

ML - BASED HEART STROKE RISK PREDICTION

A PROJECT REPORT

Submitted by

G SASI DEEKSHATH 723921104015

CH SURENDRA 723921104011

B CHENCHU NITHIN 723921104003

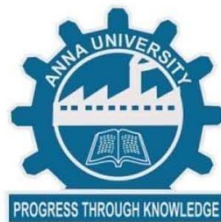
in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING



ARJUN COLLEGE OF TECHNOLOGY, COIMBATORE

ANNA UNIVERSITY :: CHENNAI 600 025

MAY 2025

ML - BASED HEART STROKE RISK PREDICTION

A PROJECT REPORT

Submitted by

G SASI DEEKSHATH 723921104015

CH SURENDRA 723921104011

B CHENCHU NITHIN 723921104003

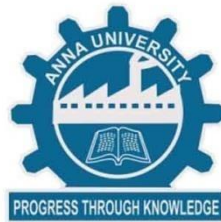
in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING



ARJUN COLLEGE OF TECHNOLOGY, COIMBATORE

ANNA UNIVERSITY :: CHENNAI 600 025

MAY 2025

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report “ **ML – Based Heart Stroke Risk Prediction** ”is the bonafide work of “ **G SASI DEEKSHATH(723921104015),CH SURENDRA (723921104011),B CHENCHU NITHIN (723921104003)**” who carried out the project work under my supervision.

SIGNATURE

**Mr.S.SATHEESH ME.,(Ph.D).,
HEAD OF THE DEPARTMENT,
ASSOCIATE PROFESSOR**

Department of Computer Science and
Engineering,
Arjun College of Technology,
Coimbatore -642 120

SIGNATURE

**Mrs.A.BARANISHRI BE.,ME.,
SUPERVISOR,
ASSOCIATE PROFESSOR**

Department of Computer Science
and Engineering,
Arjun College of Technology,
Coimbatore -642 120

Submitted for the university project viva-voice held on _____

INTERNAL EXAMINAR

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We take this opportunity to express our heartfelt gratitude to all those who have supported and guided us throughout the successful completion of our project.

First and foremost, we would like to express our sincere thanks to our respected **Chairman, Thiru R. Suriyanarayanan**, our **Managing Trustee, Thiru Er. G. Srinivasan**, our **Trustee, Thiru Raja Duraisamy**, and our **Secretary, Dr. R. Suresh Kumar** for providing us with the opportunity and infrastructure to carry out this project.

We extend our deep gratitude to our beloved **Principal, Dr. N. Janaki Manohar**, for his constant encouragement and support throughout our academic journey.

We are thankful to **Mr. S. Satheesh, Head of the Department** of Computer Science and Engineering, for his valuable support and motivation.

We would also like to extend our appreciation to our **Project Coordinator, Mr. V. M. Suresh**, for his continuous monitoring and encouragement.

A special note of thanks goes to our **Project Guide, Mrs. A. Baranishri**, for her invaluable guidance, patience, and support at every stage of this project. Her expertise and insights have been instrumental in its successful completion.

Finally, we sincerely thank all the faculty members, technical staff, and our friends of **Arjun College of Technology** who directly or indirectly contributed to the success of our project.

ABSTRACT

Stroke is a leading cause of mortality and disability worldwide, making early risk prediction crucial for preventive healthcare. This project aims to develop an ML-based predictive model capable of accurately identifying individuals at high risk of stroke using medical attributes. By leveraging advanced machine learning techniques and comprehensive patient data, the model will assist healthcare providers in implementing timely interventions, ultimately improving patient outcomes and reducing healthcare costs. The system will be designed to integrate seamlessly into clinical workflows, offering user-friendly interfaces for medical professionals while maintaining high predictive accuracy and reliability.

Table of content

Chapter no	Title	Page no
	Abstract	iv
	List of Figures	vii
	List of Abbreviations	viii
1	Introduction	1
	Types of Cardiovascular Diseases	2
2	Literature Survey	4
	2.1 Inferences Form Literature Survey	
	2.2 Open Problems in Existing System	5
3	Requirement Analysis	6
	3.1 Feasibility Studies/Risk Analysis of the Project	
	3.2 Software Requirements Specification Document	7
	3.3 System Use Case	8
	3.4 Activity Diagram	
	3.5 Sequence Diagram	9
4	Description of proposed System	11
	4.1 Architecture/ Overall Design of Proposed System	

	4.2 Selected Methodology or Process Model	13
	4.3 Description of Software for Implementation and Plan of the Proposed Model/System	19
	4.4 Transition/Software to Operations Plan	21
5	Implementation Details	22
	5.1 Development and Deployment Set up	
	5.2 Algorithm	24
	5.3 Testing	27
	5.4 Performance Analysis	28
6	Result and Discussion	30
	6.1 Output Table	31
7	Conclusion and Future Enhancement	32
	7.1 Conclusion	
	7.2 Future Work	33
	7.3 Research Issues	35
	7.4 Implementation Issues	36
A	Source Code	37
B	Screenshots	44
	References	49

List of Figures

Figure No	Title	Page No
3.1	Use Case Diagram	8
3.2	Activity Diagram	9
3.3	Sequence Diagram	10
4.1	System Architecture	11
4.2	Data Processing	15
4.3	Model training	16
4.4	Model Evaluation	18
5.1	SVM Classifier	24
5.2	Hyper Planes in 2D & 3D	
5.3	Performance Evaluation	29
B1	Implementation of Code	44
B2	Datasets and Attributes	45
B3	Targets Values	46
B4	Getting input data	47
B5	Output	48

LIST OF ABBREVIATIONS

LR	Logistic Regression
SVC	Support Vector Classification
RFC	Random Forest Classification
NNC	Neural Network Classification
XGBC	Xgboost Classification
DT	Decision Tree
NB	Naïve Bayes

CHAPTER 1

INTRODUCTION

Among various life-threatening diseases, heart disease has garnered a great deal of attention in medical research. The diagnosis of heart disease is a challenging task, which can offer automated predictions about the heart condition of a patient so that further treatment can be made effective. The diagnosis of heart disease is usually based on signs, symptoms and physical examination of the patient. There are several factors that increase the risk of heart disease, such as smoking habit, body cholesterol level, family history of heart disease, obesity, high blood pressure, and lack of physical exercise. A major challenge faced by healthcare organizations, such as hospitals and medical centers, is the provision of quality services at affordable costs.¹ The quality service implies diagnosing patients properly and administering effective treatments. The available heart disease database consists of both numerical and categorical data. Before further processing, cleaning and filtering are applied on these records in order to filter the irrelevant data from the database.² The proposed system can determine an exact hidden knowledge, i.e, patterns and relationships associated with heart disease from a historical heart disease database. It can also answer the complex queries for diagnosing heart disease; therefore, it can be helpful to health care practitioners to make intelligent clinical decisions. Results showed that the proposed system has its

unique potency in realizing the objectives of the defined mining goals.

Our motive for this project is to predict the Heart diseases for a patient with the maximum amount of accuracy in our prediction. For this we have collected a 2 dataset named Indian patient Heart disease dataset from Kaggle database of Indian Heart patients records and used that dataset in our two modules to predict Heart disease using various machine learning techniques

TYPES OF CARDIOVASCULAR DISEASES

Heart disorders are a category of problems that affect the heart and blood vessels and are sometimes referred to as cardiovascular diseases (CVD). Coronary artery disease (CAD), which includes angina and infarction, is a type of cardiovascular illness. Another type of heart illness is coronary heart disease (CHD), which develops when plaque, a waxy substance, builds up inside the coronary arteries. Such arteries deliver plasma that is rich in oxygen to the cardiovascular system. When plaque begins to accumulate in these arteries, the resulting condition is known as atherosclerosis. The formation of plaque takes several years. This plaque might harden or rupture with the passing of time (break open). Plaque calcification gradually narrows coronary arteries, reducing the flow of oxygenrich blood into the heart. If this plaque punctures, clots may form on its surface.

An arrhythmia, or irregular heartbeat, is a problem with the rate or rhythm of your heartbeat. Your heart may beat too quickly, too slowly, or with an irregular rhythm. It is normal for your heart rate to speed up during physical activity and to slow down while resting or sleeping. There are four types of arrhythmia. They are :● Ventricular fibrillation.

- Ventricular tachycardia.
- Premature ventricular beats (PVCs)
- Torsade's de pointes.

It is understood that the best ECG lead for monitoring arrhythmias is **V1**. The patient's symptoms were related to a wide QRS complex tachycardia, and **V1** is capable of distinguishing ventricular tachycardia (VT) from supraventricular tachycardia (SVT) with aberrant conduction.

