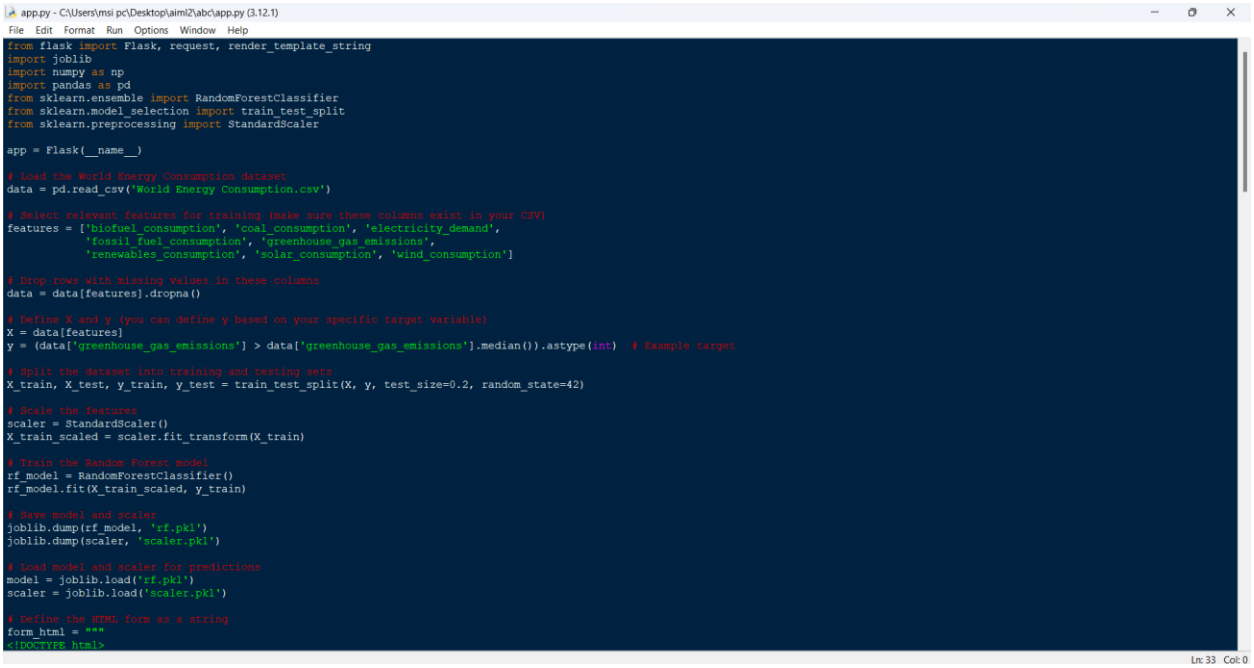
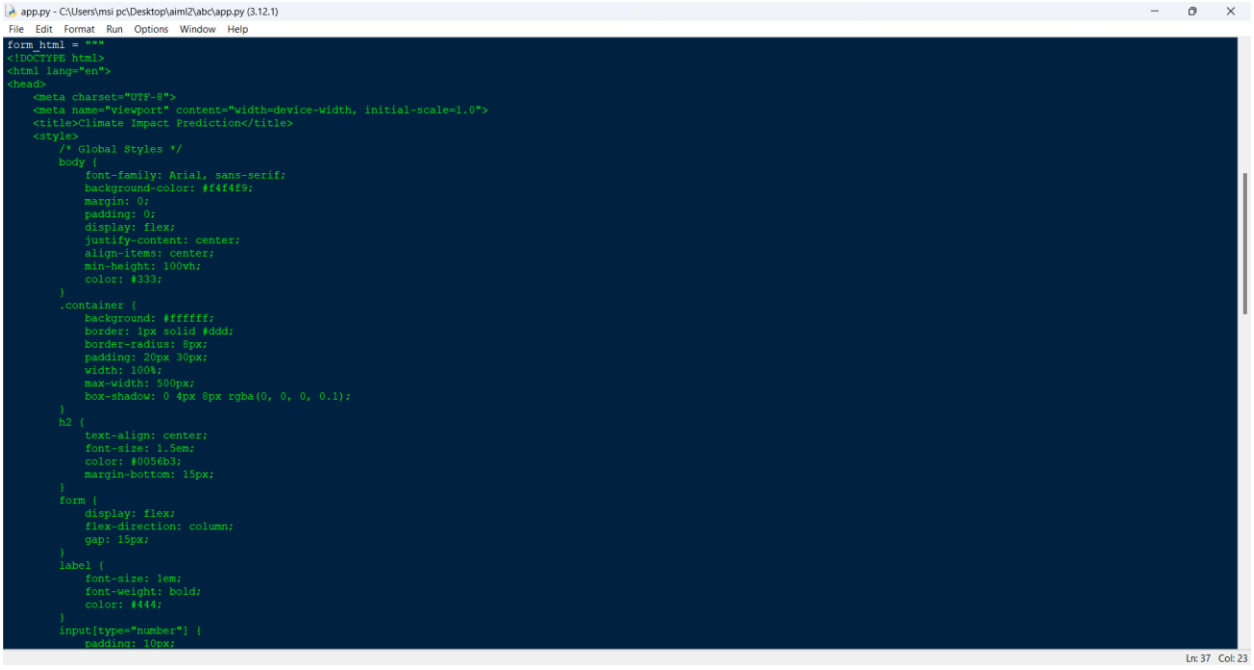
AIML ASSIGNMENT 2

