



# Tech Saksham

## Capstone Project Report

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING  
FUNDAMENTALS

**“HEART DISEASE PREDICTION ”**

**“ANNA UNIVERSITY REGIONAL CAMPUS TIRUNELVELI”**

| NM ID          | NAME              |
|----------------|-------------------|
| au950021135022 | L R ISWARYA LAXMI |

Trainer Name : RAMAR

## ABSTRACT

This project focuses on employing logistic regression, supported by AI and ML techniques, for heart disease prediction—a critical aspect in reducing cardiovascular disease (CVD) mortality rates. With the World Health Organization's alarming estimate attributing four out of five CVD deaths to heart attacks, timely identification of individuals at risk becomes imperative. By leveraging patient data, including demographic, medical, and lifestyle factors, this study aims to pinpoint the ratio of patients predisposed to CVD and predict overall risk. The proposed logistic regression model serves as a tool for healthcare professionals, facilitating early intervention strategies and ultimately contributing to the mitigation of CVD's global burden.

## INDEX

| Sr. No. | Table of Contents        | Page No. |
|---------|--------------------------|----------|
| 1       | Chapter 1: Introduction  | 4        |
| 2       | <b>Problem Statement</b> | 5        |
| 4       | Conclusion               | 6        |
| 7       | Code                     | 7        |

## CHAPTER 1

### INTRODUCTION

Cardiovascular diseases (CVDs) are a leading cause of mortality worldwide, with heart attacks accounting for a significant portion of these deaths. The World Health Organization (WHO) estimates that four out of five CVD deaths are due to heart attacks, highlighting the critical need for effective prediction and prevention strategies. In this research, we aim to develop a predictive model using logistic regression to identify individuals at risk of CVD and estimate their overall risk.

## CHAPTER 2

### Problem statement

Heart disease remains a leading cause of mortality worldwide, with cardiovascular diseases (CVD) contributing significantly to this burden. Timely identification of individuals at risk of heart disease is crucial for effective intervention and prevention strategies.

This project aims to develop a logistic regression model to predict the likelihood of heart disease occurrence based on various risk factors. By leveraging logistic regression, we seek to provide healthcare professionals with a reliable tool for early detection and intervention, ultimately reducing the morbidity and mortality associated with cardiovascular diseases.

## CONCLUSION

- Our project represents a significant advancement in real-time emotional recognition.
- By considering both facial emotions and age, the system enhances human-computer interaction.
- Leveraging cutting-edge AI/ML techniques and innovative system design, our goal is to create a versatile and efficient system for personalized user experiences across various domains and applications

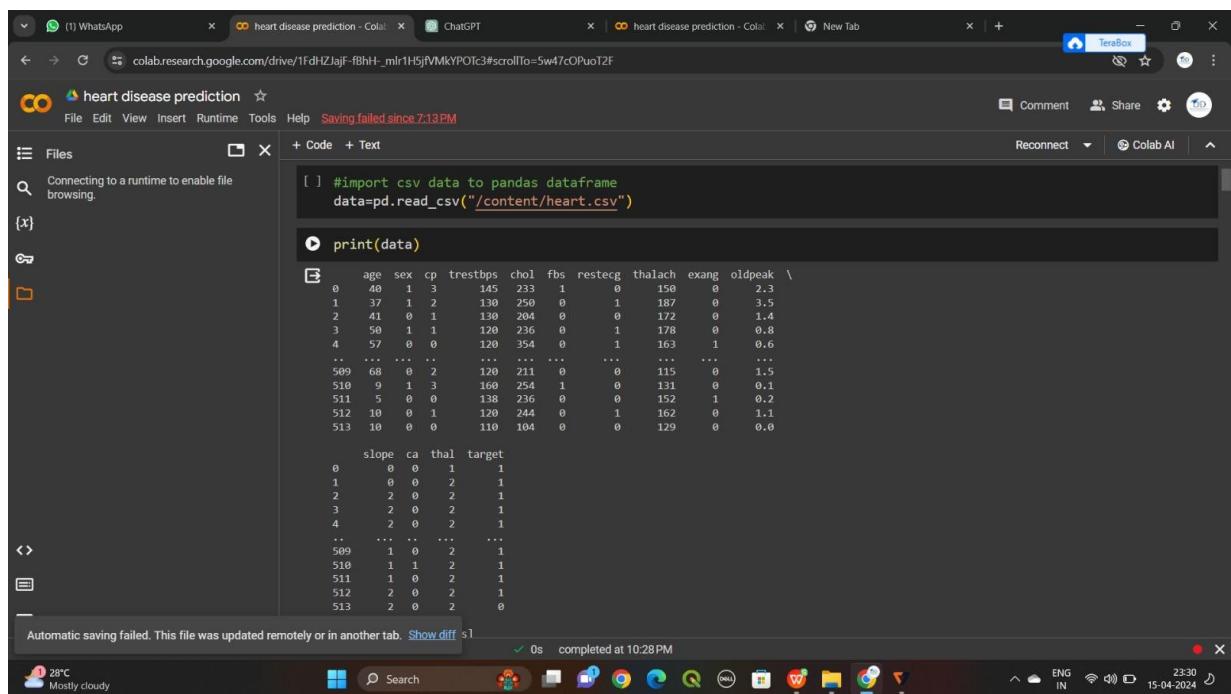
# CODE

## Dependency:

```
import numpy as np import pandas as pd import  
matplotlib.pyplot as plt import seaborn as sns from  
sklearn.model_selection import train_test_split from  
sklearn.linear_model import LogisticRegression  
from sklearn.metrics import accuracy_score
```

