FLASK

Joblib library

The `joblib` library in Python is used for **efficient serialization (saving/loading)** of large data structures, especially **NumPy arrays and machine learning models**. It's commonly used with `scikit-learn` to **save trained models** to disk and load them later for inference.

```
import joblib
from sklearn.ensemble import RandomForestClassifier

# Train your model
model = RandomForestClassifier()
model.fit(X_train, y_train)

# Save it
joblib.dump(model, 'model.pkl')

# Later... load it
loaded_model = joblib.load('model.pkl')
predictions = loaded_model.predict(X_test)
```

What is Flask?

Flask is a **lightweight web framework** for Python that allows you to easily build **web applications** and **APIs**. It is designed to be simple, flexible, and easy to use, allowing you to create powerful web servers with minimal code.

A small **web service (API)** where:

- You send some **input data** (like features of a flower),

- It sends back a **prediction** (e.g., "Iris-setosa").

FLASK API USAGE

1.Import the necessary libraries

from flask import Flask, request, jsonify
import joblib

`Flask`: Used to create the web app.

`request`: Used to receive data from the user.Is `request` written in the backend to receive data from the API?”

✅ **YES**. The `request` object in Flask is how your backend receives and understands **data sent from a client (frontend, another API, etc.**

`jsonify`: Used to send data back in a clean, JSON format.

`joblib`: Used to load your saved AI model.

2.Initalize the application

app = Flask(__name__)

Think of this like starting a new website or app. This line initializes the app.

3.Load your model

model = joblib.load('model.pkl')

This line opens your previously trained and saved AI model (`model.pkl`) so we can use it for predictions.

4.Define a prediction endpoint

@app.route('/predict', methods=['POST'])

"If someone goes to `/predict` on our website and sends a POST request (which usually includes data), run the function below."

5.The predict() function

```
def predict():
    data = request.get_json(force=True)  # Get the input data from the request
    prediction = model.predict([data['features']])  # Make a prediction
    return jsonify({'prediction': prediction.tolist()})  # Return the result
```

`request.get_json(force=True)` → reads the data the user sent (like `[5.1, 3.5, 1.4, 0.2]`).

`model.predict([...])` → uses the AI model to make a prediction.

`jsonify(...)` → converts the prediction into a format like:

{"prediction": [0]}

6.Start the app

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

This runs the app and starts a **local web server** (usually at `http://127.0.0.1:5000`) so you can send it requests.

Folder Setup

```
/prediction-app
│
├── /static
│   ├── /css
│   │   └── style.css
│   └── /js
│       └── script.js
│
├── /templates
│   └── index.html
│
├── model.pkl
├── app.py
└── requirements.txt
```