Aprendizagem 2023/24

Homework I V– Group 35

(ist1100032, ist1100070)

I. Pen-and-paper

1)

1

E-step: posterior probabilities for each observation

$$\boxed{x_1 : (1, 0.6, 0.1)}$$

likelihoods:

bernoulli

$$p(\{y_1\} \mid k=1) = 0.3^{1} \cdot (1-0.3)^{1-1} = 0.3$$

$$p(\{y_1\} \mid k=2) = 0.7^{1} \cdot (1-0.7)^{1-1} = 0.7$$

multivariate gaussian

$$p(\{y_2, y_3\} \mid k=1) = (0.6, 0.1) \sim N_1(\mu_1, \Sigma_1) = 0.06657$$

$$p(\{y_2, y_3\} \mid k=2) = (0.6, 0.1) \sim N_2(\mu_2, \Sigma_2) = 0.11961$$

- $P(x_1 \mid k=1) = P(\{y_1\} \mid k=1) \times P(\{y_2, y_3\} \mid k=1) = 0.01997$
- $P(x_1 \mid k=2) = P(\{y_1\} \mid k=2) \times P(\{y_2, y_3\} \mid k=2) = 0.083732$

posteriors:

$$p(k=1|x_1) = \frac{P(x_1|k=1) \times \tilde{\pi}_1}{\sum_{i=1}^{2} \tilde{\pi}_i \cdot P(x_1|k=i)} = 0.1926$$

0.05185

$$p(k=2|x_1) = \frac{P(x_1|K_2) \times \tilde{\pi}_2}{\sum_{i=1}^{2} \tilde{\pi}_i \cdot P(x_1|k=i)} = 0.8074$$

$\boxed{x_2 : (0, -0.4, 0.8)}$

likelihoods:

bernoulli

$$p(\{y_1\}|k=1) = 0.3^{0} \cdot (1-0.3)^{1-0} = 0.7$$

$$p(\{y_1\}|k=2) = 0.7^{0} \cdot (1-0.7)^{1-0} = 0.3$$

multivariate gaussian

$$p(\{y_2,y_3\}|k=1) = (-0.4, 0.8) \sim N_1(\mu_1, \Sigma_1) = 0.050048$$

$$p(\{y_2,y_3\}|k=2) = (-0.4, 0.8) \sim N_2(\mu_2, \Sigma_2) = 0.06819$$

- $P(x_2|k=1) = P(\{y_1\}|k=1) \times P(\{y_2,y_3\}|k=1) = 0.035034$
- $P(x_2|k=2) = P(\{y_1\}|k=2) \times P(\{y_2,y_3\}|k=2) = 0.020457$

posteriors:

$$P(K=1|x_2) = \frac{P(x_2|K=1) \times \pi_1}{\sum_{i=1}^{2} \pi_i \cdot P(x_2|K=i)} = 0.63134$$

$$P(K=2|x_2) = \frac{P(x_2|K_2) \times \pi_2}{\sum_{i=1}^{2} \pi_i \cdot P(x_2|K=i)} = 0.36865$$

$$\boxed{x_3 : (0, 0.2, 0.5)}$$

likelihoods:

bernoulli

$$p(\{y_1\}|K=1) = 0.3^{0} \cdot (1-0.3)^{1-0} = 0.7$$

$$p(\{y_1\}|K=2) = 0.7^{0} \cdot (1-0.7)^{1-0} = 0.3$$

multivariate gaussian

$$p(\{y_2, y_3\}|K=1) = (0.2, 0.5) \sim N_1(\mu_1, \Sigma_1) = 0.06837$$

$$p(\{y_2, y_3\}|K=2) = (0.2, 0.5) \sim N_2(\mu_2, \Sigma_2) = 0.12958$$

- $P(x_3 \mid k=1) = P(\{y_1\} \mid k=1) \times P(\{y_2, y_3\} \mid k=1) = 0.04786$
- $P(x_3 \mid k=2) = P(\{y_1\} \mid k=2) \times P(\{y_2, y_3\} \mid k=2) = 0.03887$

posteriors:

$$P(k=1 \mid x_3) = \frac{P(x_3 \mid k=1) \times \tilde{\pi}_1}{\sum_{i=1}^{2} \tilde{\pi}_i \cdot P(x_3 \mid k=i)} = 0.55181$$

$$P(k=2 \mid x_3) = \frac{P(x_3 \mid k=2) \times \tilde{\pi}_2}{\sum_{i=1}^{2} \tilde{\pi}_i \cdot P(x_3 \mid k=i)} = 0.44819$$

