```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

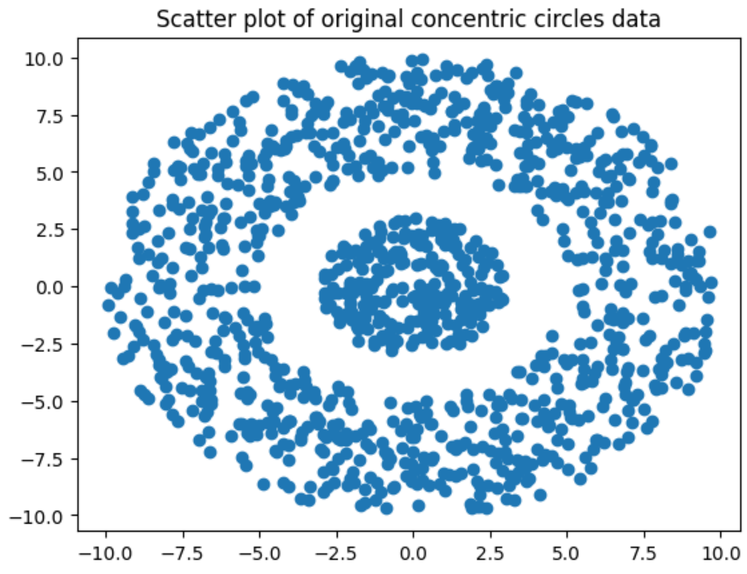
Part a

```
df=pd.read_csv("/content/ConcentricCirclesData.xls")
df = df.drop(df.columns[[0]], axis=1)
data=np.array(df)
print(data.shape)
print(df.head)
```

```
(1125, 2)
<bound method NDFrame.head of              x         y
0     -1.065198  2.416022
1      0.200612  0.050869
2     -2.159510 -0.225543
3     -0.643773  2.184968
4     -0.975247 -0.702547
...         ...       ...
1120 -8.471687  0.456926
1121 -0.161442 -5.285180
1122  8.163664  1.605210
1123  3.716827  7.172502
1124 -1.149643  5.443682

[1125 rows x 2 columns]>
```

```
plt.scatter(data[:,0],data[:,1])
plt.title("Scatter plot of original concentric circles data")
```

```
Text(0.5, 1.0, 'Scatter plot of original concentric circles data')
```



Answer

1. Describe the appearance of the points in the scatterplot?

There is one concentric circle inside of a ring of points.

2. How many clusters (clearly separated sets of points) are there?

There are two clusters of clearly separated data

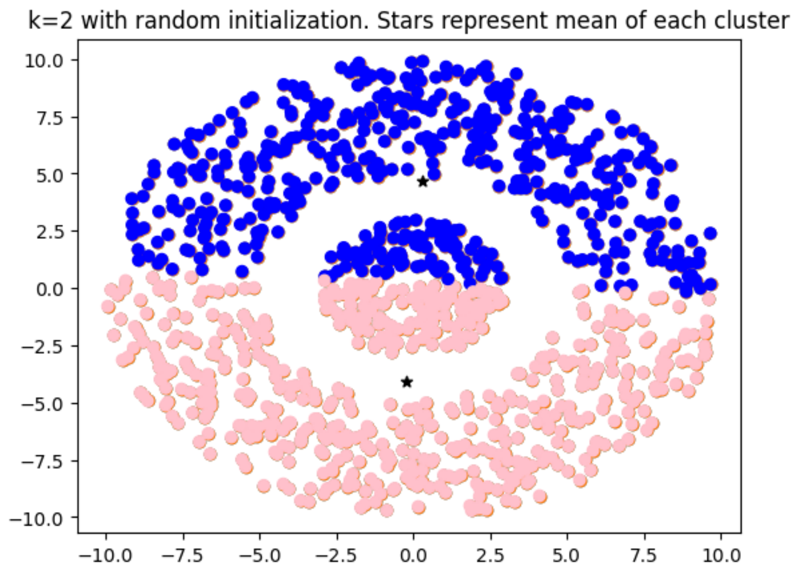
Part b

```
from sklearn.cluster import KMeans
import matplotlib
```

```
def kmeans_with_plotting(n_clusters,initialization,data_to_fit,title):

  kmeans = KMeans(n_clusters=n_clusters,init=initialization).fit(data_to_fit)
  plt.scatter(data_to_fit[:,0],data_to_fit[:,1])
  plt.scatter(data_to_fit[:,0],data_to_fit[:,1])
  p_kmeans=kmeans.predict(data_to_fit)
  colors=["blue","green","red","purple","orange","yellow","brown","pink"]
  plt.scatter(data_to_fit[:,0],data_to_fit[:,1],c=p_kmeans,cmap=matplotlib.colors.ListedColormap(colors))
  plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],marker="*",c="black")
  plt.title(title)
```

