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

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

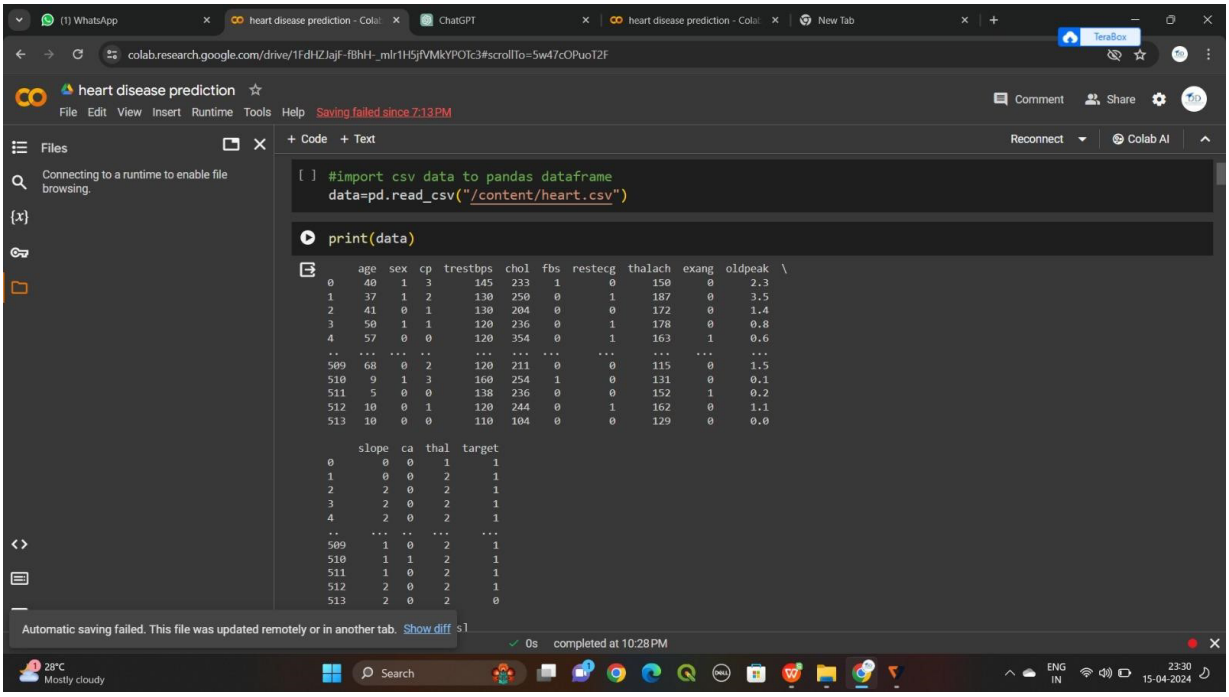
sklearn.linear_model import LogisticRegression

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)



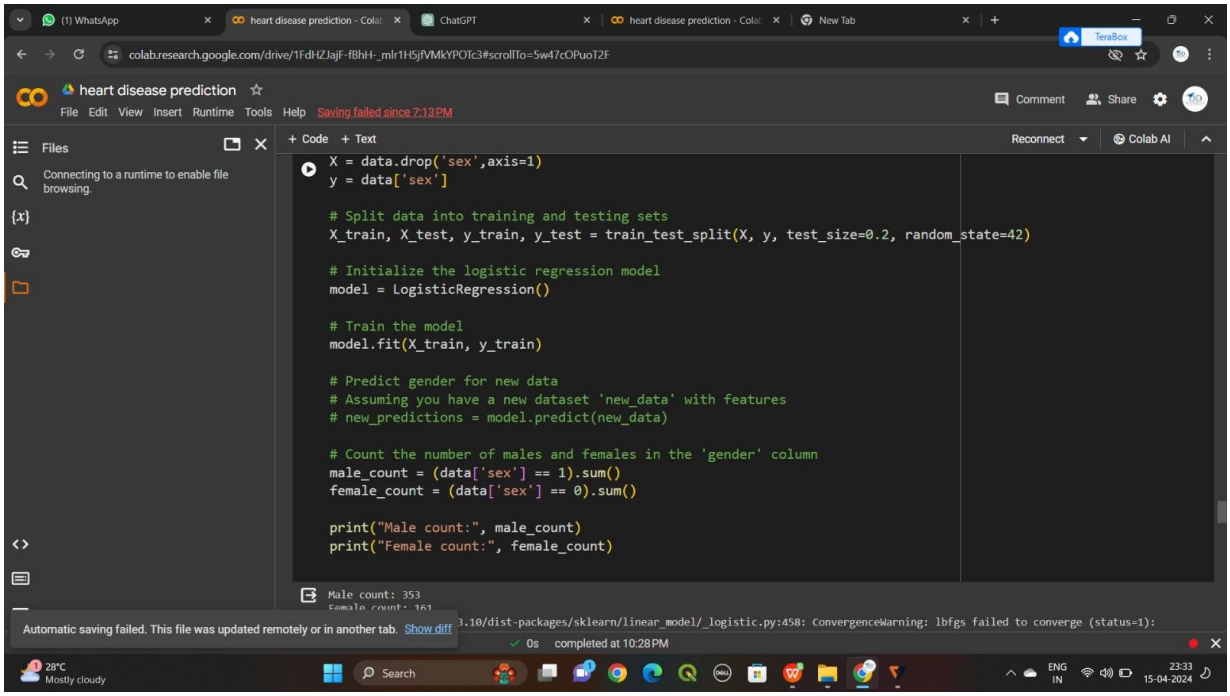
STEP 1

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)