$\boxed{x_4 : (1, 0.4, -0.1)}$

likelihoods:

bernoulli

$$p(\{y_1\} \mid k=1) = 0.3^1 \cdot (1-0.3)^{1-1} = 0.3$$

$$p(\{y_1\} \mid k=2) = 0.7^1 \cdot (1-0.7)^{1-1} = 0.7$$

multivariate gaussian

$$p(\{y_2, y_3\} \mid k=1) = (0.4, -0.1) \sim N_1(\mu_1, \Sigma_1) = 0.059047$$

$$p(\{y_2, y_3\} \mid k=2) = (0.4, -0.1) \sim N_2(\mu_2, \Sigma_2) = 0.12450$$

- $P(x_4 \mid k=1) = P(\{y_1\} \mid k=1) \times P(\{y_2, y_3\} \mid k=1) = 0.01771$
- $P(x_4 \mid k=2) = P(\{y_1\} \mid k=2) \times P(\{y_2, y_3\} \mid k=2) = 0.08715$

posteriors:

$$P(k=1 \mid x_4) = \frac{P(x_4 \mid k=1) \times \pi_1}{\sum_{i=1}^{2} \pi_i \cdot P(x_4 \mid k=i)} = 0.1689$$

$$P(k=2 \mid x_4) = \frac{P(x_4 \mid k_2) \times \pi_2}{\sum_{i=1}^{2} \pi_i \cdot P(x_4 \mid k=i)} = 0.8311$$

M-Step : Update parameters

Cluster 1

Prior : $\pi_1 = P(k=1) = \frac{\sum_{i=1}^{4} P(k=1 \mid x_i)}{4} = 0.38617$

Aprendizagem 2023/24

Homework I V– Group 35

(ist1100032, ist1100070)

Bernoulli Parameters:

$$p_1 = P(y_1=1 \mid k=1) = \frac{\sum_{i=1}^{4} P(k=1 \mid y_{1i}) \cdot y_{1i}}{\sum_{i=1}^{4} P(k=1 \mid X_i)} = \frac{0.3615}{1.5447} = 0.2340$$

Gaussian Parameters:

$$\mu_1 = \frac{\sum_{i=1}^{4} P(k=1 \mid X_i) \cdot \{y_2, y_3\}_i}{\sum_{i=1}^{4} P(k=1 \mid X_i)} = \begin{pmatrix} 0.0409 \\ 0.7833 \end{pmatrix} \div 1.5447 = \begin{bmatrix} 0.026509 \\ 0.507129 \end{bmatrix}$$

$$\Sigma^{(i,j)} = \frac{\sum_{m=1}^{4} P(k=1 \mid X_m) \cdot (\{y_2, y_3\}_i - \mu_k) \cdot (\{y_2, y_3\}_i - \mu_k)^T}{\sum_{i=1}^{4} P(k=1 \mid X_i)} =$$

$$\begin{bmatrix} 0.21836 & -0.16282 \\ -0.16282 & 0.14837 \end{bmatrix} \div 1.5447 = \begin{bmatrix} 0.14136 & -0.10541 \\ -0.10541 & 0.09605 \end{bmatrix}$$

Cluster 2

$$\text{Prior}: \pi_1 = P(k=2) = \frac{\sum_{i=1}^{4} P(k=2 \mid x_i)}{4} = 0.61383$$