```
kmeans_with_plotting(2,"random",data,"k=2 with random initialization. Stars represent mean of each cluster")
```

```
    /usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: Futu
      warnings.warn(
```



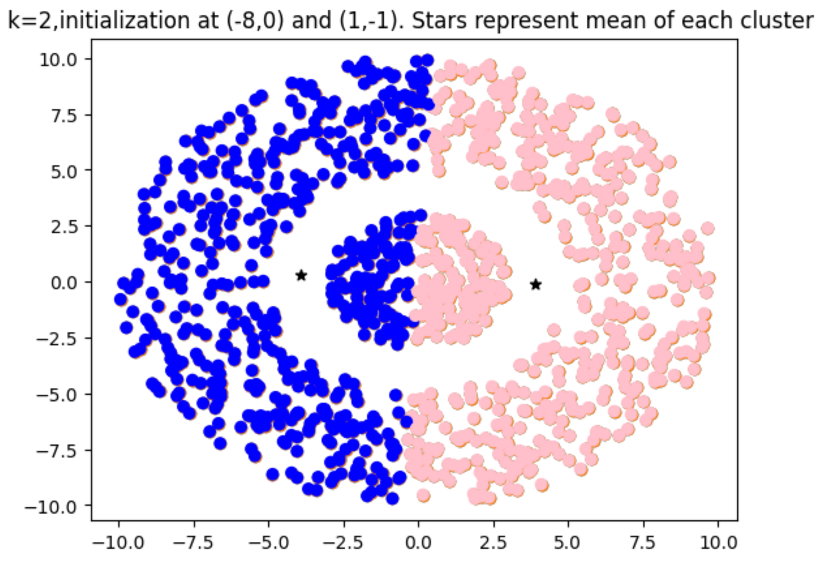
Answer

How well does this method identify the clusters in the plot? **It is not a good method . The method does not identify the two concentric clusters. It just makes division in upper points and bottom points. The cluster predicted by this method does not represent the true clusters in the data**

Part c

```
kmeans_with_plotting(2,[[-8,0],[1,-1]],data,"k=2,initialization at (-8,0)·and·(1,-1). Stars represent mean of each cluster ")
```

```
    /usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: Futu
      warnings.warn(
    /usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:1362: Run
      super()._check_params_vs_input(X, default_n_init=10)
```



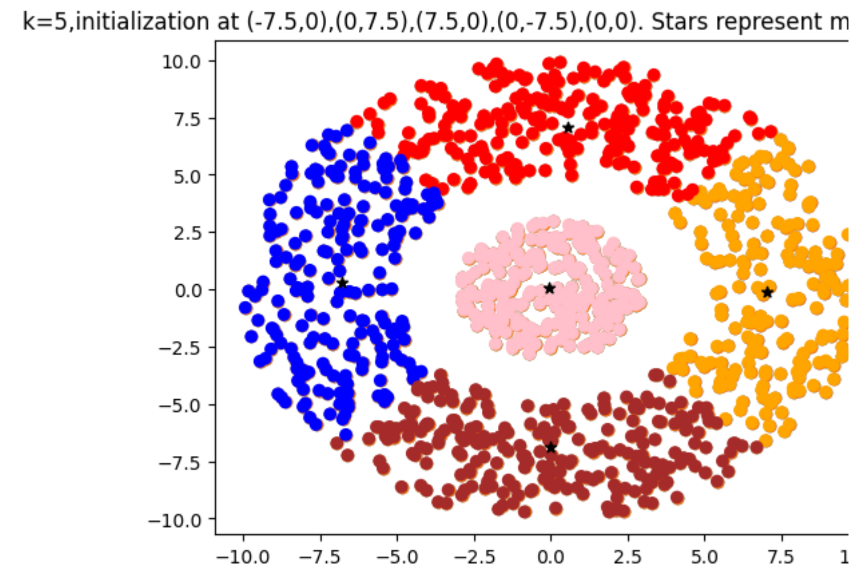
Answer

How well does this method identify the clusters in the plot? **This method does a poor work identifiying the two clusters of the plot. It just creates a left and right cluster, without identifying a cluster in the center and another cluster as the ring out of the inside circle.** Does choosing this initialization help in identifying the true clusters? **No. This initialization does not help identifying the true clusters**

Part d

```
kmeans_with_plotting(5,[[-7.5,0],[0,7.5],[7.5,0],[0,-7.5],[0,0]],data,"k=5,initialization at (-7.5,0),(0,7.5),(7.5,0),(0,-7.5),(0,0). Star
```

