🧠 Clustering Dietary Patterns Using K-Means in Python

📂 GitHub Repository:
https://github.comjohnmicky1Dietary-Behavior-Clustering-with-KMeans

📓 Overview
This Python script demonstrates how to apply **K-Means Clustering** to group individuals based on their **dietary preferences** using synthetic data.
The dataset contains three key features representing consumption scores for different food categories:

1. **Meat_Score** – Measures the level of meat consumption *(0 = none, 10 = high)*
2. **Dairy_Egg_Score** – Measures the consumption of dairy and egg products *(0 = none, 10 = high)*
3. **Plant_Protein_Score** – Measures the reliance on plant-based protein sources *(0 = low, 10 = high)*

🏵 Workflow Summary
- **Data Preparation:**
  Created a small illustrative dataset using **Pandas**.
- **Data Standardization:**
  Applied **StandardScaler** from `sklearn.preprocessing` to normalize all features so that each contributes equally to the clustering process.
- **Clustering:**
  Used the **KMeans** algorithm from `sklearn.cluster` to group the data into **three clusters**, representing distinct dietary patterns — likely **Non-Vegetarian**, **Vegetarian**, and **Vegan** categories.
- **Cluster Interpretation:**
  Calculated and displayed the **mean feature values** for each cluster to interpret the key dietary characteristics of each group.

□ Purpose
This example provides a simple yet powerful demonstration of how **unsupervised learning** can be applied to uncover hidden patterns and similarities in behavioral or lifestyle data — in this case, **dietary habits**.
It illustrates the end-to-end workflow from data preprocessing to clustering and result interpretation.

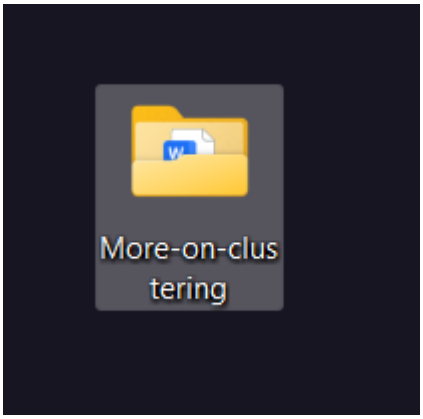
□ Prerequisites
Before running this script, ensure you have the following tools and libraries installed:

- 🧠 **Python 3.10+** *(latest version recommended)*
- 🤖 **TensorFlow** *(optional for advanced extensions)*
- 🔢 **NumPy** and **Pandas** for data handling
- 📊 **Matplotlib** *(optional for visualizations)*
- 💻 **Code Editor / IDE** — e.g., VS Code, PyCharm, or Jupyter Notebook

💡 Installation Commands
Use the following commands to install the required Python packages:
```
pip install numpy pandas scikit-learn matplotlib tensorflow
```