Bernoulli Parameters:

$$p_2 = P(y_1=1 \mid k=2) = \frac{\sum_{i=1}^{4} P(k=2 \mid y_{1i}) \cdot y_{1i}}{\sum_{i=1}^{4} P(k=2 \mid X_i)} = \frac{1.6385}{2.4553} = 0.66731$$

Gaussian Parameters:

$$\mu_2 = \frac{\sum_{i=1}^{4} P(k=2 \mid X_i) \cdot \{y_2, y_3\}_i}{\sum_{i=1}^{4} P(k=2 \mid X_i)} = \begin{pmatrix} 0.\ 75905 \\ 0.\ 0.51665 \end{pmatrix} \div 2.4553 = \begin{bmatrix} 0.30914 \\ 0.21042 \end{bmatrix}$$

$$\Sigma^{(i,j)} = \frac{\sum_{m=1}^{4} P(k=2 \mid X_m) \cdot (\{y_2, y_3\}_i - \mu_k) \cdot (\{y_2, y_3\}_i - \mu_k)^T}{\sum_{i=1}^{4} P(k=2 \mid X_i)} =$$

$$= \begin{bmatrix} 0.26589 & -0.217669 \\ -0.217669 & 0.255657 \end{bmatrix} \div 2.4553 = \begin{bmatrix} 0.10829 & -0.08865 \\ -0.08865 & 0.10412 \end{bmatrix}$$

