T4 22510064

PARSHWA HERWADE

ML Assignment 9 : Clustering

Download the following dataset :

https://www.kaggle.com/datasets/erdemtaha/cancer-data/data

1. Drop Id and Diagnosis columns
2. Cluster the data using K means clustering :
    a. Determine optimal number of clusters (between 2-10) using
        i. Elbow method with Inertia and
        ii. Silhouette Analysis
    b. Use the diagnosis column to calculate homogeneity score and evaluate quality of clustering
3. Cluster the data using DB scan clustering.
    a. Determine appropriate values of eps and min_samples parameters
    b. Use the diagnosis column to calculate homogeneity score and evaluate quality of clustering

CODE:

```
# Import necessary libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans, DBSCAN
from sklearn.metrics import silhouette_score, homogeneity_score
from sklearn.neighbors import NearestNeighbors
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

##############################################
# 1. Data Loading and Preprocessing
##############################################

# Read the dataset
data = pd.read_csv("Cancer_Data.csv")

columns_to_drop = []
```

```
if "id" in data.columns:
    columns_to_drop.append("id")
if "diagnosis" in data.columns:
    columns_to_drop.append("diagnosis")
if "Unnamed: 32" in data.columns:
    columns_to_drop.append("Unnamed: 32")

if "diagnosis" in data.columns:
    diagnosis = data["diagnosis"]
    diagnosis_numeric = diagnosis.map({'M': 1, 'B': 0})
else:
    diagnosis = None
    diagnosis_numeric = None

features = data.drop(columns=columns_to_drop, errors='ignore')
features = features.dropna()

# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(features)

################################################
# 2. K-Means: Elbow and Silhouette on Original Data
################################################
print("\n--- K-Means Clustering on Original Data ---")
inertias = []
silhouette_avgs = []
range_n_clusters = range(2, 11)

for n_clusters in range_n_clusters:
    kmeans = KMeans(n_clusters=n_clusters, random_state=42)
    labels = kmeans.fit_predict(X_scaled)
    inertias.append(kmeans.inertia_)
    silhouette_avgs.append(silhouette_score(X_scaled, labels))

plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)
plt.plot(range_n_clusters, inertias, marker='o')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('Inertia')
plt.title('Elbow Method (Original Data)')
```

```
plt.subplot(1, 2, 2)
plt.plot(range_n_clusters, silhouette_avgs, marker='o', color='red')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('Silhouette Score')
plt.title('Silhouette Analysis (Original Data)')
plt.tight_layout()
plt.show()

kmeans_optimal = KMeans(n_clusters=2, random_state=42)
kmeans_labels = kmeans_optimal.fit_predict(X_scaled)
if diagnosis_numeric is not None:
    score = homogeneity_score(diagnosis_numeric.loc[features.index],
kmeans_labels)
    print(f"K-Means Homogeneity Score (Original Data): {score:.4f}")

##############################################
# 3. DBSCAN on Original Data
##############################################
print("\n--- DBSCAN Clustering on Original Data ---")
neighbors = 5
neigh = NearestNeighbors(n_neighbors=neighbors)
nbrs = neigh.fit(X_scaled)
distances, _ = nbrs.kneighbors(X_scaled)
kth_distances = np.sort(distances[:, -1])
plt.figure(figsize=(8, 4))
plt.plot(kth_distances)
plt.xlabel("Sorted Data Points")
plt.ylabel(f"{neighbors}-th NN Distance")
plt.title("K-distance Graph (Original Data)")
plt.show()

best_score = -1
best_params = {}
for eps in np.linspace(0.5, 2.0, 16):
    for min_samples in range(3, 10):
        dbscan = DBSCAN(eps=eps, min_samples=min_samples)
        labels = dbscan.fit_predict(X_scaled)
        unique_labels = set(labels)
        if len(unique_labels) <= 1 or (len(unique_labels) == 1 and -
1 in unique_labels):
            continue
```