## DATA COLLECTION AND PROCESSING :

```
#import csv data to pandas dataframe  
data=pd.read_csv("/content/heart.csv")  
  
print(data)
```



The screenshot shows a Google Colab notebook titled "heart disease prediction". The code cell contains the following Python code:

```
[ ] #import csv data to pandas dataframe  
data=pd.read_csv("/content/heart.csv")  
  
[ ] print(data)
```

The output of the code is displayed below the cell, showing the first 13 rows of the dataset:

|     | age   | sex | cp   | trestbps | chol | fbfs | restecg | thalach | exang | oldpeak | \ |
|-----|-------|-----|------|----------|------|------|---------|---------|-------|---------|---|
| 0   | 40    | 1   | 3    | 145      | 233  | 1    | 0       | 150     | 0     | 2.3     |   |
| 1   | 37    | 1   | 2    | 130      | 250  | 0    | 1       | 187     | 0     | 3.5     |   |
| 2   | 41    | 0   | 1    | 130      | 204  | 0    | 0       | 172     | 0     | 1.4     |   |
| 3   | 58    | 1   | 1    | 120      | 236  | 0    | 1       | 178     | 0     | 0.8     |   |
| 4   | 57    | 0   | 0    | 120      | 354  | 0    | 1       | 163     | 1     | 0.6     |   |
| ..  | ..    | ..  | ..   | ..       | ..   | ..   | ..      | ..      | ..    | ..      |   |
| 509 | 68    | 0   | 2    | 120      | 211  | 0    | 0       | 115     | 0     | 1.5     |   |
| 510 | 9     | 1   | 3    | 160      | 254  | 1    | 0       | 131     | 0     | 0.1     |   |
| 511 | 5     | 0   | 0    | 138      | 236  | 0    | 0       | 152     | 1     | 0.2     |   |
| 512 | 10    | 0   | 1    | 120      | 244  | 0    | 1       | 162     | 0     | 1.1     |   |
| 513 | 10    | 0   | 0    | 110      | 104  | 0    | 0       | 129     | 0     | 0.0     |   |
|     | slope | ca  | thal | target   |      |      |         |         |       |         |   |
| 0   | 0     | 0   | 1    | 1        |      |      |         |         |       |         |   |
| 1   | 0     | 0   | 2    | 1        |      |      |         |         |       |         |   |
| 2   | 2     | 0   | 2    | 1        |      |      |         |         |       |         |   |
| 3   | 2     | 0   | 2    | 1        |      |      |         |         |       |         |   |
| 4   | 2     | 0   | 2    | 1        |      |      |         |         |       |         |   |
| ..  | ..    | ..  | ..   | ..       |      |      |         |         |       |         |   |
| 509 | 1     | 0   | 2    | 1        |      |      |         |         |       |         |   |
| 510 | 1     | 1   | 2    | 1        |      |      |         |         |       |         |   |
| 511 | 1     | 0   | 2    | 1        |      |      |         |         |       |         |   |
| 512 | 2     | 0   | 2    | 1        |      |      |         |         |       |         |   |
| 513 | 2     | 0   | 2    | 0        |      |      |         |         |       |         |   |

# STEP 1

## EDA

```
# Split the dataset into features and  
labels X = data.drop('sex',axis=1) y =  
data['sex']
```

```
# Split data into training and testing sets
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,  
random_state=42)
```

```
# Initialize the logistic regression model
```

```
model = LogisticRegression()
```

```
# Train the model model.fit(X_train,  
y_train)
```

```
# Predict gender for new data
```

```
# Assuming you have a new dataset 'new_data' with features  
# new_predictions = model.predict(new_data)
```

```

# Count the number of males and females in the 'gender' column

male_count = (data['sex'] == 1).sum() female_count = (data['sex']

== 0).sum()

print("Male count:", male_count) print("Female

count:", female_count)

```

```

File Edit View Insert Runtime Tools Help Saving failed since 7:13PM
File Edit View Insert Runtime Tools Help Reconnect Comment Share Colab AI
+ Code + Text
X = data.drop('sex',axis=1)
y = data['sex']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the logistic regression model
model = LogisticRegression()

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

# Predict gender for new data
# Assuming you have a new dataset 'new_data' with features
# new_predictions = model.predict(new_data)

# Count the number of males and females in the 'gender' column
male_count = (data['sex'] == 1).sum()
female_count = (data['sex'] == 0).sum()

print("Male count:", male_count)
print("Female count:", female_count)

Male count: 353
Female count: 161
Automatic saving failed. This file was updated remotely or in another tab. Show diff 3.10/dist-packages/sklearn/linear_model/logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):

```

Male count: 353

Female count: 161

## AFFECTEDS

```
# Split data into features and target  
variable X = data[['sex']] y = data['target']
```

```
# Split data into training and testing sets  
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Create logistic regression model  
log_reg = LogisticRegression()
```

```
# Train the model  
log_reg.fit(X_train, y_train)
```

```
# Initialize the logistic regression model  
model = LogisticRegression()
```

```
# Train the model  
model.fit(X_train, y_train)
```

```
# Count the number of males and females in the 'gender'  
column male_count = (data['sex'] == 1).sum() female_count  
= (data['sex'] == 0).sum()
```

```
print("Male count:", male_count)

print("Female count:", female_count)

# Predict probabilities
probabilities = log_reg.predict_proba(X_test)[:,1]

# Classify individuals as having the disease if their predicted probability is above 0.5
predictions = (probabilities > 0).astype(int)

# Combine predictions with actual gender for analysis
predictions_df = pd.DataFrame({'sex': X_test['sex'], 'prediction': predictions, 'actual': y_test})

# Count the number of males and females who are predicted to have the disease
male_disease_count = predictions_df[(predictions_df['sex'] == 1) &
                                      (predictions_df['prediction'] == 1)].shape[0]
female_disease_count = predictions_df[(predictions_df['sex'] ==
                                         0) & (predictions_df['prediction'] == 1)].shape[0]

print("Number of males with heart disease:", male_disease_count)
print("Number of females with heart disease:", female_disease_count)
```

```
# healthy male male=males_count-
males_disease_count

#healthy female female=females_count-
females_disease_count print( " Number of males
without heart disease:",male) print( " Number of
males without heart disease:",female)
```

The image shows two screenshots of the Google Colab interface side-by-side. Both screenshots display a Jupyter-style notebook with a sidebar on the left containing 'Files' and a main workspace with code cells.