CHAPTER - 2

LITERATURE SURVEY

Recent research has demonstrated the effectiveness of ML in healthcare applications, particularly for stroke prediction:

1. LSTM models have shown exceptional sensitivity in identifying stroke risk, with one study achieving 96.15% sensitivity.
2. Random Forest models have outperformed both traditional ML and deep learning models in some cases, achieving up to 99% accuracy in stroke prediction.
3. Neural networks have demonstrated superior performance in capturing complex data relationships, with accuracies reaching 95.16% in certain studies.
4. Existing systems primarily focus on either ML or DL models, with limited comparative analysis between approaches.
5. Most studies utilize hospital datasets with 400-700 patients, highlighting the need for larger datasets.
6. Key features identified across studies include platelet levels, cholesterol, and alkaline phosphatase.

2.1 INFERENCES FROM LITERATURE SURVEY

There are some logically strong inferences that can be made from the literature review. The thesis is to composite the ideology of using machine learning algorithms for the

prognosis, diagnosis and study of heart diseases and their predictability, it is important to deal majorly with the kind of machine learning algorithms that would suit the purpose and be centric on the major objectives - being able to predict the presence of a heart disease in the most accurate possible way. The literature surveys conclude the use of Naive Bayes and Support Vector Machine algorithms for the prediction of heart diseases. There are two major parameters that are involved in understanding the suitability of the respective methodologies and they are - the time taken to execute the prediction process and the accuracy of the predictive result. It is clear through various studies and experimentations that the SVM classifier is the best of all the algorithms owing to the extremely high accuracy rates. But when it comes to the time taken to execute the predictive process, the Naive Bayes classifier reflects higher suitability since it takes the least possible time to execute the process.

2.2 OPEN PROBLEMS IN EXISTING SYSTEM

Although prediction results achieved are promising, these traditional approaches are still far from being highly accurate and efficient. The existing systems are simple and effective but are extremely vulnerable to impact. Although prediction results achieved are promising, these traditional approaches are still far from being highly accurate and efficient.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDIES/RISK ANALYSIS OF THE PROJECT

FEASIBILITY STUDY

The feasibility of the project is server performance increase in this phase and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system is to be carried out. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Economic feasibility
- Technical feasibility
- Operational feasibility

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of funds that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand

on the available technical resources. This will lead to high demands being placed on the client. The developed system must have modest requirements, as only minimal or null changes are required for implementing this system.

OPERATIONAL FEASIBILITY

The aspect of the study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

3.2 SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

Hardware specifications: to educate the user about the system and to make him familiar with it. His level of

- Stable Version of Google Chrome
- Higher RAM, of about 4GB or above
- Processor of frequency 1.5GHz or above

Software specifications:

- Google Collab
- PyStream

3.3 SYSTEM USE CASE

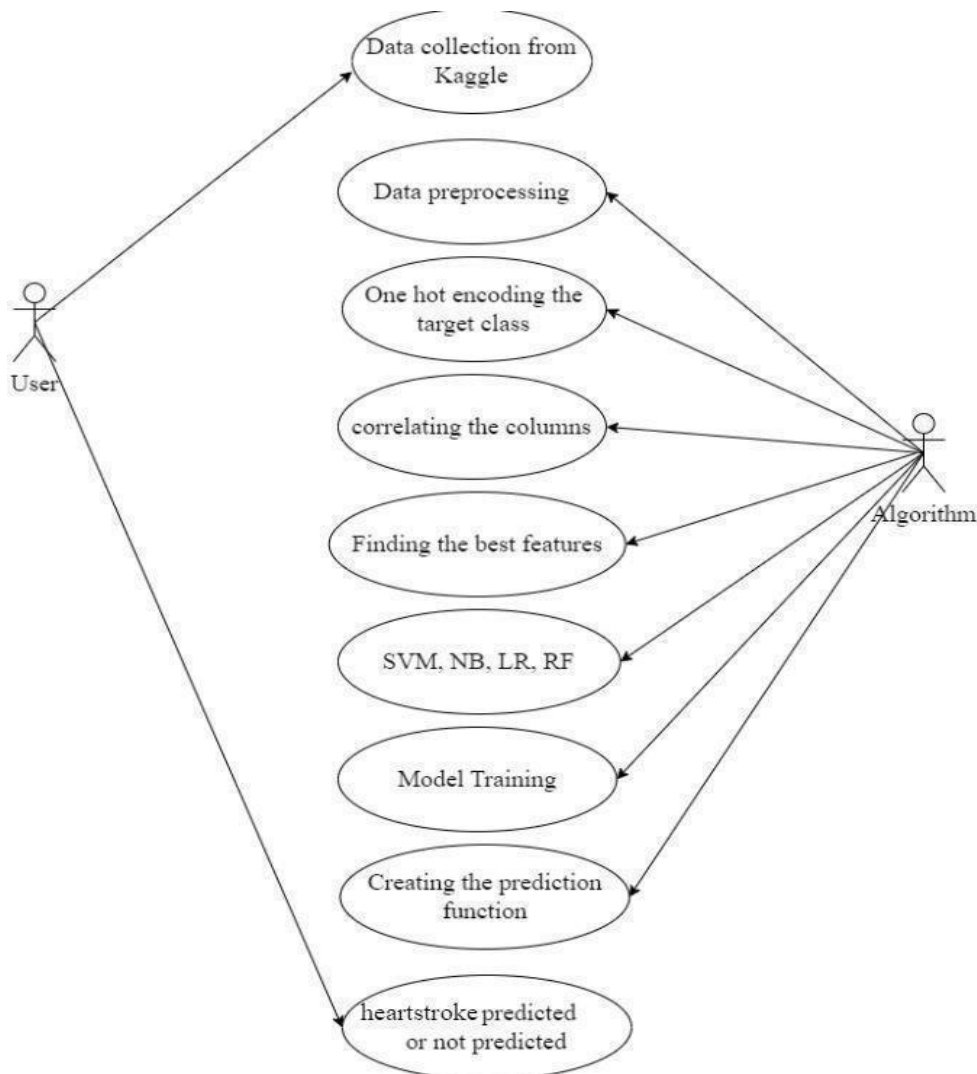


Fig 3.1 : Use Case Diagram

3.4 ACTIVITY DIAGRAM

An activity diagram is a kind of graphical representation that may be used to depict events visually. It is made up

of a group of nodes that are linked to one another by means of edges. They are able to be connected to any other modelling element, which enables the behaviour of activities to be replicated using that methodology.

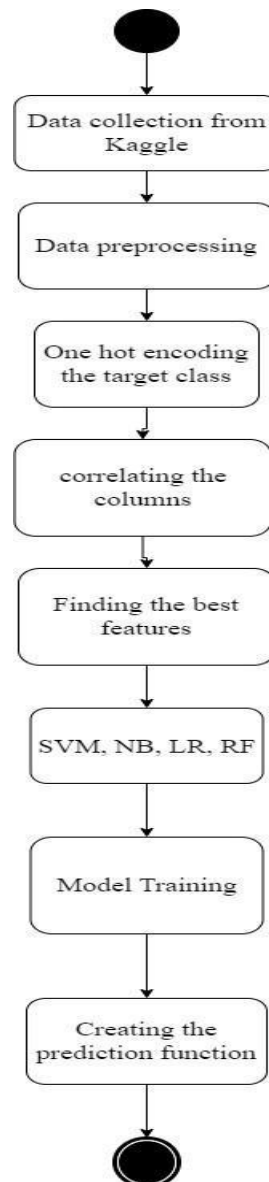


Fig 3.2 Activity Diagram

3.5 SEQUENCE DIAGRAM

The sequence diagram, also called the event diagram, describes the flow of messages in the system. It helps to visualize various dynamic parameters. He describes the

communication between two rescue lines as a series of events arranged in time in which these rescue lines participated during the performance. The lifeline is represented by a vertical bar in UML while the message flow is represented by a vertical dotted line that crosses the bottom of the page. It includes both repetitions and branches.

