STATISTICAL MACHINE LEARNING APPROACHES TO LIVER DISEASE PREDICTION

TEAM ID: PNT2022TMID15752

BUILD PYTHON CODE
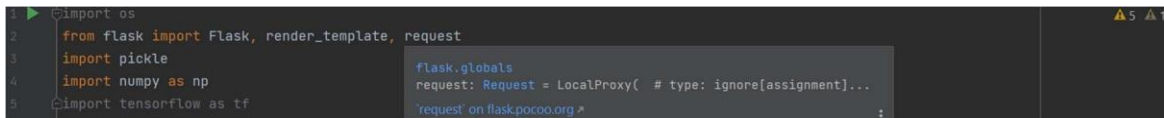
Python package builds are the product of coordination between a few

different tools driven by a standardized process. One of the biggest choices you

have as a package author is which set of tools to use. It can be difficult to assess

the nuances of each, especially if you are new to packaging. Fortunately, tools

are standardizing around the same core workflow, so once you learn it you have

got the agility to switch between tools with minimal effort This article covers

what you first need to learn about the pieces of the Python build system itself.

IMPORTING LIBRARIES

```
1  ▶  import os
2     from flask import Flask, render_template, request
3     import pickle                    flask.globals
4     import numpy as np               request: Request = LocalProxy(  # type: ignore[assignment]...
5     import tensorflow as tf          'request' on flask.pocoo.org ↗
```

Libraries required for the app to run are to be imported.

Creating our flask app and loading the model

```
app = Flask(__name__)
```

Now after all the libraries are imported, we will be creating our flask app. and the

load our model into our flask app.

ROUTING TO HTML PAGE

@app.route is used to route the application where it should route to.

'/' URL is bound with the home.html function. Hence, when the home page of

the web server is opened in the browser, the html page is rendered.

Whenever you enter the values from the html page the values can be retrieved using POST method.

Here, "home.html" is rendered when the home button is clicked on the UI and

"liver.html" is rendered when the predict button is clicked.

```python
@app.route("/")
def home():
    return render_template('home.html')

@app.route("/liver", methods=['GET', 'POST'])
def liverPage():
    return render_template('liver.html')

@app.route("/predict", methods = ['POST', 'GET'])
def predictPage():
```

Firstly, we are rendering the home.html template and from there we are

navigating to our prediction page that is predict.html.

We enter input values here and these values are sent to the loaded model and the resultant output is displayed on home.html.

Here the route

```python
@app.route("/predict", methods = ['POST', 'GET'])
def predictPage():
    try:
        if request.method == 'POST':
            to_predict_dict = request.form.to_dict()

            for key, value in to_predict_dict.items():
                try:
                    to_predict_dict[key] = int(value)
                except ValueError:
                    to_predict_dict[key] = float(value)

            to_predict_list = list(map(float, list(to_predict_dict.values())))
            pred = predict(to_predict_list, to_predict_dict)
    except:
        message = "Please enter valid data"
        return render_template("home.html", message=message)

    return render_template('predict.html', pred=pred)

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

Here the route for prediction is given and necessary steps are performed in order to get the predicted output.

Lastly, we run our app on the local host. Here we are running it on localhost:5000