Código auxiliar:

```python
import numpy as np
from scipy.stats import bernoulli, multivariate_normal

# Given data points
data = np.array([
    [1, 0.6, 0.1],
    [0, -0.4, 0.8],
    [0, 0.2, 0.5],
    [1, 0.4, -0.1]
])

# Given parameters
pi = np.array([0.5, 0.5])
p = np.array([0.3, 0.7])
N1 = {
    'mean': np.array([1, 1]),
    'cov': np.array([[2, 0.5], [0.5, 2]])
}
N2 = {
    'mean': np.array([0, 0]),
    'cov': np.array([[1.5, 1], [1, 1.5]])
}
```

```python
# Initialize arrays to store posterior probabilities
posterior_probs = np.zeros((len(data), 2))

# E-step: Compute posterior probabilities
for i in range(len(data)):
    print(f"\n\nX{ i }\n")
    likelihood_bernoulli_1 = p[0]**data[i, 0] * (1-p[0])**(1-data[i, 0])
    likelihood_bernoulli_2 = p[1]**data[i, 0] * (1-p[1])**(1-data[i, 0])
    print(f"Bernoulli Likelihood (Cluster 1): {likelihood_bernoulli_1}")
    print(f"Bernoulli Likelihood (Cluster 2): {likelihood_bernoulli_2}")

    likelihood_gaussian_1 = multivariate_normal.pdf(data[i, 1:], mean=N1['mean'], cov=N1['cov'])
    likelihood_gaussian_2 = multivariate_normal.pdf(data[i, 1:], mean=N2['mean'], cov=N2['cov'])
    print(f"Gaussian Likelihood (Cluster 1): {likelihood_gaussian_1}")
    print(f"Gaussian Likelihood (Cluster 2): {likelihood_gaussian_2}")

    likelihood_1 = likelihood_bernoulli_1 * likelihood_gaussian_1
    likelihood_2 = likelihood_bernoulli_2 * likelihood_gaussian_2
    print(f"likelihood (Cluster 1): {likelihood_1}")
    print(f"likelihood (Cluster 2): {likelihood_2}")

    denominator = pi[0] * likelihood_1 + pi[1] * likelihood_2
    print(f"Denominator): {denominator}")

    posterior_probs[i, 0] = (pi[0] * likelihood_bernoulli_1 * likelihood_gaussian_1) / denominator
    posterior_probs[i, 1] = (pi[1] * likelihood_bernoulli_2 * likelihood_gaussian_2) / denominator

print("Posterior Probabilities:\n", posterior_probs)
```

```
X0

Bernoulli Likelihood (Cluster 1): 0.3
Bernoulli Likelihood (Cluster 2): 0.7
Gaussian Likelihood (Cluster 1): 0.06657529920303393
Gaussian Likelihood (Cluster 2): 0.11961837142058572
likelihood (Cluster 1): 0.019972589760910178
likelihood (Cluster 2): 0.08373285999441
Denominator): 0.05185272487766009


X1

Bernoulli Likelihood (Cluster 1): 0.7
Bernoulli Likelihood (Cluster 2): 0.30000000000000004
Gaussian Likelihood (Cluster 1): 0.05004888824270901
Gaussian Likelihood (Cluster 2): 0.0681905803254947
likelihood (Cluster 1): 0.035034221769896304
likelihood (Cluster 2): 0.020457174097648412
Denominator): 0.027745697933772358
```

```
X2

Bernoulli Likelihood (Cluster 1): 0.7
Bernoulli Likelihood (Cluster 2): 0.30000000000000004
Gaussian Likelihood (Cluster 1): 0.06837452355368487
Gaussian Likelihood (Cluster 2): 0.12958103481626038
likelihood (Cluster 1): 0.047862166487579405
likelihood (Cluster 2): 0.038874310444878116
Denominator): 0.04336823846622876


X3

Bernoulli Likelihood (Cluster 1): 0.3
Bernoulli Likelihood (Cluster 2): 0.7
Gaussian Likelihood (Cluster 1): 0.059046993443730274
Gaussian Likelihood (Cluster 2): 0.12450008976589248
likelihood (Cluster 1): 0.01771409803311908
likelihood (Cluster 2): 0.08715006283612474
Denominator): 0.052432080434621914
Posterior Probabilities:
 [[0.19258959 0.80741041]
 [0.63134512 0.36865488]
 [0.55181128 0.44818872]
 [0.16892423 0.83107577]]
```

```python
# M-step: Update model parameters
new_pi = np.mean(posterior_probs, axis=0)
print("PI:", new_pi)
print("\n\n")

# Update p
print("BERNOULLI")

new_p = np.zeros(2)
for j in range(2): # 2 clusters
    print("cluster:", j)
    p_sum = 0
    for i in range(len(data)):
        p_sum += data[i, 0] * posterior_probs[i, j]
    print("p_sum:", p_sum)
    denominator = np.sum(posterior_probs[:, j])
    print("denominator:", denominator)
    new_p[j] = p_sum / denominator
print("p:", new_p)
print("\n\n")
```

```python
# Update N1 and N2
new_N1_mean = np.zeros(2)
new_N2_mean = np.zeros(2)
new_N1_cov = np.zeros((2, 2))
new_N2_cov = np.zeros((2, 2))

print("\nGAUSS")
for j in range(2): # 2 clusters
    print("\ncluster:", j)
    mean_sum = np.zeros(2)
    cov_sum = np.zeros((2, 2))
    for i in range(len(data)):
        mean_sum += posterior_probs[i, j] * data[i, 1:]
        cov_sum += posterior_probs[i, j] * np.outer(data[i, 1:] - [N1['mean'], N2['mean']][j], data[i, 1:] - [N1['mean'], N2['mean']][j])
    print("mean_sum:", mean_sum)
    print("cov_sum:", cov_sum)
    if j == 0:
        new_N1_mean = mean_sum / np.sum(posterior_probs[:, j])
        new_N1_cov = cov_sum / np.sum(posterior_probs[:, j])
    else:
        new_N2_mean = mean_sum / np.sum(posterior_probs[:, j])
        new_N2_cov = cov_sum / np.sum(posterior_probs[:, j])
```

```
# Update the model parameters
pi = new_pi
p = new_p

N1['mean'] = new_N1_mean
N2['mean'] = new_N2_mean
N1['cov'] = new_N1_cov
N2['cov'] = new_N2_cov

print("\n")
print("N1 mean:", N1['mean'])
print("N1 cov:\n", N1['cov'])
print("\n")
print("N2 mean:", N2['mean'])
print("N2 cov:\n", N2['cov'])
```

```
PI: [0.38616755 0.61383245]
```

```
BERNOULLI
cluster: 0
p_sum: 0.36151382107026986
denominator: 1.5446702187154808
cluster: 1
p_sum: 1.63848617892973
denominator: 2.455329781284519
p: [0.23403948 0.66731817]
```

```
GAUSS

cluster: 0
mean_sum: [0.04094766 0.78334827]
cov_sum: [[ 0.21836232 -0.16281668]
 [-0.16281668  0.1483696 ]]

cluster: 1
mean_sum: [0.75905234 0.51665173]
cov_sum: [[ 0.26589514 -0.21766927]
 [-0.21766927  0.25565705]]


N1 mean: [0.026509   0.50712978]
N1 cov:
 [[ 0.14136501 -0.10540546]
 [-0.10540546  0.0960526 ]]


N2 mean: [0.30914476 0.2104205 ]
N2 cov:
 [[ 0.10829305 -0.08865175]
 [-0.08865175  0.1041233 ]]
```

