2η Σειρά Ασκήσεων Μηχανική Μάθηση 03121089 Γιάννης Πολυχρονόπουλος

Άσκηση 2.1

a) • $Gini(Root) = 1 - \sum_{i \in labels(D)} p_i^2 = 1 - \left(\overset{\text{Κλάση 1}}{\frac{5}{9}}\right)^2 - \left(\overset{\text{Κλάση -1}}{\frac{4}{9}}\right)^2 = 1 - \frac{25}{81} - \frac{16}{81} = \frac{81-41}{81} = \frac{40}{81}$

$\Rightarrow Gini(Root) \simeq 0.4938$ (1)

• $Gini(X_1) = \sum_{v \in values(X_1)} \frac{|X_1 = v|}{|Root|} \cdot gini(X_1 = v) = \frac{|X_1 = 0|}{|Root|} \cdot gini(X_1 = 0) + \frac{|X_1 = 1|}{|Root|} \cdot gini(X_1 = 1)$

$\Rightarrow Gini(X_1) = \frac{4}{9} \cdot gini(X_1 = 0) + \frac{5}{9} gini(X_1 = 1)$ (2)

- $Gini(X_1 = 0) = 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 = 0{,}5$ (2.a)
- $Gini(X_1 = 1) = 1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2 = 0{,}48$ (2.b)

(2) $\xrightarrow[2.b]{2.a} \boxed{Gini(X_1) = 0.488}$ (2)

• $Gini(X_2) = \frac{4}{9} gini(X_2 = 0) + \frac{5}{9} gin(X_2 = 1)$ (3)

- $gini(X_2 = 0) = 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 = 0{,}5$ (3.a)
- $gini(X_2 = 1) = 1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2 = 0{,}48$ (3.b)

(3) $\xrightarrow[3.b]{3.a} \boxed{Gini(X_2) = 0.488}$ (3)

• $Gini(X_3) = \frac{5}{9} gin(X_3 = 0) + \frac{4}{9} gini(X_3 = 1)$ (4)

- $gini(X_3 = 0) = 1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2 = 0{,}48$ (4.a)
- $gini(X_3 = 1) = 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 = 0{,}5$ (4.b)

(4) $\xrightarrow[4.b]{4.a} Gini(X_3) = 0.491$

• $Gini(X_4) = \frac{3}{9} gini(X_4 = 0) + \frac{6}{9} gini(X_4 = 1)$ (5)

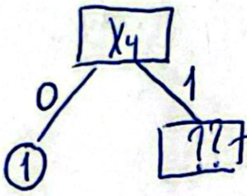
- $gini(X_4 = 0) = 1 - \left(\frac{3}{3}\right)^2 - \left(\frac{0}{3}\right)^2 = 0$ (5.a)
- $gini(X_4 = 1) = 1 - \left(\frac{2}{6}\right)^2 - \left(\frac{4}{6}\right)^2 = 0{,}444$ (5.b)

(5) $\Rightarrow \boxed{Gini(X_4) = 0.296}$ (5)

Άρα Information gain για X_1, X_2, X_3, X_4 είναι:

- $ig(X_1) = gini(Root) - gini(X_1) = 0.0058$
- $ig(X_2) = gini(Root) - gini(X_2) = 0.0058$
- $ig(X_3) = gini(Root) - gini(X_3) = 0.0498$
- $ig(X_4) = gini(Root) - gini(X_4) = \boxed{0.1972}$ Άρα με την αρχική διακλάδωση του X_4 αντλούμε την περισσότερη πληροφορία:

Δέντρο Απόφασης:



→ Υπολογίζουμε $gini(X_i)$ για $X_4 = 1$, $i = 1,2,3$

Για $X_4 = 1$:

- $Gini(Root) = 1 - \left(\frac{2}{6}\right)^2 - \left(\frac{4}{6}\right)^2 = 1 - \frac{1}{9} - \frac{4}{9} = \frac{9-5}{9} = \frac{4}{9} \Rightarrow Gini(Root) \simeq 0.444$ (1)'
- $Gini(X_1) = \frac{|X_1=0|}{|Root|} \cdot gini(X_1=0) + \frac{|X_1=1|}{|Root|} gini(X_1=1) = \frac{3}{6} gini(X_1=0) + \frac{3}{6} gini(X_1=1)$ (2)'
  - $gini(X_1=0) = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 \simeq 0.444$ (2.α)'
  - $gini(X_1=1) = 1 - \left(\frac{1}{3}\right) - \left(\frac{2}{3}\right)^2 \simeq 0.444$ (2.β)'

(2)' $\xrightarrow[(2.β)']{(2.α)'}$ $Gini(X_1) = 0.444 = Gini(Root)$ άρα $\boxed{ig(X_1) = 0}$

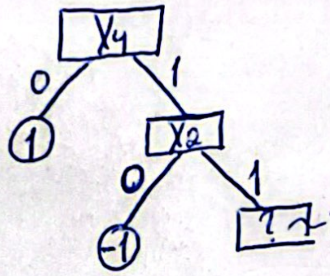
- $Gini(X_2) = \frac{2}{6} gini(X_2=0) + \frac{4}{6} gini(X_2=1)$ (3)'
  - $gini(X_2=0) = 1 - \left(\frac{2}{2}\right)^2 + \left(\frac{0}{2}\right)^2 = 0$ 3.α'
  - $gini(X_2=1) = 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 = 0{,}5$ 3β'

(3)' $\xrightarrow[3.β']{3.α'}$ $gini(X_2) = \frac{1}{3} \cdot 0 + \frac{2}{3} \cdot 0.5 = \frac{1}{3} = 0{,}3333$ Άρα $ig(X_2) = 0.444 - 0.333$
$\Rightarrow \boxed{ig(X_2) = 0.111}$

- $gini(X_3) = \frac{3}{6} gini(X_3=0) + \frac{3}{6} gini(X_3=1)$ (4)'
  - $gini(X_3=0) = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = 0.444$
  - $gini(X_3=1) = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = 0.444$

$\}$ (4)' $\Rightarrow gini(X_3) = 0.444 = gini(Root)$
Άρα $\boxed{ig(X_3) = 0}$

Άρα δέντρο απόφασης:



→ Gini indexes για $X_4=1$ και $X_2=1$

$$gini(Root) = 1-\left(\frac{2}{4}\right)^2+\left(\frac{2}{4}\right)^2 = 0,5$$

- $gini(X_1) = \frac{2}{4} gini(X_1=0) + \frac{2}{4} gini(X_1=1)$

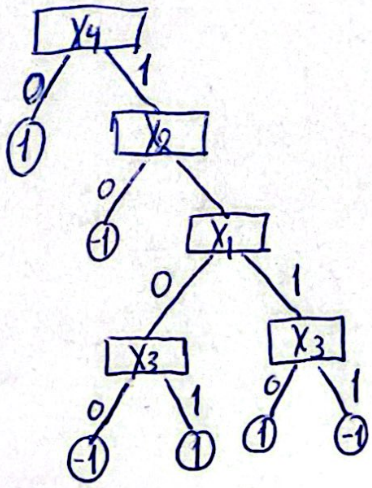
$$\left.\begin{array}{l} gini(X_1=0) = 1-\left(\frac{1}{2}\right)^2-\left(\frac{1}{2}\right)^2 = 0,5 \\ gin(X_1=1) = 1-\left(\frac{1}{2}\right)^2-\left(\frac{1}{2}\right)^2 = 0,5 \end{array}\right\} gini(X_1) = 0,5 \Rightarrow ig(X_1) = 0$$

