

Week 4: Deployment on Flask

Name: Saanjanna Yuvaraj

Batch Code: LISUM23:30

Submission Date: 28/07/2023

Submitted to: Data Glacier

Step 1: Writing Flask Application

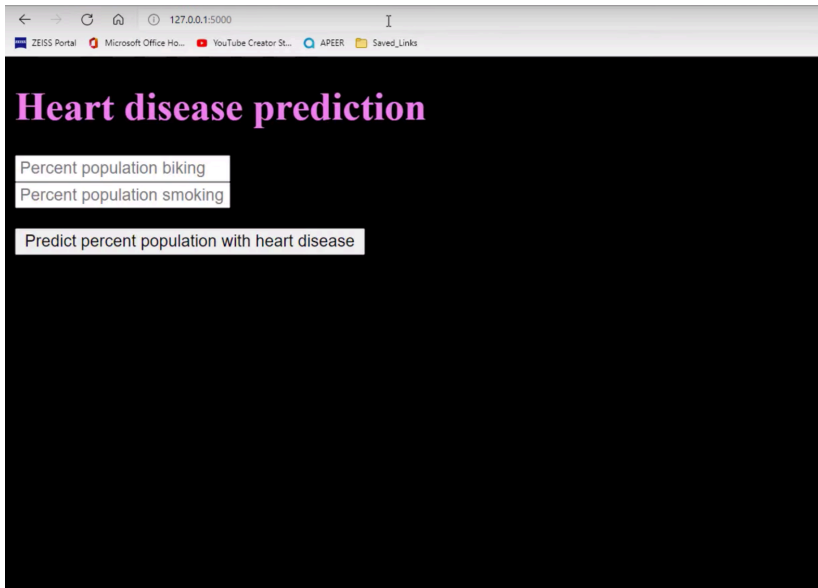
```
Heart.py index.html app.py
11 Application that predicts heart disease percentage in the population of a town
12 based on the number of bikers and smokers.
13
14 Trained on the data set of percentage of people biking
15 to work each day, the percentage of people smoking, and the percentage of
16 people with heart disease in an imaginary sample of 500 towns.
17
18 """
19
20
21 import numpy as np
22 from flask import Flask, request, render_template
23 import pickle
24
25 #Create an app object using the Flask class.
26 app = Flask(__name__)
27
28 #Load the trained model. (Pickle file)
29 model = pickle.load(open('model.pkl', 'rb'))
30
31 #Define the route to be home.
32 #The decorator below links the relative route of the URL to the function it is decorating.
33 #Here, home function is with '/', our root directory.
34 #Running the app sends us to index.html.
35 #Note that render_template means it looks for the file in the templates folder.
36 |
37 #Use the route() decorator to tell Flask what URL should trigger our function.
38 @app.route('/')
39 def home():
40     return render_template('index.html')
41
42 #You can use the methods argument of the route() decorator to handle different HTTP methods.
43 #GET: A GET message is send, and the server returns data
44 #POST: Used to send HTML form data to the server.
45 #Add Post method to the decorator to allow for form submission.
46 #Redirect to /predict page with the output
47 @app.route('/predict', methods=['POST'])
48 def predict():
49
50     int_features = [float(x) for x in request.form.values()] #Convert string inputs to float.
51     features = np.array(int_features) #Convert to the form [[a, b]] for input to the model
52     prediction = model.predict(features) # features Must be in the form [[a, b]]
53
54     output = round(prediction[0], 2)
55
56     return render_template('index.html', prediction_text='Percent with heart disease is {}'.format(output))
57
58
59 #When the Python interpreter reads a source file, it first defines a few special variables.
60 #For now, we care about the __name__ variable.
61 #If we execute our code in the main program, like in our case here, it assigns
62 # __main__ as the name (__name__).
63 #So if we want to run our code right here, we can check if __name__ == __main__
64 #if so, execute it here.
65 #If we import this file (module) to another file then __name__ == app (which is the name of this python file).
66
67 if __name__ == "__main__":
68     app.run()
69
```

