

Name: Yashwanth Reddy Bomma Reddy

Batch Code: LISUM20

Submission Date: 28-04-2023

Submitted To: Data Glacier

Imported Linnerud toy data and used it for model prediction

```
In [36]: import pickle
models = {"model":model,"model2":model2}
with open("models.pkl","wb") as file:
    pickle.dump(models,file)

In [42]: with open("models.pkl","rb") as file:
models = pickle.load(file)
print(models["model"].predict([[0.923824,16.507468,1.488591]]))

ra 821762001

In [3]: data = load_linnerud().data
target = load_linnerud().target
feature_names = load_linnerud().feature_names
target_names = load_linnerud().target_names

In [4]: df1 = pd.DataFrame(data, columns = feature_names)
df2 = pd.DataFrame(target, columns = target_names)

In [5]: df1.head()

Out[5]:
```

	Chins	Situps	Jumps
0	5.0	162.0	60.0
1	2.0	110.0	60.0
2	12.0	101.0	101.0
3	12.0	105.0	37.0
4	13.0	155.0	58.0

```
In [6]: df2.head()

Out[6]:
```

	Weight	Waist	Pulse
0	191.0	36.0	50.0
1	189.0	37.0	52.0
2	193.0	38.0	58.0
3	162.0	35.0	62.0
4	189.0	35.0	46.0

Saving the prediction model to pickle file

```
In [36]: import pickle
models = {"model":model,"model2":model2}
with open("models.pkl","wb") as file:
    pickle.dump(models,file)

In [42]: with open("models.pkl","rb") as file:
models = pickle.load(file)
print(models["model"].predict([[0.923824,16.507468,1.488591]]))

ra 821762001
```

Deployed the model on flask

```
flask_week3.py > ...
1  from flask import Flask, request, jsonify, json
2  import pickle
3  import warnings
4  warnings.filterwarnings("ignore")
5
6  app = Flask(__name__)
7
8  #with open("models.pkl","rb") as file:
9  |   #models = pickle.load(file)
10
11 @app.route("/week3/flask",methods = ["POST"])
12
13
14
15 def prediction_model1():
16     #weight_model = request.get_json()
17     #prediction = models["model"].predict(weight_model)
18     #return jsonify({"Prediction": prediction.tolist()})
19
20     with open("models.pkl","rb") as file:
21         models = pickle.load(file)
22         x=[]
23         for keys,values in json.loads(request.data).items():
24             x.append(values)
25     return(models["model"].predict([x]).tolist())
26
```

Flask run

```
PS C:\Users\yashw\OneDrive\Desktop\Data Glacier\Flask deployment> $env:FLASK_APP = "flask_week3.py"
PS C:\Users\yashw\OneDrive\Desktop\Data Glacier\Flask deployment> python -m flask run
* Serving Flask app 'flask_week3.py'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [28/Apr/2023 23:00:29] "POST /week3/flask HTTP/1.1" 200 -
```

Tested on postman

The screenshot shows a Postman interface for a POST request to `http://127.0.0.1:5000/week3/flask`. The request body is a JSON object: `{ "a": 7.782527, "b": 15.793620, "c": 1.523996 }`. The response is a 200 OK status with a response time of 2.12s and a body of `0.822010703802444`.

Request Details:

- Method: POST
- URL: `http://127.0.0.1:5000/week3/flask`
- Body Type: raw (JSON)
- Body Content:

```
1 {
2   "a": 7.782527,
3   "b": 15.793620,
4   "c": 1.523996
5 }
```

Response Details:

- Status: 200 OK
- Time: 2.12 s
- Size: 186 B
- Body Type: Pretty (JSON)
- Body Content:

```
1 [
2   0.822010703802444
3 ]
```

Cors Error

```
8 from flask_restful import reqparse, abort, Api, Resource
9 from flask_cors import CORS, cross_origin
10
11 app = Flask(__name__)
12 CORS(app, resources={r"/": {"origins": "*"}})
13
14 @cross_origin
```

Prediction Model

Chins:

Situps:

Jumps:

The screenshot shows a web browser's developer tools interface. The Network tab is active, displaying a list of requests. The first request, 'flask/', is highlighted, showing a status of 404 and a type of 'CO...'. The Console tab shows an error message: 'Access to fetch at 'http://127.0.0.1:5000/week3/fla flask_week3_js.html:138' from origin 'null' has been blocked by CORS policy: Response to preflight request doesn't pass access control check: No 'Access-Control-Allow-Origin' header is present on the requested resource. If an opaque response serves your needs, set the request's mode to 'no-cors' to fetch the resource with CORS disabled.' Below this, a 'TypeError: Failed to fetch' message is shown, indicating that the fetch operation failed due to the CORS error.

Name	Stat...	Type	Initiator	Size	Time	Waterfall
task/	404	pre...	Preflight...	0 B	8 ms	
flask/	CO...	fetch	flask_we...	0 B	8 ms	

5 requests | 1.6 kB transferred | 1.6 kB resources | Finish: 46.64 s | DOMContentLoaded: 12 ms

Console

2 Issues: 2

Access to fetch at 'http://127.0.0.1:5000/week3/fla flask_week3_js.html:138' from origin 'null' has been blocked by CORS policy: Response to preflight request doesn't pass access control check: No 'Access-Control-Allow-Origin' header is present on the requested resource. If an opaque response serves your needs, set the request's mode to 'no-cors' to fetch the resource with CORS disabled.

POST http://127.0.0.1:5000/week3/flask/ flask_week3_js.html:38 net::ERR_FAILED

TypeError: Failed to fetch flask_week3_js.html:43 at HTMLFormElement.<anonymous> (flask_week3_js.html:38:19)