**Instructions for Running the Code and API**

**1. Introduction**

This document provides detailed, step-by-step instructions for setting up, running, and testing the machine learning code and the Flask API developed for the Airbnb Market Analysis and Real Estate Sales Data project. All project files—including the source code (app.py), the serialized model (best\_model.pkl), and supporting documentation—are available on GitHub at the following URL:

<https://github.com/Pran-San/EAI6020-Assignment-3-Pranav-Sankar>

By following these instructions, you will be able to clone the repository, install the necessary dependencies, run the Flask server locally, and test the API via a web browser or using cURL commands.

**2. Prerequisites**

Before proceeding, ensure you have the following:

* **Python 3.x** installed on your system.
* Git installed (to clone the repository).
* Required Python packages:
  + flask
  + scikit-learn
  + numpy
  + pandas
* The project repository downloaded from GitHub contains all required files (e.g., app.py and best\_model.pkl).

**3. Cloning the Repository**

**Step 1:** Open Command Prompt, PowerShell, or your preferred terminal.

**Step 2:** Clone the repository from GitHub by running:

git clone https://github.com/Pran-San/EAI6020-Assignment-3-Pranav-Sankar.git

This command creates a local copy of the repository in a folder named EAI6020-Assignment-3-Pranav-Sankar.

**Step 3:** Navigate to the repository directory:

cd EAI6020-Assignment-3-Pranav-Sankar

**4. Installation of Dependencies**

Install the necessary packages using pip. In your terminal, run the following command from the repository directory:

pip install flask scikit-learn numpy pandas

This command installs all required libraries to run both the machine learning code and the Flask API.

**5. Running the Flask API**

**Step 1:** Start the Flask server. In your terminal, within the project directory, run:

python app.py

Upon execution, you should see output similar to the following:

\* Serving Flask app 'app'

\* Debug mode: on

WARNING: This is a development server. Do not use it in a production deployment.

\* Running on http://127.0.0.1:5000

Press CTRL+C to quit

**Step 2:** Access the API by opening your web browser and navigating to:

http://127.0.0.1:5000/

The home route is configured to return a JSON response with a default prediction and a descriptive message. A sample response may look like this:

{

"message": "Prediction of property revenue based on the input values. The input includes normalized features for bedrooms, bathrooms, guests, openness, occupancy, nightly rate, lead time, and length stay, plus one-hot encoded categorical features.",

"prediction": [ -0.05461 ]

}

**Note:** The prediction value is normalized. To convert it back to the original revenue units, you must apply the inverse transformation using the scaler used during preprocessing.

**6. Testing the API Using cURL in PowerShell**

Below are two examples of cURL commands you can use in PowerShell to test the API endpoints:

**6.1. GET Request (Default Home Route):**

To test the default home route that returns the sample prediction, run the following command in PowerShell:

curl.exe http://127.0.0.1:5000/

This command sends a GET request to the home route and displays the JSON response with the default prediction.

**6.2. POST Request (Predict Endpoint):**

If your API includes a POST endpoint (e.g., /predict), you can test it using the following command. This command sends a JSON payload with an array of 51 features (8 normalized numeric features followed by 43 zeros representing one-hot encoded categorical features):

curl.exe -X POST -H "Content-Type: application/json" -d "{\"input\": [-0.5, -0.3, 0.2, 0.1, 0.0, 0.05, -0.1, 0.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]}" http://127.0.0.1:5000/predict

This command sends a POST request to the /predict endpoint. The API should return a JSON response containing the predicted value.

**7. Code Structure Overview**

* **app.py:** Contains the Flask API code. This file:
  + Loads the pre-trained machine learning model from best\_model.pkl.
  + Defines a default home route (/) that returns a sample prediction along with an explanatory message.
  + Optionally defines a /predict endpoint that accepts POST requests with custom input data.
* **best\_model.pkl:** The serialized machine learning model file.

All project files are organized within the GitHub repository, ensuring that the project is fully reproducible.

**8. Troubleshooting**

* **404 Not Found Errors:**  
  If you receive a 404 error when navigating to http://127.0.0.1:5000/, verify that your app.py includes a route for / and that the file is saved and executed correctly.
* **Module Not Found Errors:**  
  Ensure that all required Python packages are installed by running the pip install command again if needed.
* **File Not Found Errors:**  
  Confirm that best\_model.pkl is in the same directory as app.py or update the file path in app.py accordingly.
* **Browser Cache Issues:**  
  If the expected output is not displayed, perform a hard refresh (Ctrl+F5) or use an incognito/private window.

**9. Summary**

By following these instructions, you will be able to:

* Clone the project repository from GitHub.
* Install the necessary dependencies.
* Run the Flask API server locally.
* Access and test the API using a web browser and cURL commands in PowerShell.

This document provides clear, concise guidance for setting up and interacting with the machine learning model and its associated API. For further assistance, please refer to the troubleshooting section.