Step 2: Running the Flask Application

```
File ~/Users/sanjukarthick/opt/anaconda3/lib/python3.8/site-packages/flask/app.py, line 1200, in dispatch_request
return self.view_functions[rule.endpoint](**req.view_args)
File ~/Users/sanjukarthick/Desktop/app.py, line 40, in home
return render_template('index.html')
File ~/Users/sanjukarthick/opt/anaconda3/lib/python3.8/site-packages/flask/templating.py, line 138, in render_template
ctx.app.jinja_env.get_or_select_template(template_name_or_list),
File ~/Users/sanjukarthick/opt/anaconda3/lib/python3.8/site-packages/jinja2/environment.py, line 930, in
get_or_select_template
return self.get_template(template_name_or_list, parent, globals)
File ~/Users/sanjukarthick/opt/anaconda3/lib/python3.8/site-packages/jinja2/environment.py, line 883, in get_template
return self._load_template(name, self.make_globals(globals))
File ~/Users/sanjukarthick/opt/anaconda3/lib/python3.8/site-packages/jinja2/environment.py, line 857, in _load_template
template = self.loader.load(self, name, globals)
File ~/Users/sanjukarthick/opt/anaconda3/lib/python3.8/site-packages/jinja2/loaders.py, line 115, in load
source, filename, uptodate = self.get_source(environment, name)
File ~/Users/sanjukarthick/opt/anaconda3/lib/python3.8/site-packages/flask/templating.py, line 60, in get_source
return self._get_source_fast(environment, template)
File ~/Users/sanjukarthick/opt/anaconda3/lib/python3.8/site-packages/flask/templating.py, line 89, in _get_source_fast
raise TemplateNotFound(template)
jinja2.exceptions.TemplateNotFound: index.html
127.0.0.1 - - [28/Jul/2023 21:30:58] "GET / HTTP/1.1" 500 -

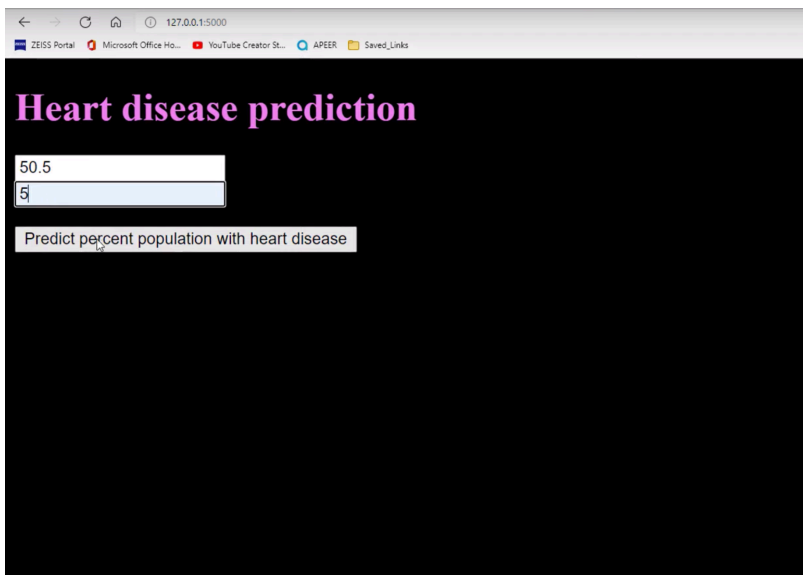
In [87]: runfile('/Users/sanjukarthick/Desktop/app.py', wdir='/Users/sanjukarthick/Desktop')
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Step 3: Open the link in the browser



A screenshot of a web browser window. The address bar shows the URL '127.0.0.1:5000'. The browser's tab bar includes 'ZEISS Portal', 'Microsoft Office Ho...', 'YouTube Creator St...', 'APEER', and 'Saved Links'. The main content area has a black background with the title 'Heart disease prediction' in pink. Below the title are three white input fields: 'Percent population biking', 'Percent population smoking', and 'Predict percent population with heart disease'.

Step 4: Testing the model



A screenshot of the same web browser window as in Step 3. The 'Heart disease prediction' form is now filled with data. The 'Percent population biking' field contains the value '50.5'. The 'Percent population smoking' field contains the value '5'. The 'Predict percent population with heart disease' field is empty. The browser's address bar and tab bar remain the same.

Step 5: Getting the result

← → ↻ 🏠 127.0.0.1:5000/predict

ZEISS Portal Microsoft Office Ho... YouTube Creator St... APEER Saved_Links

Heart disease prediction

Percent population biking

Percent population smoking

Predict percent population with heart disease

Percent with heart disease is 5.79