

Customer Segmentation Project

Table of Contents

1. [Project Overview](#)
2. [Technologies Used](#)
3. [Installation](#)
4. [Project Structure](#)
5. [Implementation Steps](#)
6. [Running the Project](#)
7. [Usage Instructions](#)
8. [Conclusion](#)

Project Overview

The Customer Segmentation project aims to categorize customers based on their purchasing behavior using K-Means clustering. The project provides a web interface that allows users to upload their CSV data, select the clustering model, and visualize the resulting customer segments through a scatter plot.

Technologies Used

- **Flask:** A lightweight web framework for Python.
- **Pandas:** For data manipulation and analysis.
- **Matplotlib:** For creating static, animated, and interactive visualizations in Python.
- **Scikit-learn:** A library for machine learning in Python that provides simple and efficient tools for data mining and data analysis.

1. Ensure the directory structure includes:

- `app.py`: The main Flask application file.
- `templates/`: Folder containing HTML templates (e.g., `index.html`).
- `static/`: Folder to store static files like images.

2. Project Structure

```
Customer Segmentation Project/  
├─ app.py  
├─ templates/  
│   └─ index.html  
└─ static/
```

Implementation Steps

1. Set up Flask App:

- Create a Flask application instance and configure the static folder for storing uploaded files and generated plots.

2. Define Routes:

- **Home Route (/):** Renders the main page with the file upload form.
- **Upload Route (/upload):** Handles file uploads, applies K-Means clustering, generates a plot, and displays the results.

3. Handle CSV File Upload:

- Use Pandas to read the uploaded CSV file and apply K-Means clustering using Scikit-learn.

4. Generate Cluster Plot:

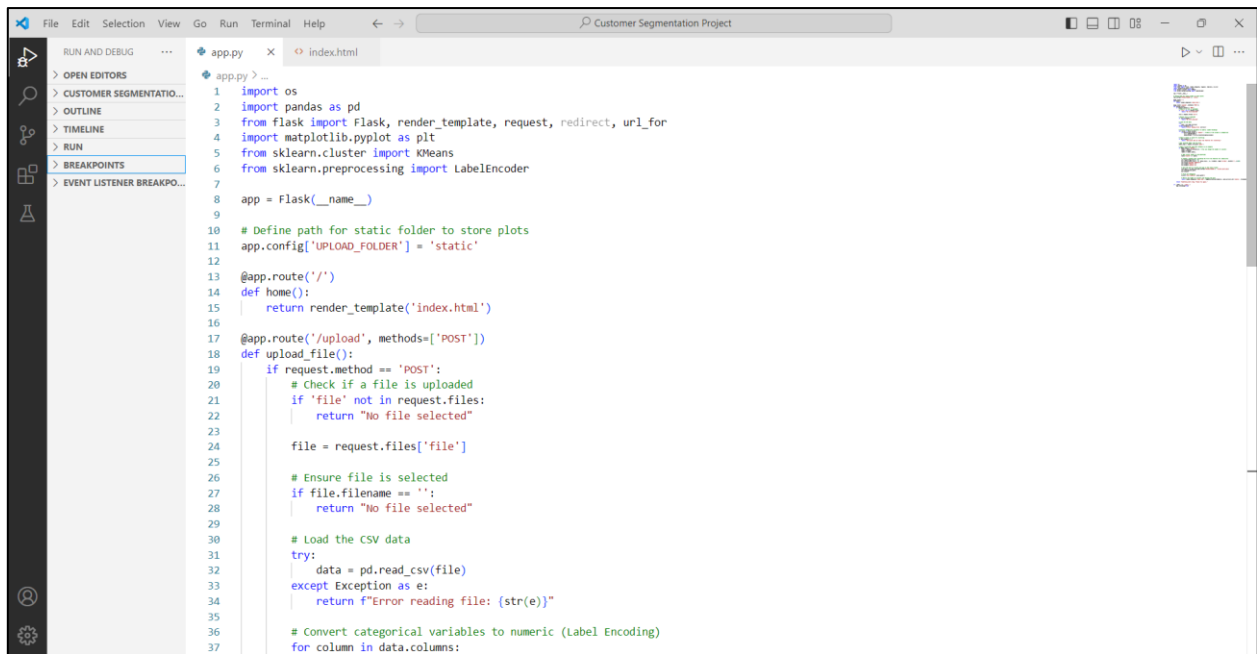
- Use Matplotlib to create a scatter plot of the clustered data and save it in the static folder for later display.

5. Render Results:

- Use Jinja2 templating to pass the number of clusters and the plot URL back to the HTML page.