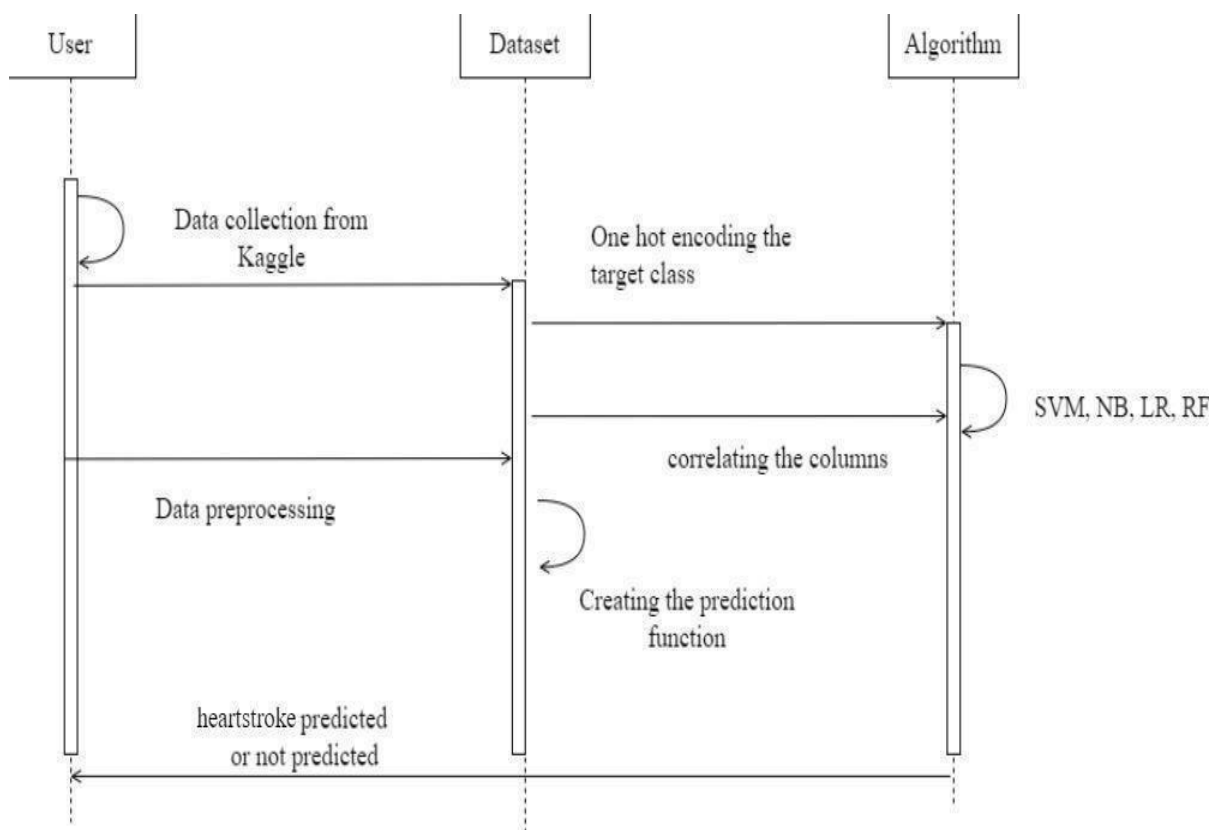


Fig 3.3 Sequence diagram

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

4.1 ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM

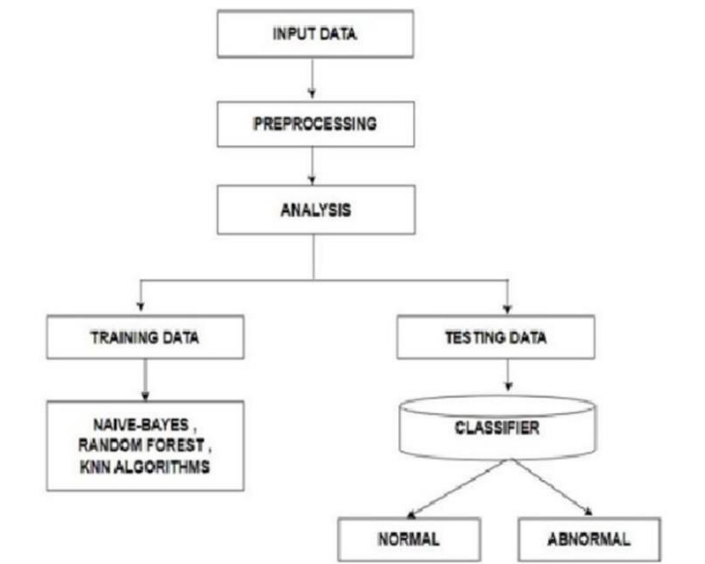


Fig 4.1: System Architecture

Data Collection: The first step is to collect data from various sources such as electronic health records, medical imaging, and patient history. This data will be used to create a comprehensive view of the patient's health status.

Data Preprocessing: The collected data needs to be preprocessed to remove noise and outliers. This may involve data cleaning, feature extraction, and feature selection.

Machine Learning Model: The preprocessed data is fed into a machine learning model that uses various algorithms such as decision tree, random forests, support vector machine, Naïve bayes, K - nearest neighbour to predict the likelihood of a heart stroke. The model may also take into account the patient's demographics, medical history, and lifestyle factors.

Model Training and Validation: The machine learning model needs to be trained using a dataset with known outcomes. The model's performance is then validated using a different dataset to ensure that it is accurate and generalizable.

Deployment: Once the model has been trained and validated, it can be deployed in a real- world setting such as a hospital or clinic. The model will analyze patient data in real-time and provide predictions on the likelihood of a heart stroke

Firstly, we just had to acquire all the datasets. Read the dataset, the information should have several characteristics as sex, age, restBP, cp, cholesterol, FB, goal etc. The information should be investigated to ensure that the data is confirmed. In this case, the sigmoid function is used, which aids in the visual depiction of the labeled data. Methods, such as the DT, LR, CNN algorithms enhance the accuracy.

The proposed system of classifying pulse rate using ECG data values using machine learning techniques is an improvement over existing methods in several ways.

Flexibility: The proposed system can use various ml techniques, such as SVM, LR,

CNN, DT, and KNN, providing more flexibility in choosing the most suitable algorithm for a given dataset.

Improved performance: With the use of advanced deep learning models, such as CNN or RNN, the proposed system has the potential to achieve better performance compared to traditional machine learning algorithms.

Incorporation of additional data sources: For better categorization results, the suggested system may take into account other sources of data as demographics.

Real-time implementation: The proposed system can be implemented in realtime, making it more suitable for practical use in healthcare compared to existing methods that may have limitations in terms of processing speed

4.2 SELECTED METHODOLOGY OR PROCESS MODEL

Implementing a system for heart stroke prediction using various machine learning algorithms such as K-Nearest Neighbour (KNN), Linear Regression, Support Vector Classification (SVC), Decision Tree, Random Forest, and Naive Bayes can be a powerful tool in predicting and preventing heart strokes.

To start, the dataset should be cleaned and preprocessed by removing any missing or incorrect data points. The data should then be split into training and testing sets to evaluate the

performance of the algorithms. Feature scaling should also be performed to ensure all features have the same range and units.

The first algorithm that can be implemented is KNN, which works by finding the k nearest neighbours to a data point based on a similarity metric. The k neighbours' output labels are then used to predict the output label of the new data point. The best value of k can be determined using cross-validation.

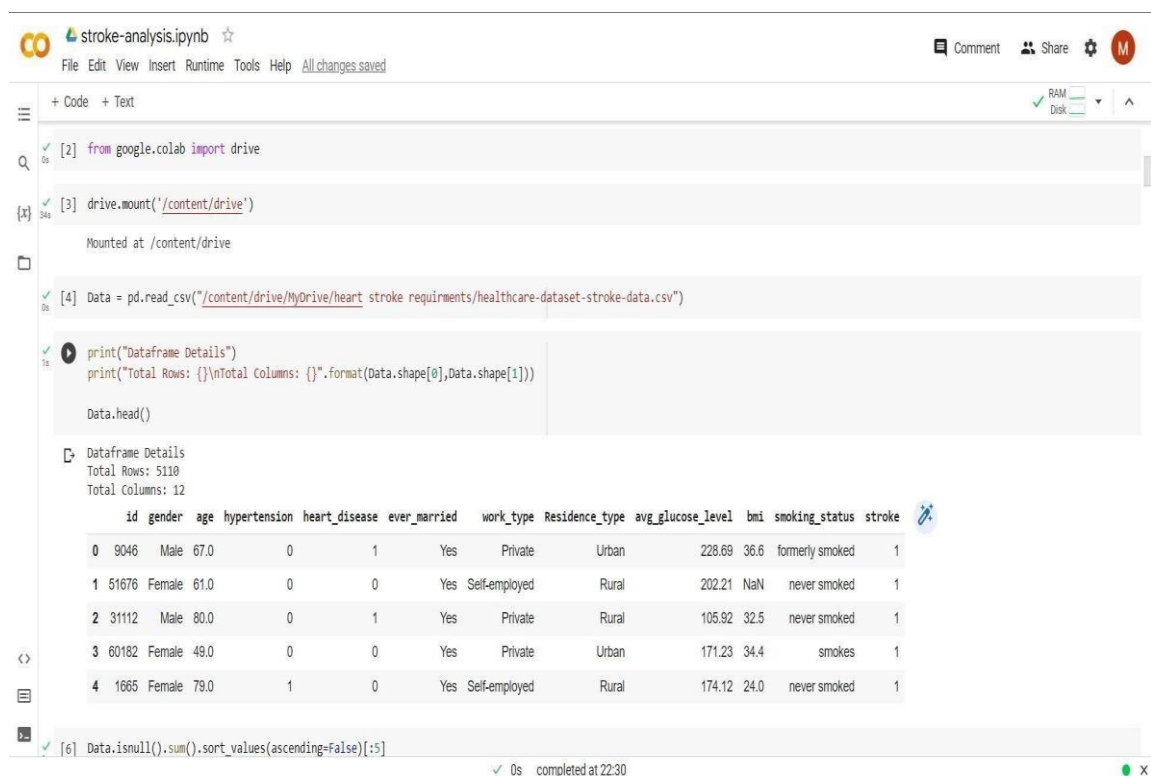
Linear regression is another algorithm that can be used for heart stroke prediction. This algorithm works by fitting a linear equation to the data, where the dependent variable is the output label, and the independent variables are the input features. The coefficients of the linear equation can be determined using gradient descent or other optimization algorithms.

