M11-L2-P1

November 26, 2023

0.1 M11-L2 Problem 1

In this problem you will implement the elbow method using three different sklearn clustering algorithms: (`KMeans`, `SpectralClustering`, `GaussianMixture`). You will use the algorithms to find the number of natural clusters for two different datasets, one "blob" shaped dataset, and one concetric circle dataset.

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
plt.rcParams['figure.dpi'] = 200

from sklearn.datasets import make_blobs, make_circles
from sklearn.cluster import KMeans, SpectralClustering
from sklearn.mixture import GaussianMixture

def plot_loss(loss, ax = None, title = None):
    if ax is None:
        ax = plt.gca()
    ax.plot(np.arange(2, len(loss)+2), loss, 'k-o')
    ax.set_xlabel('Number of Clusters')
    ax.set_ylabel('Loss')
    if title:
        ax.set_title(title)
    return ax

def plot_pred(x, labels, ax = None, title = None):
    if ax is None:
        ax = plt.gca()
    n_clust = len(np.unique(labels))
    for i in range(n_clust):
        ax.scatter(x[labels == i,0], x[labels == i,1], alpha = 0.5)
    ax.set_title(title)
    return ax

def compute_loss(x, labels):
    # Initialize loss
    loss = 0
    # Number of clusters
```