Aprendizagem 2023/24

Homework I V– Group 35

(ist1100032, ist1100070)

2)

2) $x_{new} = \begin{pmatrix} 1 \\ 0{,}3 \\ 0{,}7 \end{pmatrix}$

After EM update

$\pi_1 = 0{,}38617$ $\qquad \pi_2 = 0{,}61383$

$p_1 = 0{,}23404$ $\qquad p_2 = 0{,}66732$

$$N_1\left(\mu_1 = \begin{pmatrix} 0{,}02651 \\ 0{,}50713 \end{pmatrix}, \Sigma_1 = \begin{pmatrix} 0{,}14137 & -0{,}10541 \\ -0{,}10541 & 0{,}09605 \end{pmatrix} \right)$$

$$N_2\left(\mu_2 = \begin{pmatrix} 0{,}30914 \\ 0{,}21042 \end{pmatrix}, \Sigma_2 = \begin{pmatrix} 0{,}10829 & -0{,}08865 \\ -0{,}08865 & 0{,}10412 \end{pmatrix} \right)$$

likelihood cluster 1 $= p(\{y_1\} \mid k=1) \times p(\{y_2, y_3\} \mid k=1) =$

$= 0{,}006337$

likelihood cluster 2 $= p(\{y_1\} \mid k=2) \times p(\{y_2, y_3\} \mid k=2) =$

$= 0{,}04567$

posteriors:

$$p(k=1 \mid x_{new}) = \frac{P(x_{new} \mid k=1) \times \pi_1}{\sum_{i=1}^{2} \pi_i \cdot P(x_{new} \mid k=i)} = 0{,}08029$$

$$p(k=2 \mid x_{new}) = \frac{P(x_{new} \mid k=2) \times \pi_2}{\sum_{i=1}^{2} \pi_i \cdot P(x_{new} \mid k=i)} = 0{,}91971$$

Código auxiliar:

```
import numpy as np
from scipy.stats import multivariate_normal

# Given initial values
pi1 = new_pi[0]
pi2 = new_pi[1]
p1 = new_p[0]
p2 = new_p[1]
mu1 = new_N1_mean
sigma1 = new_N1_cov
mu2 = new_N2_mean
sigma2 = new_N2_cov
x_new = np.array([1, 0.3, 0.7])
```

```python
# Likelihood of x_new under each cluster
likelihood_c1 = (p1**x_new[0]) * ((1 - p1)**(1 - x_new[0])) * multivariate_normal.pdf(x_new[1:], mean=mu1, cov=sigma1)
likelihood_c2 = (p2**x_new[0]) * ((1 - p2)**(1 - x_new[0])) * multivariate_normal.pdf(x_new[1:], mean=mu2, cov=sigma2)

print("Likelihood for Cluster 1: ",likelihood_c1)
print("Likelihood for Cluster 2: ",likelihood_c2)

# Posterior probabilities
posterior_c1 = (likelihood_c1 * pi1) / (likelihood_c1 * pi1 + likelihood_c2 * pi2)
posterior_c2 = (likelihood_c2 * pi2) / (likelihood_c1 * pi1 + likelihood_c2 * pi2)

print("Posterior Probability for Cluster 1:", posterior_c1)
print("Posterior Probability for Cluster 2:", posterior_c2)
```

```
Likelihood for Cluster 1:  0.006336753293816409
Likelihood for Cluster 2:  0.045665170414993406
Posterior Probability for Cluster 1: 0.08028950846197516
Posterior Probability for Cluster 2: 0.9197104915380249
```