SVC is a powerful algorithm for classification tasks, and it works by finding the optimal hyperplane that separates the data into different classes. The algorithm tries to find the hyperplane with the largest margin between the classes, and it can handle nonlinearly separable data by using kernel functions.

Decision trees are another algorithm that can be used for heart stroke prediction. This algorithm works by splitting the data into subsets based on the values of the input features. The algorithm tries to maximize the information gain at each split to create a tree that accurately predicts the output labels.

MODULE 1: Data Preprocessing

The data preprocessing module is a crucial part of any machine learning project as it involves cleaning and preparing the data before it is used for training the models. The quality of the data and its preparation can significantly impact the performance of the machine learning models. In the case of heart stroke prediction, the data preprocessing module can be used to clean and preprocess the medical data to ensure accurate predictions.



```
stroke-analysis.ipynb
File Edit View Insert Runtime Tools Help All changes saved

[2] from google.colab import drive

[3] drive.mount('/content/drive')
Mounted at /content/drive

[4] Data = pd.read_csv("/content/drive/MyDrive/heart stroke requirements/healthcare-dataset-stroke-data.csv")

print("Dataframe Details")
print("Total Rows: {} \n Total Columns: {}".format(Data.shape[0], Data.shape[1]))

Data.head()
```

Dataframe Details
Total Rows: 5110
Total Columns: 12

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural	202.21	NaN	never smoked	1
2	31112	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
3	60182	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1

```
[6] Data.isnull().sum().sort_values(ascending=False)[:5]
```

0s completed at 22:30

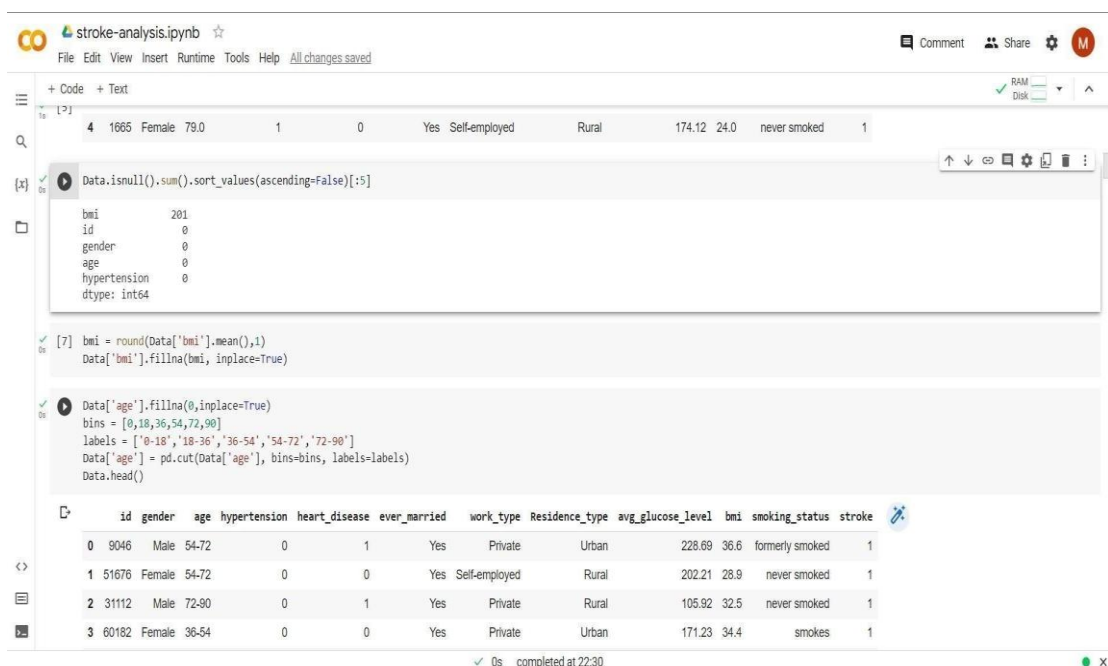
Fig -4.2-DATA PROCESSING

MODULE 2: Model training

The model training module can be used to train the machine learning models using the preprocessed data. This module can include functions to train different machine learning models such as linear regression, SVM, decision tree, random forest, and Neural networks.

1. Split the dataset into training and testing sets.
2. Train models using the training set.
3. Fine-tune hyperparameters to optimize performance.

For example, hyperparameter tuning can be performed using techniques such as grid search or random search to find the best hyperparameters for the models. Cross-validation can be performed to evaluate the performance of the models on different subsets of the data. The trained models can be saved in separate files for testing and evaluation.



The screenshot displays a Jupyter Notebook titled 'stroke-analysis.ipynb'. The interface includes a top menu bar with options like File, Edit, View, Insert, Runtime, Tools, and Help. Below the menu, there's a toolbar with icons for saving, running, and other actions. The main area shows a series of code cells. The first cell contains a line of code: `Data.isnull().sum().sort_values(ascending=False)[:5]`. The output of this cell is a dictionary showing the number of missing values for each column: `bmi: 201, id: 0, gender: 0, age: 0, hypertension: 0`. The second cell contains code to calculate the mean of the 'bmi' column and fill missing values: `bmi = round(Data['bmi'].mean(),1)` and `Data['bmi'].fillna(bmi, inplace=True)`. The third cell contains code to fill missing values in the 'age' column and create age bins: `Data['age'].fillna(0, inplace=True)`, `bins = [0,18,36,54,72,90]`, `labels = ['0-18','18-36','36-54','54-72','72-90']`, and `Data['age'] = pd.cut(Data['age'], bins=bins, labels=labels)`. The final output is a preview of the dataset, showing columns: id, gender, age, hypertension, heart_disease, ever_married, work_type, Residence_type, avg_glucose_level, bmi, smoking_status, and stroke. The data is presented in a table with 4 rows (index 0 to 3).

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	Male	54-72	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
1	51676	Female	54-72	0	0	Yes	Self-employed	Rural	202.21	28.9	never smoked	1
2	31112	Male	72-90	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
3	60182	Female	36-54	0	0	Yes	Private	Urban	171.23	34.4	smokes	1

Fig – 4.3 – MODEL TRAINING

MODULE 3: Model Evaluation

The model evaluation module is a critical part of a heart stroke prediction project, as it allows us to test and evaluate the performance of the trained machine learning models on new data. The module should include functions to load the preprocessed data and the trained models, and use them to predict the output labels for the test data. It should also include functions to calculate the performance metrics such as accuracy, precision, recall, and F1-score for each of the models.

Accuracy is a common performance metric that measures the proportion of correctly classified instances. It can be calculated as the number of correctly classified instances divided by the total number of instances. However, accuracy can be misleading if the data is imbalanced, as the model may simply predict the majority class without learning anything about the minority class. Therefore, precision, recall, and F1-score can provide more insights into the model's performance.

```

# Train and evaluate model
def fit_eval_model(model, train_features, y_train, test_features, y_test):
    """
    Function: train and evaluate a machine learning classifier.
    Args:
        model: machine learning classifier
        train_features: train data extracted features
        y_train: train data labels
        test_features: train data extracted features
        y_test: train data labels
    Return:
        results(dictionary): a dictionary of classification report
    """
    results = {}

    # Train the model
    model.fit(train_features, y_train)

    # Test the model
    train_predicted = model.predict(train_features)
    test_predicted = model.predict(test_features)

    # Classification report and Confusion Matrix
    results['classification_report'] = classification_report(y_test, test_predicted)
    results['confusion_matrix'] = confusion_matrix(y_test, test_predicted)

    return results

```

Fig-4.4-Model Evaluation

```

# Initialize the models
sv = SVC(random_state = 1)
rf = RandomForestClassifier(random_state = 1)
ab = AdaBoostClassifier(random_state = 1)
gb = GradientBoostingClassifier(random_state = 1)

# Fit and evaluate models
results = {}
for cls in [sv, rf, ab, gb]:
    cls_name = cls.__class__.__name__
    results[cls_name] = {}
    results[cls_name] = fit_eval_model(cls, X_train, y_train, X_test, y_test)

# Print classifiers results
for result in results:
    print(result)
    print()
    for i in results[result]:
        print(i, ':')
        print(results[result][i])
        print()
    print('-----')
    print()

```

Output exceeds the [size limit](#). Open the full output data [in a text editor](#)

Fig – 4.4 – Model Evaluation

4.3 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

Testing the model is an important step in heart stroke prediction as it evaluates the model's performance on new and unseen data. Here is a summary of the steps involved in testing the model for heart stroke prediction:

1. **Split the data:** The first step is to split the data into training and testing sets. The training set is used to train the model, while the testing set is used to evaluate the model's performance. The recommended split is usually 80:20 or 70:30 for training and testing, respectively.
2. **Train the model:** Next, the model is trained on the training set using the optimized hyperparameters and feature selection techniques. The model's performance is evaluated using appropriate evaluation metrics such as accuracy, precision, recall, F1-score, and area under the ROC curve.
3. **Test the model:** Once the model is trained, it is tested on the testing set to evaluate its performance on new and unseen data. The same evaluation metrics are used to evaluate the model's performance on the testing set.
4. **Evaluate the results:** Finally, the results are evaluated to determine if the model is performing well on new and unseen data. If the model's performance is satisfactory, it can be deployed for heart stroke prediction in clinical practice.