```
    n_clust = len(np.unique(labels))
    # Loop through the clusters
    for i in range(n_clust):
        # Compute the center of a given label
        center = np.mean(x[labels == i, :], axis = 0)
        # Compute the sum of squared distances between each point and its␣
↪corresponding cluster center
        loss += np.sum(np.linalg.norm(x[labels == i, :] - center, axis = 1)**2)
    return loss
```

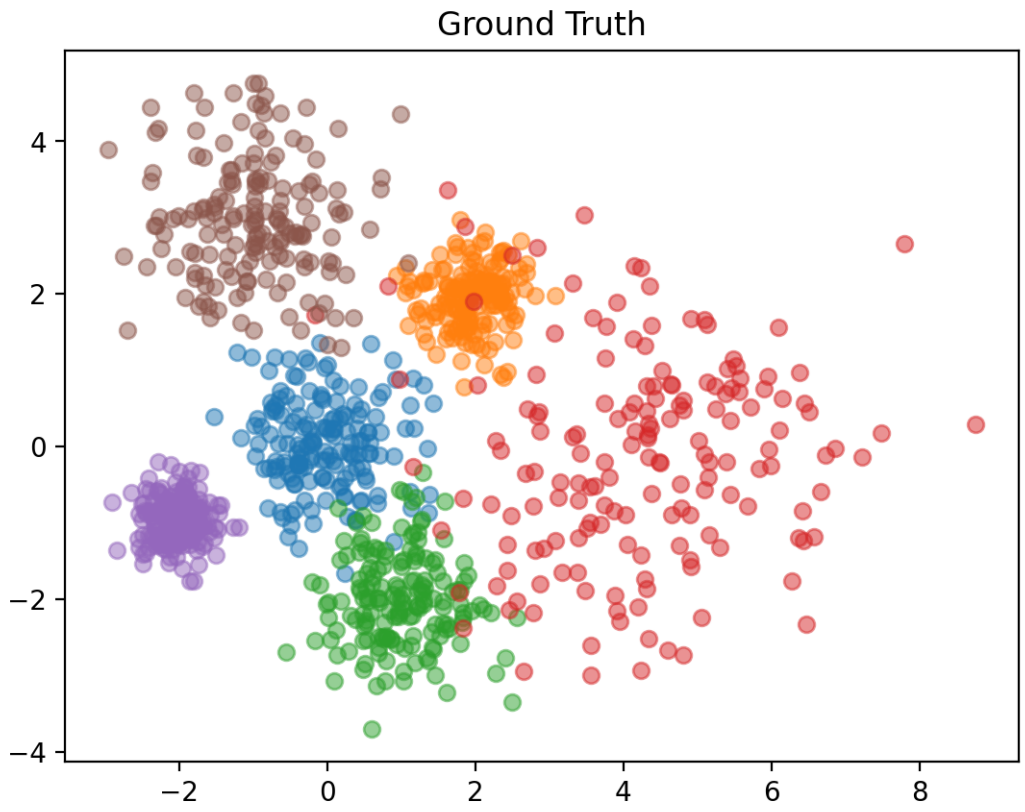
0.2 Blob dataset

Visualize the "blob" dataset generated below, using a unique color for each cluster of points, where y contains the label of each corresponding point in x.

[ ]:
```
## DO NOT MODIFY
x, y = make_blobs(n_samples = 1000, n_features = 2, centers =␣
↪[[0,0],[2,2],[1,-2],[4,0],[-2,-1],[-1,3]], cluster_std = [0.6,0.4,0.6,1.5,0.
↪3,0.8], random_state = 0)
```

[ ]:
```
fig, ax = plt.subplots()
plot_pred(x, y, ax, "Ground Truth")
```

[ ]:
```
<Axes: title={'center': 'Ground Truth'}>
```

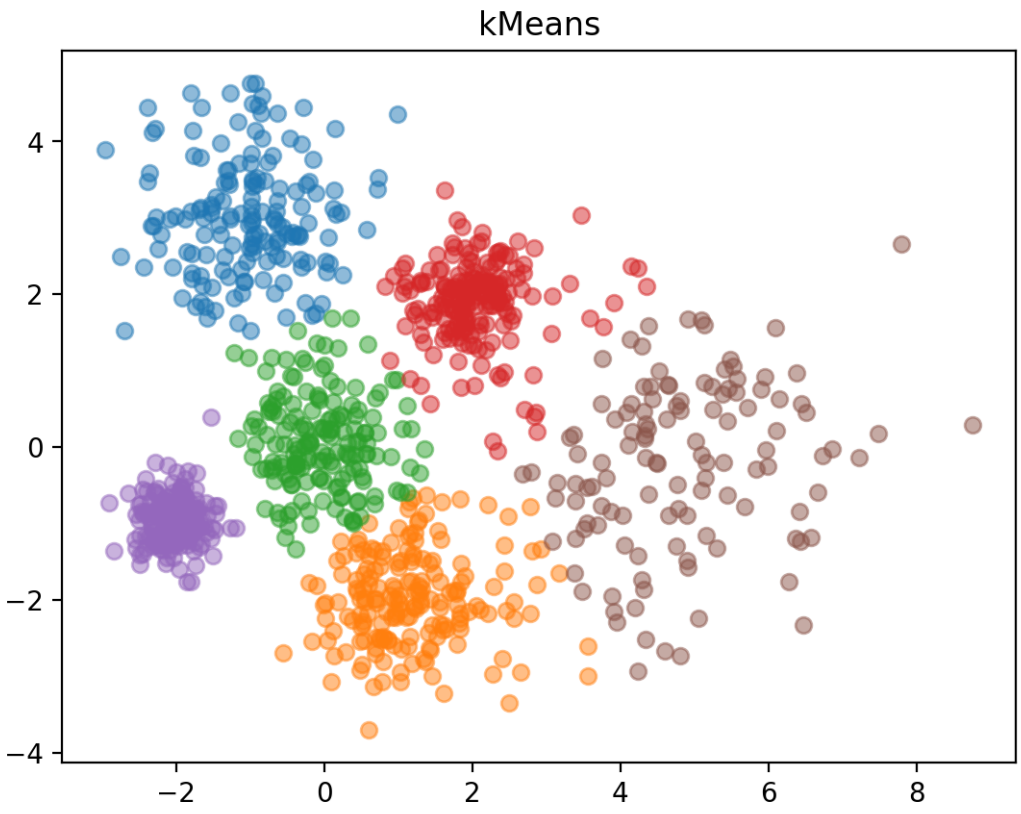


