

Nama : DESI ARTI
NPM : 2206130706

DBSCAN CLUSTERING

```
import pandas as pd
from sklearn.cluster import DBSCAN
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt

import pandas as pd

# Read the data from a CSV file
data = pd.read_csv("data.csv")

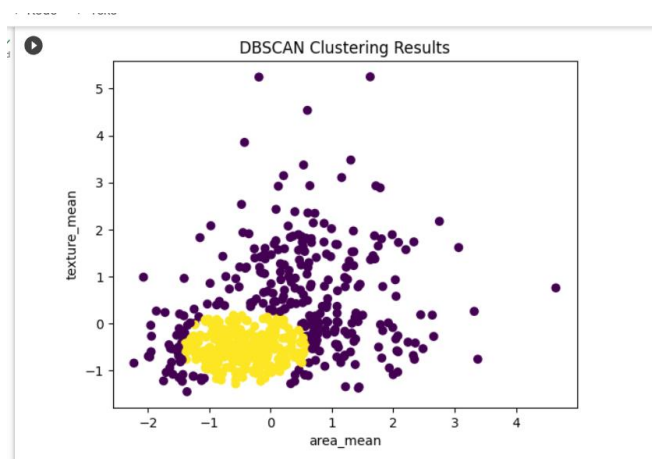
# Select columns diagonis, texture_mean, and area_mean
df= ["texture_mean", "area_mean"]
selected_data = data[df]

# Print the selected data
print(selected_data.head())

# Standardize the features
X = StandardScaler().fit_transform(selected_data)

# Run the DBSCAN algorithm
dbscan = DBSCAN(eps=0.5, min_samples=100)
clusters = dbscan.fit_predict(X)

# Visualize the results
plt.scatter(X[:, 0], X[:, 1], c=clusters)
plt.xlabel('area_mean')
plt.ylabel('texture_mean')
plt.title('DBSCAN Clustering Results')
plt.show()
```



SPECTRAL CLUSTERING

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import SpectralClustering
from sklearn.decomposition import PCA

# load dataset
data = pd.read_csv('data.csv')

# drop unnecessary columns
data = data.drop(['id', 'Unnamed: 32'], axis=1)

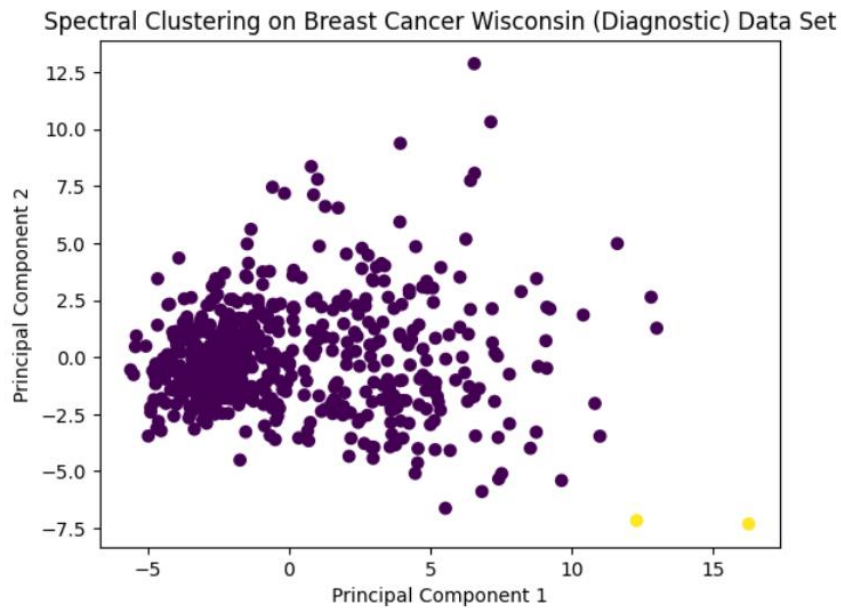
# encode the target variable
data['diagnosis'] = np.where(data['diagnosis'] == 'M', 1, 0)

# standardize the data
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data)

# apply PCA to reduce the number of features to 2
pca = PCA(n_components=3)
X_pca = pca.fit_transform(data_scaled)

# apply spectral clustering
spectral = SpectralClustering(n_clusters=2, affinity='rbf', gamma=0.1)
y_pred = spectral.fit_predict(data_scaled)

# plot the results
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=y_pred)
plt.title('Spectral Clustering on Breast Cancer Wisconsin (Diagnostic) Data Set')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```



K-MEANS CLUSTERING

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

# Load dataset
data = pd.read_csv('data.csv')

# Replace NaN values with mean of each column
data = data.fillna(data.mean(numeric_only=True))

# Remove non-numeric columns
data = data.drop(['diagnosis'], axis=1)

# Convert remaining columns to numeric type
data = data.apply(pd.to_numeric)

# Visualize data points
plt.scatter(data['texture_mean'], data['area_mean'])
plt.show()

# Initialize K-Means clustering
kmeans = KMeans(n_clusters=3)
```

