1. **Train and Save the Model**: First, train your machine learning model on your dataset. Once the model is trained and evaluated, save it to a file using a library like **joblib** or **pickle**.
2. **Create a Flask Web Application**: You can use the Flask framework to create a web application that will serve your model. Install Flask if you haven't already:Then create a Flask web application:
3. **Run the Web Application**: Run your Flask application. It will start a web server and listen for incoming requests.
4. **Send Requests for Predictions**: You can now send HTTP POST requests to your Flask application with input data to get real-time predictions. You can use libraries like **requests** in Python to send these requests.
5. **Scalability and Deployment**: For production use, consider deploying your Flask application on a production server. Popular options include AWS, Azure, Google Cloud, Heroku, or using containerization tools like Docker and Kubernetes for scalability.

import joblib

model = train\_your\_model()

joblib.dump(model, 'your\_model.pkl')

from flask import Flask, request, jsonify

import joblib

app = Flask(\_\_name\_\_)

# Load the model

model = joblib.load('your\_model.pkl')

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

def predict():

data = request.json # Assuming you are sending JSON data

prediction = model.predict(data)

return jsonify({'prediction': prediction.tolist()})

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

import requests

import json

data = {"feature1": 1.2, "feature2": 3.4} # Your input data

response = requests.post("http://localhost:5000/predict", json=data)

result = response.json()

print("Prediction:", result["prediction"])