Develop a Flask-based UI to use the ML/DL model developed in the Assignment – 1. Upload your all the resources on GitHub.

Code –



```
app.py - C:\Users\msi pc\Desktop\aimlZ\abc\app.py (3.12.1)
File  Edit  Format  Run  Options  Window  Help
from flask import Flask, request, render_template_string
import joblib
import numpy as np
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

app = Flask(__name__)

# Load the World Energy Consumption dataset
data = pd.read_csv('World Energy Consumption.csv')

# Select relevant features for training (make sure these columns exist in your CSV)
features = ['biofuel_consumption', 'coal_consumption', 'electricity_demand',
            'fossil_fuel_consumption', 'greenhouse_gas_emissions',
            'renewables_consumption', 'solar_consumption', 'wind_consumption']

# Drop rows with missing values in these columns
data = data[features].dropna()

# Define X and y (you can define y based on your specific target variable)
X = data[features]
y = (data['greenhouse_gas_emissions'] > data['greenhouse_gas_emissions'].median()).astype(int)  # Example 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)

# Scale the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)

# Train the Random Forest model
rf_model = RandomForestClassifier()
rf_model.fit(X_train_scaled, y_train)

# Save model and scaler
joblib.dump(rf_model, 'rf.pkl')
joblib.dump(scaler, 'scaler.pkl')

# Load model and scaler for predictions
model = joblib.load('rf.pkl')
scaler = joblib.load('scaler.pkl')

# Define the HTML form as a string
form_html = """
<!DOCTYPE html>
Ln: 33  Col: 0
```



```
app.py - C:\Users\msi pc\Desktop\aimlZ\abc\app.py (3.12.1)
File  Edit  Format  Run  Options  Window  Help
form_html = """
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Climate Impact Prediction</title>
    <style>
        /* Global Styles */
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f9;
            margin: 0;
            padding: 0;
            display: flex;
            justify-content: center;
            align-items: center;
            min-height: 100vh;
            color: #333;
        }
        .container {
            background: #ffffff;
            border: 1px solid #ddd;
            border-radius: 8px;
            padding: 20px 30px;
            width: 100%;
            max-width: 500px;
            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
        }
        h2 {
            text-align: center;
            font-size: 1.5em;
            color: #0056b3;
            margin-bottom: 15px;
        }
        form {
            display: flex;
            flex-direction: column;
            gap: 15px;
        }
        label {
            font-size: 1em;
            font-weight: bold;
            color: #444;
        }
        input[type="number"] {
            padding: 10px;
Ln: 37  Col: 23
```

app.py - C:\Users\msi pc\Desktop\aiml2\abc\app.py (3.12.1)

