Introduction:

We investigate how to use clustering approaches to examine data from simulated fitness trac ker.

We want to find different user groups according to how many steps they do each day and ho w long they sleep by using the KMeans algorithm.This analysis illustrates the usefulness of cl ustering in exploratory data analysis by assisting us in identifying trends in activity levels and sleeping patterns.

Learning Objective:

Through this task, we aim to:

-Use Jupyter Notebook or Google Colab to manipulate and analyze data.

-Get practical experience using scikit-learn and NumPy.

-Use Matplotlib to efficiently visualize datasets.

-Use clustering algorithms to group users according to their sleep and activity habits.

Data Visualization:

The raw fitness tracker data is displayed as a scatter plot, with steps per day on the x-axis and sleep time on the y-axis.

Prior to grouping, the visualization aids in the identification of first trends.

KMeans Clustering Implementations:

We divide users into three groups according to their rest and activity patterns using scikit-learn's KMeans clustering algorithm. Each user is assigned to a cluster by the system, which enables us to compare behavioural variations among groups.

Cluster Visualization:

We make a scatter plot with different colours for each of them to help visualize the clustering findings, this picture shows how the clustering algorithm grouping is according to their sleeping patterns and level of fitness.

Cluster Analysis:

The findings indicate:

-**Cluster 1 (Highly Active):** Users in this group sleep for seven to nine hours every night and

walk a lot.
-**Cluster 2 (Moderately Active):** Users slept for 6–7.5 hours and were moderately active.
-**Cluster 3 (Least Active):** These individuals sleep for an average of 5 to 6.5 hours each night and take the fewest daily steps.
-Each cluster's average step count and sleep duration reveal details on sleep and exercise habits.

Observations & Insights:

Higher activity levels are often associated with longer sleep durations.
The clustering algorithm efficiently differentiates between user groups based on movement and sleep data.This method can be applied to realworld scenarios, such as individualized fitn ess advice.

Conclusion:

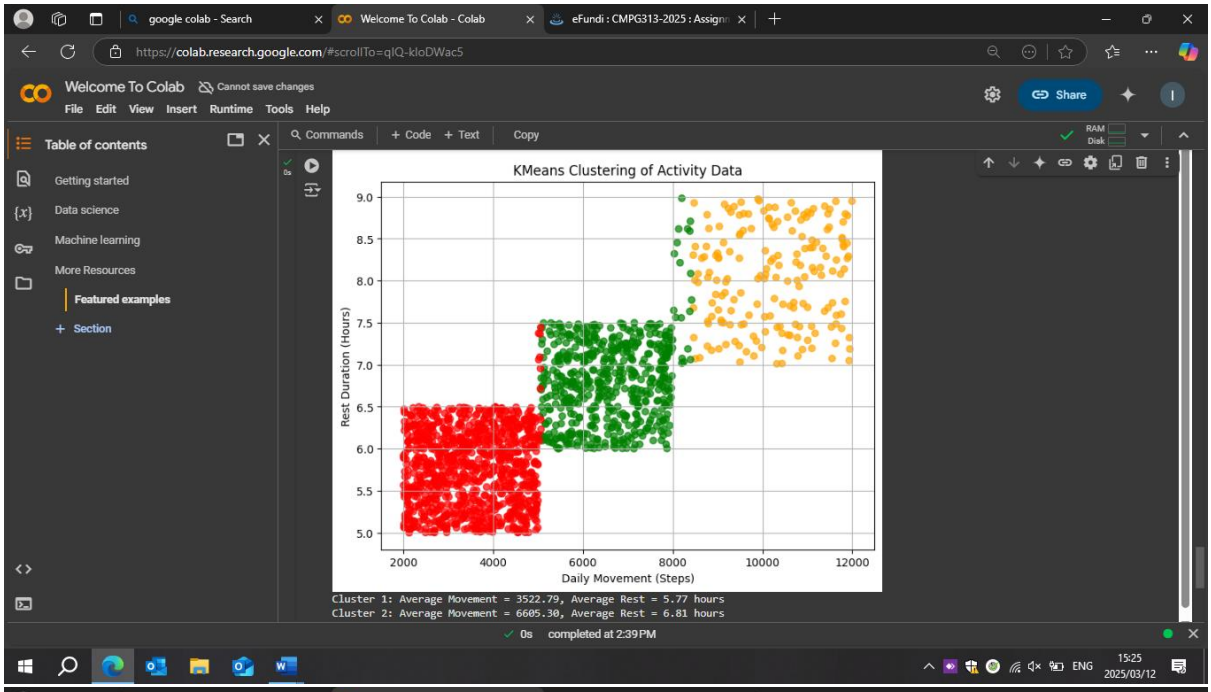
Participants gained invaluable experience with data visualization, clustering techniques, and using Python tools for data analysis during this practical assessment. The information gained emphasizes how important data-driven decision-making is for fitness and health-related applications.