- $gini(X_3) = \frac{2}{4} gini(X_3=0) + \frac{2}{4} gini(X_3=1)$

$$\left.\begin{array}{l} gini(X_3=0) = 1-\left(\frac{1}{2}\right)^2-\left(\frac{1}{2}\right)^2 = 0,5 \\ gini(X_3=1) = 1-\left(\frac{1}{2}\right)^2-\left(\frac{1}{2}\right)^2 = 0,5 \end{array}\right\} gini(X_3) = 0,5 \Rightarrow ig(X_3) = 0$$

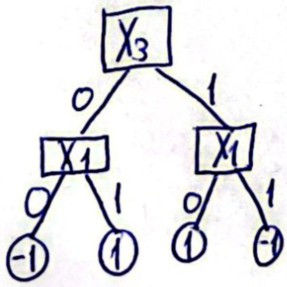
Διαλέγουμε το X_1 (αφού πρώτο αλφαβητικά)

Τελικό δέντρο Απόφασης



β) Παρατηρούμε πως όταν $X_3 (xor) X_1 = 0$ τότε $Y=-1$ ενώ όταν $X_1 (xor) X_3 = 1, Y=$

Άρα ξεκινώντας από το X_3:



γ) Για το πρώτο υποσύνολο:

Γενικά παρατηρήσαμε πως όσο μικρότερο το $gini(X_i)$ τόσο μεγαλύτερο το $ig(X_i)$

Άρα • $gini(X_1) = \frac{2}{3} gini(X_1=0) + \frac{1}{3} gini(X_1=1)$

με $gini(X_1=0) = 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2 = 0,5$

$gini(X_1=1) = 1 - \left(\frac{1}{1}\right)^2 - \left(\frac{0}{1}\right)^2 = 0$

$\Big\} \; gini(X_1) = \frac{2}{3} \cdot \frac{1}{2} = \frac{1}{3} = 0,333 \Rightarrow \boxed{gini(X_1) = 0.333}$

• $gini(X_2) = gini(X_2=1) = 1 - \left(\frac{2}{3}\right)^2 - \left(\frac{1}{3}\right)^2 = 1 - \frac{4}{9} - \frac{1}{9} = \frac{9-5}{9} = \frac{4}{9} = \boxed{0.444 = gini(X_2)}$

• $gini(X_3) = \frac{2}{3} gini(X_3=0) + \frac{1}{3} gini(X_3=1)$

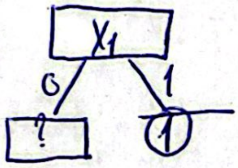
με $gini(X_3=0) = 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2 = 0,5$

$gini(X_3=1) = 1 - \left(\frac{1}{1}\right)^2 - \left(\frac{0}{1}\right)^2 = 0$

$\Big\} \; \boxed{gini(X_3) = 0.333}$

• $gini(X_4) = \frac{1}{3} \cancel{gini(X_4=0)}^{\to 0} + \frac{2}{3} gini(X_4=1) = 0.333$ (ίδια λογική με τα προηγούμενα)

Επιλέγω αλφαβητικά το X_1:



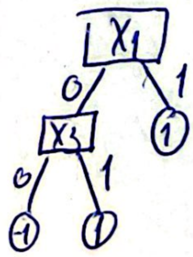
Για $X_1 = 0$: (Κατευθείαν το X_3 παρατηρούμε πως διαχωρίζει τις κλάσεις)

$gini(X_2) = gini(X_2=1) = 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2 = 0,5$

$gini(X_3) = \frac{1}{2} \cancel{gini(X_3=0)}^{\to 0} + \frac{1}{2} \cancel{gini(X_3=1)}^{\to 0} = 0$

$gini(X_4) = gini(X_4=1) = \frac{1}{1} - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2 = 0,5$

Άρα για το πρώτο υποσύνολο



Δευτερο υποσύνολο:

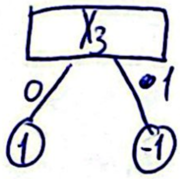
$gini(X_1) = gini(X_1=1) = 1-\left(\frac{1}{3}\right)^2-\left(\frac{2}{3}\right)^2 = 0.444$

$gini(X_2) = \frac{1}{3}\cancel{gini(X_2=0)}^{\to 0} + \frac{2}{3}gini(X_2=1) = \frac{2}{3}\left(1-\left(\frac{1}{2}\right)^2-\left(\frac{1}{2}\right)^2\right) = 0.333$

$gini(X_3) = \frac{2}{3}gini(X_3=0) + \frac{1}{3}\cancel{gini(X_3=1)}^{\to 0} = \frac{2}{3}\left(1-\left(\frac{2}{2}\right)^2-\left(\frac{0}{2}\right)^2\right) = 0$

$gini(X_4) = 0.333$ (ίδια λογική με $gini(X_2)$)

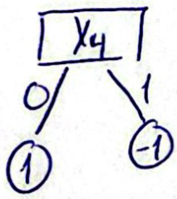
Άρα για δεύτερο υποσύνολο:



Τρίτο Υποσύνολο:

$gini(X_4) = 0$ (Αφού για $X_4=0 \to Y=1$ και $X_4=1 \to Y=-1$)

Όλα οι άλλα δείκτες θα είναι μεγαλύτεροι του μηδενός, άρα



Άσκηση 2.2

a) $X = \{(x_1,y_1),\dots,(x_8,y_8)\}$ x_i, y_i αντιστοιχούν με τα νούμερα στην εκφώνηση (Υπολο

Υπολογίζουμε απόσταση (x_i,y_i) με τα δύο κέντρα και αναθέτουμε αντίστοιχ

το σημείο στην ομάδα με την μικρότερη απόσταση. Κέντρα: $C_1=(3,3)$, $C_2=(4,4)$

Δηλαδή υπολογίζουμε όλες τις: $d((x_i,y_i), C_j) = \sqrt{(x_i-C_{j,x})^2+(y_i-C_{j,y})^2}$

Έφτιαξα python script για αυτό.

Άσκηση 2.2

Ερώτημα Α

```
PS G:\My Drive\Ece Ntua\7th Semester\Machine Learning\Series of Exercises\Exercise 2> python3 .\Exercise_2a.py

=== Iteration 0 ===
Point: (2, 3), Distances: [1.0, 2.23606797749979], Assigned to Cluster: 1
Point: (3, 2), Distances: [1.0, 2.23606797749979], Assigned to Cluster: 1
Point: (1, 2), Distances: [2.23606797749979, 3.605551275463989], Assigned to Cluster: 1
Point: (4, 5), Distances: [2.23606797749979, 1.0], Assigned to Cluster: 2
Point: (5, 4), Distances: [2.23606797749979, 1.0], Assigned to Cluster: 2
Point: (3, 4), Distances: [1.0, 1.0], Assigned to Cluster: 1
Point: (6, 4), Distances: [3.1622776601683795, 2.0], Assigned to Cluster: 2
Point: (6, 5), Distances: [3.605551275463989, 2.23606797749979], Assigned to Cluster: 2
New Centroids: [[2.25 2.75]
 [5.25 4.5 ]]

=== Iteration 1 ===
Point: (2, 3), Distances: [0.3535533905932738, 3.5794552658190883], Assigned to Cluster: 1
Point: (3, 2), Distances: [1.0606601717798212, 3.3634060117684275], Assigned to Cluster: 1
Point: (1, 2), Distances: [1.4577379737113252, 4.930770730829005], Assigned to Cluster: 1
Point: (4, 5), Distances: [2.850438562747845, 1.346291201783626], Assigned to Cluster: 2
Point: (5, 4), Distances: [3.020761493398643, 0.5590169943749475], Assigned to Cluster: 2
Point: (3, 4), Distances: [1.4577379737113252, 2.3048861143232218], Assigned to Cluster: 1
Point: (6, 4), Distances: [3.952847075210474, 0.9013878188659973], Assigned to Cluster: 2
Point: (6, 5), Distances: [4.373213921133975, 0.9013878188659973], Assigned to Cluster: 2
New Centroids: [[2.25 2.75]
 [5.25 4.5 ]]

Convergence reached.

=== Final Output ===
Final Centroids:
  Cluster 1: [2.25 2.75]
  Cluster 2: [5.25 4.5 ]

Final Cluster Assignments:
  Point 1: Assigned to Cluster 1
  Point 2: Assigned to Cluster 1
  Point 3: Assigned to Cluster 1
  Point 4: Assigned to Cluster 2
  Point 5: Assigned to Cluster 2
  Point 6: Assigned to Cluster 1
  Point 7: Assigned to Cluster 2
  Point 8: Assigned to Cluster 2
PS G:\My Drive\Ece Ntua\7th Semester\Machine Learning\Series of Exercises\Exercise 2>
```