Running the Project

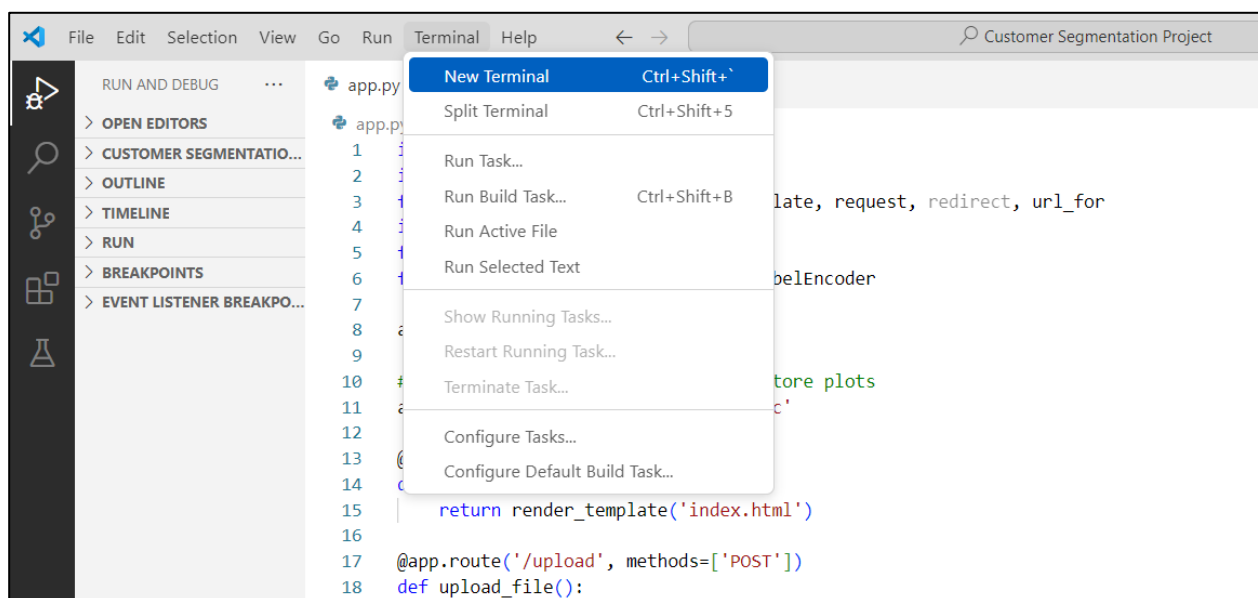
1. Open the app.py code in “visual code studio/ vs code” then in vs code open new terminal run **Run the Flask application:**



The screenshot shows the Visual Studio Code interface with the 'app.py' file open in the editor. The file contains Python code for a Flask application. A terminal window is open at the bottom, showing the command 'python app.py' and its output.

```
1 import os
2 import pandas as pd
3 from flask import Flask, render_template, request, redirect, url_for
4 import matplotlib.pyplot as plt
5 from sklearn.cluster import KMeans
6 from sklearn.preprocessing import LabelEncoder
7
8 app = Flask(__name__)
9
10 # Define path for static folder to store plots
11 app.config['UPLOAD_FOLDER'] = 'static'
12
13 @app.route('/')
14 def home():
15     return render_template('index.html')
16
17 @app.route('/upload', methods=['POST'])
18 def upload_file():
19     if request.method == 'POST':
20         # Check if a file is uploaded
21         if 'file' not in request.files:
22             return "No file selected"
23
24         file = request.files['file']
25
26         # Ensure file is selected
27         if file.filename == '':
28             return "No file selected"
29
30         # Load the CSV data
31         try:
32             data = pd.read_csv(file)
33         except Exception as e:
34             return f"Error reading file: {str(e)}"
35
36         # Convert categorical variables to numeric (Label Encoding)
37         for column in data.columns:
```

