Name: Muhammad Nuril Huda

Batch Code: LISUM30

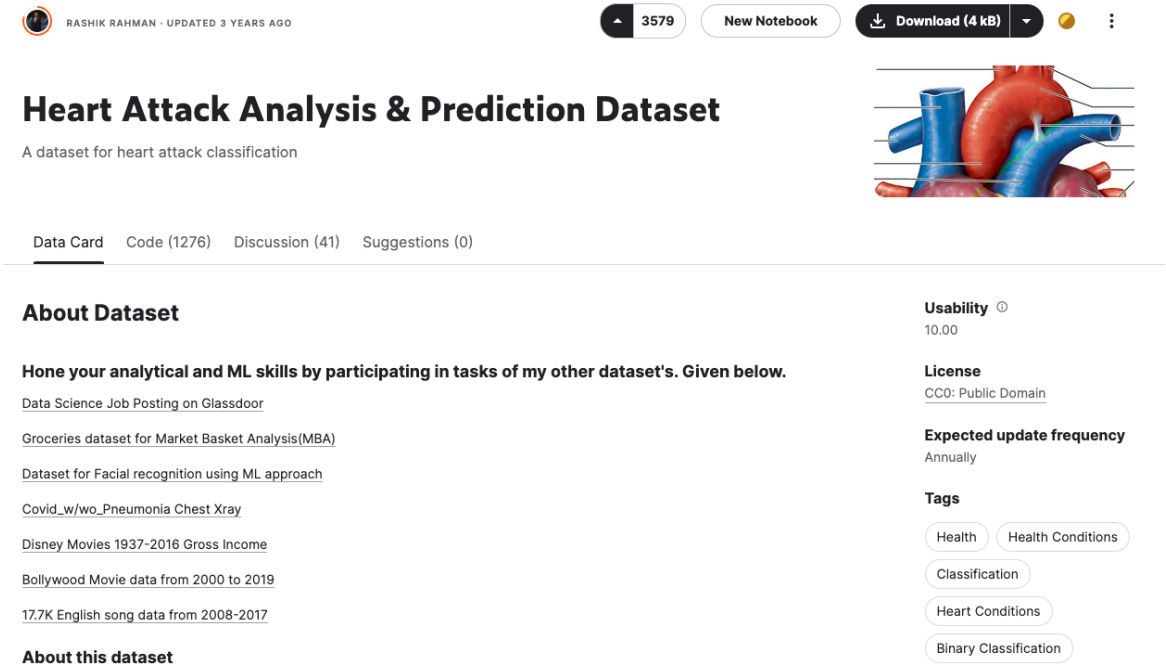
Submission date: 26 Feb 2024

Submitted to: https://github.com/MuhammadNurilHuda/Deploy---Heart-Attack

Heart Attack Prediction Deployment

1. Prepare the dataset.

   The dataset was downloaded from kaggle (https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset/data).



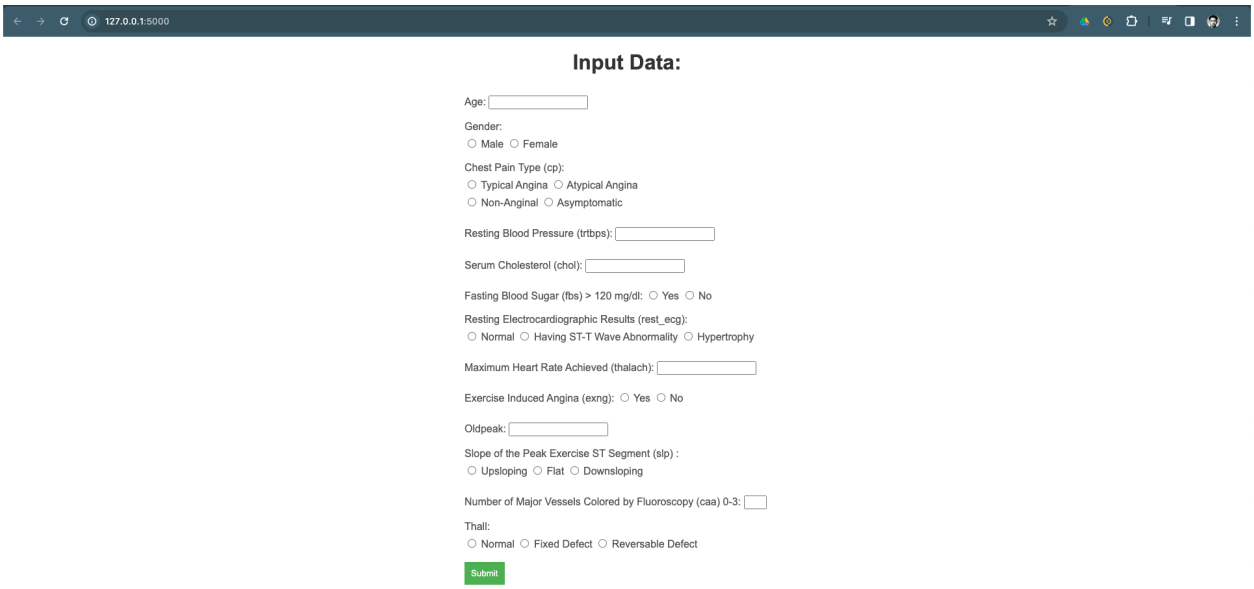
   Check this Data Intake report for the complete dataset information.