```
    /usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: Futu
      warnings.warn(
    /usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:1362: Run
      super()._check_params_vs_input(X, default_n_init=10)
```



Answer

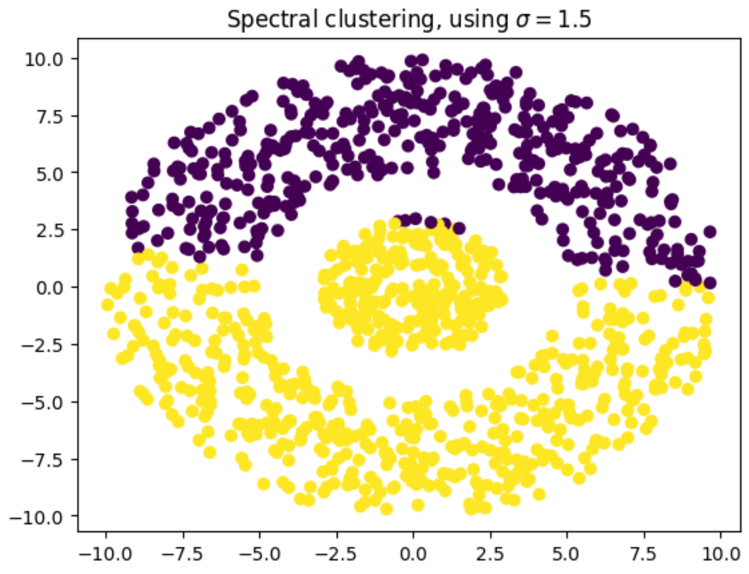
How well does this method identify the clusters in the plot? **This method does not identify the true clusters because we are using more number of clusters than the true number of clusters. The method separetes the center circle from the outer ring. However, the ring should be considered as just one cluster and the kmeans algorithm is splitting it in several clusters.** Does choosing this initialization help in identifying the true clusters? **No, because we are using a number of clusters greater than the number of true clusters.**

Part e

```
from scipy.spatial.distance import pdist, squareform
from sklearn import metrics
from numpy.linalg import eig
from sklearn.preprocessing import normalize

def spectral_with_plotting(n_eigen,s,X):
  D=metrics.pairwise_distances(X,X)
  K=np.exp(-D**2/(2*s**2))
  for i in range(K.shape[0]):
    for j in range(K.shape[1]):
      if i==j:
        K[i,j]=0.0

  D1=np.diag(np.sum(K,axis=1)**(-0.5))
  K1=D1.dot(K).dot(D1)
  w,v=eig(K1)

  top2=[-20,-22]
  index=[-10,-10]
  for i in range(w.shape[0]):
      if(w[i]>=top2[0]):
        top2[0]=w[i]
        index[0]=i
      if(w[i]>=top2[1] and w[i]<top2[0]):
        top2[1]=w[i]
        index[1]=i
  top=np.array([v[:,index[0]],v[:,index[1]]]).transpose()
  sum_of_rows = np.sqrt(np.sum(top**2,axis=1))

  normalized_array =normalize(top)
  kmeans = KMeans(n_clusters=n_eigen,init="random").fit(normalized_array)
  p_kmeans=kmeans.predict(normalized_array)
  colors=["blue","green","red","purple","orange","yellow","brown","pink"]
  fig1, ax1 = plt.subplots()
  plt.scatter(X[:,0],X[:,1],c=p_kmeans)#,cmap=matplotlib.colors.ListedColormap(colors))
  ax1.set_title("Spectral clustering, using $\sigma=$"+str(s))
```