Βοηθητικός κώδικας που χρησιμοποιήθηκε:

```python
import numpy as np
# Data
points = np.array([[2, 3], [3, 2], [1, 2], [4, 5],
                   [5, 4], [3, 4], [6, 4], [6, 5]])

# Initialization of centroids
centroids = np.array([[3, 3], [4, 4]])

def euclidean_distance(point, centroid):
      """Calculation of the Euclidean distance between a point and a
centroid"""
    return np.sqrt(np.sum((point - centroid) ** 2))
```

```python
def assign_points_to_clusters(points, centroids):
    """Assign each point to the nearest centroid"""
    clusters = []
    for point in points:
        distances = [euclidean_distance(point, centroid) for centroid in centroids]
        cluster = np.argmin(distances)  # Returns the index of the nearest centroid
        clusters.append(cluster)
    return np.array(clusters)

def update_centroids(points, clusters, k):
    """Calculation of new centroids as the mean of the points in each cluster"""
    new_centroids = []
    for i in range(k):
        cluster_points = points[clusters == i]
        if len(cluster_points) > 0:  # If there are points in the cluster
            new_centroid = np.mean(cluster_points, axis=0)
        else:  # If there are no points, retain the old centroid
            new_centroid = centroids[i]
        new_centroids.append(new_centroid)
    return np.array(new_centroids)
# Execution of the algorithm
iteration = 0
k = len(centroids)
while True:
    print(f"\n=== Iteration {iteration} ===")
    # Calculation of distances and assignment
    clusters = assign_points_to_clusters(points, centroids)
    for i, point in enumerate(points):
        distances = [euclidean_distance(point, centroid) for centroid in centroids]
        print(f"Point: ({point[0]}, {point[1]}), Distances: {distances}, Assigned to Cluster: {clusters[i] + 1}")
    # Calculation of new centroids
    new_centroids = update_centroids(points, clusters, k)
    print(f"New Centroids: {new_centroids}")
    # Convergence check
    if np.allclose(centroids, new_centroids):
        print("\nConvergence reached.")
        break
    centroids = new_centroids
    iteration += 1
# Final output
print("\n=== Final Output ===")
print("Final Centroids:")
for i, centroid in enumerate(centroids, 1):
    print(f"  Cluster {i}: {centroid}")
print("\nFinal Cluster Assignments:")
for i, cluster in enumerate(clusters, 1):
    print(f"  Point {i}: Assigned to Cluster {cluster + 1}")
```

Ερώτημα Β, Γ

Ακολουθεί τα αποτελέσματα του αλγορίθμου k-means με όλα τα χαρακτηριστικά και με μόνο δύο, καθώς και ο αντίστοιχος βοηθητικός κώδικας που χρησιμοποιήθηκε.

```
PS G:\My Drive\Ece Ntua\7th Semester\Machine Learning\Series of Exercises\Exercise 2> python .\Exercise_2b_c_d.py

Convergence reached!

=== (b) Full Feature Set ===
Confusion Matrix:
[[50  0  0]
 [ 0 48  2]
 [ 0 14 36]]

Accuracy (Full Features): 89.33%

Convergence reached!

=== (c) Two Features ===
Confusion Matrix:
[[50  0  0]
 [ 0 48  2]
 [ 0  6 44]]
Accuracy: 94.67%

=== (d) Comparison ===
Accuracy with Full Features: 89.33%
Accuracy with Two Features: 94.67%
```

```python
import pandas as pd
import numpy as np
from sklearn.metrics import confusion_matrix, accuracy_score
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
from collections import Counter

# Load the dataset
data = pd.read_csv('iris.csv')
data = data.drop(columns=['Id'])

# Encode Species to numerical values for comparison
label = LabelEncoder()
data['SpeciesEncoded'] = label.fit_transform(data['Species'])

# K-means algorithm implementation
def k_means(X, k, epsilon=1e-5, max_iterations=100):
    # Random initialization of centroids
    np.random.seed(42)
    centroids = X[np.random.choice(X.shape[0], k, replace=False)]

    for iteration in range(max_iterations):
        # Assign points to the nearest centroid
        distances = np.linalg.norm(X[:, np.newaxis] - centroids, axis=2)
```

```
        clusters = np.argmin(distances, axis=1)

        # Update centroids
        new_centroids = np.array(
            [X[clusters == i].mean(axis=0) for i in range(k)])

        # Convergence criterion
        if np.linalg.norm(new_centroids - centroids) < epsilon:
            print("\nConvergence reached!")
            break

        centroids = new_centroids

    return clusters, centroids

# Function to remap clusters to match true labels
def remap_clusters(true_labels, clusters):
    # Create a mapping from clusters to true labels based on majority vote
    mapping = {}
    for cluster_id in np.unique(clusters):
        # Find true labels corresponding to points in this cluster
        true_labels_in_cluster = true_labels[clusters == cluster_id]

        # Get the most common true label for this cluster
        most_common_label = Counter(
            true_labels_in_cluster).most_common(1)[0][0]
        mapping[cluster_id] = most_common_label

    # Remap the clusters
    remapped_clusters = np.array([mapping[cluster] for cluster in clusters])
    return remapped_clusters

# (b) Apply k-means with all features
X_full = data[['SepalLengthCm', 'SepalWidthCm',
               'PetalLengthCm', 'PetalWidthCm']].values
clusters_full, centroids_full = k_means(X_full, k=3)

# Map cluster labels to the true labels for comparison
true_labels = data['SpeciesEncoded'].values
clusters_full = remap_clusters(true_labels, clusters_full)
conf_matrix_full = confusion_matrix(true_labels, clusters_full)
accuracy_full = accuracy_score(true_labels, clusters_full)
```