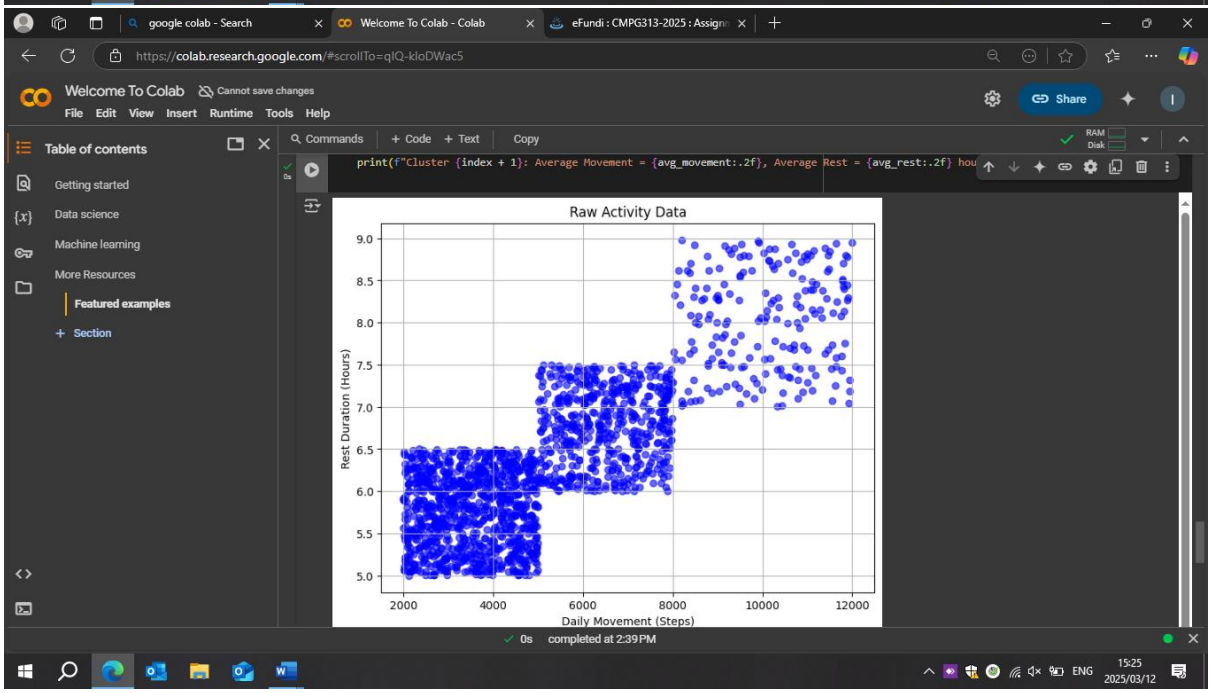
Full Code:

```
import numpy as np

import matplotlib.pyplot as plt

from sklearn.cluster import KMeans
```

```
# Setting a random seed for consistency
np.random.seed(42)

# Define the number of individuals in each category (random range)
members_group1 = np.random.randint(100, 1000)
members_group2 = np.random.randint(100, 1000)
members_group3 = np.random.randint(100, 1000)

# Generate movement and rest data for each category
movement_group1 = np.random.randint(8000, 12001, members_group1)
rest_group1 = np.random.uniform(7, 9, members_group1)

movement_group2 = np.random.randint(5000, 8001, members_group2)
rest_group2 = np.random.uniform(6, 7.5, members_group2)

movement_group3 = np.random.randint(2000, 5001, members_group3)
rest_group3 = np.random.uniform(5, 6.5, members_group3)

# Combine all data
movement = np.concatenate((movement_group1, movement_group2, movement_group3))
rest = np.concatenate((rest_group1, rest_group2, rest_group3))
data_matrix = np.column_stack((movement, rest))

# Scatter plot of the generated data
plt.figure(figsize=(8,6))
plt.scatter(movement, rest, alpha=0.6, color='blue')
plt.xlabel("Daily Movement (Steps)")
plt.ylabel("Rest Duration (Hours)")
plt.title("Raw Activity Data")
plt.grid(True)
```

```
plt.show()


# Apply KMeans clustering
kmeans_model = KMeans(n_clusters=3, random_state=42, n_init=10)
kmeans_model.fit(data_matrix)
cluster_labels = kmeans_model.labels_

# Custom cluster colors
colors_palette = ['red', 'green', 'orange']
cluster_colors = [colors_palette[label] for label in cluster_labels]

# Scatter plot of clustered data
plt.figure(figsize=(8,6))
plt.scatter(movement, rest, c=cluster_colors, alpha=0.7)
plt.xlabel("Daily Movement (Steps)")
plt.ylabel("Rest Duration (Hours)")
plt.title("KMeans Clustering of Activity Data")
plt.grid(True)
plt.show()

# Print average movement and rest duration for each cluster
for index in range(3):
    cluster_points = data_matrix[cluster_labels == index]
    avg_movement = np.mean(cluster_points[:, 0])
    avg_rest = np.mean(cluster_points[:, 1])
    print(f"Cluster {index + 1}: Average Movement = {avg_movement:.2f}, Average Rest = {avg_rest:.2f}
hours")
```


google colab - Search
Welcome To Colab - Colab
eFundi : CMPG313-2025 : Assign
https://colab.research.google.com/#scrollTo=qlQ-kloDWac5
Welcome To Colab
Cannot save changes
File Edit View Insert Runtime Tools Help
Share
Table of contents
Getting started
Data science
Machine learning
More Resources
Featured examples
Section
Commands
+ Code
+ Text
Copy
KMeans Clustering of Activity Data
Rest Duration (Hours)
Daily Movement (Steps)
Cluster 1: Average Movement = 3522.79, Average Rest = 5.77 hours
Cluster 2: Average Movement = 6605.30, Average Rest = 6.81 hours
0s completed at 2:39PM
ENG
15:25
2025/03/12


google colab - Search
Welcome To Colab - Colab
eFundi : CMPG313-2025 : Assign
https://colab.research.google.com/#scrollTo=qlQ-kloDWac5
Welcome To Colab
Cannot save changes
File Edit View Insert Runtime Tools Help
Share
Table of contents
Getting started
Data science
Machine learning
More Resources
Featured examples
Section
Commands
+ Code
+ Text
Copy
print(f"Cluster {index + 1}: Average Movement = {avg_movement:.2f}, Average Rest = {avg_rest:.2f} hou
Raw Activity Data
Rest Duration (Hours)
Daily Movement (Steps)
0s completed at 2:39PM
ENG
15:25
2025/03/12