2. Train the ML model.

   The dataset is good enough, so the only preprocessing method I use is MinMax normalization. The algorithm used is Logistic Regression. With that method, the accuracy is 85% and recall is 84%.

3. Web Application

   The full HTML is on this link and css is on this link. And here is the screenshot of UI of the website.



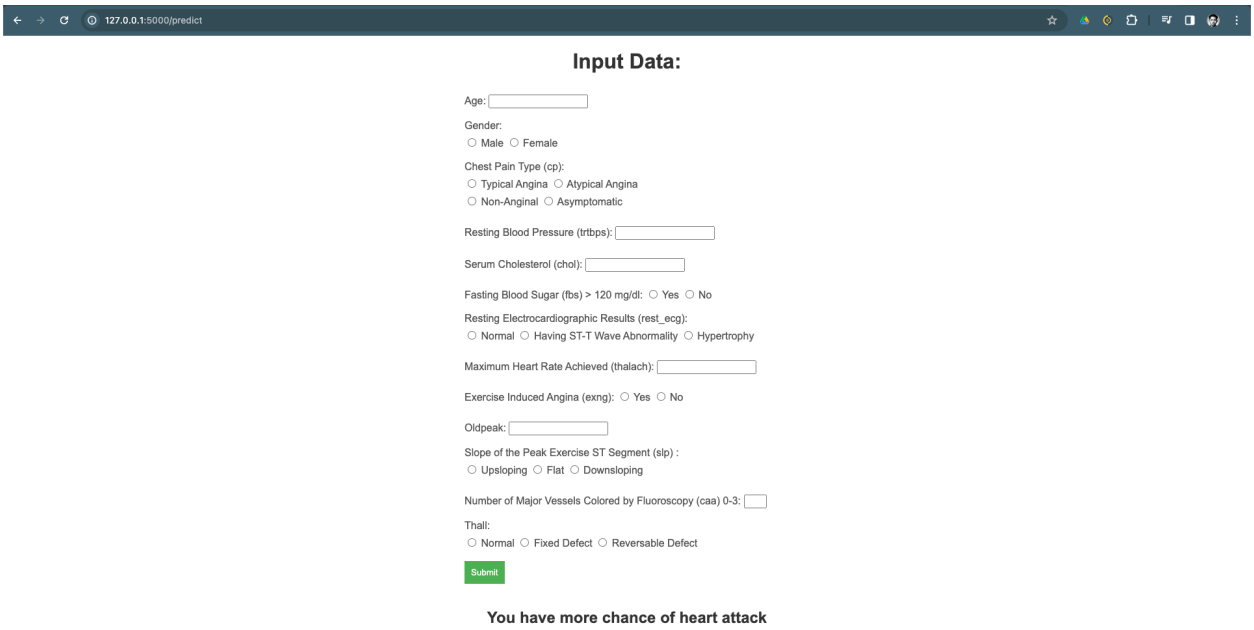
4. Flask endpoint

   The full flask code is on this link. Here the snapshot

```python
from flask import Flask, request, render_template, jsonify
import pickle
import numpy as np

app = Flask(__name__)
model = pickle.load(open('model.pkl', 'rb'))  # Load the trained model (pickle file)
scaler = pickle.load(open('minmax_scaler.pkl', 'rb'))

@app.route('/')
def home():
    return render_template('index.html')
    # return jsonify({'Message':'Hello, World!'})

@app.route( '/predict', methods=['POST'])
def predict():
    try:
        feature = ['age', 'sex', 'cp', 'trtbps', 'chol', 'fbs', 'restecg', 'thalachh','exng', 'oldpeak', 'slp', 'caa', 'thall']
        data = [float(request.form[f]) for f in feature]
        data_array = np.array([data]).reshape(1, -1)
        data_normalized = scaler.fit_transform(data_array)
        prediction = model.predict(data_normalized)[0]

        if prediction == 1:
            msg = 'You have more chance of heart attack'
        else:
            msg = 'No heart attack expected'

        response = {
            'prediction':msg
        }

        return render_template('index.html', prediction = msg)
        # return jsonify(response)

    except Exception as e:
        return jsonify({'error': str(e)})

if __name__ == "__main__":
    app.run(debug=True)
```