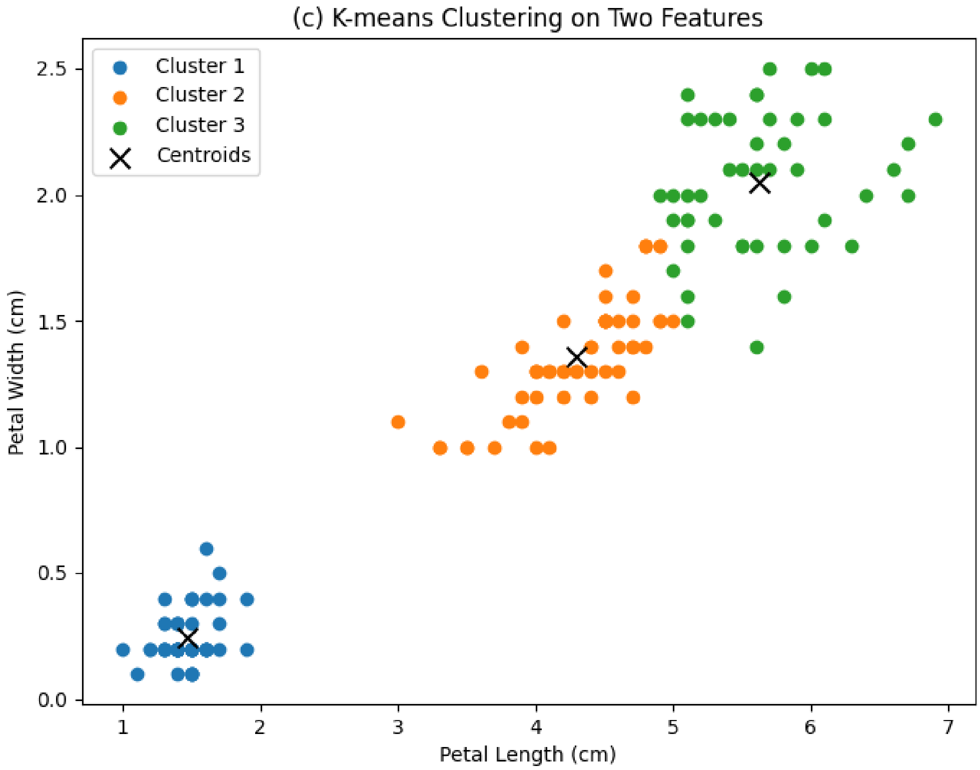
```python
print("\n=== (b) Full Feature Set ===")
print("Confusion Matrix:")
print(conf_matrix_full)
print(f"\nAccuracy (Full Features): {accuracy_full * 100:.2f}%")

# (c) Apply k-means with only two features
X_two = data[['PetalLengthCm', 'PetalWidthCm']].values
clusters_two, centroids_two = k_means(X_two, k=3)
clusters_two = remap_clusters(true_labels, clusters_two)
conf_matrix_two = confusion_matrix(true_labels, clusters_two)
accuracy_two = accuracy_score(true_labels, clusters_two)

print("\n=== (c) Two Features ===")
print("Confusion Matrix:")
print(conf_matrix_two)
print(f"Accuracy: {accuracy_two * 100:.2f}%")
# (d) Compare results
print("\n=== (d) Comparison ===")
print(f"Accuracy with Full Features: {accuracy_full * 100:.2f}%")
print(f"Accuracy with Two Features: {accuracy_two * 100:.2f}%")

# Plotting for (c)
plt.figure(figsize=(8, 6))
for i in range(3):
    cluster_points = X_two[clusters_two == i]
    plt.scatter(cluster_points[:, 0],
                cluster_points[:, 1], label=f'Cluster {i+1}')
plt.scatter(centroids_two[:, 0], centroids_two[:, 1],
            color='black', marker='x', s=100, label='Centroids')
plt.xlabel('Petal Length (cm)')
plt.ylabel('Petal Width (cm)')
plt.title('(c) K-means Clustering on Two Features')
plt.legend()
plt.show()
```

Επίσης, ακολουθεί γραφική απεικόνιση του ζητούμενου.



Ερώτημα Δ

Χρησιμοποιώντας και τα 4 χαρακτηριστικά, το success rate ήταν 89.33%, ενώ με μόνο το μήκος και το πλάτος των πετάλων αυξήθηκε στο 94.67%. Αυτό δείχνει ότι τα χαρακτηριστικά των πετάλων είναι πιο διακριτικά για τις κλάσεις, ενώ η προσθήκη των χαρακτηριστικών των σεπάλων εισάγει θόρυβο και μειώνει την ακρίβεια.

Συνεπώς, η χρήση όλων των χαρακτηριστικών δεν οδηγεί πάντα σε καλύτερα αποτελέσματα, καθώς μπορεί να περιλαμβάνει περιττή ή παραπλανητική πληροφορία που επηρεάζει αρνητικά την ταξινόμηση.

Άσκηση 2.3

α) Για μέτρο ανομοιότητας πρέπει: $\boxed{d: X \times X \to \mathbb{R}}$

- $\exists d_0 \in \mathbb{R}: -\infty < d_0 \leq d(x,y) < \infty, \ \forall x,y \in X$
- $d(x,x) = d_0, \ \forall x \in X$
- $d(x,y) = d(y,x), \ \forall x,y \in X$

Έχουμε $d_c(x,y) = 1 - \cos\theta_{xy}$ με $\cos\theta_{xy} = \dfrac{x \cdot y}{\|x\| \cdot \|y\|}$ (Από γραμμική Άλγεβρα)

- $-1 \leq \cos\theta_{xy} \leq 1 \Rightarrow -1 \leq -\cos\theta_{xy} \leq 1 \Rightarrow \boxed{0 \leq 1 - \cos\theta_{xy}}$
- $d_c(x,x) = 1 - \cos\theta_{xx} = 1 - \dfrac{x \cdot x}{\|x\| \cdot \|x\|} = 1 - \dfrac{\|x\|^2}{\|x\| \cdot \|x\|} = 0 \Rightarrow \boxed{d_c(x,x) = 0}$
- $d_c(x,y) = 1 - \cos\theta_{xy} = 1 - \dfrac{x \cdot y}{\|x\| \cdot \|y\|} \underset{\text{εσ. γινόμενου}}{\overset{\text{ιδιότητα}}{=}} 1 - \dfrac{y \cdot x}{\|y\| \cdot \|x\|} = 1 - \cos\theta_{yx} = d_c(y,x)$

$\Rightarrow \boxed{d_c(x,y) = d_c(y,x)}$

Από τα παραπάνω η $d_c(x,y)$ είναι μέτρο ανομοιότητας.

Για να είναι μετρική ανισότητα, πρέπει: $d_c(x,y) = 0$ αν και μόνο αν $x = y$

όμως αν $y = 2x$ τότε

$d_c(x,2x) = 1 - \cos\theta_{x,2x} = 1 - \dfrac{x \cdot 2x}{\|x\| \cdot \|2x\|} = 1 - \dfrac{2(x \cdot x)}{2 \cdot \|x\| \cdot \|x\|} = 0$ άρα δεν είναι μετρική ανομοιότητας.

Άσκηση 2.3

Ερώτημα Β

- Ο πίνακας προτύπων (pattern matrix) $D(X)$ είναι ο $N \times l$ η i-οστή γραμμή του οποίου είναι το (αντεστραμμένο) i-οστό διάνυσμα του X.
- Ο πίνακας ομοιότητας (ανομοιότητας), $P(X)$, είναι ένας $N \times N$, του οποίου το στοιχείο (i, j) ισούται με το βαθμό ομοιότητας $s(\boldsymbol{x}_i, \boldsymbol{x}_j)$ (ανομοιότητας $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$) των διανυσμάτων $\boldsymbol{x}_i$ και $\boldsymbol{x}_j$. Ο πίνακας αυτός είναι γνωστός και ως πίνακας εγγύτητας.

```
PS G:\My Drive\Ece Ntua\7th Semester\Machine Learning\Series of Exercises\Exercise 2> python3 .\Exercise_3b_c_d.py

D(X) - Distance Matrix:
[[1 5]
 [3 4]
 [0 2]
 [5 4]
 [2 6]
 [3 3]
 [2 3]
 [4 2]]

P(X) - Proximity Matrix:
        Point 1  Point 2  Point 3  Point 4  Point 5  Point 6  Point 7  Point 8
Point 1   0.000    0.098    0.019    0.234    0.008    0.168    0.075    0.386
Point 2   0.098    0.000    0.200    0.032    0.051    0.010    0.002    0.106
Point 3   0.019    0.200    0.000    0.375    0.051    0.293    0.168    0.553
Point 4   0.234    0.032    0.375    0.000    0.160    0.006    0.047    0.022
Point 5   0.008    0.051    0.051    0.160    0.000    0.106    0.035    0.293
Point 6   0.168    0.010    0.293    0.006    0.106    0.000    0.019    0.051
Point 7   0.075    0.002    0.168    0.047    0.035    0.019    0.000    0.132
Point 8   0.386    0.106    0.553    0.022    0.293    0.051    0.132    0.000
```