File Edit Format Run Options Window Help

```
        label {
            font-size: 1em;
            font-weight: bold;
            color: #444;
        }
        input[type="number"] {
            padding: 10px;
            font-size: 1em;
            border: 1px solid #ccc;
            border-radius: 4px;
            transition: border-color 0.2s;
        }
        input[type="number"]:focus {
            border-color: #0056b3;
            outline: none;
        }
        input[type="submit"] {
            padding: 12px;
            font-size: 1em;
            background-color: #0056b3;
            color: white;
            border: none;
            border-radius: 4px;
            cursor: pointer;
            text-align: center;
            font-weight: bold;
            transition: background-color 0.3s;
        }
        input[type="submit"]:hover {
            background-color: #003d80;
        }
        .result, .error {
            margin-top: 20px;
            padding: 12px;
            font-size: 0.9em;
            border-radius: 4px;
        }
        .result {
            background-color: #e8f5e9;
            color: #2e7d32;
            border: 1px solid #c8e6c9;
        }
        .error {
            background-color: #ffebee;
            color: #c62828;
            border: 1px solid #ef9a9a;
        }
```

Ln: 37 Col: 23

app.py - C:\Users\msi pc\Desktop\aiml2\abc\app.py (3.12.1)

File Edit Format Run Options Window Help

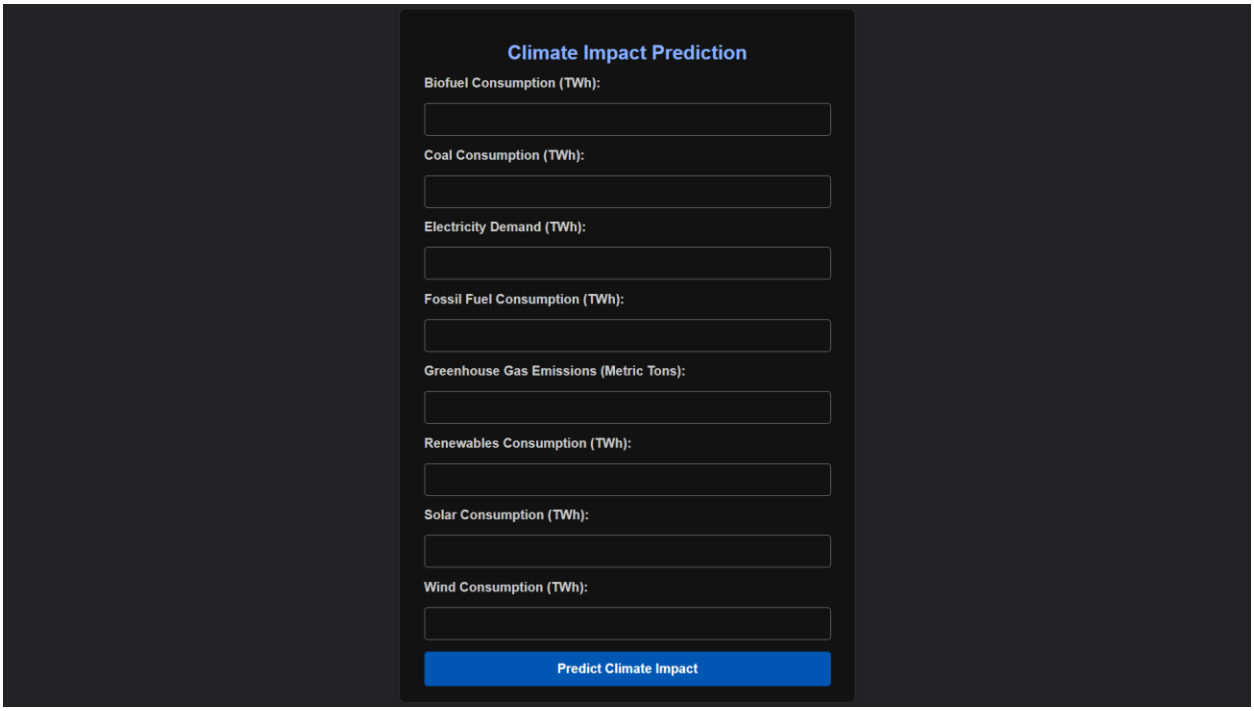
```
            border: 1px solid #ef9a9a;
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Climate Impact Prediction</h2>
        <form method="POST" action="/predict">
            <label for="biofuel_consumption">Biofuel Consumption (TWh):</label>
            <input type="number" name="biofuel_consumption" id="biofuel_consumption" required>

            <label for="coal_consumption">Coal Consumption (TWh):</label>
            <input type="number" name="coal_consumption" id="coal_consumption" required>

            <label for="electricity_demand">Electricity Demand (TWh):</label>
            <input type="number" name="electricity_demand" id="electricity_demand" required>

            <label for="fossil_fuel_consumption">Fossil Fuel Consumption (TWh):</label>
            <input type="number" name="fossil_fuel_consumption" id="fossil_fuel_consumption" required>

            <label for="greenhouse_gas_emissions">Greenhouse Gas Emissions (Metric Tons):</label>
            <input type="number" name="greenhouse_gas_emissions" id="greenhouse_gas_emissions" required>

            <label for="renewables_consumption">Renewables Consumption (TWh):</label>
            <input type="number" name="renewables_consumption" id="renewables_consumption" required>

            <label for="solar_consumption">Solar Consumption (TWh):</label>
            <input type="number" name="solar_consumption" id="solar_consumption" required>

            <label for="wind_consumption">Wind Consumption (TWh):</label>
            <input type="number" name="wind_consumption" id="wind_consumption" required>

            <input type="submit" value="Predict Climate Impact">
        </form>

        {% if prediction_text %}
            <div class="result">{{ prediction_text }}</div>
        {% endif %}

        {% if error_text %}
            <div class="error">{{ error_text }}</div>
        {% endif %}
    </div>
</body>
</html>
"""
```