5. Test the web app

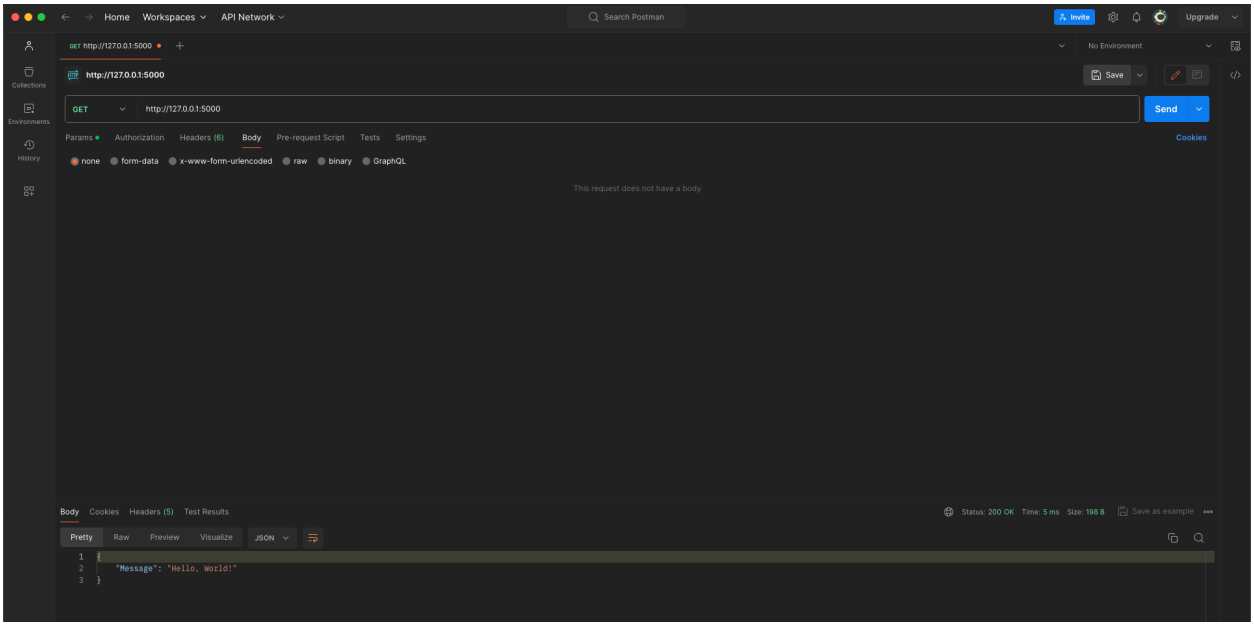
   After we insert the form, here is the output



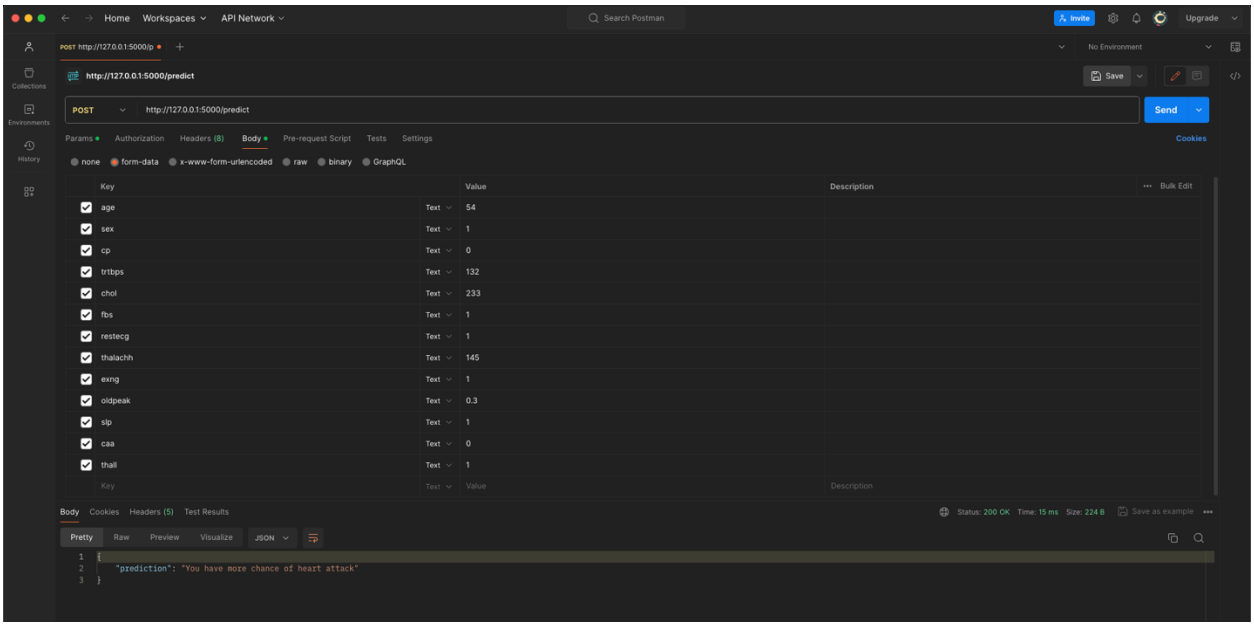
6. Postman test

   By changing the return value of the function, I do the api test using postman, and here's the result.

   main:



Predict:



It’s working fine…