BLACK BUCKS ASSIGNMENT-5

PART-2:

Deploying a Simple AI Model

Task :

- Use a basic ML model (e.g., a Decision Tree classifier on Iris dataset using scikit-learn)
- Deploy the model using Flask or FastAPI
- Create a small API endpoint (/predict) that receives input and returns prediction

Model Training Code:

```python
# model.py
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
import joblib

# Load the Iris dataset
iris = load_iris()
X = iris.data
y = iris.target

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the Decision Tree Classifier
model = DecisionTreeClassifier()

# Train the model
model.fit(X_train, y_train)

# Test the model
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Model accuracy: {accuracy * 100:.2f}%")

# Save the trained model using joblib
joblib.dump(model, 'iris_model.pkl')
```

Flask/FastAPI code for deployment:

```python
# app.py
from flask import Flask, request, jsonify
import joblib
import numpy as np

# Initialize Flask app
app = Flask(__name__)

# Load the trained model
model = joblib.load('iris_model.pkl')

@app.route('/')
def home():
    return "Welcome to the Iris Prediction API!"

@app.route('/predict', methods=['POST'])
def predict():
    try:
        # Get input data from request
        data = request.get_json()

        # Extract features (assume it's a list of features)
        features = np.array(data['features']).reshape(1, -1)

        # Make prediction using the trained model
        prediction = model.predict(features)

        # Map the prediction to Iris class name
        class_names = ['Setosa', 'Versicolor', 'Virginica']
        result = class_names[prediction[0]]

        return jsonify({'prediction': result})

    except Exception as e:
        return jsonify({'error': str(e)})

# Run the Flask app
if __name__ == '__main__':
    app.run(debug=True)
```

Input:

```
curl -X POST http://127.0.0.1:5000/predict -H "Content-Type: application/json" -d
"{\"features\": [5.1, 3.5, 1.4, 0.2]}"
```

Output:

```
{
  "prediction": "Setosa"
}
```