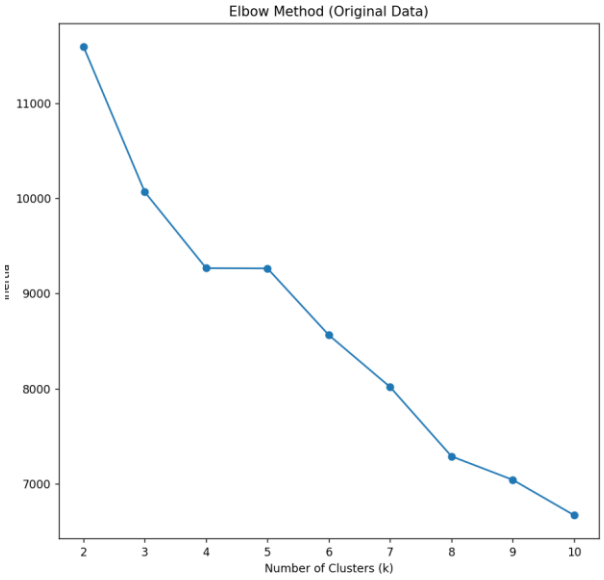
```
        score =
homogeneity_score(diagnosis_numeric.loc[features.index], labels)
        if score > best_score:
            best_score = score
            best_params = {'eps': eps, 'min_samples': min_samples}

if best_params:
    dbscan = DBSCAN(**best_params)
    dbscan_labels = dbscan.fit_predict(X_scaled)
    n_clusters = len(set(dbscan_labels)) - (1 if -1 in dbscan_labels
else 0)
    print(f"Best DBSCAN (Original): eps={best_params['eps']:.2f},
min_samples={best_params['min_samples']}")
    print(f"DBSCAN found {n_clusters} clusters. Homogeneity Score:
{best_score:.4f}")
else:
    print("No valid DBSCAN clustering found on original data.")

##############################################
# 4. PCA Transformation
##############################################
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
print("\nExplained Variance Ratio (PCA):",
pca.explained_variance_ratio_)
print("Total Variance Explained: {:.2f}%".format(100 *
np.sum(pca.explained_variance_ratio_)))

if diagnosis_numeric is not None:
    plt.figure(figsize=(7, 5))
    plt.scatter(X_pca[:, 0], X_pca[:, 1],
c=diagnosis_numeric.loc[features.index], cmap='viridis',
edgecolor='k')
    plt.xlabel("PCA 1")
    plt.ylabel("PCA 2")
    plt.title("PCA Projection (True Labels)")
    plt.colorbar(label='Diagnosis (0: Benign, 1: Malignant)')
    plt.show()

##############################################
# 5. K-Means on PCA Data
##############################################
```

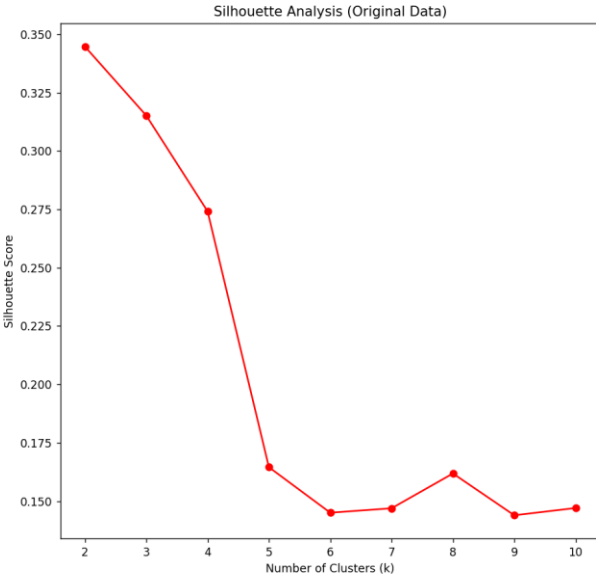
```
print("\n--- K-Means Clustering on PCA Data ---")
kmeans_pca = KMeans(n_clusters=2, random_state=42)
kmeans_pca_labels = kmeans_pca.fit_predict(X_pca)

if diagnosis_numeric is not None:
    score = homogeneity_score(diagnosis_numeric.loc[features.index],
kmeans_pca_labels)
    print(f"K-Means Homogeneity Score (PCA): {score:.4f}")

plt.figure(figsize=(7, 5))
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=kmeans_pca_labels,
cmap='viridis', edgecolor='k')
plt.xlabel("PCA 1")
plt.ylabel("PCA 2")
plt.title("K-Means Clusters (PCA)")
plt.show()

##############################################
# 6. DBSCAN on PCA Data
##############################################
print("\n--- DBSCAN Clustering on PCA Data ---")
neighbors = 5
neigh = NearestNeighbors(n_neighbors=neighbors)
nbrs = neigh.fit(X_pca)
distances, _ = nbrs.kneighbors(X_pca)
kth_distances = np.sort(distances[:, -1])
plt.figure(figsize=(8, 4))
plt.plot(kth_distances)
plt.xlabel("Sorted Points")
plt.ylabel(f"{neighbors}-th NN Distance")
plt.title("K-distance Graph (PCA Data)")
plt.show()

best_dbscan_score = -1
best_dbscan_params = {}
for eps in np.linspace(0.1, 1.5, 15):
    for min_samples in range(2, 10):
        dbscan = DBSCAN(eps=eps, min_samples=min_samples)
        labels = dbscan.fit_predict(X_pca)
        unique_labels = set(labels)
        if len(unique_labels) < 2 or (len(unique_labels) == 1 and -1
in unique_labels):
```

