# **https://www.epfl.ch/labs/cvlab/data/ski-2dpose-dataset/**

# World Happiness Report Dataset

Countries of the World

https://www.kaggle.com/datasets/fernandol/countries-of-the-world

import kagglehub

# Download latest version

path = kagglehub.dataset\_download("fernandol/countries-of-the-world")

print("Path to dataset files:", path)

* + Source: Kaggle.
  + **Use Case:** Cluster countries based on happiness scores and contributing factors.
  + **Why it’s good:** Multi-dimensional dataset for clustering techniques like k-means.

### **Why It's a Good Choice**

* **Real-World Relevance**: Understanding how happiness correlates with factors like GDP, social support, or life expectancy is highly impactful.
* **Multi-Dimensional Data**: Provides rich features for dimensionality reduction and clustering.
* **Beginner-Friendly**: Ideal for exploring clustering techniques like k-means or hierarchical clustering.

### **Project Plan**

#### ****Objective****

Cluster countries into groups based on their happiness scores and contributing factors. For example:

* Identify groups of countries with similar levels of happiness.
* Analyze what factors contribute most to these clusters.

#### ****Dataset Details****

* Source: [World Happiness Report Dataset on Kaggle](https://www.kaggle.com/).
* Typical columns:
  + Country name.
  + Happiness Score/Rank.
  + GDP per capita.
  + Social support.
  + Life expectancy.
  + Freedom to make life choices.
  + Generosity.
  + Perceptions of corruption.

### **Steps**

#### ****1. Data Collection****

* Download the dataset from Kaggle as a CSV file.
* Load it into Python using pandas:

python

Copy code

import pandas as pd

data = pd.read\_csv("world\_happiness\_report.csv")

print(data.head())

#### ****2. Data Preprocessing****

* **Check for Missing Values**:

python

Copy code

print(data.isnull().sum())

Fill missing values or drop rows if necessary.

* **Feature Selection**: Choose relevant numerical features (e.g., GDP, Social Support, etc.) for clustering. Exclude non-numeric data (like country names) from the analysis.
* **Normalization**: Standardize features to ensure all have equal weight:

python

Copy code

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

scaled\_data = scaler.fit\_transform(data[['GDP', 'Social Support', 'Life Expectancy']])

#### ****3. Exploratory Data Analysis (EDA)****

* Visualize distributions and relationships between variables (e.g., happiness vs. GDP).
* Use pair plots or correlation heatmaps to understand feature importance:

python

Copy code

import seaborn as sns

import matplotlib.pyplot as plt

sns.heatmap(data.corr(), annot=True, cmap='coolwarm')

plt.show()

#### ****4. Clustering****

* **K-Means Clustering**:
  + Choose the optimal number of clusters using the elbow method:

python

Copy code

from sklearn.cluster import KMeans

import matplotlib.pyplot as plt

inertia = []

for k in range(1, 10):

kmeans = KMeans(n\_clusters=k)

kmeans.fit(scaled\_data)

inertia.append(kmeans.inertia\_)

plt.plot(range(1, 10), inertia, marker='o')

plt.title("Elbow Method")

plt.xlabel("Number of Clusters")

plt.ylabel("Inertia")

plt.show()

* + Fit the model and assign clusters:

python

Copy code

kmeans = KMeans(n\_clusters=3)

data['Cluster'] = kmeans.fit\_predict(scaled\_data)

* **Alternative Clustering Methods**:
  + Hierarchical clustering for visual dendrograms.
  + DBSCAN for density-based clustering.

#### ****5. Visualization****

* Use scatter plots or 3D visualizations to display clusters:

python

Copy code

import matplotlib.pyplot as plt

plt.scatter(scaled\_data[:, 0], scaled\_data[:, 1], c=data['Cluster'], cmap='viridis')

plt.xlabel('GDP')

plt.ylabel('Social Support')

plt.title('Clusters of Countries')

plt.show()

* Use a world map for geographic visualization of clusters.

#### ****6. Insights and Reporting****

* Identify key characteristics of each cluster.
* Analyze which factors (e.g., GDP, social support) are most significant in differentiating clusters.

#### ****7. Evaluate and Document****

* Evaluate clustering using metrics like silhouette score:

python

Copy code

from sklearn.metrics import silhouette\_score

score = silhouette\_score(scaled\_data, data['Cluster'])

print("Silhouette Score:", score)

* Report findings in a clear and concise format

### **Time and Complexity**

#### ****For a Pro****

* **Data Collection and Preprocessing**: 2–3 hours.
* **EDA**: 2–3 hours.
* **Clustering and Optimization**: 4–6 hours.
* **Visualization**: 2–4 hours.
* **Reporting**: 3–4 hours.
* **Total**: ~15–20 hours.

#### ****For a Newbie****

* **Learning Curve**:
  + Basics of clustering (e.g., k-means): 4–6 hours.
  + Familiarity with Python libraries: 4–6 hours.
* **Data Collection and Preprocessing**: 4–6 hours.
* **EDA**: 6–8 hours.
* **Clustering and Optimization**: 6–10 hours.
* **Visualization**: 4–6 hours.
* **Reporting**: 4–6 hours.
* **Total**: ~30–40 hours.

### **Research Question Example**

**"Can countries be grouped into distinct clusters based on their happiness scores and socio-economic factors, and what are the key characteristics of these clusters?"**