Q1 - K-Means Clustering (HOUSING Dataset) - Detailed Answers

Problem statement

Apply simple K-means algorithm for clustering the HOUSING dataset. Pre-process the dataset as required. Compare the performance of clusters by varying k.

Data preprocessing

1. Load dataset (e.g., housing.csv). 2. Inspect missing values and outliers. Impute missing numeric values using median or mean. 3. Drop or encode categorical features (one-hot or label encoding). 4. Standardize/normalize numerical features (StandardScaler or MinMaxScaler). 5. Optionally reduce dimensionality using PCA for visualization.

K-means steps & evaluation

1. Use sklearn.cluster.KMeans. 2. Run KMeans for different k values (e.g., 2..8). 3. For each k record inertia (within-cluster sum of squares) and silhouette_score. 4. Plot Elbow (k vs inertia) and Silhouette (k vs silhouette score). 5. Choose k balancing inertia decrease and silhouette. 6. Interpret cluster centers and profile clusters by features.

Sample Python code

```
# K-means on HOUSING dataset (example)
import pandas as pd
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
import matplotlib.pyplot as plt

df = pd.read_csv('housing.csv')  # replace with filename
# -- preprocessing --
num_cols = df.select_dtypes(include=['int64','float64']).columns
imputer = SimpleImputer(strategy='median')
df[num_cols] = imputer.fit_transform(df[num_cols])
df = pd.get_dummies(df, drop_first=True)  # encode categoricals
scaler = StandardScaler()
X = scaler.fit_transform(df)

inertias = []
sil_scores = []
K = range(2,9)
for k in K:
    km = KMeans(n_clusters=k, random_state=42, n_init=10)
    labels = km.fit_predict(X)
    inertias.append(km.inertia_)
    sil_scores.append(silhouette_score(X, labels))
plt.figure()
plt.plot(K, inertias, '-o'); plt.xlabel('k'); plt.ylabel('Inertia (SSE)'); plt.title('Elbow Meth
plt.figure()
plt.plot(K, sil_scores, '-o'); plt.xlabel('k'); plt.ylabel('Silhouette Score'); plt.title('Silho
plt.show()

# After choosing k, inspect centers (inverse transform if scaled)
best_k = 3
km = KMeans(n_clusters=best_k, random_state=42).fit(X)
centers = scaler.inverse_transform(km.cluster_centers_)
centers_df = pd.DataFrame(centers, columns=df.columns)
print(centers_df.T)
```

Interpretation & reporting

Report: chosen k, inertia, silhouette score, cluster sizes, and cluster profiles (which features are high/low). Discuss whether clusters are meaningful for housing (e.g., high-price vs low-price regions, feature combinations). Mention limitations (KMeans assumes spherical clusters, sensitive to scaling and outliers).