REFINEMENT AND DEPLOYMENT OF MODEL

Refinement and deployment of a model for heart stroke prediction involves improving the model's performance and deploying it in a real-world setting. Here is a summary of the steps involved in refinement and deployment of the model:

1. **Refinement:** Once the model has been tested and evaluated, any necessary refinements are made to improve the model's performance. This could involve adjusting hyperparameters, improving feature selection, or exploring different algorithms or ensemble techniques. The model is then retrained and retested to evaluate its performance.
2. **Validation:** Before deployment, the model must be validated to ensure that it performs well on new data. This can be done using techniques such as cross-validation, bootstrapping, or hold-out validation. The model's performance on validation data is compared to its performance on the testing data to ensure that it generalizes well to new data.
3. **Deployment:** Once the model has been validated, it can be deployed in a real-world setting. This involves integrating the model into a software application or system that is used in clinical practice. The deployment process must be carefully planned and tested to ensure that the model is robust, reliable, and secure.

4. **Monitoring and maintenance:** After deployment, the model must be monitored and maintained to ensure that it continues to perform well overtime. This involves monitoring the model's performance, updating the model if necessary, and addressing any issues or bugs that arise.

4.4 TRANSITION/ SOFTWARE TO OPERATIONS PLAN

- Scikit learn
- Tensor flow
- Keras
- pyTorch
- XGBoot
- Collab
- Stable Google Chrome
- Intel® Core™ i5 processor 8250U at 1.60 GHz or 1.80 GHz, 8 GB of DRAM.
- Disk space 2TB. 3. Operating System: 64-bit Windows 10 Pro
- PIP and NumPy: Ubuntu*, Python 3.6.2, NumPy 1.13.1, scikit-learn 0.18.2 Windows: Python 3.6.2, PIP and NumPy 1.13.1, scikit-learn 0.18.2 Intel® Distribution for Python* 2018.

CHAPTER 5

IMPLEMENTATION DETAILS

5.1 DEVELOPMENT AND DEPLOYMENT SETUP

Input

The following factors should be considered while designing input for a comparison study on machine learning for Heart disorders diagnosis.

Data sources

In order to diagnose heart disease, it is necessary to identify which data sources will be used, such as electronic medical records, Heart function tests, imaging data, and clinical data analysis.

Data Preprocessing

The system will need to perform data preprocessing in order to prepare the data for analysis, which will include data cleaning, data transformation, and feature engineering in order to prepare the data for analysis.

Feature engineering

Identify and extract from the data the key features that will be used to train the machine learning model based on the features found in the data. In some cases, it may be necessary to do this with automated feature engineering methods or with domain expertise in order to achieve this goal.

Data storage

Determine the appropriate storage mechanism for the input data, such as a database or data warehouse. This will ensure

that the data is easily accessible and can be efficiently processed by the machine learning models.

Data integration

The aim of this project is to integrate data from different sources and formats in order to create a unique dataset that can be used for the diagnosis of Heart disorders.

Output Design

When designing output for Heart disorder diagnosis using machine learning techniques, a comparative study, it is important to consider the following factors.

Diagnosis and classification

Diagnose and classify Heart disorders based on input data. A specific disorder, such as fatty Heart, cirrhosis, or hepatitis, should be identified, along with its severity.

5.2 ALGORITHMS

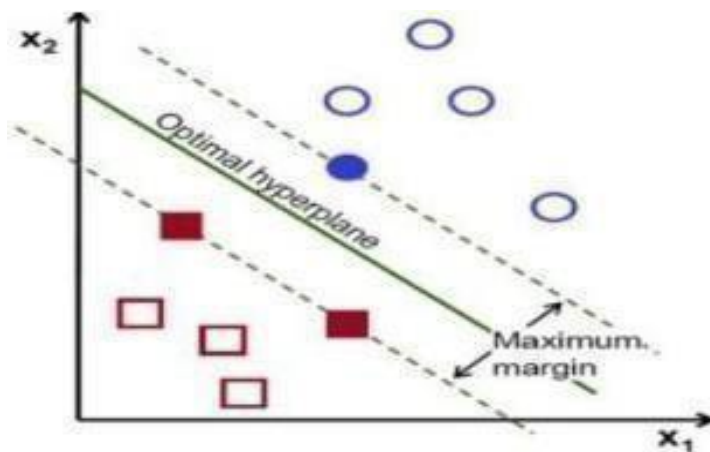
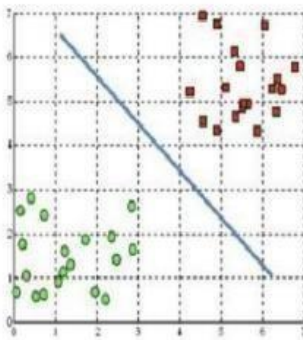


FIG 5.1 : SVM CLASSIFIER

A hyperplane in \mathbb{R}^2 is a line



A hyperplane in \mathbb{R}^3 is a plane

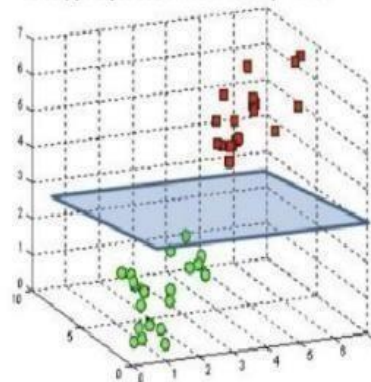


FIG 5.2 : HYPERPLANES IN 2D & 3D

Logistic Regression Algorithm :

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. 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 the 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.

Random Forest Algorithm :

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..

K-Nearest Neighbour(KNN) Algorithm :

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique. K-NN algorithm 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. K-NN algorithm 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.

Decision Tree Classification Algorithm :

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 a graphical representation for getting all the possible solutions to a problem/decision based on given conditions. 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. In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm. A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

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

5.3 TESTING

Test Case 1: Input Validation

Description: The system should validate that input data is correct and complete.

Test Steps: Enter invalid data, such as a negative age or a missing blood pressure reading.

Verify that the system provides an appropriate error message.

Test Case 2: Accuracy of Prediction

Description: The system should accurately predict the likelihood of a heart stroke.

Test Steps: Enter test data for patients who have had a heart stroke and patients who have not. Verify that the system accurately predicts the likelihood of a heart stroke for each patient.

Test Case 3: Sensitivity and Specificity Description: The system should have high sensitivity and specificity.

Test Steps: Enter test data for patients who have had a heart stroke and patients who have not. Calculate the sensitivity and specificity of the system's predictions. Verify that the sensitivity and specificity are high.

Test Case 4: Performance and Scalability

Description: The system should perform well and be scalable.

Test Steps: Enter test data for a large number of patients. Verify that the system can handle a large number of requests without significantly impacting performance. Verify that the system can be easily scaled up to handle additional requests as needed

5.4 PERFORMANCE ANALYSIS

The performance analysis for the heart stroke prediction project involves evaluating the performance of the system based on different performance metrics and comparing them to the project's goals and constraints. Here are some possible performance metrics that can be used for the analysis:

Detection Accuracy: The detection accuracy is a critical performance metric for the heart stroke prediction project. The accuracy measures how accurately the system can predict heart stroke cases. The accuracy can be calculated by comparing the predicted labels to the actual labels in the test dataset. The heart stroke prediction project should aim for an accuracy of at least 80%, and a higher accuracy is preferred. If the accuracy is lower than the target, the system's training data, model selection, or hyperparameters may need to be adjusted.

Response Time: The response time is the time taken by the system to make predictions.