Βοηθητικός κώδικας:

```python
import numpy as np

import pandas as pd
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import linkage, dendrogram
from scipy.spatial.distance import pdist, squareform

# Data
X = np.array([[1,5],[3,4],[0, 2], [5, 4], [2, 6], [3, 3], [2, 3], [4, 2]])

# D(X)
D = X

# Function to compute the proximity matrix using dc(x,y)=1 - cos(theta_xy)
def proximity_matrix(X):
    n = X.shape[0]
    P = np.zeros((n, n))  # Initialize proximity matrix
    for i in range(n):
        for j in range(n):
            if i != j:
                # Compute cosine similarity and transform to proximity
```

```python
                 cos_theta = np.dot(X[i], X[j]) / (np.linalg.norm(X[i]) *
np.linalg.norm(X[j]))
                P[i, j] = 1 - cos_theta
    return P

# P(X)
P = proximity_matrix(X)

# Convert matrices to pandas DataFrames for better formatting

P_df = pd.DataFrame(P, index=[f"Point {i+1}" for i in range(X.shape[0])],
                    columns=[f"Point {i+1}" for i in range(X.shape[0])])

# Print the matrices in a nicely formatted way
print("\nD(X) - Distance Matrix:")
print(D)

print("\nP(X) - Proximity Matrix:")
print(P_df.round(3))  # Round to 3 decimal places for better readability
```

Ερώτημα Γ

Αρχικό Στάδιο

Ξεκινάμε με 8 μονομελή συμπλέγματα:

{1},{2},{3},{4},{5},{6},{7},{8}.

Αρχικά, εντοπίζουμε την ελάχιστη τιμή εγγύτητας στον πίνακα $P(xi, xj)$. Η ελάχιστη εγγύτητα είναι $P(2,7) = \ P(7,2) = 0.002$. Συνεπώς, τα σημεία 2 και 7 ενώνονται σε ένα νέο σύμπλεγμα:

$$\{2,7\}$$

Αναζητούμε ξανά τη μέγιστη τιμή εγγύτητας. Εντοπίζουμε ότι η ελάχιστη τιμή είναι $P(4,6) = 0.006$. Συνεπώς, τα σημεία 4 και 6 ενώνονται σε ένα νέο σύμπλεγμα:

$$\{4,6\}$$

Η επόμενη ελάχιστη τιμή στον πίνακα εγγύτητας είναι $P(1,5) = 0.008$.. Συνεπώς, τα σημεία 1 και 5 ενώνονται:

$$\{1,5\}$$

Τώρα έχουμε τρία διμελή συμπλέγματα: {2,7},{4,6},{1,5} και δύο μονομελή: {3},{8}. Ελέγχουμε τις νέες εγγύτητες και βρίσκουμε την ελάχιστη:

$$P(\{2,7\},\{4,6\}) = \min P(x_2, x_4), P(x_2, x_6), P(x_7, x_4), P(x_7, x_6) = 0.010$$

Τα συμπλέγματα {2,7} και {4,6} ενώνονται σε ένα νέο σύμπλεγμα:

$$\{2,4,6,7\}$$

Τα συμπλέγματα είναι πλέον: {1,5},{2,4,6,7},{3},{8}. Η ελάχιστη εγγύτητα τώρα βρίσκεται μεταξύ {1,5} και {3}:

$$P(\{1,5\},\{3\}) = \min P(x_1, x_3), P(x_5, x_3) = 0.019$$

Συνεπώς, ενώνονται:

$$\{1,3,5\}$$

Τα συμπλέγματα είναι τώρα: {1,3,5},{2,4,6,7},{8}. Εντοπίζουμε τη ελάχιστη εγγύτητα μεταξύ {2,4,6,7}και {8}:

$$P(\{2,4,6,7\},\{8\}) = \min P(x_2, x_8), P(x_4, x_8), P(x_6, x_8), P(x_7, x_8) = 0.022$$

Συνεπώς, ενώνονται:

$$\{2,4,6,7,8\}$$

Απομένουν δύο συμπλέγματα: {1,3,5} και {2,4,6,7,8}. Η ελάχιστη εγγύτητα είναι:

$$P(\{2,4,6,7,8\},\{1,3,5\})$$
$$= \min P(x_2, x_1), P(x_2, x_3), P(x_2, x_5), P(x_4, x_1), P(x_4, x_3), P(x_4, x_5), P(x_6, x_1), P(x_6, x_3), P(x_6, x_5) \ldots,$$
$$= 0.035$$

Όλα τα σημεία ενώνονται σε ένα τελικό σύμπλεγμα:

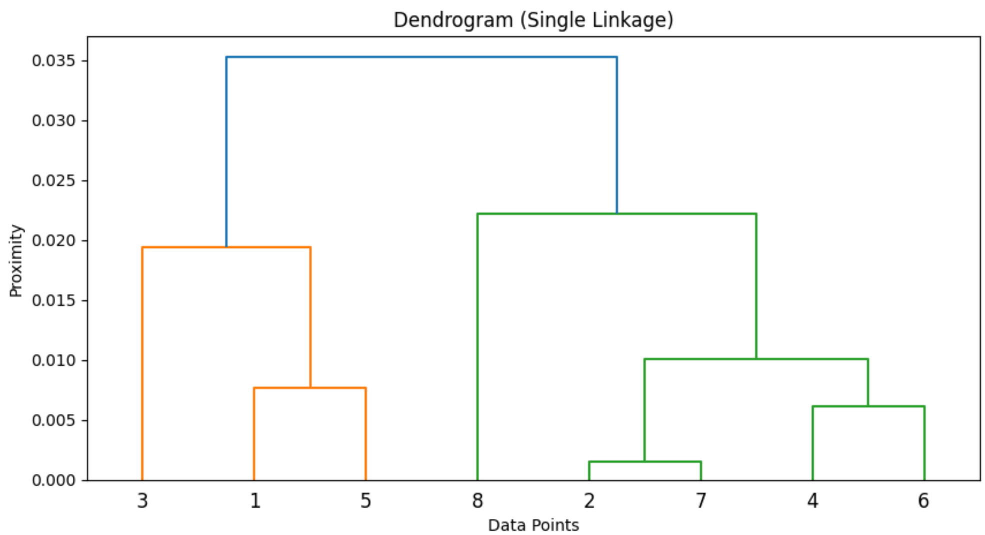
$$\{1,2,3,4,5,6,7,8\}$$

Παρακάτω δίνεται ο βοηθητικός κώδικας (έτοιμη βιβλιοθήκη χρησιμοποιήθηκε) και το αντίστοιχο δενδρόγραμμα.

```
# Convert the proximity matrix to a condensed form
P_condensed = squareform(P)

# Hierarchical Clustering with Single Linkage
linkage_matrix_single = linkage(P_condensed, method='single')
```

```python
# Plot dendrogram for Single Linkage
plt.figure(figsize=(10, 5))
dendrogram(linkage_matrix_single, labels=np.arange(1, X.shape[0] + 1))
plt.title('Dendrogram (Single Linkage)')
plt.xlabel('Data Points')
plt.ylabel('Proximity')
plt.show()
```



Ερώτημα Δ

Αρχικό Στάδιο

Ξεκινάμε με 8 μονομελή συμπλέγματα:

{1},{2},{3},{4},{5},{6},{7},{8}.