Male count: 353

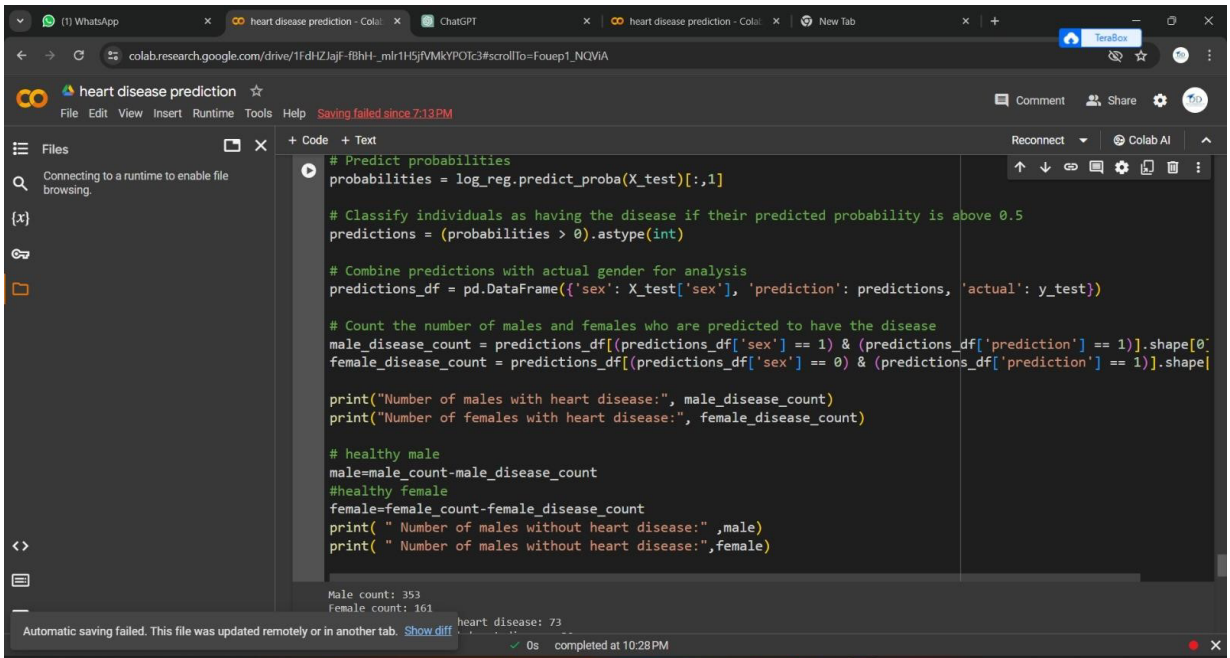
Female count: 161

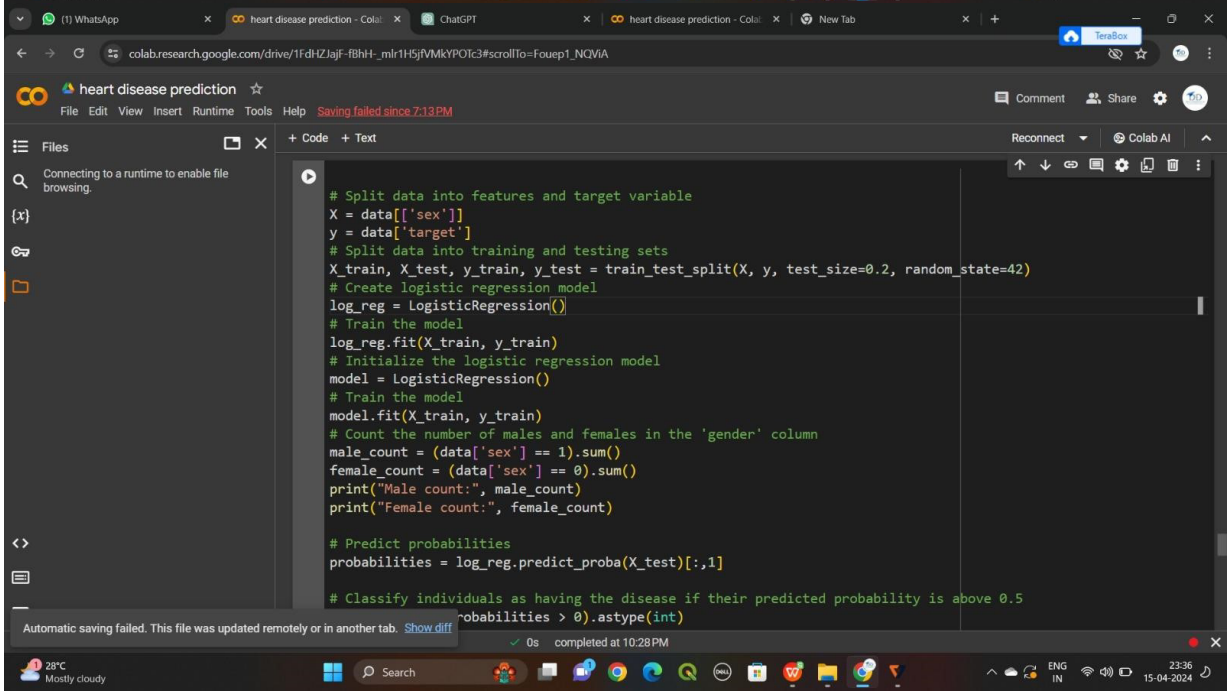
AFFECTEDS

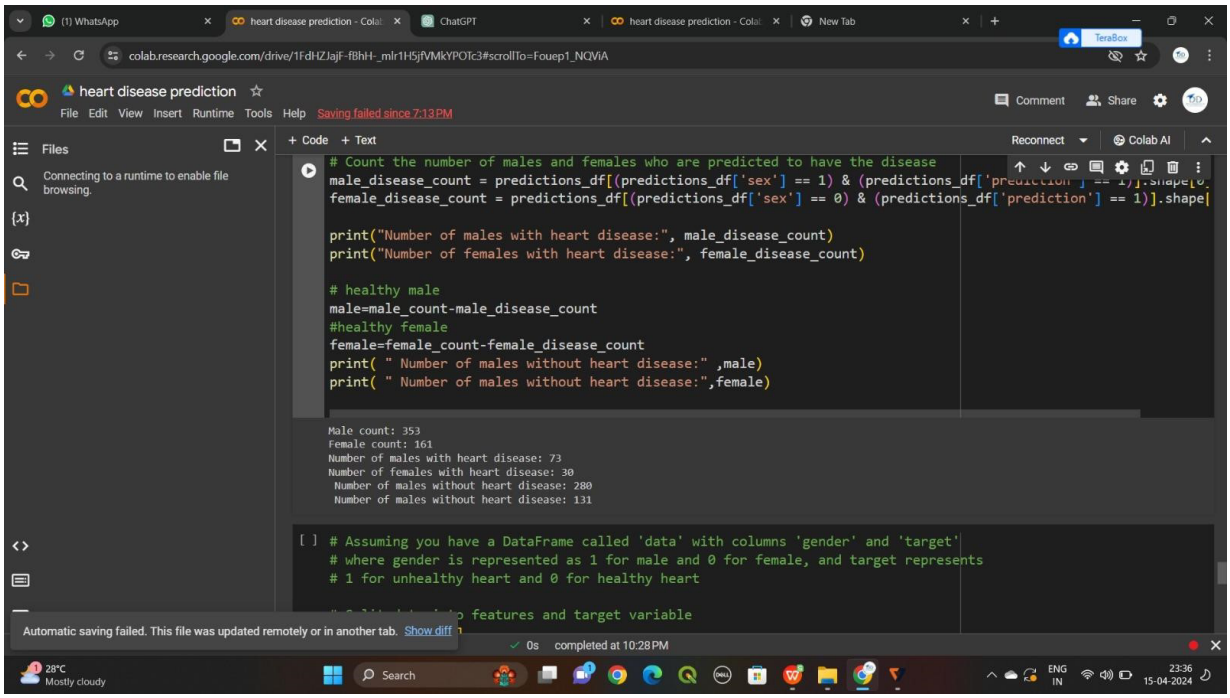
```
# Split data into features and target

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=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)
```