**Top Screenshot:**

```

# Predict probabilities
probabilities = log_reg.predict_proba(X_test)[:,1]

# Classify individuals as having the disease if their predicted probability is above 0.5
predictions = (probabilities > 0).astype(int)

# Combine predictions with actual gender for analysis
predictions_df = pd.DataFrame({'sex': X_test['sex'], 'prediction': predictions, 'actual': y_test})

# Count the number of males and females who are predicted to have the disease
male_disease_count = predictions_df[(predictions_df['sex'] == 1) & (predictions_df['prediction'] == 1)].shape[0]
female_disease_count = predictions_df[(predictions_df['sex'] == 0) & (predictions_df['prediction'] == 1)].shape[0]

print("Number of males with heart disease:", male_disease_count)
print("Number of females with heart disease:", female_disease_count)

# healthy male
male=males_count-male_disease_count
#healthy female
female=females_count-female_disease_count
print( " Number of males without heart disease:" ,male)
print( " Number of females without heart disease:" ,female)

```

**Bottom Screenshot:**

```

# Split data into features and target variable
X = data[['sex']]
y = data['target']
# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Create logistic regression model
log_reg = LogisticRegression()
# Train the model
log_reg.fit(X_train, y_train)
# Initialize the logistic regression model
model = LogisticRegression()
# Train the model
model.fit(X_train, y_train)
# Count the number of males and females in the 'gender' column
male_count = (data['sex'] == 1).sum()
female_count = (data['sex'] == 0).sum()
print("Male count:", male_count)
print("Female count:", female_count)

# Predict probabilities
probabilities = log_reg.predict_proba(X_test)[:,1]

# Classify individuals as having the disease if their predicted probability is above 0.5
predictions = (probabilities > 0).astype(int)

```

```

heart disease prediction
File Edit View Insert Runtime Tools Help Saving failed since 7:13PM
Reconnect Comment Share Settings
Files + Code + Text
Connecting to a runtime to enable file browsing.
[x]
[ ] # Count the number of males and females who are predicted to have the disease
male_disease_count = predictions_df[(predictions_df['sex'] == 1) & (predictions_df['prediction'] == 1)].shape[0]
female_disease_count = predictions_df[(predictions_df['sex'] == 0) & (predictions_df['prediction'] == 1)].shape[0]

print("Number of males with heart disease:", male_disease_count)
print("Number of females with heart disease:", female_disease_count)

# healthy male
male=male_count-male_disease_count
#healthy female
female=female_count-female_disease_count
print( " Number of males without heart disease:",male)
print( " Number of males without heart disease:",female)

Male count: 353
Female count: 161
Number of males with heart disease: 73
Number of females with heart disease: 30
Number of males without heart disease: 280
Number of males without heart disease: 131

[ ] # Assuming you have a DataFrame called 'data' with columns 'gender' and 'target'
# where gender is represented as 1 for male and 0 for female, and target represents
# 1 for unhealthy heart and 0 for healthy heart
Automatic saving failed. This file was updated remotely or in another tab. Show diff 0s completed at 10:28PM

```

Creating the features and target

```

x=data.drop(columns='target',axis=1) y=data['target'] # to
print x print(x) age sex cp trestbps chol fbs restecg
thalach exang oldpeak \

```

|     |     |     |    |     |     |     |     |     |     |          |
|-----|-----|-----|----|-----|-----|-----|-----|-----|-----|----------|
| 1   | 40  | 1   | 3  | 145 | 233 | 1   | 0   | 150 | 0   | 2.3      |
| 2   | 37  | 1   | 2  | 130 | 250 | 0   | 1   | 187 | 0   | 3.5      |
| 3   | 41  | 0   | 1  | 130 | 204 | 0   | 0   | 172 | 0   | 1.4      |
| 4   | 50  | 1   | 1  | 120 | 236 | 0   | 1   | 178 | 0   | 0.8 4 57 |
|     | 0   | 0   |    | 120 | 354 | 0   | 1   | 163 | 1   | 0.6      |
| ..  | ... | ... | .. | ... | ... | ... | ... | ... | ... | ...      |
| 509 | 68  | 0   | 2  | 120 | 211 | 0   | 0   | 115 | 0   | 1.5      |

|     |    |   |   |     |     |   |   |     |   |     |     |
|-----|----|---|---|-----|-----|---|---|-----|---|-----|-----|
| 510 | 9  | 1 | 3 | 160 | 254 | 1 | 0 | 131 | 0 | 0.1 |     |
| 511 | 5  | 0 | 0 | 138 | 236 | 0 | 0 | 152 | 1 | 0.2 |     |
| 512 | 10 | 0 | 1 | 120 | 244 | 0 | 1 | 162 | 0 | 1.1 | 513 |
|     | 10 | 0 | 0 | 110 | 104 | 0 | 0 | 129 | 0 | 0.0 |     |