```
spectral_with_plotting(2,1.5,data)
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: Futu
  warnings.warn(
```



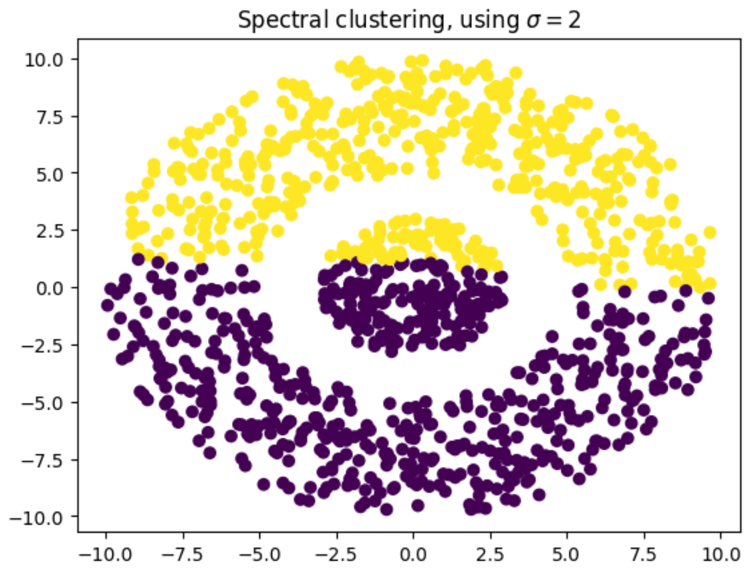
Answer

How well does this method identify the clusters in the plot? **The method does a poor job. It does not classify correctly the two clusters. It just divides the upper points and the bottom poiunts in two clusters.** How do the types of clusters found in this method differ from those found using K-Means? **When using random initialization for k=2 means, the result is similar. The difference is that there are more points in the upper cluster than with spectral clustering. For the case when using initialization at (-8,0) and (1,-1) with k means is that the clusters separetes as left and right points, while in the case of spectral clustering the points are separeted in two clusters as bottom and top points. Finally, comparing to the initialization with k=5 means, is that in this case there are only two clusters, while with k means there are 5 clusters, one in the middle circle of points and four in the outer ring.**

Part f

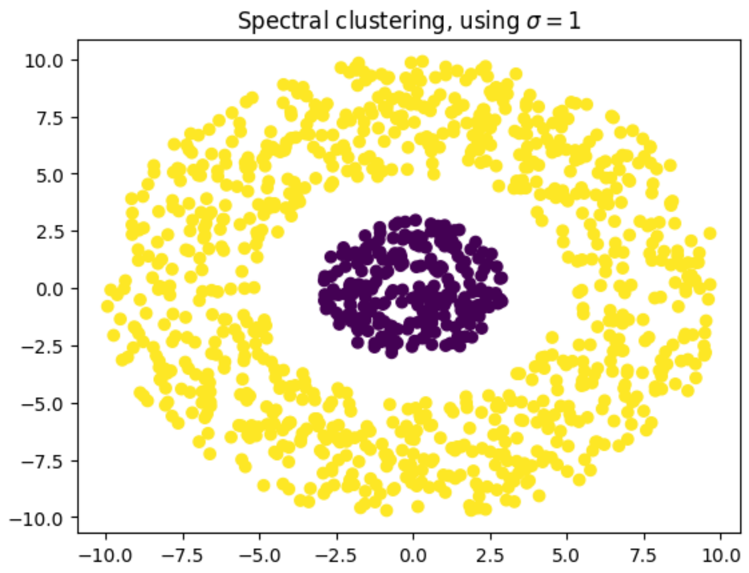
```
spectral_with_plotting(2,2,data)
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from
  warnings.warn(
```



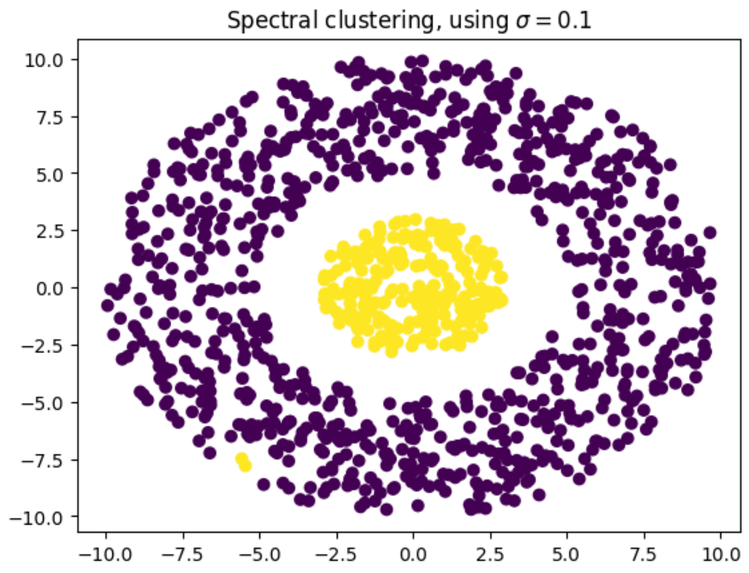
```
spectral_with_plotting(2,1,data)
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from
  warnings.warn(
```



```
spectral_with_plotting(2,0.1,data)
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: Futu
  warnings.warn(
```



Answer

For which of these values of σ do the most appropriate clusters get created?**For $\sigma = 1$ the most appropiate clusters get created as shown in the plot. Perfectly all the points in the inner circular group are taken as just one cluster, at the same time all the outer points in the ring as taken as just one cluster. This method with $\sigma = 1$ accurately predicts the true clusters** How well does this method identify the clusters in the plot? **When using $\sigma = 1$ the spectral clustering method identifies perfectly the two true clusters.When using $\sigma = 2$ the method does a poor job, separating the points in a upper and bottom cluster and missidentifying the true clusters. When using $\sigma = 0.1$ the job is almost perfect, the method just misidentify two points in the outer ring.**

✓ 2s completed at 9:52 AM