2. Open New Terminal:



3. Run the Flask application: **app.py**

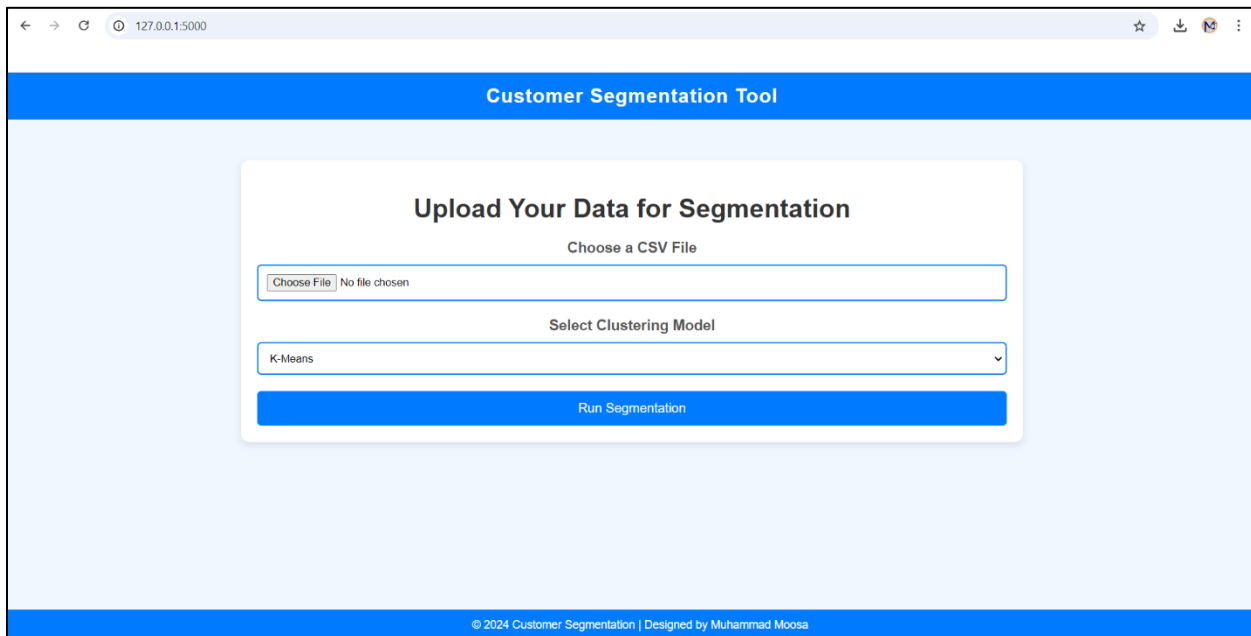


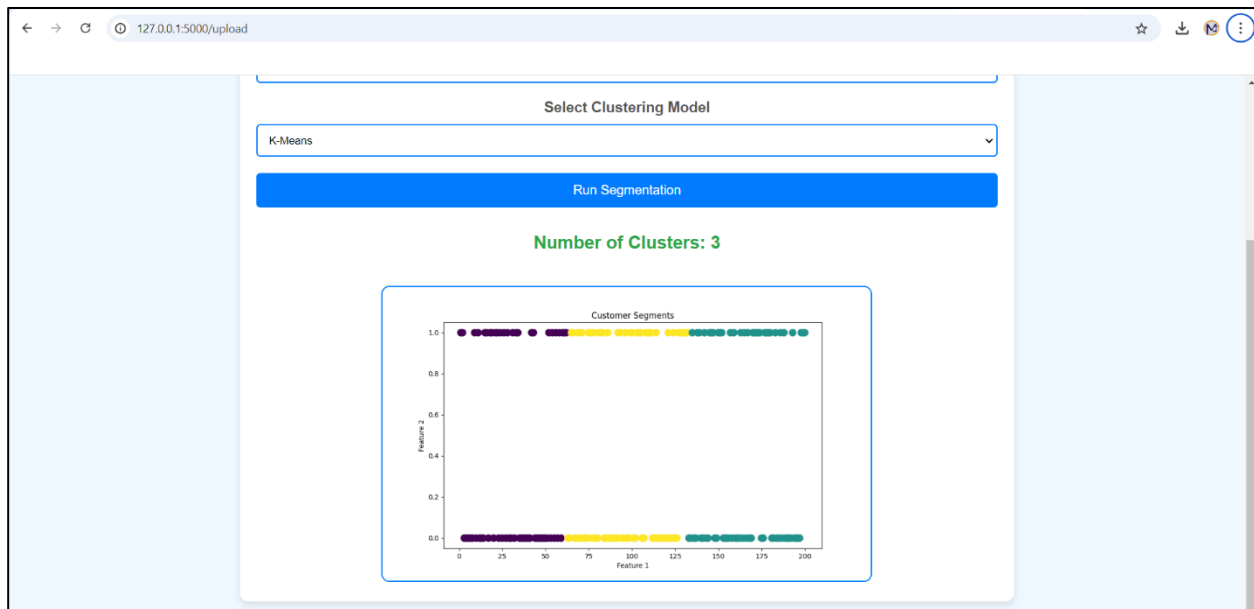
```
13 @app.route('/')  
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
PS C:\Users\PMLS\Desktop\Customer Segmentation Project>  
python app.py
```

4. Open your web browser and navigate to

* Running on `http://127.0.0.1:5000`

On Web browser output:





Choose the file of the your “dataset” file then click on button **On Segmentation**

Usage Instructions

1. Upload Data:

- Click on the "Choose a CSV File" button to select your data file. Ensure the file is in CSV format with at least two numeric features for clustering.

2. Select Clustering Model:

- Choose the K-Means model from the dropdown menu.

3. Run Segmentation:

- Click the "Run Segmentation" button to submit the form.

4. View Results:

- After processing, the page will display the number of clusters identified and the corresponding cluster plot.

Conclusion

This Customer Segmentation project serves as a practical implementation of K-Means clustering, providing a user-friendly interface for analyzing customer behavior. Users can easily visualize segmentation results, aiding in better decision-making for marketing and sales strategies.