Αρχικά, εντοπίζουμε την ελάχιστη τιμή εγγύτητας στον πίνακα $P(xi, xj)$. Η ελάχιστη εγγύτητα είναι $P(2,7) = P(7,2) = 0.002$. Συνεπώς, τα σημεία 2 και 7 ενώνονται σε ένα νέο σύμπλεγμα:

{2,7}

Αναζητούμε ξανά τη μέγιστη τιμή εγγύτητας. Εντοπίζουμε ότι η ελάχιστη τιμή είναι $P(4, 6) = 0.006$. Συνεπώς, τα σημεία 4 και 6 ενώνονται σε ένα νέο σύμπλεγμα:

{4,6}

Η επόμενη ελάχιστη τιμή στον πίνακα εγγύτητας είναι $P(1,5) = 0.008$. Συνεπώς, τα σημεία 1 και 5 ενώνονται:

$$\{1,5\}$$

Τώρα έχουμε τρία διμελή συμπλέγματα: $\{2,7\},\{4,6\},\{1,5\}$ και δύο μονομελή: $\{3\},\{8\}$. Ελέγχουμε τις νέες εγγύτητες και βρίσκουμε την μέγιστη:

$$P(\{2,7\},\{4,6\}) = \max P(x_2,x_4), P(x_2,x_6), P(x_7,x_4), P(x_7,x_6) = 0.047$$

Τα συμπλέγματα {2,7} και {4,6} ενώνονται σε ένα νέο σύμπλεγμα:

$$\{2,4,6,7\}$$

Τα συμπλέγματα είναι πλέον: {1,5},{2,4,6,7},{3},{8}. Η μέγιστη εγγύτητα τώρα μεταξύ {1,5} και {3}:

$$P(\{1,5\},\{3\}) = \max P(x_1,x_3), P(x_5,x_3) = 0.051$$

Συνεπώς, ενώνονται:

$$\{1,3,5\}$$

Τα συμπλέγματα είναι τώρα: {1,3,5},{2,4,6,7},{8}. Εντοπίζουμε τη μέγιστη εγγύτητα μεταξύ {2,4,6,7}και {8}:

$$P(\{2,4,6,7\},\{8\}) = max\, P(x_2,x_8), P(x_4,x_8), P(x_6,x_8), P(x_7,x_8) = 0.132$$

Συνεπώς, ενώνονται:

$$\{2,4,6,7,8\}$$

Απομένουν δύο συμπλέγματα: {1,3,5} και {2,4,6,7,8}. Η ελάχιστη εγγύτητα είναι:

$$\begin{aligned} &P(\{2,4,6,7,8\},\{1,3,5\}) \\ &= \max P(x_2,x_1), P(x_2,x_3), P(x_2,x_5), P(x_4,x_1), P(x_4,x_3), P(x_4,x_5)\,, P(x_6,x_1), P(x_6,x_3), P(x_6,x_5) \ldots, \\ &= 0.553 \end{aligned}$$

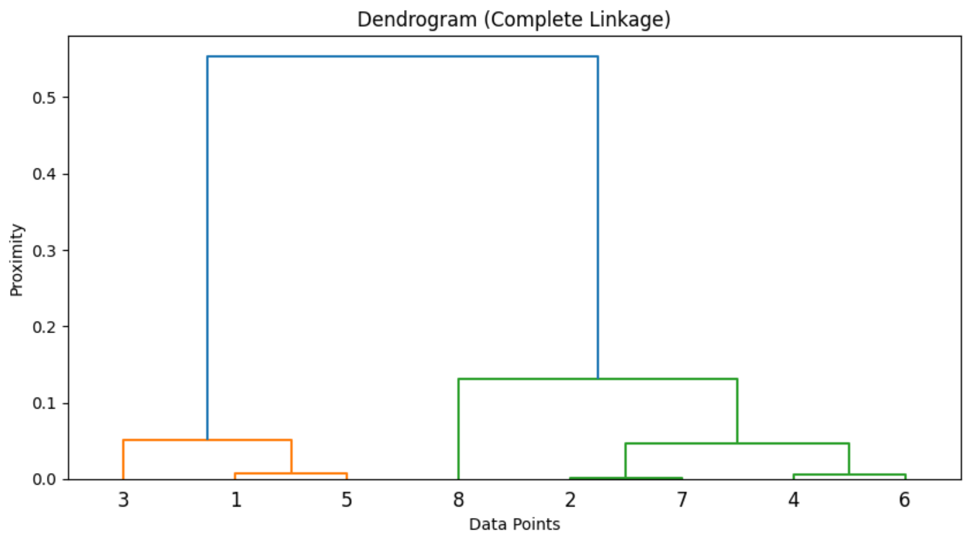
Όλα τα σημεία ενώνονται σε ένα τελικό σύμπλεγμα:

$$\{1,2,3,4,5,6,7,8\}$$

Παρακάτω δίνεται ο βοηθητικός κώδικας (έτοιμη βιβλιοθήκη χρησιμοποιήθηκε) και το αντίστοιχο δενδρόγραμμα.

```
# Hierarchical Clustering with Complete Linkage
linkage_matrix_complete = linkage(P_condensed, method='complete')

# Plot dendrogram for Complete Linkage
plt.figure(figsize=(10, 5))
dendrogram(linkage_matrix_complete, labels=np.arange(1, X.shape[0] + 1))
plt.title('Dendrogram (Complete Linkage)')
plt.xlabel('Data Points')
plt.ylabel('Proximity')
plt.show()
```



Ερώτημα Ε

Παρατηρούμε ότι τα δύο δενδρογράμματα που προκύπτουν από τις μεθόδους απλού και πλήρους δεσμού είναι πανομοιότυπα. Και στις δύο περιπτώσεις, οι ομαδοποιήσεις που προκύπτουν είναι οι εξής: ένα cluster για τα σημεία {3,1,5} και ένα για τα σημεία {2,7,4,6,8}. Το γεγονός ότι οι δύο μέθοδοι οδηγούν στο ίδιο αποτέλεσμα μπορεί να αποδοθεί στη φύση των δεδομένων. Πιο συγκεκριμένα, οι αποστάσεις μεταξύ των σημείων πρέπει να είναι τέτοιες ώστε η ελάχιστη και η μέγιστη απόσταση εντός των ομάδων να είναι παρόμοιες, ώστε τόσο ο απλός όσο και ο πλήρης δεσμός να καταλήγουν σε παρόμοια ή ταυτόσημα αποτελέσματα.

Άσκηση 2.4 (Πηγή: ekamperi.github.io/mathematics/2020/11/01/principal-component-analysis-lagrange-multiplier.html

α) Για m=1: Έχουμε $\|\vec{u}_1\| = 1$, και $\vec{u}_1^T \cdot \vec{u}_1 = 1$

Η προβολή του $\vec{x}_n$ διανύσματος στο $\vec{u}_1$ μοναδιαίο διάνυσμα θα είναι $\boxed{\vec{y}_n = \vec{x}_n^T \cdot \vec{u}_1}$

$$Var(\vec{y}) = \frac{1}{N}\sum_{n=1}^{N} \vec{y}_n^{\,2} = \frac{1}{N}\sum_{n=1}^{N} (\vec{x}_i^T \cdot \vec{u}_1)^2 = \frac{1}{N}\sum_{n=1}^{N} (\vec{x}_n^T \cdot \vec{u}_1) \overset{\text{dot product}}{\circ} (\vec{x}_n^T \vec{u}_1) = \frac{1}{N}\sum_{n=1}^{N} \vec{u}_1^T (\vec{x}_n \circ \vec{x}_n^T)\vec{u}_1$$

$$\Rightarrow Var(\vec{y}) = \vec{u}_1^T \underbrace{\left(\frac{1}{N}\sum_{n=1}^{N} \vec{x}_n \cdot \vec{x}_n^T\right)}_{\text{Covariance matrix}} \cdot \vec{u}_1 = \boxed{\vec{u}_1^T \hat{\Sigma} \cdot \vec{u}_1}$$

Θέλουμε να μεγιστοποιήσουμε το variance, με $\boxed{\|u_1\| = 1}$. ($\max \underbrace{u_1^T \cdot \hat{\Sigma} u_1}_{f}$) με $u_1^T u_1 \equiv 1$