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

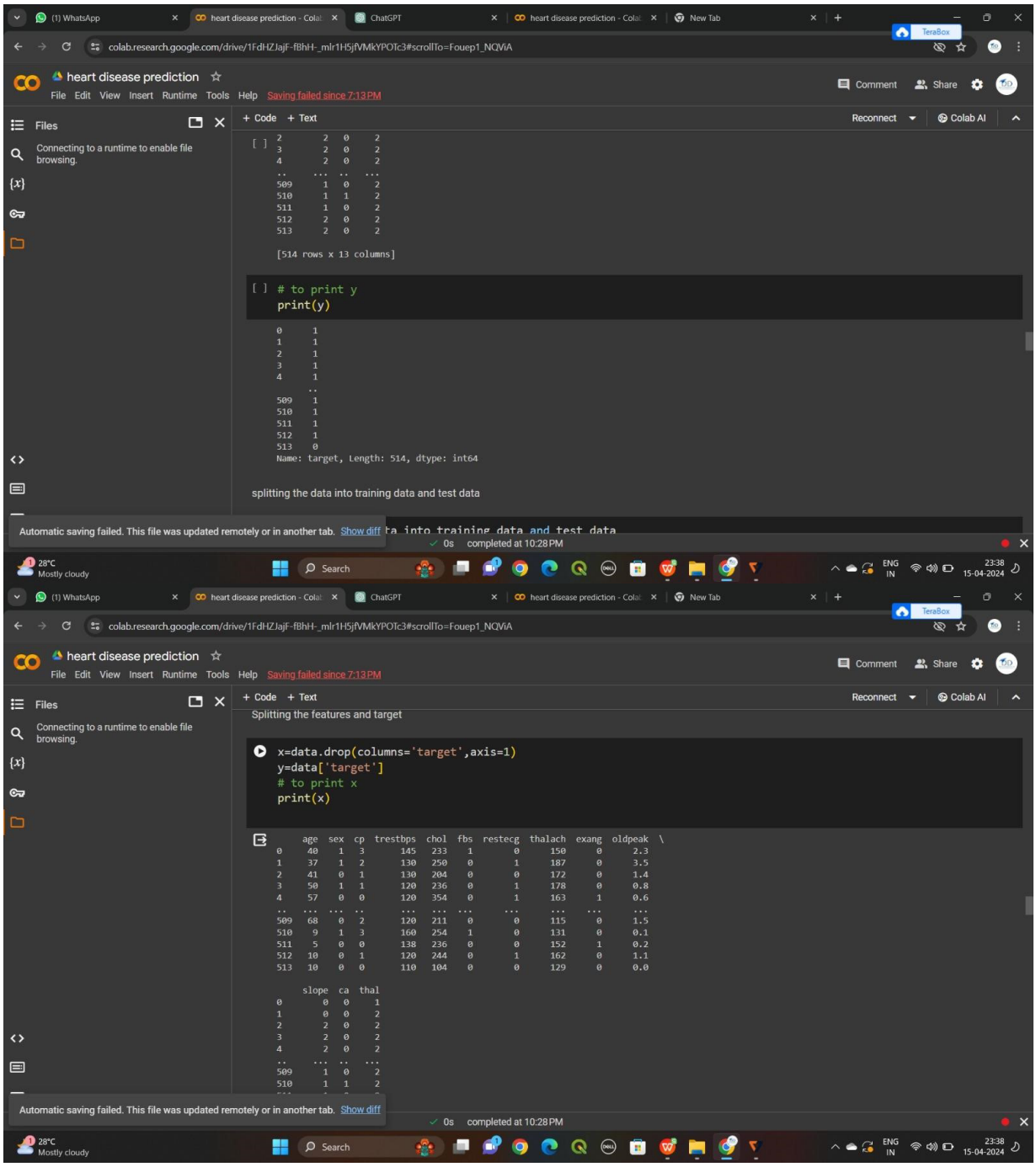
510 1

511 1

512 1

513 0

Name: target, Length: 514, dtype: int64



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,st

rat ify=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: 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