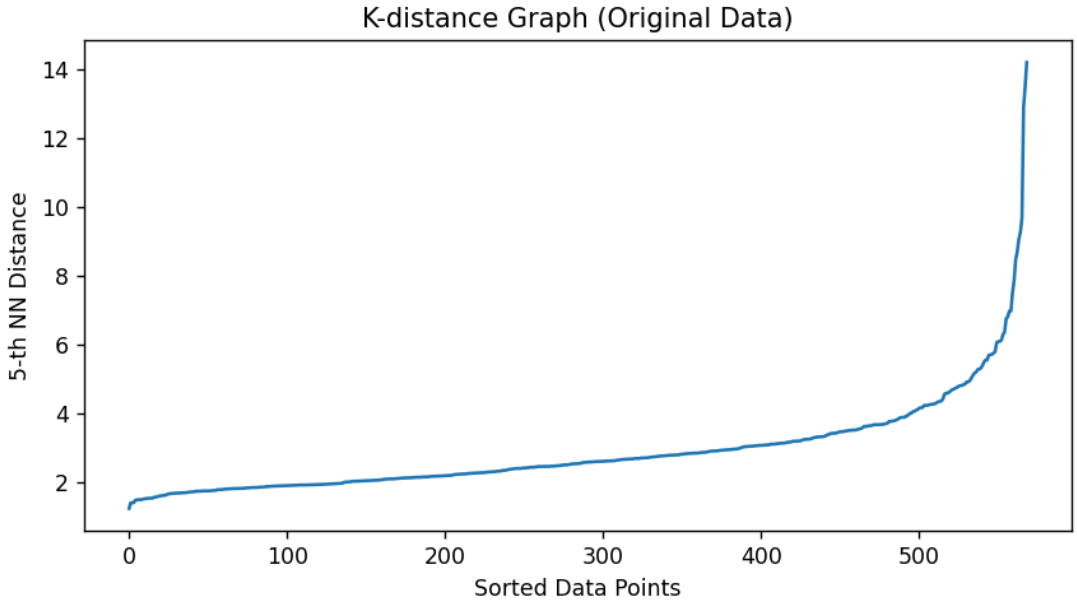
```
            continue
        score =
homogeneity_score(diagnosis_numeric.loc[features.index], labels)
        if score > best_dbscan_score:
            best_dbscan_score = score
            best_dbscan_params = {"eps": eps, "min_samples":
min_samples}

if best_dbscan_params:
    dbscan = DBSCAN(**best_dbscan_params)
    labels = dbscan.fit_predict(X_pca)
    n_clusters = len(set(labels)) - (1 if -1 in labels else 0)
    print(f"Best DBSCAN (PCA): eps={best_dbscan_params['eps']:.2f},
min_samples={best_dbscan_params['min_samples']}")
    print(f"DBSCAN found {n_clusters} clusters. Homogeneity Score:
{best_dbscan_score:.4f}")

    plt.figure(figsize=(7, 5))
    plt.scatter(X_pca[:, 0], X_pca[:, 1], c=labels, cmap='viridis',
edgecolor='k')
    plt.xlabel("PCA 1")
    plt.ylabel("PCA 2")
    plt.title("DBSCAN Clustering (PCA)")
    plt.show()
else:
    print("No valid DBSCAN clustering found on PCA data.")


##############################################
# 7. Observations:
##############################################
print("\n##############################################")
print("# Observations:")
print("##############################################")
print("# 1. K-Means Clustering on Original Data:")
print("#    - The elbow plot shows that inertia decreases as the
number of clusters increases,")
print("#      while the silhouette plot peaks near k=2, suggesting
an optimal choice for two clusters.")
print("#    - The homogeneity score indicates moderate alignment
between the K-Means clusters and")
print("#      the true diagnosis labels in the original data.")
```