slope ca thal

|     |     |     |     |     |   |   |   |  |  |  |  |
|-----|-----|-----|-----|-----|---|---|---|--|--|--|--|
| 0   | 0   | 0   | 1   |     |   |   |   |  |  |  |  |
| 1   | 0   | 0   | 2   |     |   |   |   |  |  |  |  |
| 2   | 2   | 0   | 2   |     |   |   |   |  |  |  |  |
| 3   | 2   | 0   | 2   |     |   |   |   |  |  |  |  |
| 4   | 2   | 0   | 2   |     |   |   |   |  |  |  |  |
| ..  | ... | ... | ... |     |   |   |   |  |  |  |  |
| 509 | 1   | 0   | 2   |     |   |   |   |  |  |  |  |
| 510 | 1   | 1   | 2   |     |   |   |   |  |  |  |  |
| 511 | 1   | 0   | 2   |     |   |   |   |  |  |  |  |
| 512 | 2   | 0   | 2   | 513 | 2 | 0 | 2 |  |  |  |  |

[514 rows x 13 columns]

# to print y print(y)

1 1

|     |   |    |
|-----|---|----|
| 2   | 1 |    |
| 3   | 1 |    |
| 4   | 1 |    |
| 5   | 1 | .. |
| 509 | 1 |    |
| 510 | 1 |    |
| 511 | 1 |    |
| 512 | 1 |    |
| 513 | 0 |    |

**Name: target, Length: 514, dtype: int64**

```

heart disease prediction
File Edit View Insert Runtime Tools Help Saving failed since 7:13PM
Reconnect Comment Share Colab AI
[ ] 2 2 0 2
[ ] 3 2 0 2
[ ] 4 2 0 2
[ ] .. ...
[ ] 509 1 0 2
[ ] 510 1 1 2
[ ] 511 1 0 2
[ ] 512 2 0 2
[ ] 513 2 0 2
[ ] [514 rows x 13 columns]

[ ] # to print y
[ ] print(y)
[ ] 0 1
[ ] 1 1
[ ] 2 1
[ ] 3 1
[ ] 4 1
[ ] ..
[ ] 509 1
[ ] 510 1
[ ] 511 1
[ ] 512 1
[ ] 513 0
[ ] Name: target, Length: 514, dtype: int64

splitting the data into training data and test data

Automatic saving failed. This file was updated remotely or in another tab. Show diff 0s completed at 10:28PM 23:38 ENG IN 15-04-2024

heart disease prediction
File Edit View Insert Runtime Tools Help Saving failed since 7:13PM
Reconnect Comment Share Colab AI
[ ] 2 2 0 2
[ ] 3 2 0 2
[ ] 4 2 0 2
[ ] .. ...
[ ] 509 1 0 2
[ ] 510 1 1 2
[ ] 511 1 0 2
[ ] 512 2 0 2
[ ] 513 2 0 2
[ ] [514 rows x 13 columns]

[ ] # to print x
[ ] print(x)

[ ] x=data.drop(columns='target',axis=1)
[ ] y=data['target']
[ ] # to print x
[ ] print(x)

[ ] 0 40 1 3 145 233 1 0 150 0 2.3
[ ] 1 37 1 2 130 250 0 1 187 0 3.5
[ ] 2 41 0 1 130 204 0 0 172 0 1.4
[ ] 3 50 1 1 120 236 0 1 178 0 0.8
[ ] 4 57 0 0 120 354 0 1 163 1 0.6
[ ] ..
[ ] 509 68 0 2 120 211 0 0 115 0 1.5
[ ] 510 9 1 3 160 254 1 0 131 0 0.1
[ ] 511 5 0 0 138 236 0 0 152 1 0.2
[ ] 512 10 0 1 120 244 0 1 162 0 1.1
[ ] 513 10 0 0 110 104 0 0 129 0 0.0

[ ] slope ca thal
[ ] 0 0 0 1
[ ] 1 0 0 2
[ ] 2 2 0 2
[ ] 3 2 0 2
[ ] 4 2 0 2
[ ] ..
[ ] 509 1 0 2
[ ] 510 1 1 2
[ ] ..

Automatic saving failed. This file was updated remotely or in another tab. Show diff 0s completed at 10:28PM 23:38 ENG IN 15-04-2024

```

splitting the data into training data and test data

```

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,stratify=y,random_state=42)

print(x.shape,x_train.shape,x_test.shape) op

```

(514, 13) (411, 13) (103, 13)

