ASSIGNMENT - 2

(Q1) You're a botanist on a research mission to classify rare iris flowers into three species: **Setosa**, **Versicolor**, and **Virginica**, based on their **sepal length**, **sepal width**, **petal length**, and **petal width**. Using the famous Iris dataset from Kaggle, you'll build a KNN classifier to identify these species accurately. However, measurement errors in the field introduce noise, challenging your model's robustness.

Dataset

Use the **Iris Species** dataset from Kaggle:
https://www.kaggle.com/datasets/uciml/iris

- **Features**: sepal_length, sepal_width, petal_length, petal_width (all in cm, continuous).
- **Target**: species (Setosa, Versicolor, Virginica).
- **Size**: 150 samples, balanced across three classes.
- **Note**: Assume 5% of the data contains minor noise (e.g., measurement errors).

Tasks

1. **Data Exploration (10 points)**:
    - Load the dataset using Pandas.
    - Create a 2D scatter plot (Matplotlib/Seaborn) of petal_length vs. petal_width, color-coded by species, to visualize class separation (Page 15: similarity principle).
2. **KNN Classification (20 points)**:
    - Split the data into 80% training and 20% testing sets (random_state=42, Page 19).
    - Train two KNN models (n_neighbors=5) using **Euclidean** and **Manhattan** distances (Page 21).
    - Report test accuracy for both using score().
3. **Hyperparameter Tuning (15 points)**:
    - Test n_neighbors from 1 to 10 for the Euclidean model.
    - Plot test accuracy vs. k to find the optimal k.
4. **Reflection (15 points)**:
    - Explain which distance metric performed better and why, referencing KNN's sensitivity to outliers (Page 18) and the Iris dataset's characteristics.
    - Suggest one preprocessing step (e.g., scaling) to improve KNN performance and justify it.

(Q2) Dataset:- synthetic_customers

Dataset

- Rows: 300
- Columns: CustomerID, Name, Age, Gender, Income, Spending Score, Signup Date, Notes

Objective

Your goal is to **clean the raw dataset** and **apply K-Means clustering** to segment customers based on their attributes.

1. Data Cleaning

Clean the dataset to make it suitable for K-Means:

- Drop irrelevant columns (e.g., CustomerID, Name, Signup Date, Notes).
- Fix inconsistent categorical values in Gender (e.g., M vs Male, F vs Female).
- Handle missing values (either drop or impute appropriately).
- Convert string-based numbers (e.g., "forty thousand") to integers.
- Convert non-numeric fields (e.g., "missing" in Spending Score) to NaN.
- Ensure correct data types for all columns.

2. Data Preprocessing

- Encode categorical values using one-hot encoding (e.g., Gender).
- Normalize/standardize all numeric columns using StandardScaler.

3. K-Means Clustering

- Use the Elbow Method to determine an optimal value of K.

- Apply KMeans clustering on the cleaned and scaled dataset.
- Add the resulting cluster label as a new column called Cluster.

4. Visualization & Insights

- Plot a 2D scatter plot using Income and Spending Score colored by clusters.
- Print the count of customers in each cluster.
- Analyze any patterns (e.g., high-income low-spending cluster?).

Submission Requirements

- Python script or Jupyter Notebook (.ipynb) with all code and explanations.
- Cleaned version of the dataset (CSV or in-memory as a DataFrame).
- A short write-up (markdown or cell text) explaining:
    - Cleaning decisions
    - Chosen K and why
    - Meaningful insights from clusters

Hints

- Use pandas, numpy, matplotlib, seaborn, and scikit-learn.
- Check for unusual values or string labels where numbers should be.
- If a column contains more than 20% missing data, consider dropping it