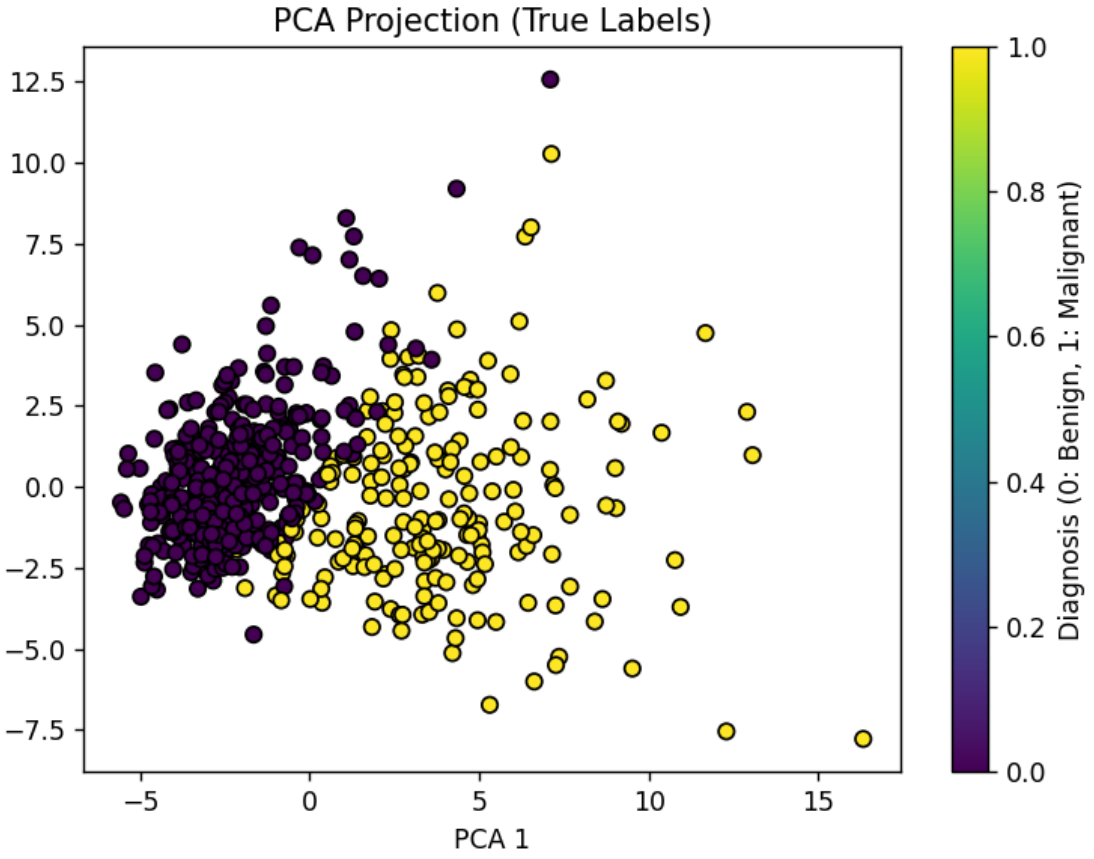
```
print("")
print("# 2. DBSCAN Clustering on Original Data:")
print("#    - The k-distance graph assists in choosing an
appropriate eps value; the knee of the graph")
print("#      serves as a visual cue for selecting eps.")
print("#    - DBSCAN labels noise points (typically as -1), and the
resulting homogeneity score")
print("#      reflects how purely the clusters correspond to the
true classes.")
print("")
print("# 3. PCA Transformation and Clustering:")
print("#    - PCA reduces the dimensionality and captures a
significant amount of the data variance,")
print("#      allowing clearer visualization of the cluster
structure with true labels.")
print("#    - K-Means on PCA data yields a higher or comparable
homogeneity score, with well-separated")
print("#      clusters visible in the 2D projection.")
print("#    - DBSCAN applied in the PCA space benefits from reduced
noise and dimensionality,")
print("#      resulting in more interpretable clusters based on the
tuning of eps and min_samples.")
print("")
print("# Overall, the comparison between clustering on the original
data and PCA-transformed data")
print("# demonstrates that PCA can enhance cluster separation and
improve the alignment of clusters")
print("# with the underlying diagnosis labels. Fine-tuning
parameters in both methods further")
print("# helps in achieving better clustering performance.")
print("################################################")
```