## Step 4

### MOdel training

logistic regression

```
model=LogisticRegression()
```

```
#training the logistic regression model with training
```

```
data model.fit(x_train,y_train) op
```

```
/usr/local/lib/python3.10/distpackages/sklearn/linear_model/  
_logistic.py:458:
```

```
ConvergenceWarning: lbfsgs 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:

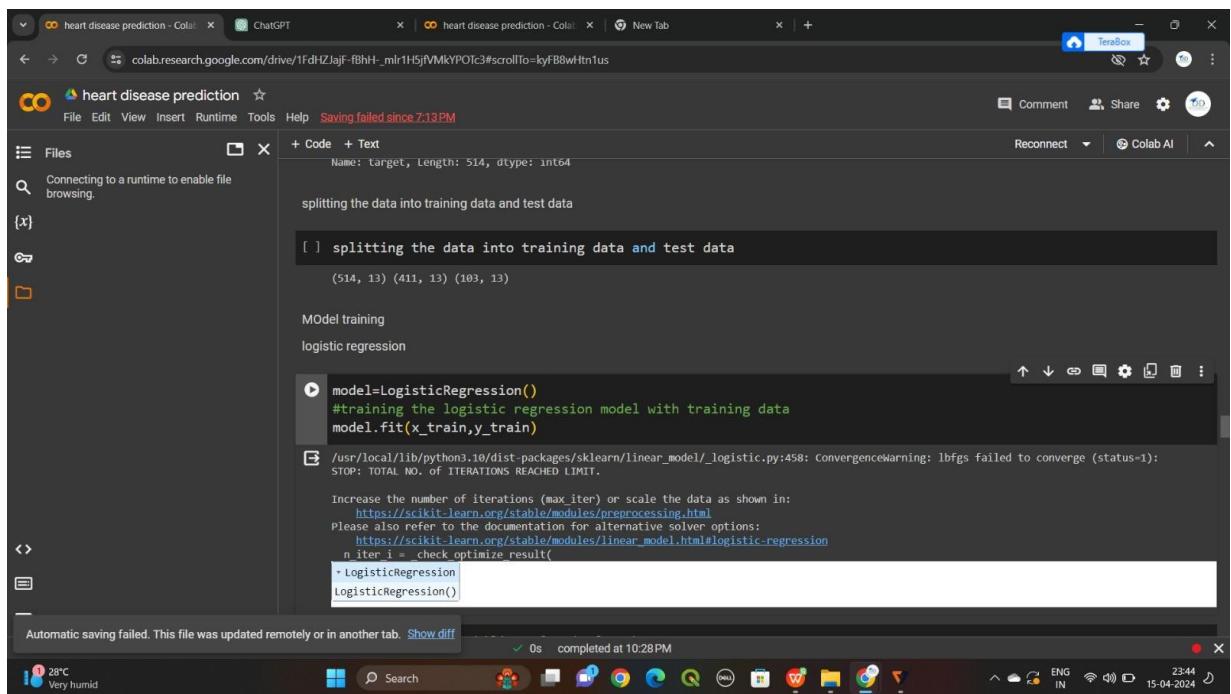
<https://scikit-learn.org/stable/modules/preprocessing.html>

Please also refer to the documentation for alternative solver options:

[https://scikit-learn.org/stable/modules/linear\\_model.html#logistic-regression](https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression) n\_iter\_i = \_check\_optimize\_result()

## LogisticRegression

### LogisticRegression()



```
heart disease prediction
File Edit View Insert Runtime Tools Help Saving failed since 7:13PM
+ Code + Text
Name: target, Length: 514, dtype: int64
splitting the data into training data and test data
[ ] splitting the data into training data and test data
(514, 13) (411, 13) (103, 13)
MOdel training
logistic regression
model=LogisticRegression()
#training the logistic regression model with training data
model.fit(x_train,y_train)
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: 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()
LogisticRegression()

Automatic saving failed. This file was updated remotely or in another tab. Show diff
0s completed at 10:28PM
23:44 15-04-2024
```

## Create function for evaluating matrix

#accuracy of training data

```
x_train_prediction=model.predict(x_train)
```

```
training_data_accuracy=accuracy_score(x_train_prediction,y_train)
print('tarining data accuracy:',training_data_accuracy)
```

tarining data accuracy: 0.8442822384428224

```
#accuracy of test data
```

```
x_test_prediction=model.predict(x_test)
```

```
test_data_accuracy=accuracy_score(x_test_prediction,y_test)
print('test data accuracy:',test_data_accuracy)
```

st data accuracy:

0.8640776699029126

```
#accuracy of training data
x_train_prediction=model.predict(x_train)
training_data_accuracy=accuracy_score(x_train_prediction,y_train)
print('taining data accuracy:',training_data_accuracy)

taining data accuracy: 0.8442822384428224

[108] #accuracy of test data
x_test_prediction=model.predict(x_test)
test_data_accuracy=accuracy_score(x_test_prediction,y_test)
print('test data accuracy:',test_data_accuracy)

test data accuracy: 0.8640776699029126
```

Looking at the evaluation metrics for our best model

```
input_data=(28,1,3,110,211,0,0,144,1,1.8,1,0,2) #
```

change the input data to numpy array

```
input_data_as_numpy_array=np.asarray(input_data)
```

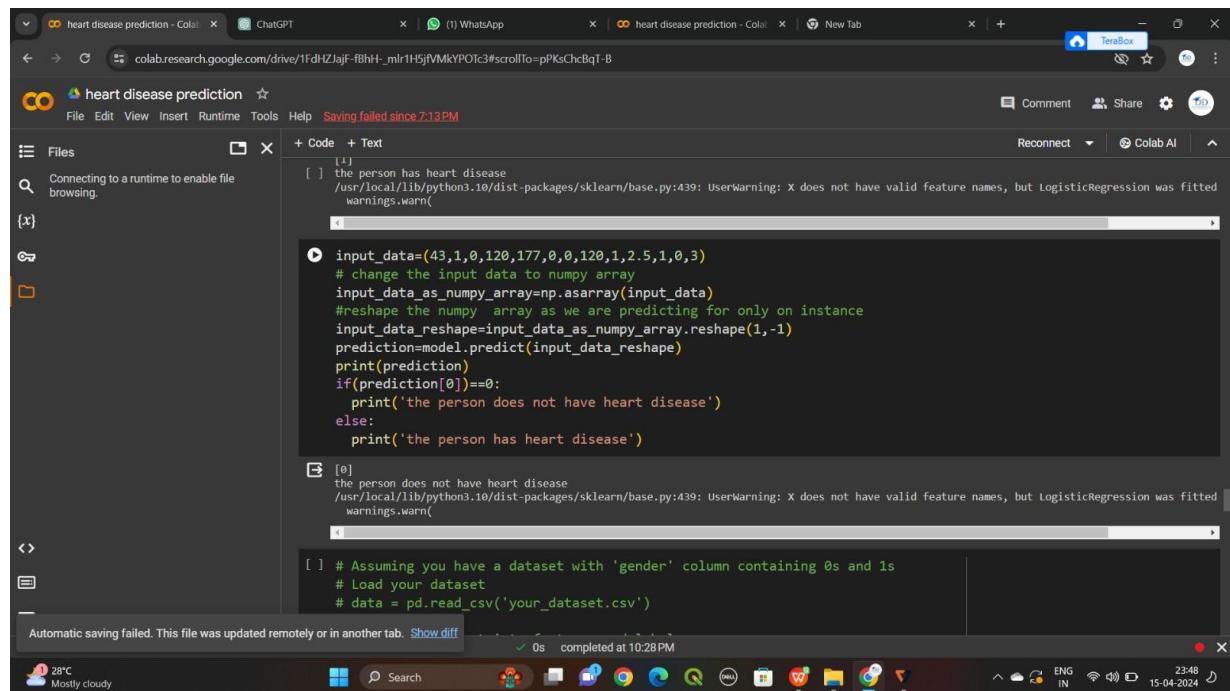
```
#reshape the numpy array as we are predicting for only on
```

instance