Θα αξιοποιήσουμε την μέθοδο των πολλαπλασιαστών Lagrange. $\Rightarrow \underbrace{u_1^T u_1 - 1 = 0}_{g}$

Ορίζουμε συνάρτηση Lagrange: $\mathcal{L}(\vec{u}_1, \lambda) = \vec{u}_1^T \hat{\Sigma} \cdot \vec{u}_1 - \lambda_1(\vec{u}_1^T \cdot \vec{u}_1 - 1)$

$$\frac{\partial \mathcal{L}}{\partial \vec{u}_1} = 0 \Rightarrow 2\hat{\Sigma} \cdot \vec{u}_1 - 2\lambda_1 \vec{u}_1 = 0 \Leftrightarrow \hat{\Sigma} \cdot \vec{u}_1 = \lambda_1 \vec{u}_1 \quad (1)$$

Σημείωση: $\frac{\partial(\vec{x}^T A \vec{x})}{\partial \vec{x}} = \vec{x}^T(A + A^T)$, για τον $A = \hat{\Sigma}$, ο $\hat{\Sigma}$ είναι συμμετρικός άρα $\boxed{\hat{\Sigma}^T = \hat{\Sigma}}$

$$\frac{\partial \mathcal{L}}{\partial \lambda_1} = \vec{u}_1^T \vec{u}_1 - 1 = 0$$ δηλαδή τηρείται η προϋπόθεσή μας

Από την (1) βλέπουμε πως το διάνυσμα $\vec{u}_1$ είναι ιδιοδιάνυσμα και λ_1 η αντίστοιχη ιδιοτιμή.

Δεδομένου ότι θέλουμε να μεγιστοποιήσουμε το Var άρα επιλέγουμε την μεγαλύτερη ιδιοτιμή λ_1.

Άλλος ένας λόγος είναι:

Ότι αν πάρουμε ιδιοτιμές του $\hat{\Sigma}$, $\lambda_1, \dots, \lambda_\ell$ ταξινομημένες κατά φθίνουσα σειρά ($\lambda_1 \geq \lambda_2 \geq \lambda_3 \dots \geq \lambda_\ell$) (θετικά ημι-ορισμένος άρα $\lambda_i \geq 0$). Γράφουμε το $\vec{u}_1$ ως γραμμικό συνδυασμό των ιδιοδιανυσμάτων $\vec{v}_1, \vec{v}_2, \dots, \vec{v}_\ell$ (orthogonal). άρα $\vec{u}_1 = c_1 \cdot \vec{v}_1 + c_2 \vec{v}_2 + \dots + c_n \cdot \vec{v}_\ell \Rightarrow \vec{u}_1^T \hat{\Sigma} \vec{u}_1 = c_1^2 \lambda_1 + \dots + c_\ell^2 \lambda_\ell$

Όταν μεγιστοποιούμε αυτή τη ποσότητα(*): $c_2 = c_3 = \dots = c_\ell = 0$, $c_1 = 1$ άρα όταν μεγιστοποιείται έχει την μεγαλύτερη ιδιοτιμή: λ_1 (αυτό συμβαίνει και με την λύση Lagrange)

(*) όταν θέλουμε να εστιάσουμε σε ένα ιδιοδιάνυσμα β) Για $m=2$ παρόμοια έχουμε $\vec{y}_{n1} = \vec{u}_1^T \cdot \vec{x}_n$ και $\vec{y}_{n2} = \vec{u}_2^T \cdot \vec{x}_n$

$$Var_{total} = \frac{1}{N}\sum_{n=1}^{N}(\vec{y}_{n1}^{\,2} + \vec{y}_{n2}^{\,2}) = \frac{1}{N}\sum_{n=1}^{N}\vec{y}_{n1}^{\,2} + \frac{1}{N}\sum \vec{y}_{n2}^{\,2} \xlongequal[\text{ερωτ. α}]{\text{Όπως στο}} \vec{u}_1^T \hat{\Sigma} \vec{u}_1 + \vec{u}_2^T \cdot \hat{\Sigma} \cdot \vec{u}_2$$

Έχουμε πως $\boxed{\vec{u}_1^T \cdot \vec{u}_1 = \vec{u}_2^T \cdot \vec{u}_2 = 1}$ και $\boxed{\vec{u}_1^T \cdot \vec{u}_2 = 0}$

συντελεστής ορθογωνιότητας

$$\mathcal{L}(\vec{u}_1, \vec{u}_2, \lambda_1, \lambda_2, \mu) = \vec{u}_1^T \hat{\Sigma} \vec{u}_1 + \vec{u}_2^T \hat{\Sigma} \vec{u}_2 - \lambda_1(\vec{u}_1^T \vec{u}_1 - 1) - \lambda_2(\vec{u}_2^T \vec{u}_2 - 1) - \boxed{\mu}\,\vec{u}_1^T \cdot \vec{u}_2$$

$$\frac{\partial \mathcal{L}}{\partial \lambda_1} = 0 \Rightarrow \boxed{\vec{u}_1^T \vec{u}_1 = 1}, \quad \frac{\partial \mathcal{L}}{\partial \lambda_2} = 0 \Rightarrow \boxed{\vec{u}_2^T \vec{u}_2 = 1} \text{ και } \frac{\partial \mathcal{L}}{\partial \mu} = 0 \Rightarrow \boxed{\vec{u}_1^T \cdot \vec{u}_2 = 0}$$

Εξασφαλίσαμε τους περιορισμούς μας.

$$\frac{\partial \mathcal{L}}{\partial \vec{u}_1} = 2 \cdot \Sigma \vec{u}_1 - 2\lambda_1 \vec{u}_1 - \mu \cdot \vec{u}_2 = 0 \Leftrightarrow \Sigma \vec{u}_1 = \lambda_1 \cdot \vec{u}_1 + \frac{\mu}{2}\vec{u}_2$$

, επειδή $u_1 \perp u_2$ πρέπει $\mu = 0$

$$\Rightarrow \boxed{\Sigma \vec{u}_1 = \lambda_1 \cdot \vec{u}_1}$$

$$\frac{\partial \mathcal{L}}{\partial \vec{u}_2} = 0 \Rightarrow 2\Sigma \vec{u}_2 - 2\lambda_2 \vec{u}_2 - \mu \vec{u}_1 = 0 \Leftrightarrow \Sigma \vec{u}_2 = \lambda_2 \cdot \vec{u}_2 + \frac{\mu}{2}\vec{u}_1$$

, $u_1 \perp u_2$ άρα $\mu = 0$

$$\Rightarrow \boxed{\Sigma \vec{u}_2 = \lambda_2 \vec{u}_2}$$

Άρα u_1, u_2 ιδιοδιανύσματα του $\hat{\Sigma}$ με λ_1, λ_2 ιδιοτιμές. Επειδή μεγιστοποιούμε την διασπορά με τα λ_1, λ_2, όπως και για $m=1$ έτσι κι εδώ τα λ_1, λ_2 είναι τα μέγιστα ιδιοδιανύσματα

Απόδειξη: ίδια με ερώτημα α αλλά τώρα $c_1 = c_2 = 1$ και $c_3 = c_4 = \dots = c_n = 0$. Δηλαδή εστιάζουμε σε 2 διανύσματα. Επίσης, θέλουμε να μεγιστοποιήσουμε το Var_{total}, άρα τα λ_1, λ_2 είναι τα μέγιστα ιδιοδιανύσματα.