? Step 1: Project Setup

1. Create a new folder or directory named `TensorFlow-Customer-Classification-Model`.



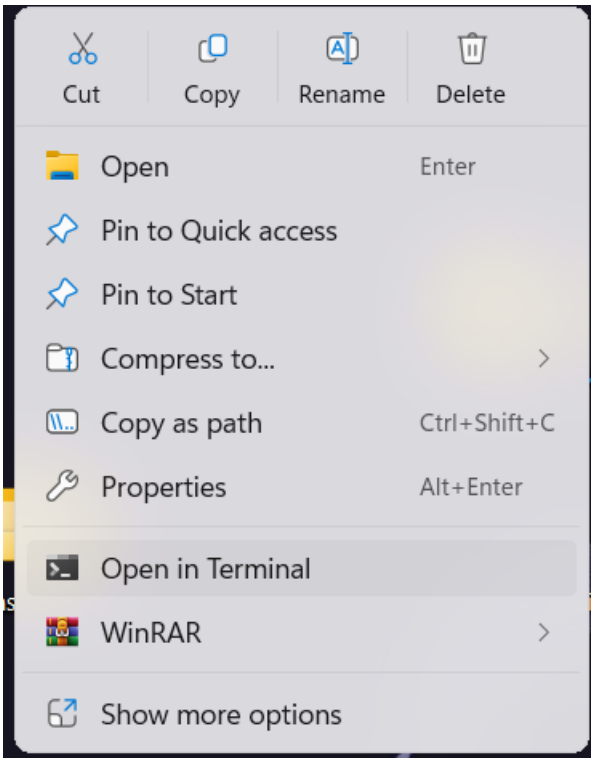
2. Open your preferred code editor (VS Code, Notepad++, etc.).
3. Create a new Python file and name it `tensorflow-customer-classification-model.py using the code below`. Then save it in the folder.

```
import pandas as pd

from sklearn.preprocessing import StandardScaler

from sklearn.cluster import KMeans

import numpy as np



# Illustrative Dataset (Synthetic Data)

# Features:

# 1. Meat_Score (0=None, 10=High)

# 2. Dairy_Egg_Score (0=None, 10=High)

# 3. Plant_Protein_Score (0=Low, 10=High)



data = {

    'Meat_Score': [8, 9, 7, 0, 0, 0, 0, 0, 0, 5, 4, 6],

    'Dairy_Egg_Score': [7, 6, 8, 9, 8, 7, 0, 1, 0, 5, 6, 4],

    'Plant_Protein_Score': [4, 3, 5, 7, 8, 6, 9, 8, 7, 5, 6, 4]

}

df = pd.DataFrame(data)
```

```
#  Prepare Data for Clustering
# Scaling is crucial for K-Means to prevent features with larger values from dominating.
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df)

# 🛠️ Execute K-Means
# We set n_clusters=3 to match your 3 target groups (Vegetarian, Non-Vegetarian, Vegan)
kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)
df['Cluster'] = kmeans.fit_predict(df_scaled)

#  Interpret Cluster Centers
# Check the mean of the original (unscaled) features for each cluster
cluster_centers = df.groupby('Cluster').mean().sort_values(by='Meat_Score', ascending=False)

print("--- Cluster Centers (Mean Feature Values) ---")
print(cluster_centers)

print("\n--- Final Assignments ---")
print(df)
```

▶ Step 2: Run the Code

1. **Right Click on folder/Directory (TensorFlow-Customer-Classificatio-Model)- Select Open in Terminal**



2. **PowerShell will Open**

ls

```
PS C:\Users\johnm\OneDrive\Desktop\More-on-clustering> ls


    Directory: C:\Users\johnm\OneDrive\Desktop\More-on-clustering


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        10/28/2025   8:20 PM           1266 eating_food.py
```

⚙ Step 3: Run the Model

Use PowerShell or your terminal to run and verify the model setup.

python -c "import tensorflow as tf; print('TensorFlow version:', tf.__version__); print('GPUs:', tf.config.list_physical_devices('GPU'))"

```
PS C:\Users\johnm\OneDrive\Desktop\More-on-clustering> python -c "import tensorflow as tf; print('TensorFlow version:', tf.__version__); print('GPUs:', tf.config.list_physical_devices('GPU'))"
2025-10-28 20:20:57.718027: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2025-10-28 20:21:01.361393: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
TensorFlow version: 2.20.0
GPUs: []
```

python eating_food.py

```
PS C:\Users\johnm\OneDrive\Desktop\More-on-clustering> python eating_food.py
--- Cluster Centers (Mean Feature Values) ---
         Meat_Score  Dairy_Egg_Score  Plant_Protein_Score
Cluster
0               6.5         6.000000                  4.5
1               0.0         0.333333                  8.0
2               0.0         8.000000                  7.0

--- Final Assignments ---
    Meat_Score  Dairy_Egg_Score  Plant_Protein_Score  Cluster
0            8                7                    4        0
1            9                6                    3        0
2            7                8                    5        0
3            0                9                    7        2
4            0                8                    8        2
5            0                7                    6        2
6            0                0                    9        1
7            0                1                    8        1
8            0                0                    7        1
9            5                5                    5        0
10           4                6                    6        0
11           6                4                    4        0
```

☑ End of Guide — TensorFlow Customer Classification Model successfully documented.

GitHub Link: https://github.comjohnmicky1Dietary-Behavior-Clustering-with-KMeans