```
input_data_reshape=input_data_as_numpy_array.reshape(1,-1)
prediction=model.predict(input_data_reshape) print(prediction)
if(prediction[0])==0:
    print('the person does not have heart
disease') else: print('the person has heart
disease')
```

[1] the person has heart

disease



```
+ Code + Text
[1]
the person has heart disease
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but LogisticRegression was fitted
warnings.warn(
[2]
input_data=(43,1,0,120,177,0,0,120,1,2.5,1,0,3)
# change the input data to numpy array
input_data_as_numpy_array=np.asarray(input_data)
#reshape the numpy array as we are predicting for only one instance
input_data_reshape=input_data_as_numpy_array.reshape(1,-1)
prediction=model.predict(input_data_reshape)
print(prediction)
if(prediction[0]==0:
    print('the person does not have heart disease')
else:
    print('the person has heart disease')
[3]
the person does not have heart disease
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but LogisticRegression was fitted
warnings.warn(
[4]
# Assuming you have a dataset with 'gender' column containing 0s and 1s
# Load your dataset
# data = pd.read_csv('your_dataset.csv')
```

Lets save our model using pickle

Welcome to Colaboratory

```
import pickle  
# Assuming 'log_reg' is your trained logistic Regression model  
# Save the model to a file  
with open('logistic_regression_model.pkl', 'wb') as file:  
    pickle.dump(log_reg, file)  
from google.colab import files  
files.download('logistic_regression_model.pkl')  
  
!pip install streamlit pyngrok==4.1.1  
# Download and install ngrok  
!wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-amd64.zip  
!unzip ngrok-stable-linux-amd64.zip
```

Connected to Python 3 Google Compute Engine backend

## Import streamlit ,pyngrok, and ngrok modules

```
!pip install streamlit pyngrok==4.1.1  
# Download and install Ngrok  
!wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-amd64.zip  
!unzip ngrok-stable-linux-amd64.zip  
  
Collecting streamlit  
  Downloading streamlit-1.33.0-py2.py3-none-any.whl (8.1 MB)  
  8.1/8.1 MB 16.0 MB/s eta 0:00:00  
Collecting pyngrok==4.1.1  
  Downloading pyngrok-4.1.1.tar.gz (18 kB)  
  Preparing metadata (setup.py) ... done  
Requirement already satisfied: future in /usr/local/lib/python3.10/dist-packages (from pyngrok==4.1.1) (0.18.3)  
Requirement already satisfied: gitpython!=3.1.19,<4,>=3.0.7 (from streamlit)
```

```

Collecting gitpython>=3.1.19,<4.0.0
  Downloading GitPython-3.1.43-py3-none-any.whl (207 kB)
Collecting pydeck<1.0,>0.8.0b4
  Downloading pydeck-0.8.1b0-py3-none-any.whl (4.8 MB)
Requirement already satisfied: tornado<7.0,>=6.0.3 in /usr/local/lib/python3.10/dist-packages (from streamlit) (6.3.3)
Collecting watchdog>=2.1.5
  Downloading Watchdog-2.1.5-py3-none-any.whl (9.6 kB)
Requirement already satisfied: gitdb<5.0,>=4.0.1 (from gitpython>=3.1.19,<4.0.0)
  Downloading gitdb-4.0.11-py3-none-any.whl (62 kB)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas<3.0,>=1.3.0>streamlit) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas<3.0,>=1.3.0>streamlit) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas<3.0,>=1.3.0>streamlit) (2024.1)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3.0,>=2.27>streamlit) (3.3.2)
Requirement already satisfied: idna<4.0,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3.0,>=2.27>streamlit) (3.6)
Requirement already satisfied: urllib3<3.0,>=2.1.1 in /usr/local/lib/python3.10/dist-packages (from requests<3.0,>=2.27>streamlit) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3.0,>=2.27>streamlit) (2024.2.2)
Requirement already satisfied: markdown-it-py>=2.0 in /usr/local/lib/python3.10/dist-packages (from richc14>=10.14.0>streamlit) (3.0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (from richc14>=10.14.0>streamlit) (2.16.1)
Collecting s mmap6,>=3.0.0
  Downloading s mmap-5.0.1-py3-none-any.whl (24 kB)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from Jinja2>=2.11.0>streamlit) (2.1.5)
Requirement already satisfied: attrs>=22.2.0 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=3.0.0>streamlit) (23.1.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=3.0.0>streamlit) (2023.03.6)
Requirement already satisfied: referencing>=0.28.4 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=3.0.0>streamlit) (0.28.4)

```

Connected to Python 3 Google Compute Engine backend