γ) Στην γενική περίπτωση $m \leq l$ έχουμε παρομοίως: $\vec{y}_{ni} = \vec{u}_i^T \vec{x}_n$, $i = 1, \dots, m$

$$Var_{total} = \sum_{i=1}^{m} \vec{u}_i^T \cdot \hat{\Sigma} \cdot \vec{u}_i$$

, έχουμε $u_i^T u_i = 1$ και $u_i^T u_j = 0$, για $i \neq j$

Lagrange Συνάρτηση:

$$L(u_1,\dots,u_m,\lambda_1,\dots,\lambda_m,\mu_{ij}) = \sum_{i=1}^{m} \vec{u}_i^T \Sigma \cdot \vec{u}_i - \sum_{i=1}^{m} \lambda_i(\vec{u}_i^T \cdot \vec{u}_i - 1) - \sum_{i<j} \mu_{ij} \cdot \vec{u}_i^T \vec{u}_j$$

$$\frac{\partial L}{\partial \lambda_i} = 0 \Rightarrow \boxed{\vec{u}_i^T \cdot \vec{u}_i = 1}, \quad \frac{\partial L}{\partial \mu_{ij}} = 0 \Rightarrow \boxed{u_i^T u_j = 0}$$, Εξασφαλίσαμε τους περιορισμούς

- $\frac{\partial L}{\partial u_i} = 0 \Rightarrow 2\Sigma\vec{u}_i - 2\lambda_i \vec{u}_i - \sum_{j\neq j} \mu_{ij} \vec{u}_i^T \vec{u}_j = 0$, με $u_i \perp u_j$ άρα $\boxed{\mu_{ij} = 0}$

$$\Rightarrow \Sigma\vec{u}_i = \lambda_i \vec{u}_i + \frac{1}{2}\sum_{i\neq j} \mu_{ij} \vec{u}_i^T \vec{u}_j \Rightarrow \boxed{\Sigma\vec{u}_i = \lambda_i \vec{u}_i}$$ ~> u_i ιδιοδιανύσματα με λ_i ιδιοτιμές

Και αποτελούν τις μεγαλύτερες $\lambda_1, \dots \lambda_m$ ιδιοτιμές. Απόδειξη: Γενίκευση του β αντί για 2 διανύσματα έχουμε $c_1 = 1, c_2 = 1, \dots, c_m = 1$, $c_{m+1} = c_{m+2} = \dots = c_l = 0$

$Var_{total_m} = \lambda_1 + \dots + \lambda_m$. Προφανώς, όταν έχουμε $\lambda_1, \dots, \lambda_m$ να είναι μέγιστα τότε έχουμε και μέγιστο Var_{total}

δ) Η συνολική διασπορά των δεδομένων x_n είναι: $Var = \frac{1}{N}\sum_{n=1}^{N} \|x_n\|^2$

Οι ιδιοτιμές του Σ αντιπροσωπεύουν την διασπορά σε κάθε συνιστώσα. Επομένως στις m συνιστώσες: $Var_m = \sum_{i=1}^{m} \lambda_i$. Αυτό ισχύει διότι $trace(\Sigma) = \sum_{i=1}^{l} \lambda_i = \frac{1}{N}\sum_{n=1}^{N} \|x_n\|^2$, άρα για variance στις m συνιστώσες:

Συνεπώς το ποσοστό p της διασποράς των δεδομένων που "ερμηνεύεται" από m κύριες συνιστώσες είναι:

$$p = \frac{Var_m}{Var} = \frac{\sum_{i=1}^{m} \lambda_i}{\frac{1}{N}\sum_{n=1}^{N} \|x_n\|^2} = N \cdot \frac{\sum_{i=1}^{m} \lambda_i}{\sum_{n=1}^{N} \|x_n\|^2}$$

Άσκηση 2.5

α) Καταστάσεις: Όλα τα τετράγωνα του 4x4 πλέγματος (16 καταστάσεις)

$$S = \{(x,y) \mid x,y \in \{0,1,2,3\}\}$$

Ενέργειες: Οι δυνατές κατευθύνσεις που μπορεί να κινηθεί το ρομπότ

$A = \{$πάνω, κάτω, αριστερά, δεξιά$\}$

Ανταμοιβές: $R: S \times A \times S \to \mathbb{R}$, Η ανταμοιβή εξαρτάται από την τωρινή κατάσταση s, την ενέργεια a και την επόμενη κατάσταση s'.

- $R(s,a,s') = +10$ όταν $s' = (3,3)$ (η επόμενη κίνηση μας πάει στον στόχο)
- $R(s,a,s') = -10$, όταν $s' = (2,2)$ (" " " " σε παγίδα)
- $R(s,a,s') = -1$, οποιαδήποτε άλλη κίνηση.

Πιθανότητες Μετάβασης:

$$P(s'|s,a) = \begin{cases} 1, & \text{αν η ενέργεια } a \text{ οδηγεί από το } s \text{ στο } s' \text{ αν } s' \neq s \text{ (για } s=(3,3) \text{ ή } (2,2)) \\ 0, & \text{αλλιώς,} \\ 1, & s = s' \text{ για } s(3,3),(2,2)\ \forall a. \end{cases}$$

(3,3) -> στόχος
(2,2) -> παγίδα

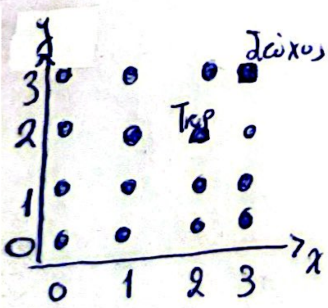
Οι κινήσεις είναι ντετερμινιστικές, συνεπώς και οι πιθανότητες μετάβασης. Επίσης, αν το ρομπότ βρίσκεται στα άκρα και επιχειρήσει να κινηθεί εκτός ορίων, τότε μένει στην ίδια θέση.

β) Επανάληψη τιμών:

$$V^\pi(s) = \mathbb{E}_{s'}[r + \gamma V^\pi(s)] = \mathbb{E}_{s'}[R(s,a,s') + \overset{\text{παράγοντας έκπτωσης}}{\boxed{\gamma}} V^\pi(s')] = r_0 + \gamma \max_{s_0} r_1 + \gamma^2 \max_{s_1} r_2 + \dots$$

Όπου ξεκινάμε από την εκτίμηση V_0 και υπολογίζουμε επαναληπτικά εκτιμήσεις V_i μέχρι το $r_0 + \dots$ να συγκλίνει στο $E_{s'}[r + \gamma V^\pi(s)]$ (εξίσωση Bellman)

γ) Με πολιτική απόσταση Manhattan αρχίζουμε από $(0,0)$ και αν έχουμε ισοπαλία σε απόσταση επιλέγουμε την κατεύθυνση με το μικρότερο x, έπειτα με το μικρότερο y: Άρα θα ακολουθήσουμε διαδρομή:



$$\underset{-1}{(0,0)} \to \underset{-1}{(0,1)} \to \underset{-1}{(0,2)} \to \underset{-1}{(0,3)} \to \underset{-1}{(1,3)} \to \underset{-1}{(2,3)} \to \underset{+10}{(3,3)}$$

Άρα $V^\pi(s) = r_0 + \gamma \max_{s_0} r_1 + \gamma^2 \max_{s_1} r_2 + \dots$

$= -1 + 0{,}9(-1) + 0.9^2 \cdot (-1) + 0{,}9^3(-1) + 0.9^4(-1) + 0.9^5(-1)$
$+ 0.9^6 \cdot 10 = 0.62882$

Το ίδιο αποτέλεσμα θα βρίσκαμε και με τον αναδρομικό τύπο.

δ) Εξαρτάται από την πολιτική που έχουμε διαλέξει. Για την πολιτική στο (γ)

έχουμε πως το $V(2,1) = r_{21} + \gamma \cdot V(2,2) \underset{\downarrow}{=} r_{21} + \gamma \cdot r_{22} = -1 + 0.9 \cdot (-1) = \boxed{-10}$

αφού δεν
υπάρχει στοχαστική κατάσταση