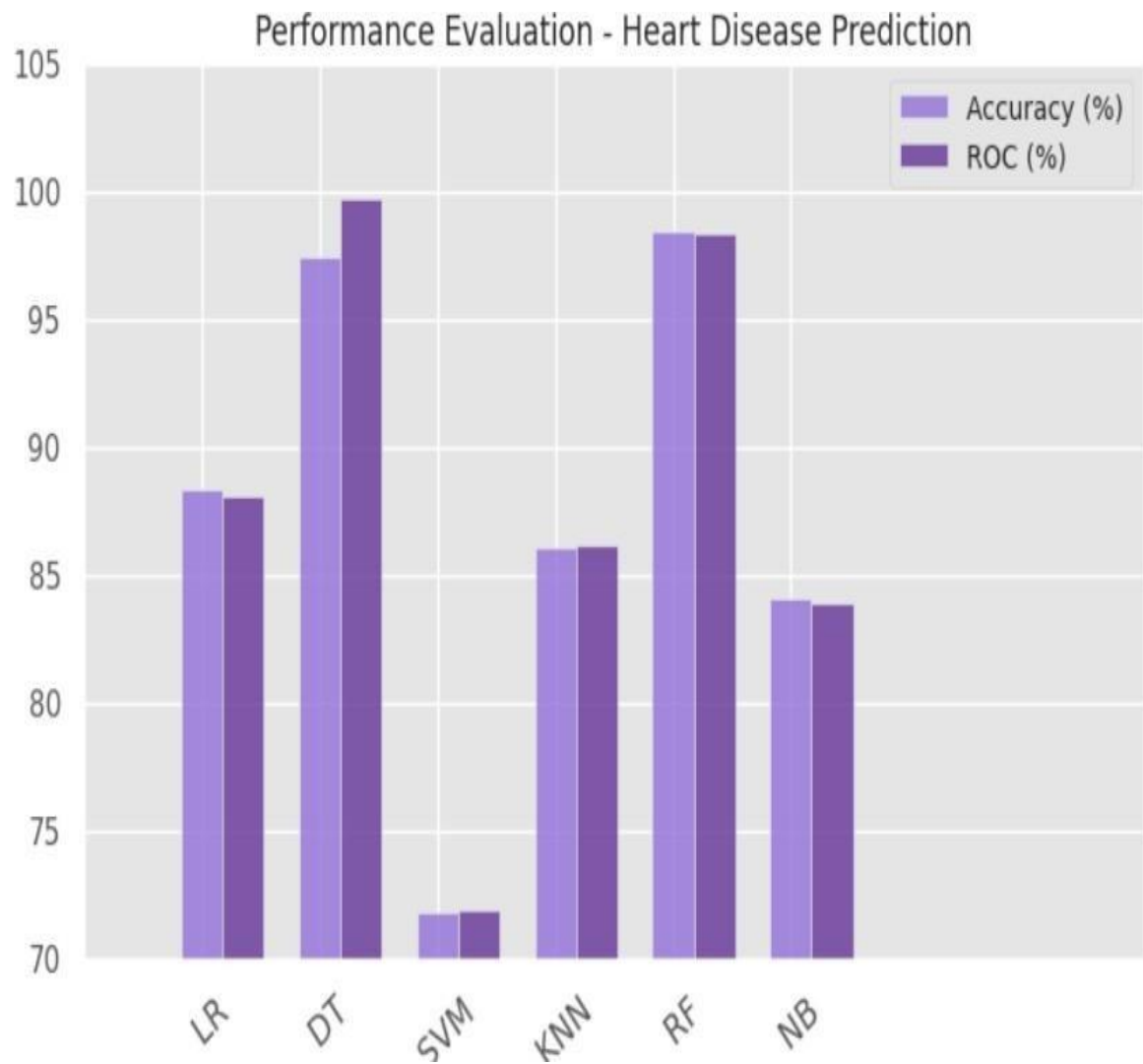


Fig – 5.3 – Performance Evaluation

CHAPTER 6

RESULTS AND DISCUSSION

In order to detect the likelihood that a user has heart disease, a ml model was developed in this research. For this machine learning model, the input data consists of 13 sets of human test results. During the processing and cleaning of this dataset, it was verified that no missing or invalid data values existed. The data set was broken down into two separate halves, x and y. Where the output (represented by the dependent variable y) is a function of the x parameter (which includes the 13 characteristics representing the various test outcomes). Standard Scaler was used to transform the x-axis value. Additional categories were created for the x and y variables: y-test, x-test, x-train & y- train.

These y-train & x-train were learned with the aid of the supervised learning methods CNN, K-NN, LR, SVM, & RF. Using varying values for n, the four methods were utilized to determine the accuracy %. The maximum accuracy is achieved by K Neighbours at n = 1, by CNN at n = 1 by SVM at n=1, and by Random Forest at n = 10. After conducting extensive tests, we settled on using SVM because of its reputation as one among the most reliable forecasting models.

Flask, an Api, was used to save and upload the SVM model to the web. We constructed an HTML website with Thirteen variables using Flask so that people may submit the findings

of their different tests and let the model assess the likelihood that they have pulse rate. The model's results will be shown on a web- based portal for the patient's convenience

TEST ID	ALGORITHM	ACTUAL OUTPUT	PREDICTED OUTPUT	STATUS
1	Logistic Regression	100	85.0	SUCCESS
2	Random Forest	100	80.0	SUCCESS
3	KNN	100	85.0	SUCCESS
4	Decision Tree	100	69.0	SUCCESS
5	SVM	100	80.0	SUCCESS

TABLE 6.1 : OUTPUT TAB

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 CONCLUSION

Machine learning techniques have shown promising results in predicting heart stroke. Several studies have compared the performance of different machine learning algorithms in predicting heart stroke, and the results have varied depending on the dataset, features, and evaluation metrics used. In general, studies have found that ensemble methods such as random forests and gradient boosting classifiers perform better than other algorithms, such as logistic regression, support vector machines, and neural networks. However, the performance differences between these algorithms may not be statistically significant in all cases. It is also important to note that the quality of the data and the choice of features can have a significant impact on the performance of the model. Preprocessing and feature selection techniques, such as imputation, normalization, and feature engineering, can improve the performance of the model by reducing noise and extracting relevant information. Overall, machine learning techniques have the potential to improve the accuracy and efficiency of heart stroke prediction, and further research is needed to identify the most effective algorithms and feature selection techniques for different datasets and evaluation metrics. The prevention and disease progression can be aided by early identification. Early diagnosis and the finding of

important causal factors can be aided by machine learning technologies. The proposed technique generates a deep learning model that can predict cardiovascular illnesses and heart attacks. The optimal solution for the job is the SVM algorithm. The model suggests that new machine learning algorithms usually lead to better prediction accuracy. The prevention and disease progression can be aided by early identification. Early diagnosis and the finding of important causal factors can be aided by machine learning technologies. The proposed technique generates a deep learning model that can predict cardiovascular illnesses and heart attacks. The optimal solution for the job is the SVM algorithm. The model suggests that new machine learning algorithms usually lead to better prediction accuracy.

7.2 FUTURE WORK

The project can be further enhanced by deploying the machine learning model obtained using a web application and a larger dataset could be used for prediction to give higher accuracy and produce better results..

1. **Incorporating new risk factors:** Current heart stroke prediction models use arrange of risk factors such as age, gender, blood pressure, cholesterol levels, smoking habits, and medical history. Future research can explore

incorporating new risk factors such as genetic information, diet, and lifestyle habits.

2. **Improving prediction accuracy:** While machine learning models have shown promise in heart stroke prediction, further research can explore new techniques to improve the accuracy and reliability of the prediction models. This can involve the use of advanced algorithms, data preprocessing techniques, and feature engineering methods.
3. **Developing personalized prediction models:** Personalized heart stroke prediction models can help healthcare professionals identify patients who are at a higher risk of heart stroke and provide targeted interventions. Future research can explore developing personalized prediction models that consider individual differences in risk factors and lifestyles.
4. **Integration with electronic health records (EHRs):** Integrating heart stroke prediction models with EHRs can allow healthcare professionals to access patient data in real-time and make informed decisions about patient care. Future research can explore developing EHR-integrated prediction models that can be integrated into clinical decision support systems.

5. **Evaluating the impact of prediction models:** It is essential to evaluate the impact of heart stroke prediction models in real-world settings to understand their effectiveness in clinical practice. Future research can focus on evaluating the impact of prediction models on patient outcomes, healthcare costs, and healthcare provider decision-making.

7.3 RESEARCH ISSUES

Few Heart problems cannot lead to change in ECG images like;

- Atrial fibrillation/flutter. ☐ Heart attack.
- Heart failure.
- Multifocal atrial tachycardia.
- Paroxysmal supraventricular tachycardia.
- Sick sinus syndrome.
- Wolff-Parkinson-White syndrome.

Common mistakes are: Left-right arm reversals lead to a negative complex in lead I with a negative P wave in lead I. They are one of the most common causes of right axis deviation on the ECG! Arm-foot switches lead to a very small or 'far field' signal in leads II or III.