```

Requirement already satisfied: rpsd-py>=0.7.1 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=3.0.0>streamlit) (0.7.1)
Requirement already satisfied: mdurl>=0.1 in /usr/local/lib/python3.10/dist-packages (from markdown-it-py>=2.2.0>streamlit) (0.1.0)
Requirement already satisfied: six<1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2>pandas<3.0,>=1.3.0>streamlit) (1.5.4)
Building wheels for collected packages: pyngrok
  Building wheel for pyngrok (setup.py) ...
  Created wheel for pyngrok: filename=pyngrok-4.1.1-py3-none-any.whl size=15964 sha256=6d94e98bf5fa632e312cf3397a06618d3ddecd8c81c46ab35ee2
  Stored in directory: /root/.cache/pip/wheels/4c/7c/4c/632fb2e8a88d8890102eb07bc922e1ca8fa14db5962c01a8
Successfully built pyngrok
Installing collected packages: watchdog, s mmap, pyngrok, pydeck, gitdb, gitpython, streamlit
Successfully installed gitdb-4.0.11-gitpython-3.1.43 pydeck-0.8.1b0 pyngrok-4.1.1 s mmap-5.0.1 streamlit-1.33.0 watchdog-4.0.0
--2024-04-15 17:24:26 - https://bin.equinox.io/c/AVD/A7a1ab/ngrok-stable-linux-amd64.zip
Resolving bin.equinox.io (bin.equinox.io)... 54.161.241.46, 52.202.168.65, 54.237.133.81, ...
Connecting to bin.equinox.io (bin.equinox.io)[54.161.241.46]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 13921656 (13M) [application/octet-stream]
Saving to: "ngrok-stable-linux-amd64.zip"
ngrok-stable-linux: 100%[=====] 13.28M 55.8MB/s in 0.2s
2024-04-15 17:24:20 (55.8 MB/s) - 'ngrok-stable-linux-amd64.zip' saved [13921656/13921656]

[43]: # Write your Streamlit app
%%writefile my_app.py
import streamlit as st

def main():
    st.title("My Streamlit App")
    st.write("Hello, this is my first Streamlit app!")


```

Connected to Python 3 Google Compute Engine backend

Welcome to Colaboratory

File Edit View Insert Runtime Tools Help Cannot save changes

Files

- sample\_data
- logistic\_regression\_model.pkl
- my\_app.py
- ngrok
- ngrok-stable-linux-amd64.zip

+ Code + Text Copy to Drive

```
Stored in directory: /root/.cache/pip/wheels/4c/7c/4c/632fba2ea8e88d8890102e07bc92e1ca8fa14db590zc91a8
Successfully built pyngrok
Installing collected packages: watchdog, s mmap, pyngrok, pydeck, gitdb, gipython, streamlit
Successfully installed gitdb-4.0.11 gipython-3.1.43 pydeck-0.8.1b0 pyngrok-4.1.1 s mmap-5.0.1 streamlit-1.33.0 watchdog-4.0.0
-- 2024-04-15 17:24:20 -- https://bin.equinox.io/c/4VmDzA7ihB/ngrok-stable-linux-amd64.zip
Resolving bin.equinox.io (bin.equinox.io)... 54.161.241.46, 52.202.168.65, 54.237.133.81, ...
Connecting to bin.equinox.io (bin.equinox.io)|54.161.241.46|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 13921656 (13M) [application/octet-stream]
Saving to: 'ngrok-stable-linux-amd64.zip'

ngrok-stable-linux- 100%[=====] 13.28M 55.8MB/s in 0.2s
2024-04-15 17:24:20 (55.8 MB/s) - 'ngrok-stable-linux-amd64.zip' saved [13921656/13921656]

Archive: ngrok-stable-linux-amd64.zip
inflating: ngrok
```

```
[43]: # Write your Streamlit app
%writefile my_app.py
import streamlit as st

def main():
    st.title("My Streamlit App")
    st.write("Hello, this is my first Streamlit app!")

if __name__ == "__main__":
    main()

# Run Streamlit app with Ngrok
get_ipython().system_raw('./ngrok http 8501 &')
!curl -s http://localhost:4040/api/tunnels | python3 -c \
"import sys, json; print(json.load(sys.stdin)['tunnels'][0]['public_url'])"
!streamlit run my_app.py
```

Connected to Python 3 Google Compute Engine backend

Disk 81.38 GB available

85°F Partly cloudy

ENG IN 10:57 PM 4/15/2024

Welcome to Colaboratory

File Edit View Insert Runtime Tools Help Cannot save changes

Files

- sample\_data
- logistic\_regression\_model.pkl
- my\_app.py
- ngrok
- ngrok-stable-linux-amd64.zip

+ Code + Text Copy to Drive

```
inflating: ngrok
```

```
[43]: # Write your Streamlit app
%writefile my_app.py
import streamlit as st

def main():
    st.title("My Streamlit App")
    st.write("Hello, this is my first Streamlit app!")

if __name__ == "__main__":
    main()

# Run Streamlit app with Ngrok
get_ipython().system_raw('./ngrok http 8501 &')
!curl -s http://localhost:4040/api/tunnels | python3 -c \
"import sys, json; print(json.load(sys.stdin)['tunnels'][0]['public_url'])"
!streamlit run my_app.py