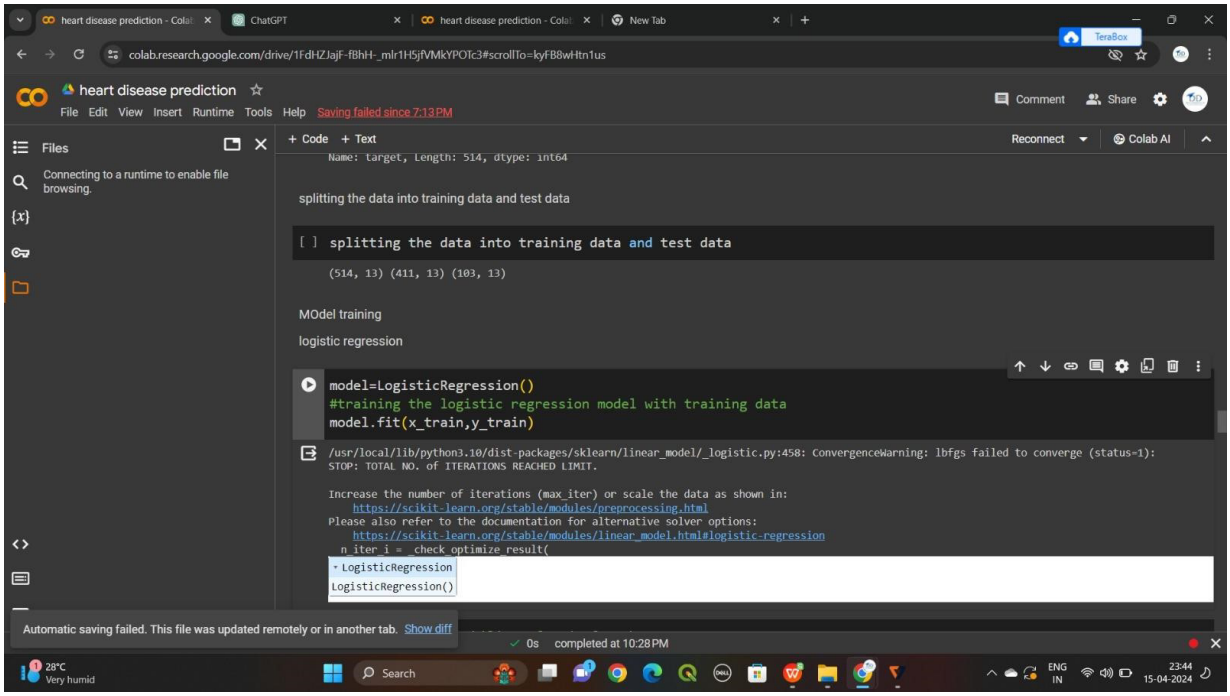
Please also refer to the documentation for alternative solver options:

https://scikitlearn.org/stable/modules/linear_model.html#logi

stic-regression n_iter_i = _check_optimize_result(

LogisticRegression

LogisticRegression()



Create fuction for evaluating matrix

#accuracy of training data

x_train_prediction=model.predict(x_train)

training_data_accuracy=accuracy_score(x_train_prediction,y_tr

ai n) print('tarining data accuracy:',training_data_accuracy)

tarining data accuracy: 0.8442822384428224

#accuracy of test data

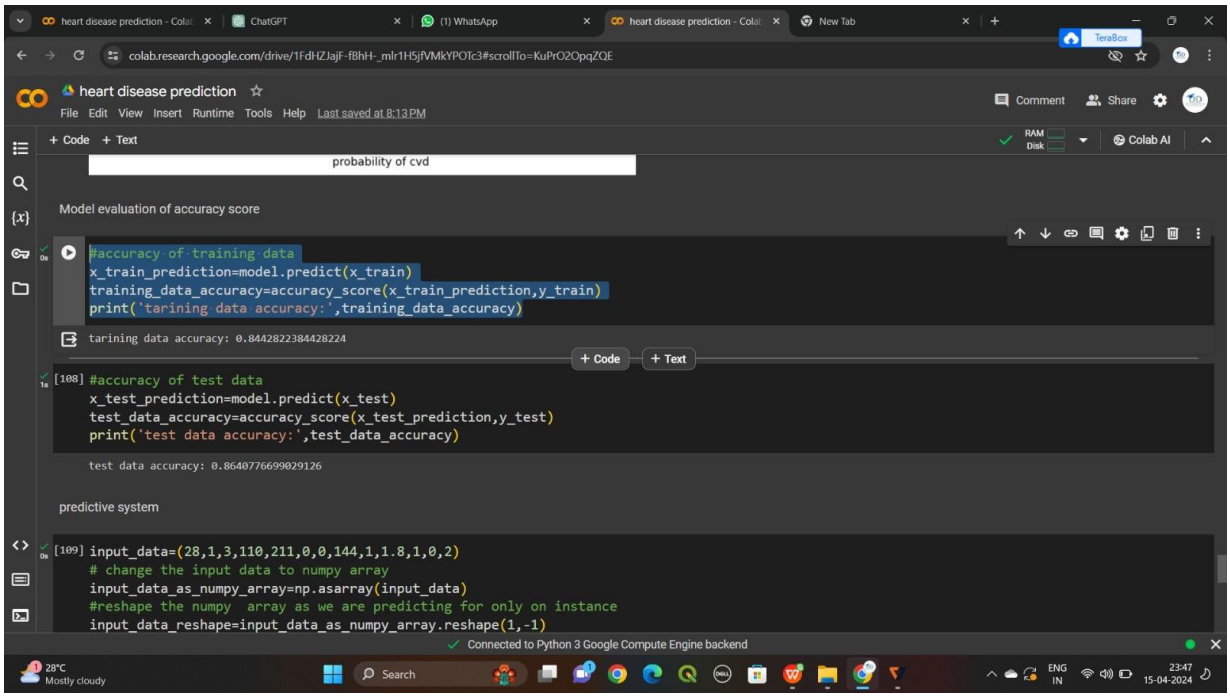
x_test_prediction=model.predict(x_test)

test_data_accuracy=accuracy_score(x_test_prediction,y_t

est) print('test data accuracy:',test_data_accuracy) st 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:

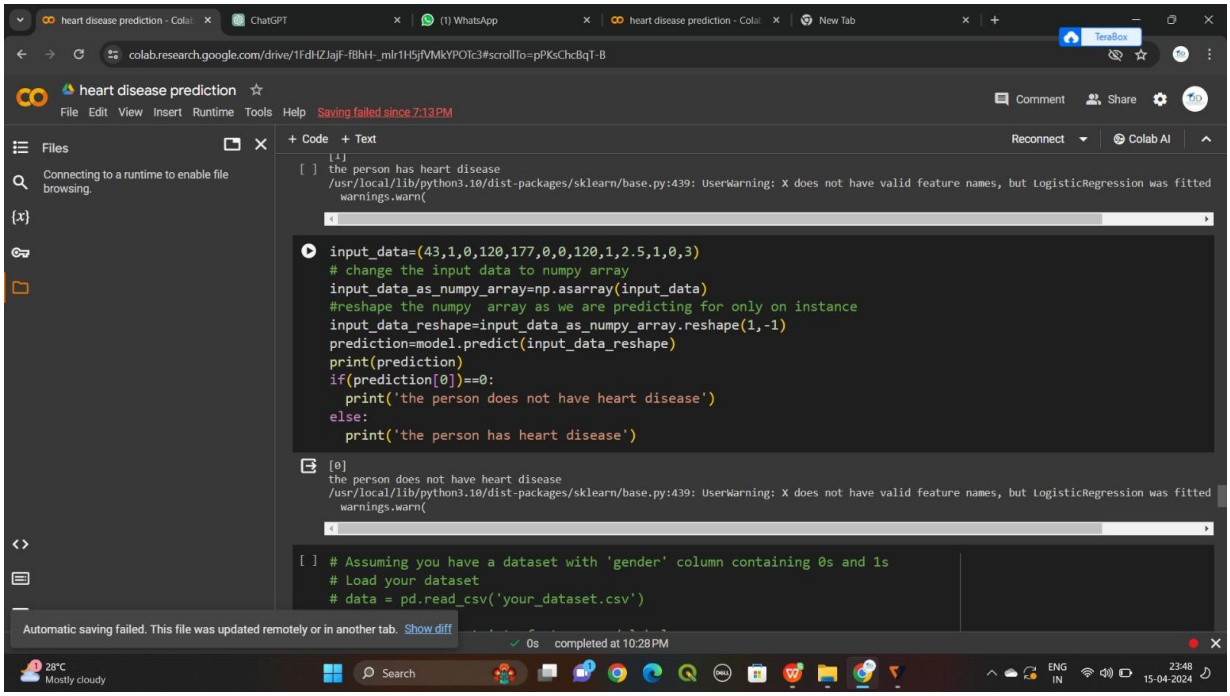
print('the person does not have heart

disease') else: print('the person has heart

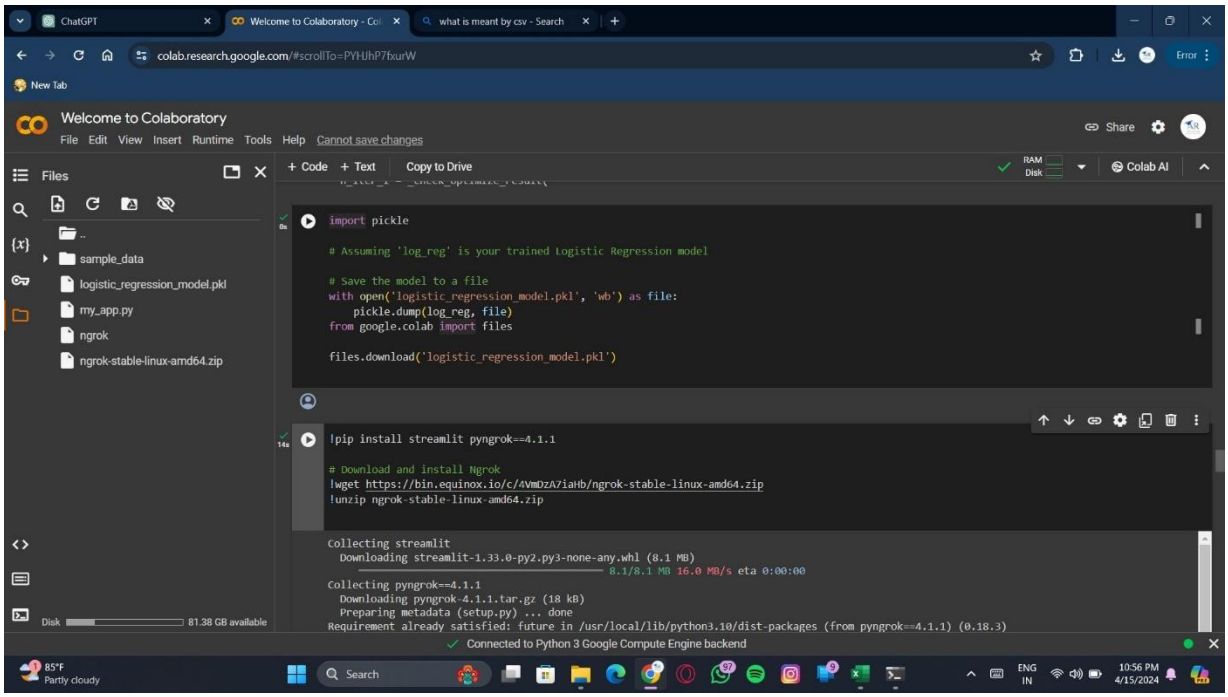
disease')

