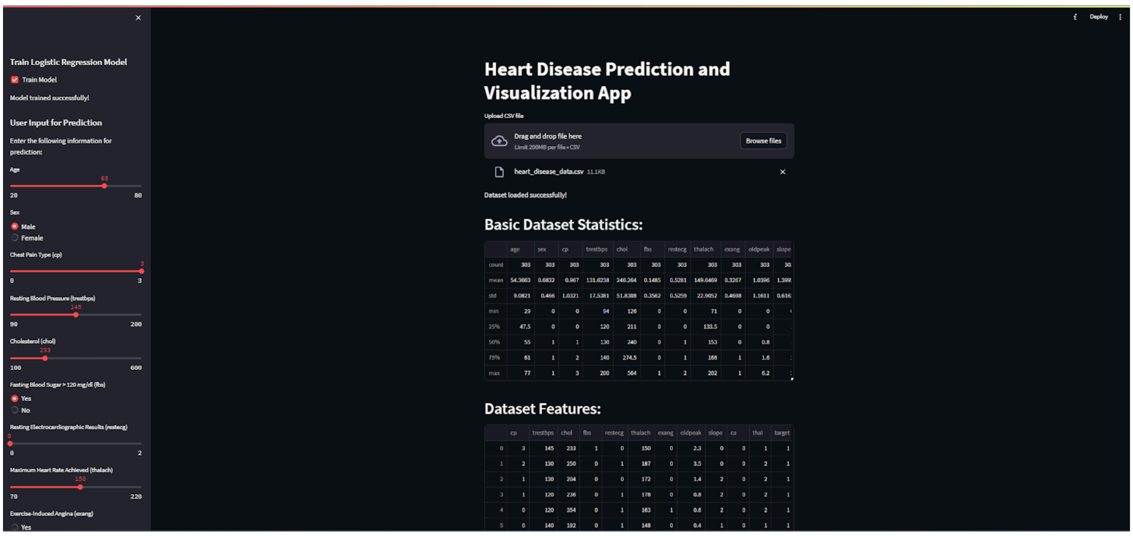


Figure 16: Final Output

Chapter 6
Conclusion and Future Scope

In conclusion, this project signifies a significant stride towards enhancing heart disease diagnosis and prognosis through the integration of machine learning technology. Our model exhibited exceptional promise in accurately identifying individuals at risk of heart disease while minimizing false positives and false negatives. As we move forward, further refinements, validations, and real-world clinical implementations will be essential to ascertain the model's robustness and generalizability. By embracing the potential of machine learning in healthcare, particularly in the realm of cardiovascular health, we contribute to the broader mission of advancing early disease detection, improving patient care, and ultimately saving lives.

References

[1] A. H. M. S. U. Marjia Sultana, "Analysis of Data Mining Techniques for Heart Disease Prediction," 2018.

[2] M. I. K. ,. A. I. ,. S. Musfiq Ali, "Heart Disease Prediction Using Machine Learning Algorithms".

[3] K. Bhanot, "towarddatascience.com," 13 Feb 2019. [Online]. Available: https://towardsdatascience.com/predicting-presence-of-heart-diseases-using-machine-learning-36f00f3edb2c. [Accessed 2 March 2020].

[4] M. A. K. S. H. K. M. a. V. P. M Marimuthu, "A Review on Heart Disease Prediction using Machine Learning and Data Analytics Approach".

Acknowledgement

The satisfaction that accompanies the successful completion of this project would be incomplete without the mention of the people who made it possible, without whose constant guidance and encouragement would have made efforts go in vain. We would like to express our gratitude and respect towards all those who guided us through the completion of this project. We would like to thank our project guide Mrs. Nazneen Ansari of the Computer Engineering Department for providing encouragement, constant support and guidance which was of great help to complete this project successfully.