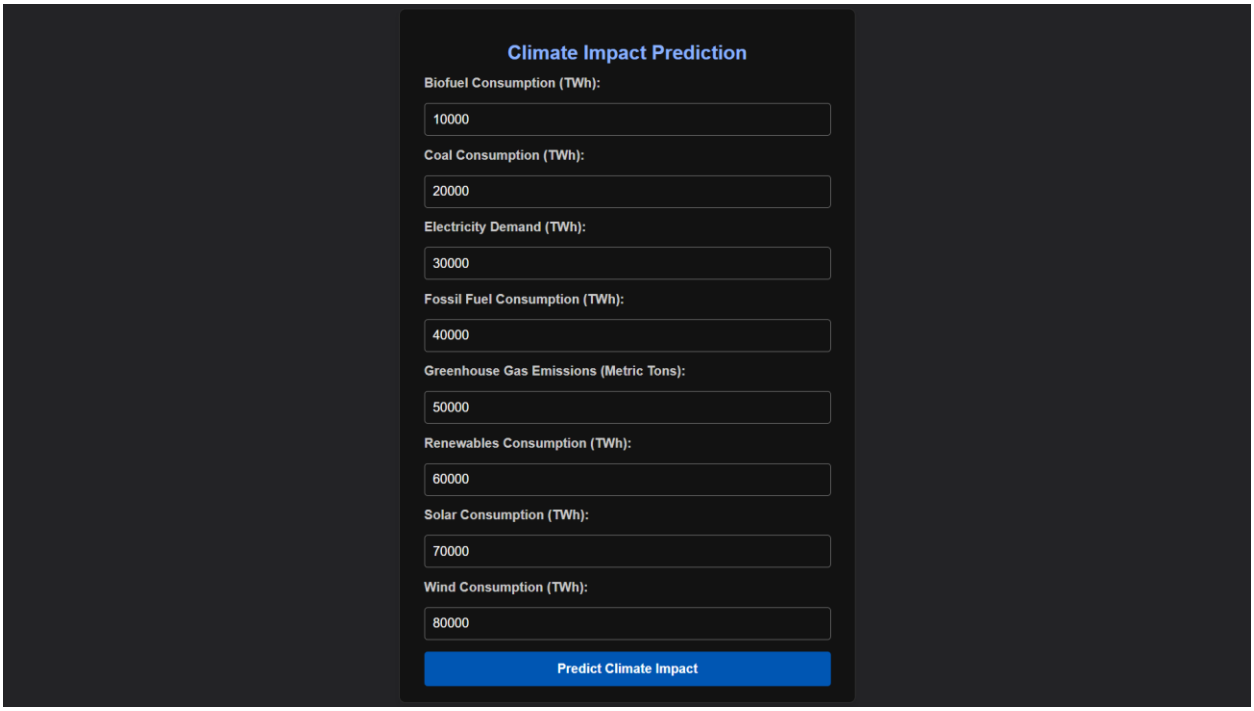
Ln: 160 Col: 0

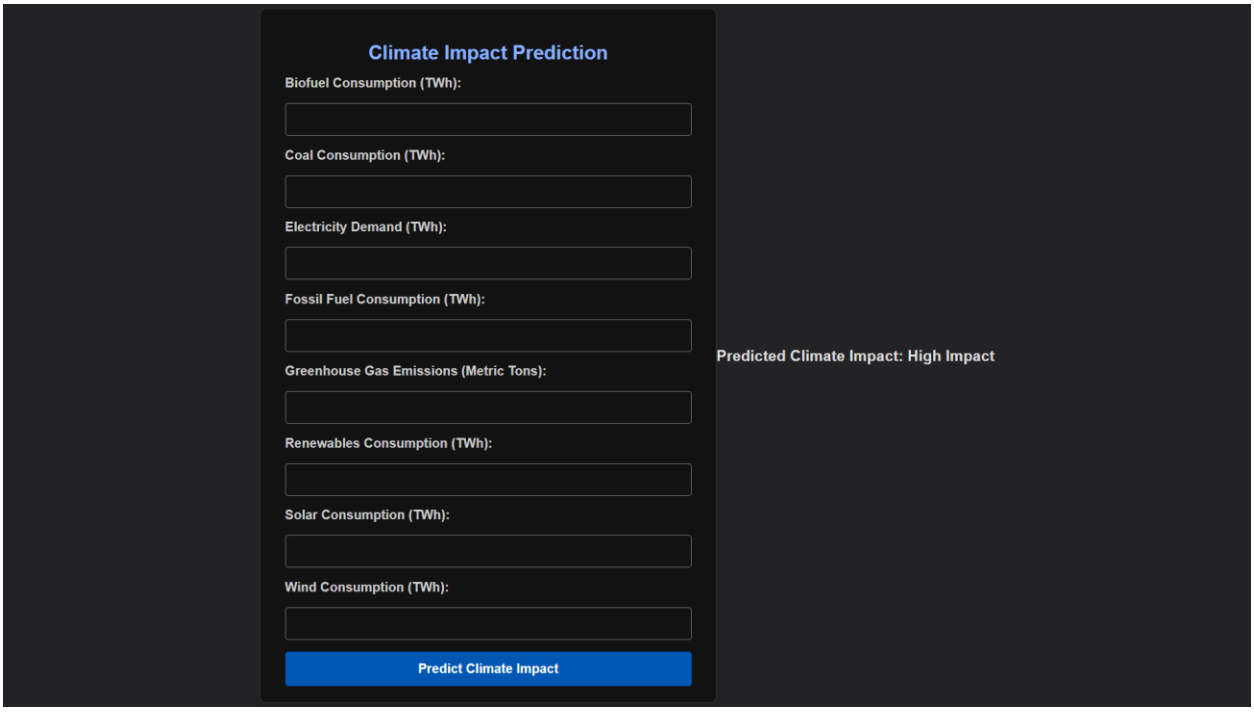
```
app.py - C:\Users\msi pc\Desktop\aiml2\abc\app.py (3.12.1)
File  Edit  Format  Run  Options  Window  Help
        </form>

        {% if prediction_text %}
            <div class="result">{{ prediction_text }}</div>
        {% endif %}

        {% if error_text %}
            <div class="error">{{ error_text }}</div>
        {% endif %}
    </div>
</body>
</html>
"""

@app.route('/')
def index():
    return render_template_string(form_html)

@app.route('/predict', methods=['POST'])
def predict():
    try:
        # Collect input features from the form
        features = np.array([float(request.form['biofuel_consumption']),
                             float(request.form['coal_consumption']),
                             float(request.form['electricity_demand']),
                             float(request.form['fossil_fuel_consumption']),
                             float(request.form['greenhouse_gas_emissions']),
                             float(request.form['renewables_consumption']),
                             float(request.form['solar_consumption']),
                             float(request.form['wind_consumption'])]).reshape(1, -1)

        # Scale the features
        features_scaled = scaler.transform(features)

        # Make the prediction
        prediction = model.predict(features_scaled)[0]

        # Translate prediction to a meaningful label
        result = "High Impact" if prediction == 1 else "Low Impact"

        return render_template_string(form_html + f"<h3>Predicted Climate Impact: {result}</h3>")

    except Exception as e:
        return render_template_string(form_html + "<h3>Error occurred: " + str(e) + "</h3>")

if __name__ == '__main__':
    app.run(debug=True)
                                                                                                    Ln: 160  Col: 0
```