3)

4)

II. Programming and critical analysis

1)

```
from scipy.io import arff
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import metrics
from sklearn.preprocessing import MinMaxScaler
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import warnings

warnings.filterwarnings("ignore", category=FutureWarning) # Ignore FutureWarnings


data = arff.loadarff('column_diagnosis.arff')
df = pd.DataFrame(data[0])

# Features
X = df.drop(columns='class').values

# Normalization
scaler = MinMaxScaler()
X_normalized = scaler.fit_transform(X)
```

```python
def purity_score(y_true, y_pred):
    confusion_matrix = metrics.cluster.contingency_matrix(y_true, y_pred)
    return np.sum(np.amax(confusion_matrix, axis=0)) / np.sum(confusion_matrix)


k_values = [2, 3, 4, 5]

silhouette_scores = {}
purity_scores = {}

for k in k_values:
    kmeans = KMeans(n_clusters=k, random_state=0)
    cluster_labels = kmeans.fit_predict(X_normalized)

    silhouette = metrics.silhouette_score(X_normalized, cluster_labels)
    silhouette_scores[k] = silhouette

    purity = purity_score(df['class'], cluster_labels)
    purity_scores[k] = purity

for k, silhouette in silhouette_scores.items():
    print(f"K = {k}, Silhouette Score: {silhouette}")
for k, purity in purity_scores.items():
    print(f"K = {k}, Purity Score: {purity}")
```

```
K = 2, Silhouette Score: 0.36044124340441114
K = 3, Silhouette Score: 0.29579055730002257
K = 4, Silhouette Score: 0.27442402122340176
K = 5, Silhouette Score: 0.23823928397844843
K = 2, Purity Score: 0.632258064516129
K = 3, Purity Score: 0.667741935483871
K = 4, Purity Score: 0.6612903225806451
K = 5, Purity Score: 0.6774193548387096
```

2)

```python
n_components = 2

pca = PCA(n_components=n_components)
principal_components = pca.fit_transform(X_normalized)

explained_variance = pca.explained_variance_ratio_
print("Explained Variance for Top Two Components:", explained_variance)

for i, component in enumerate(range(1, n_components + 1)):
    print(f"\nTop Variables for Principal Component {component} (sorted by relevance):")
    component_weights = pca.components_[i]
    sorted_indices = np.argsort(np.abs(component_weights))[::-1]

    for idx in sorted_indices:
        print(f"Variable: {df.columns[idx]}, Weight: {component_weights[idx]}")
```

```
Explained Variance for Top Two Components: [0.56181445 0.20955953]

Top Variables for Principal Component 1 (sorted by relevance):
Variable: pelvic_incidence, Weight: 0.5916206177372234
Variable: lumbar_lordosis_angle, Weight: 0.5150847620730926
Variable: pelvic_tilt, Weight: 0.46703943896727157
Variable: sacral_slope, Weight: 0.3256888625569196
Variable: degree_spondylolisthesis, Weight: 0.21692963450485403
Variable: pelvic_radius, Weight: -0.11582397626328887

Top Variables for Principal Component 2 (sorted by relevance):
Variable: pelvic_tilt, Weight: -0.6703727595553627
Variable: pelvic_radius, Weight: -0.5810738370953603
Variable: sacral_slope, Weight: 0.4433029949470745
Variable: pelvic_incidence, Weight: 0.10003707489152235
Variable: lumbar_lordosis_angle, Weight: 0.08004745059088263
Variable: degree_spondylolisthesis, Weight: 0.004582909709399325
```

