|  |
| --- |
| Team ID : LTVIP2025TMID32402 |
| Project Name : Enchanted Wings: Marvels of Butterfly Species |

**Application Building**

In this section, we have built a web application that integrates the machine learning model we developed. The application features a user interface (UI) that allows users to input data for prediction. The entered values are passed to the saved machine learning model, and the prediction result is displayed on the UI.

**Tasks Performed**

1. **Building HTML Pages**

We designed a simple and user-friendly HTML page that includes form elements. These forms collect input values from the user, which are required for making predictions using the model.

* + The HTML page consists of:
    - Text input fields for each required feature
    - A submit button to trigger the prediction
    - A result display area showing the model's output

1. **Building Server-side Script**

We developed the server-side logic using Python's Flask framework. The script:

* + Loads the saved machine learning model (e.g., using pickle)
  + Receives input from the HTML form via POST method
  + Preprocesses the input data as required
  + Passes the input to the model for prediction
  + Renders the HTML page with the prediction result

**Example Technologies Used**

* **Frontend:** HTML, CSS (basic)
* **Backend:** Python, Flask
* **Model Handling:** Pickle (or Joblib)
* **Execution:** Hosted locally or on a cloud platform

**Outcome**

The integration of the machine learning model with a web application allows for real-time user interaction. Users can now input custom data and immediately receive predictions, enhancing the accessibility and usability of the model.

**1. Building HTML Pages (Frontend)**

Create a simple HTML form for users to input features.

**Example: index.html**

html

CopyEdit

<!DOCTYPE html>

<html>

<head>

<title>Prediction App</title>

</head>

<body>

<h1>Butterfly Classification</h1>

<form action="/predict" method="post">

<label for="feature1">Feature 1:</label>

<input type="text" name="feature1" required><br><br>

<label for="feature2">Feature 2:</label>

<input type="text" name="feature2" required><br><br>

<!-- Add more input fields as required -->

<input type="submit" value="Predict">

</form>

{% if prediction %}

<h2>Prediction: {{ prediction }}</h2>

{% endif %}

</body>

</html>

**2. Building the Server-Side Script (Backend)**

You can use **Flask** (Python micro web framework) for the backend.

**Example: app.py**

from flask import Flask, render\_template, request

import pickle # or joblib

import numpy as np

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

# Load the saved model

model = pickle.load(open('model.pkl', 'rb')) # Adjust if you used joblib

@app.route('/')

def home():

return render\_template('index.html')

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

def predict():

try:

# Get values from form and convert to float

features = [float(request.form[f'feature{i+1}']) for i in range(2)] # Adjust index range to your features

prediction = model.predict([np.array(features)])

return render\_template('index.html', prediction=prediction[0])

except Exception as e:

return f"Error: {str(e)}"

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

app.run(debug=True)

**3. Saving the Model (if not already done)**

Make sure your model is saved using:

import pickle

pickle.dump(model, open('model.pkl', 'wb'))

**✅ Next Steps**

* Finalize feature list and update the HTML form accordingly.
* Ensure your model accepts input in the same order.
* Test the app locally (python app.py) and go to http://127.0.0.1:5000 in your browser.

Would you like help customizing the HTML and Python script based on your specific model input (e.g., number and names of features)?

**You said:**

word

**ChatGPT said:**

Here's a **Word (DOCX)** version of the "Application Building" section for your report or documentation.

**Application Building**

In this section, we have built a web application that integrates the machine learning model we developed. The application features a user interface (UI) that allows users to input data for prediction. The entered values are passed to the saved machine learning model, and the prediction result is displayed on the UI.

**Tasks Performed**

1. **Building HTML Pages**

We designed a simple and user-friendly HTML page that includes form elements. These forms collect input values from the user, which are required for making predictions using the model.

* + The HTML page consists of:
    - Text input fields for each required feature
    - A submit button to trigger the prediction
    - A result display area showing the model's output

1. **Building Server-side Script**

We developed the server-side logic using Python's Flask framework. The script:

* + Loads the saved machine learning model (e.g., using pickle)
  + Receives input from the HTML form via POST method
  + Preprocesses the input data as required
  + Passes the input to the model for prediction
  + Renders the HTML page with the prediction result

**Example Technologies Used**

* **Frontend:** HTML, CSS (basic)
* **Backend:** Python, Flask
* **Model Handling:** Pickle (or Joblib)
* **Execution:** Hosted locally or on a cloud platform

**Outcome**

The integration of the machine learning model with a web application allows for real-time user interaction. Users can now input custom data and immediately receive predictions, enhancing the accessibility and usability of the model.