[1] the person has heart

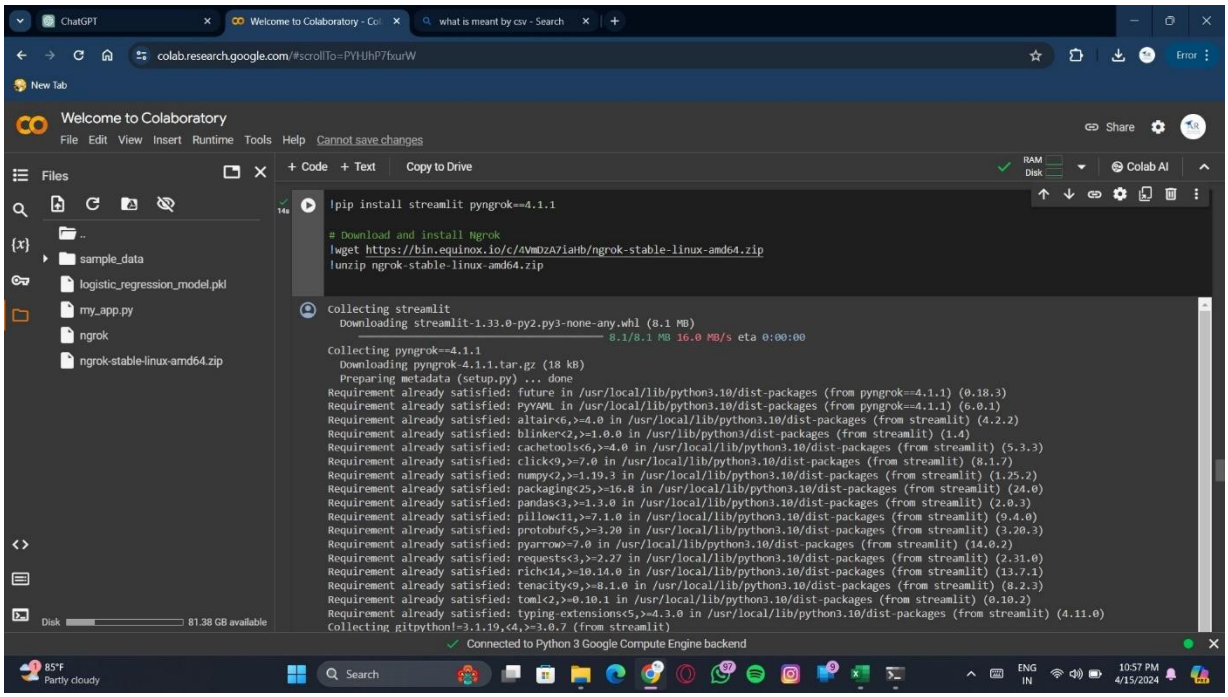
disease

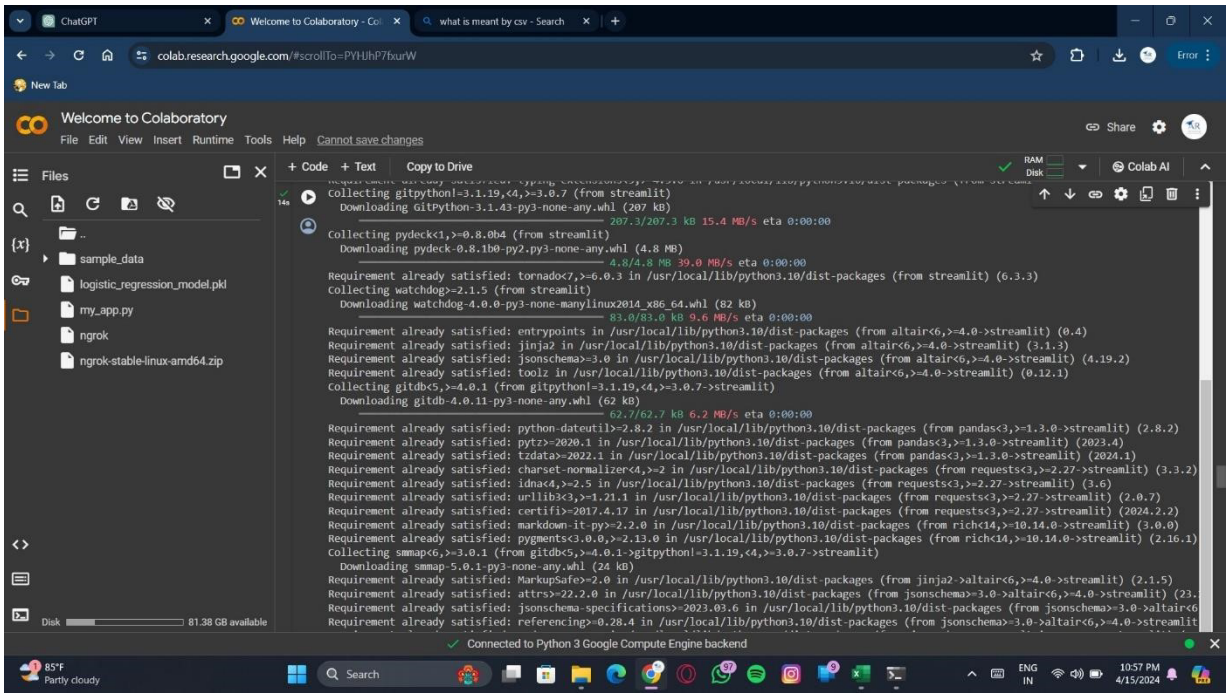


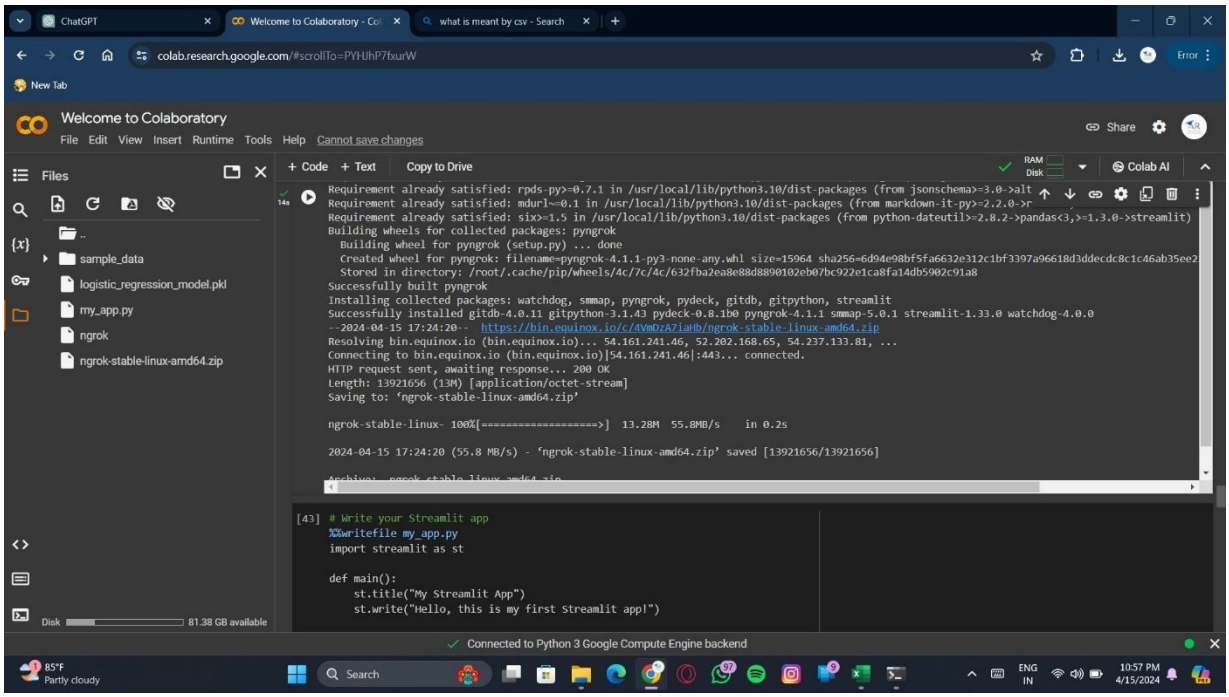