Output –

```
*IDLE Shell 3.12.1*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\msi pc\Desktop\aiml2\abc\app.py
 * Serving Flask app 'app'
 * Debug mode: on
[31m[1mWARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.[0m
 * Running on http://127.0.0.1:5000
[33mPress CTRL+C to quit[0m
 * Restarting with stat
```




Climate Impact Prediction
Biofuel Consumption (TWh):
10000
Coal Consumption (TWh):
20000
Electricity Demand (TWh):
30000
Fossil Fuel Consumption (TWh):
40000
Greenhouse Gas Emissions (Metric Tons):
50000
Renewables Consumption (TWh):
60000
Solar Consumption (TWh):
70000
Wind Consumption (TWh):
80000
Predict Climate Impact


Climate Impact Prediction
Biofuel Consumption (TWh):
Coal Consumption (TWh):
Electricity Demand (TWh):
Fossil Fuel Consumption (TWh):
Greenhouse Gas Emissions (Metric Tons):
Renewables Consumption (TWh):
Solar Consumption (TWh):
Wind Consumption (TWh):
Predict Climate Impact
Predicted Climate Impact: High Impact

For this project, we created the Flask web application to determine the climate-change effect given the energy consumption data. This app uses a Random Forest classifier to predict the climate impact as high or low depending on a number of energy consumption factors like biofuel, coal and renewable power consumption.

Code Breakdown:

Data Loading and Preprocessing:

To study this relationship, the World Energy Consumption data is imported from the web using pandas for analysis.

Outliers are deleted and only useful features are taken for the experimentation such as amount of energy consumption (biofuel, coal, electricity demand etc.).

Model Training:

We then use the train_test_split to split the data into training and testing set.

Features are scaled with StandardScaler tool in order to get better model performance on the data.

A Random Forest classifier is trained with the data which has been preprocessed for the prediction of the climate impact.

Model and Scaler Saving:

The trained model and scaler are stored using joblib from which they can be loaded into the Flask app.

Flask Web App:

An HTML form that is relatively easy to build is designed to allow the user to enter energy consumption rates.

Flask routing is employed to capture the form data and to provide the appropriate response to it by the app. The input features are further normalized and then provided to the trained model to predict the observations.

Prediction:

The app divides the input data as either high climate impact or low climate impact.

It applies SHAP or LIME (Interpretable machine learning algorithms) bringing feature-wise contributions in the model's decision to make the prognosis clear.

Deployment:

The application functions locally thus users can key in data through the user interface and get back forecasts on climate impact.

Summary:

In this project, an app is developed as a climate impact prediction with the help of machine learning to differentiate climate risks with regard to energy consumption. Besides that, XAI techniques integrated into the app also guarantee that the user is able to see how each of the features affects the model, which makes it not only useful but also explainable.