3)

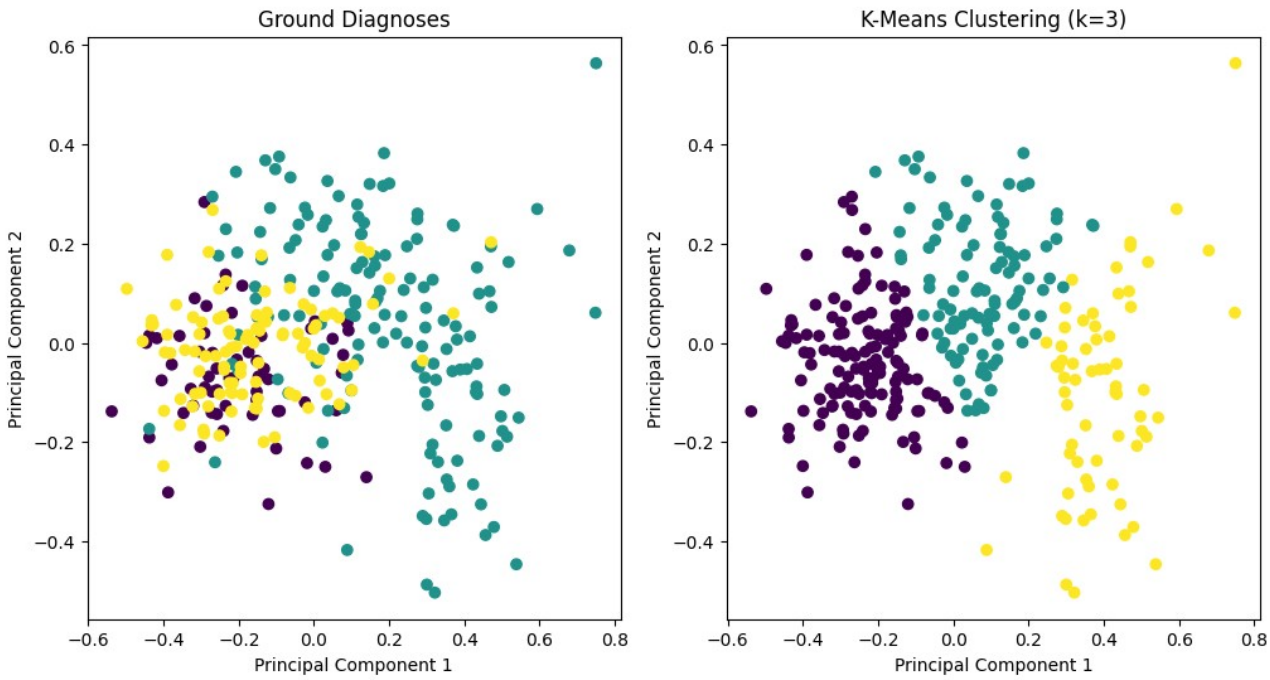
```
kmeans = KMeans(n_clusters=3, random_state=0)
cluster_labels = kmeans.fit_predict(X_normalized)

# Mapping of class labels to numerical values
y = df['class']
y = y.astype(str)
class_mapping = {'Hernia': 0, 'Spondylolisthesis': 1, 'Normal': 2}
class_numerical = y.map(class_mapping)
class_numerical

# ground diagnoses
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.scatter(principal_components[:, 0], principal_components[:, 1], c=class_numerical)
plt.title("Ground Diagnoses")
plt.xlabel("Principal Component 1")
plt.ylabel("Principal Component 2")

# k-means clustering
plt.subplot(1, 2, 2)
plt.scatter(principal_components[:, 0], principal_components[:, 1], c=cluster_labels)
plt.title("K-Means Clustering (k=3)")
plt.xlabel("Principal Component 1")
plt.ylabel("Principal Component 2")

plt.show()
```



4)

END