Lets save our model using pickle

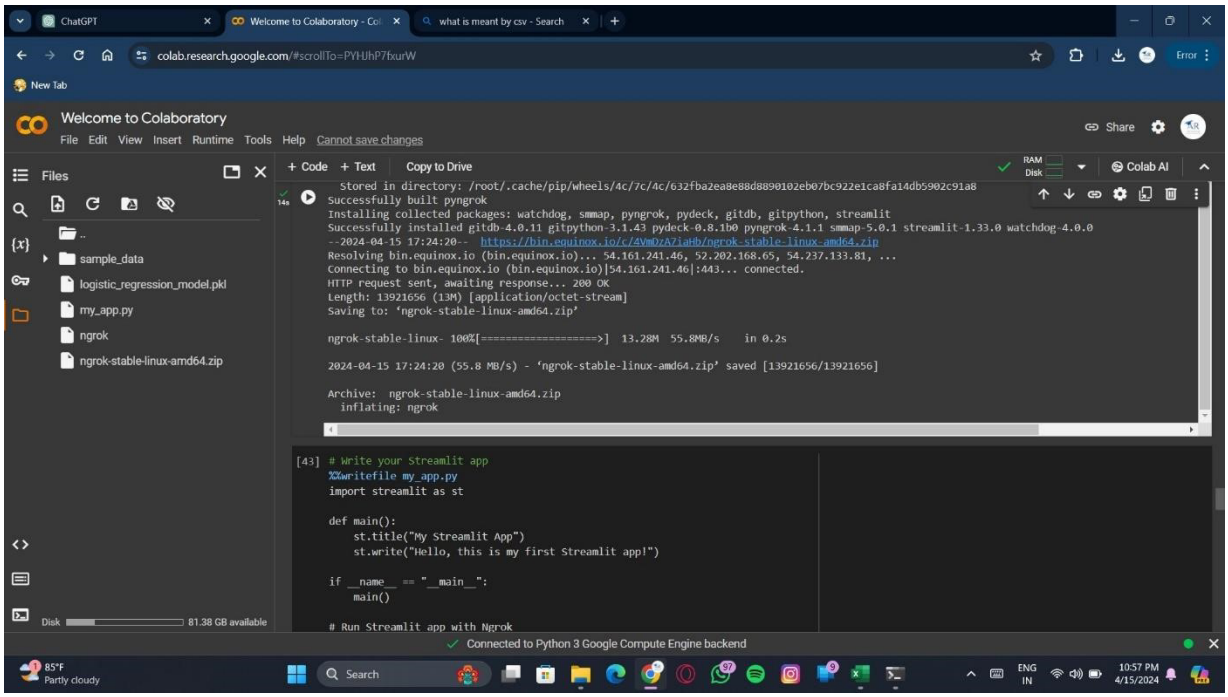


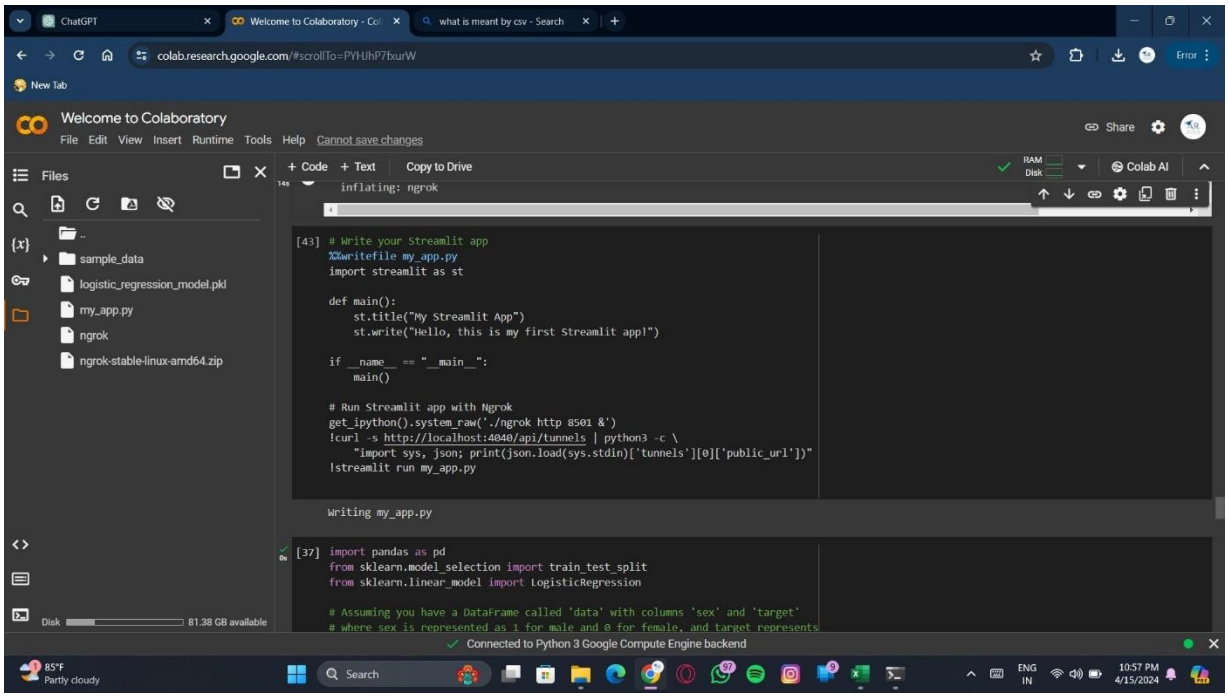
Import streamlit ,pyngrok,and ngrok modules