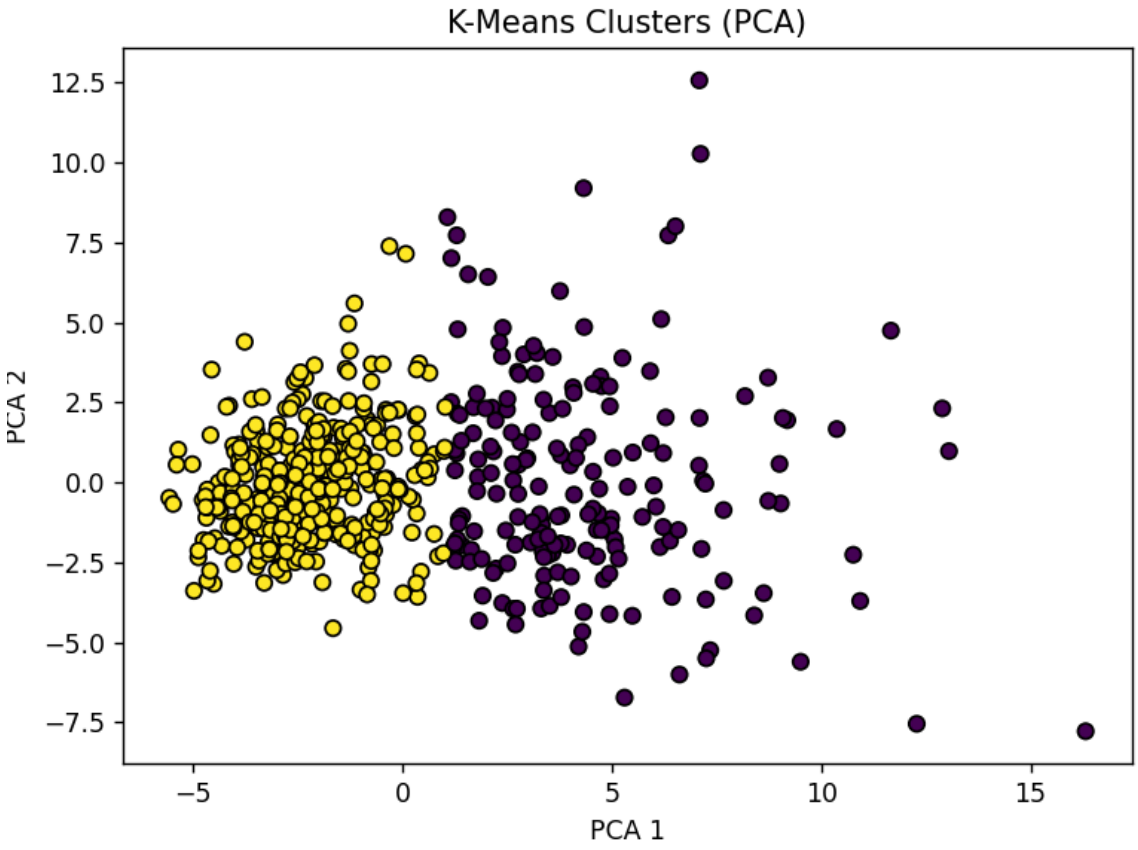
```
--- K-Means Clustering on Original Data ---
K-Means Homogeneity Score (Original Data): 0.5510

--- DBSCAN Clustering on Original Data ---
Best DBSCAN (Original): eps=2.00, min_samples=3
DBSCAN found 7 clusters. Homogeneity Score: 0.2753

Explained Variance Ratio (PCA): [0.44272026 0.18971182]
Total Variance Explained: 63.24%

--- K-Means Clustering on PCA Data ---
K-Means Homogeneity Score (PCA): 0.5313

--- DBSCAN Clustering on PCA Data ---
Best DBSCAN (PCA): eps=0.50, min_samples=2
DBSCAN found 41 clusters. Homogeneity Score: 0.6409
```