Writing my_app.py
```

```
[37]: import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

# Assuming you have a DataFrame called 'data' with columns 'sex' and 'target'
# where sex is represented as 1 for male and 0 for female, and target represents
```

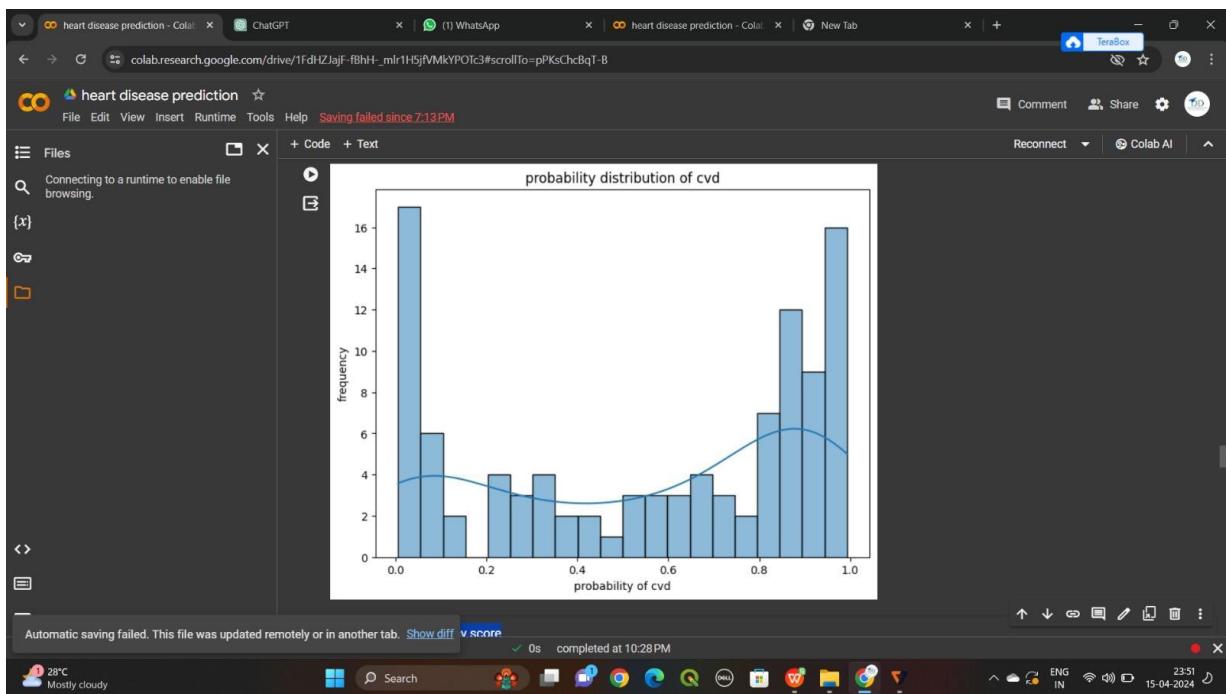
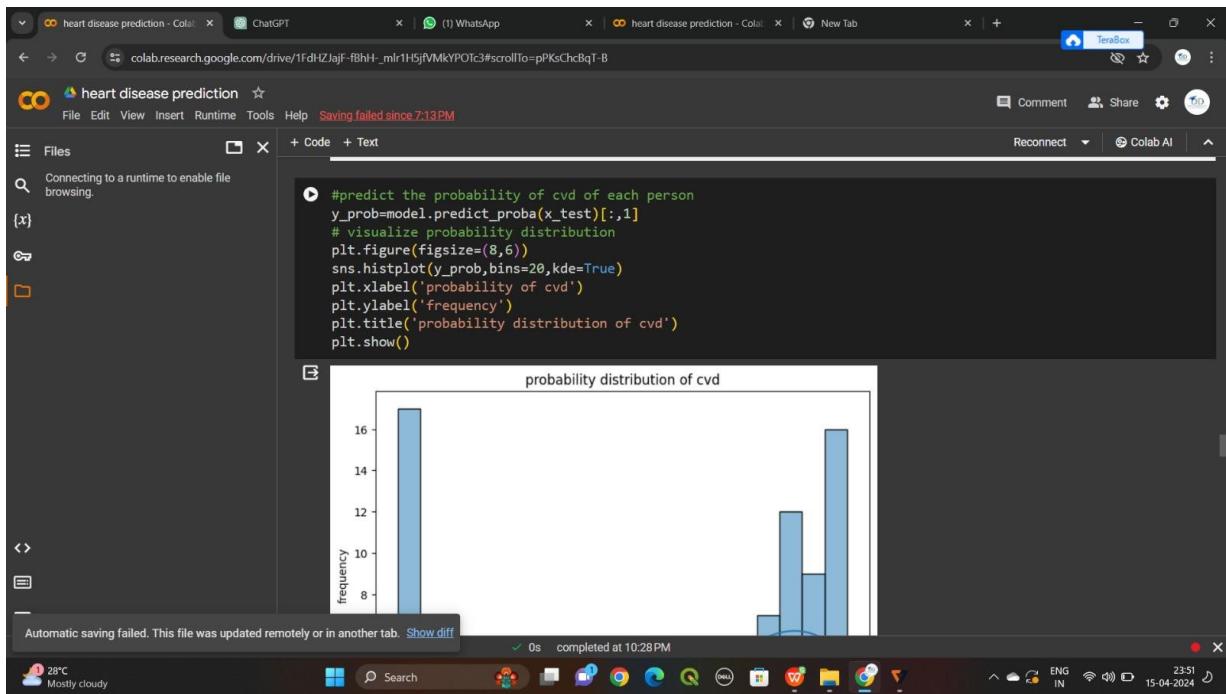
Connected to Python 3 Google Compute Engine backend

Disk 81.38 GB available

85°F Partly cloudy

ENG IN 10:57 PM 4/15/2024

## Probability of heart disease of each person



LINK : <https://youtu.be/XouSAUr8IVg>