Valvular defects cannot be detected using an ECG. Chest X-ray can be used to determine such defects. Therefore, an ECG can detect arrhythmia, myocardial infarction and also heart block but not valvular defects.

It cannot provide definite diagnosis of congestive heart failure. Prognosis during anesthesia or surgical procedures is unpredictable. Heart chamber wall thickness cannot be measured. Diseases of heart valves, endocardium, or pericardium cannot.

7.4 IMPLEMENTATION ISSUES

The Algorithms used in our project did not give a 100% accuracy, so the prediction is not 100% feasible. Clinical diagnosis and diagnosis using our project may differ slightly because the prediction is not 100% accurate. Medical diagnosis is considered as a significant yet intricate task that needs to be carried out precisely and efficiently. The automation of the same would be highly beneficial. Clinical decisions are often made based on the doctor's intuition and experience rather than on the knowledge rich data collected from the dataset.

APPENDIX

A. SOURCE CODE

```
# Import the libraries
import numpy as np # To work with arrays
import pandas as pd # To work with the data
# Importing Sk learn libraries/ Modules
from sklearn.model_selection import train_test_split # Module to split the
data into training and testing
from sklearn.linear_model import LogisticRegression # To build the model
from sklearn.metrics import accuracy_score # Evaluation

# Improting the data

df = pd.read_csv("Data/heart_disease_data.csv")

# First 5 Records
df.head()
```

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target	
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

```
df.tail()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

```
# Getting the dimension of our data
df.shape

(303, 14)
df.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

df.isnull().sum()

	0
age	0
sex	0
cp	0
trestbps	0
chol	0
fbs	0
restecg	0
thalach	0
exang	0
oldpeak	0
slope	0

dtype: int64

df.describe()

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	
count	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	0.300000
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.400000
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000

4

1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000
1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000


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

	count
target	
1	165
0	138

```
dtype: int64
```

```
X = df.drop(columns='target',axis=1)
```

```
Y = df['target']
```

X

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	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
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2
...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3

300	68	1	0	144	193	1	1	141	0	3.4	1	2	3
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2

303 rows × 13 columns

	target
0	1
1	1
2	1
3	1
4	1
...	...
298	0
299	0
300	0
301	0

303 rows × 1 columns

dtype: int64

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

```
X_train.shape
(242, 13)
```

```
X_test.shape
(61, 13)
```

```
Y_train.shape
```

```
(242,)
Y_test.shape
(61,)
```

```
model = LogisticRegression()
model.fit(X_train,Y_train) # look into the documentation
```

```
/usr/local/lib/python3.11/dist-
packages/sklearn/linear_model/_logistic.py:465: 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

```
n_iter_i = _check_optimize_result(
```

```
LogisticRegression
```

```
?i
```

```
LogisticRegression()
```

```
# accuracy on training data
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
print(f'This is the accuracy score on training data
{training_data_accuracy}')
```

This is the accuracy score on training data 0.8512396694214877

```

# accuracy on training data
X_test_prediction = model.predict(X_test)
testing_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print(f'This is the accuracy score on training data {testing_data_accuracy}')
This is the accuracy score on training data 0.819672131147541
training_data_accuracy - testing_data_accuracy
0.03156753827394665

```

```

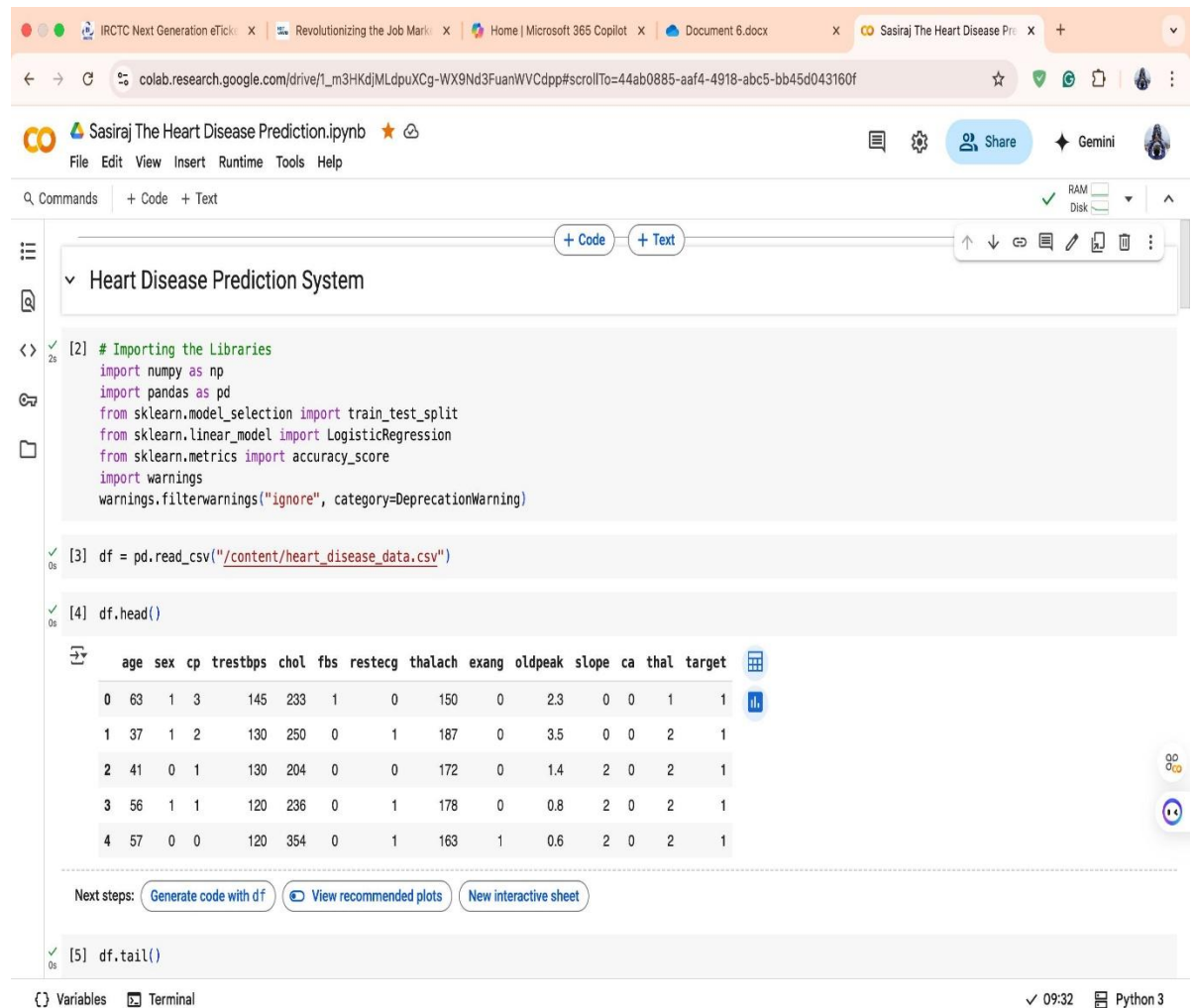
# Building a model
# Input
input_string=input()
numbers=[float(num) for num in input_string.split(",")]
target = (62,0,0,140,268,0,0,160,0,3.6,0,2,2)
# Converting the input list to numpy array.
target = np.array(target)
print(target)
print(type(target))
target = target.reshape(1,-1)
target

prediction = model.predict(target)

if prediction[0] == 0 :
print("Good News Patient doesn't have heart disease")
else:
print("Oh! Patient should visit the doctor")

```

B. SCREENSHOTS



The screenshot shows a Google Colab notebook titled "Sasiraj The Heart Disease Prediction.ipynb". The notebook is open in a web browser with the URL `colab.research.google.com/drive/1_m3HKdjMLdpuXCg-WX9Nd3FuanWVCdpp#scrollTo=44ab0885-aaf4-4918-abc5-bb45d043160f`. The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for commands, code, and text, and a sidebar with a file explorer and a search bar.

The notebook content shows the following code cells:

```
[2] # Importing the Libraries
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
```

```
[3] df = pd.read_csv("/content/heart_disease_data.csv")
```

```
[4] df.head()
```

The output of the `df.head()` command is displayed as a table with 15 columns: `age`, `sex`, `cp`, `trestbps`, `chol`, `fbs`, `restecg`, `thalach`, `exang`, `oldpeak`, `slope`, `ca`, `thal`, and `target`. The first five rows of data are shown:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

Below the table, there are three buttons: "Generate code with df", "View recommended plots", and "New interactive sheet".

The notebook also shows the following code cell:

```
[5] df.tail()
```

The bottom of the notebook interface shows a status bar with "Variables", "Terminal", and "Python 3".

Fig -B1 – implementation of code

Dataset Preview:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

Code Cell [6]:

```
[6] df.shape
```

```
(303, 14)
```

Code Cell [7]:

```
[7] df.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
```

Fig –B2 – data sets and attributes

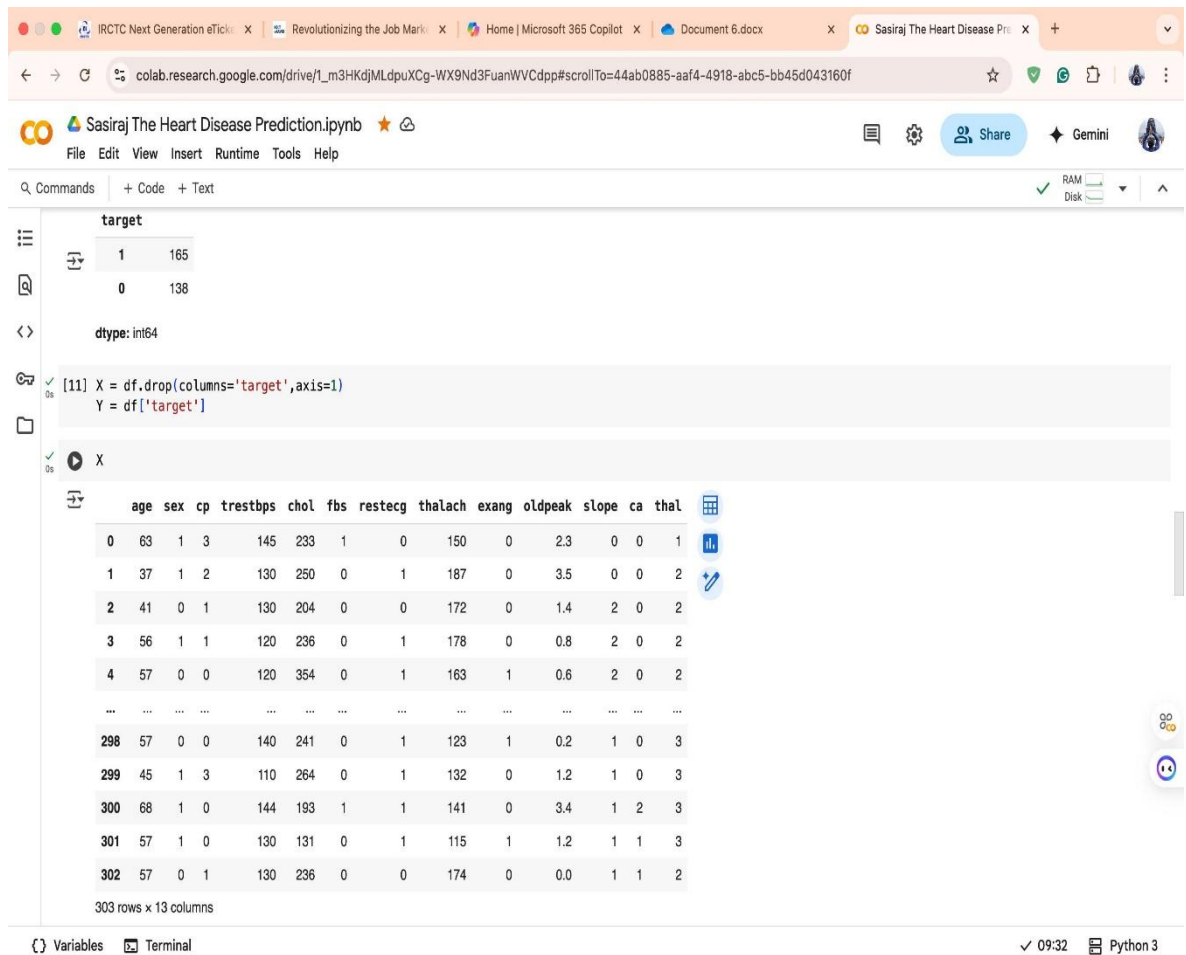
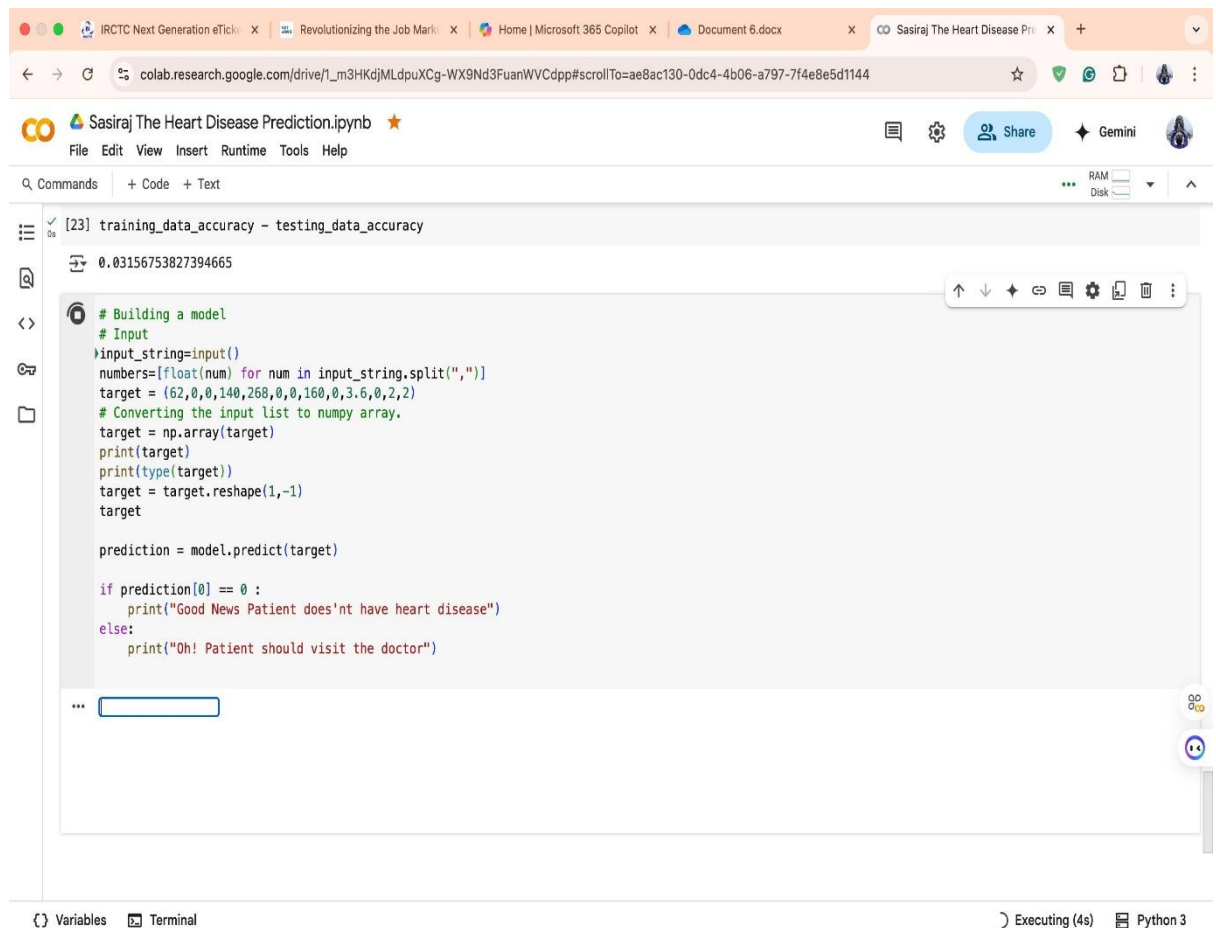


Fig -B3 – target values



```
[23] training_data_accuracy - testing_data_accuracy

0.03156753827394665

# Building a model
# Input
>input_string=input()
numbers=[float(num) for num in input_string.split(",")]
target = (62,0,0,140,268,0,0,160,0,3.6,0,2,2)
# Converting the input list to numpy array.
target = np.array(target)
print(target)
print(type(target))
target = target.reshape(1,-1)
target

prediction = model.predict(target)

if prediction[0] == 0 :
    print("Good News Patient does'nt have heart disease")
else:
    print("Oh! Patient should visit the doctor")

***
```

Fig – B4-Getting input data


```
62,0,0,140,268,0,0,160,0,3.6,0,2,2
[ 62.  0.  0. 140. 268.  0.  0. 160.  0.  3.6  0.  2.
  2. ]
<class 'numpy.ndarray'>
Good News Patient doesn't have heart disease
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but LogisticRegression was fitted
warnings.warn(
```

Fig-B5- output

REFERENCES: -

1. Explainable artificial intelligence for stroke prediction through comparison of deep learning and machine learning models. Nature, 2024
2. Predicting stroke risk: An effective stroke prediction model based on neural networks. Journal of Neurorestoratology, 2025
3. World Health Organization. (2023). Global Health Estimates: Stroke Mortality and Morbidity Report
4. Feng et al. (2023). Deep learning applications in stroke management: A systematic review

Shirsat et al. (2022). Machine learning applications in stroke prevention and management