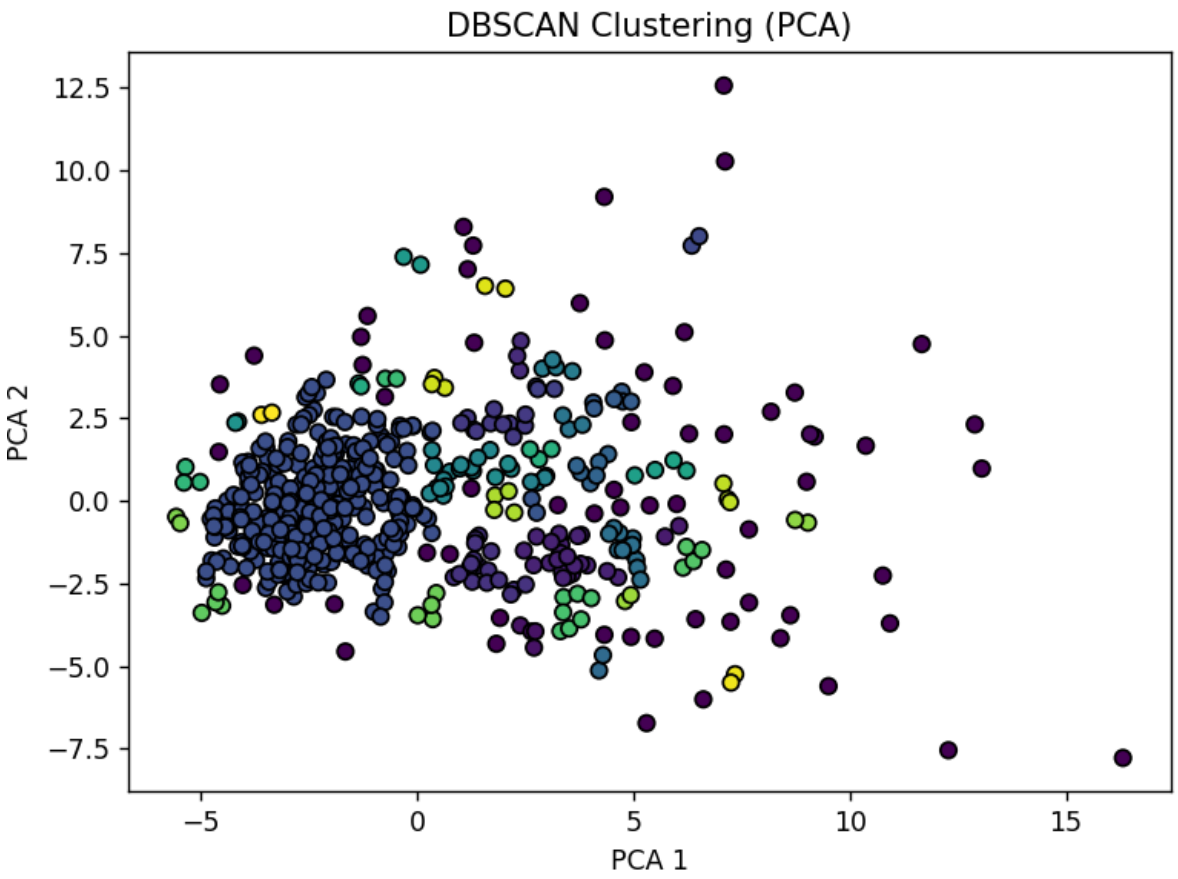





K-distance Graph (Original Data)
5-th NN Distance
Sorted Data Points
14
12
10
8
6
4
2
0
100
200
300
400
500


PCA Projection (True Labels)
PCA 1
Diagnosis (0: Benign, 1: Malignant)
12.5
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−5
0
5
10
15
1.0
0.8
0.6
0.4
0.2
0.0


K-Means Clusters (PCA)
PCA 2
PCA 1
12.5
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−5
0
5
10
15


DBSCAN Clustering (PCA)
PCA 2
PCA 1
12.5
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−5
0
5
10
15