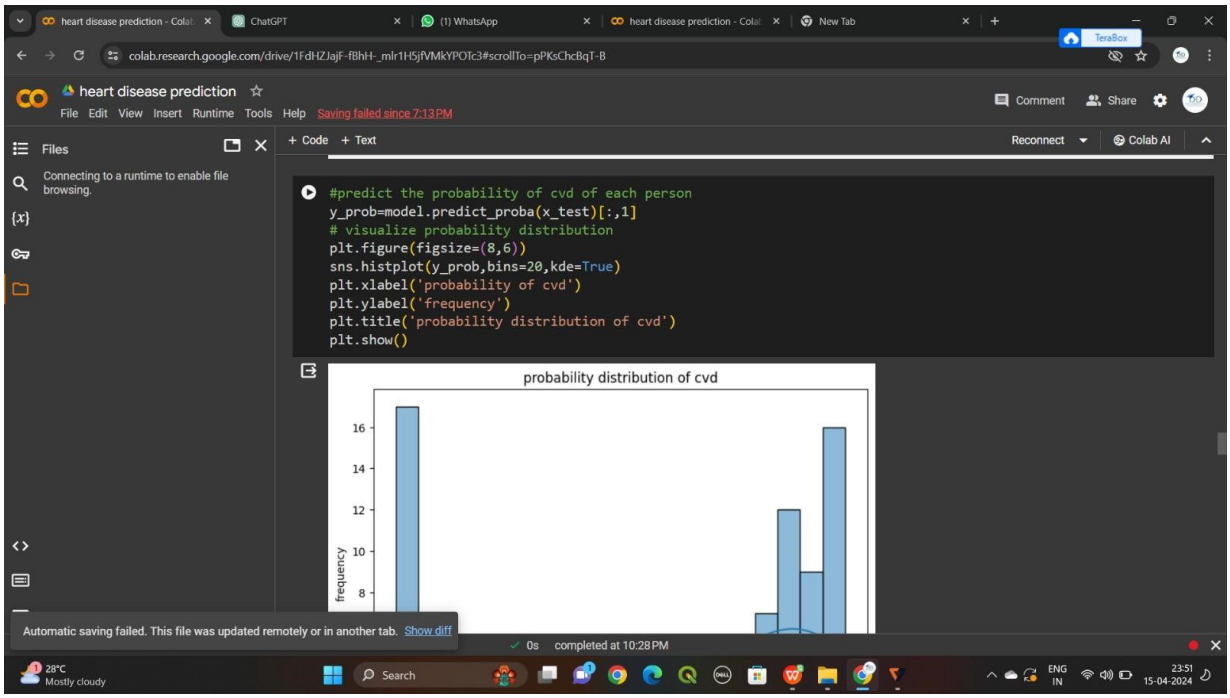


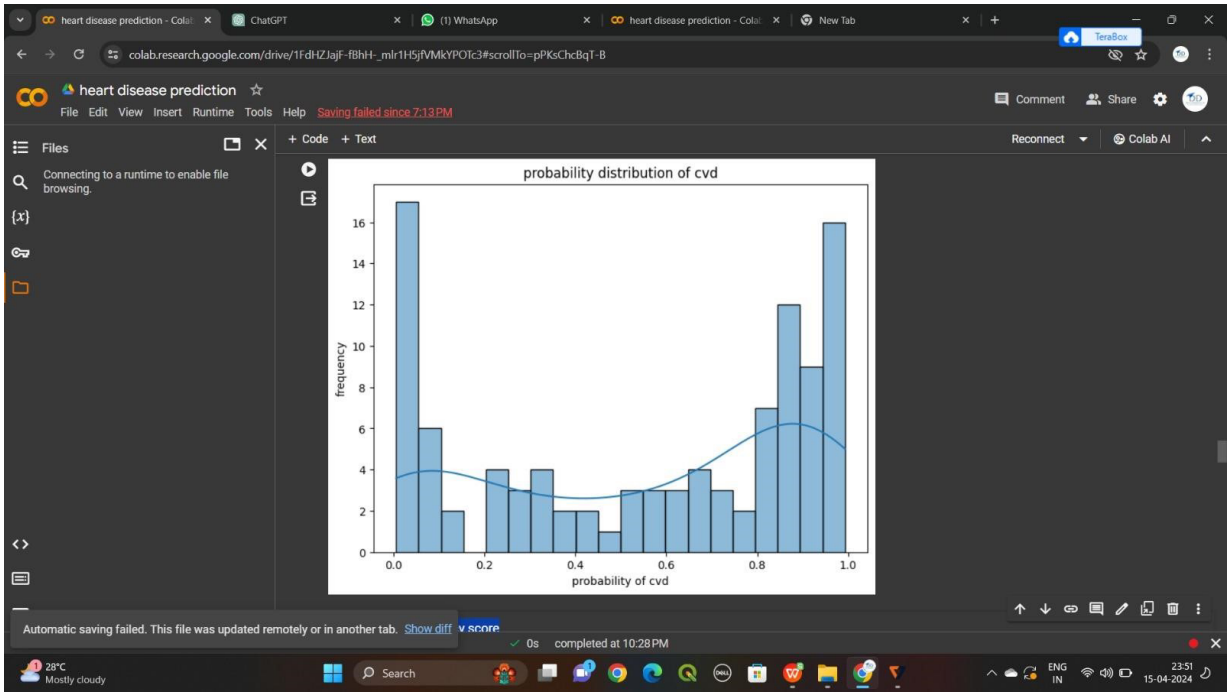






Probability of heart disease of each person





LINK : https://youtu.be/XouSAUr8IVg