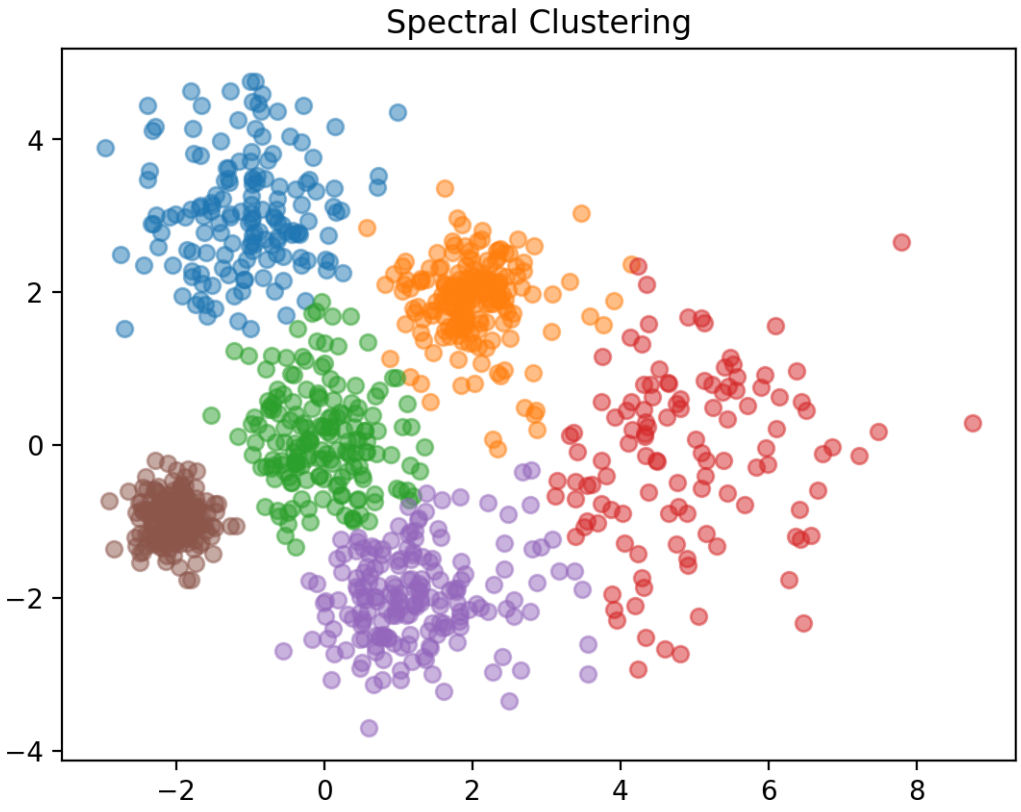
Use the `sklearn` KMeans, Spectral Clustering, and Gaussian Mixture Model functions to cluster the "blob" data with 6 clusters, and modify the parameters until you get satisfactory results. Plot the results of your three models side-by-side using `plt.subplots` and the provided `plot_pred(x, labels, ax, title)` function.

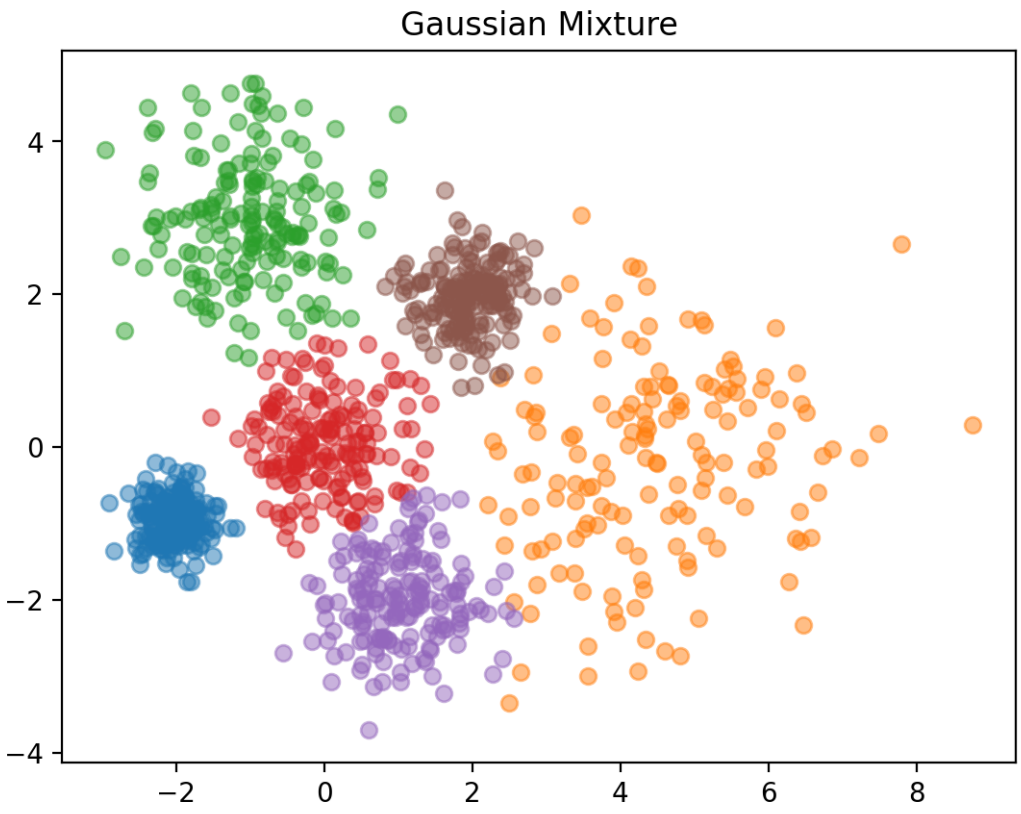
[ ]:
```
fig, ax = plt.subplots()
model1 = KMeans(n_clusters=6).fit(x)
plot_pred(x, model1.labels_, ax, "kMeans")

fig, ax = plt.subplots()
model2 = SpectralClustering(n_clusters=6).fit(x)
plot_pred(x, model2.labels_, ax, "Spectral Clustering")

fig, ax = plt.subplots()
model3 = GaussianMixture(n_components=6).fit(x)
plot_pred(x, model3.predict(x), ax, "Gaussian Mixture")
```

```
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-packages/threadpoolctl.py:1019:
RuntimeWarning: libc not found. The ctypes module in Python 3.8 is maybe too old
for this OS.
  warnings.warn(
```

[ ]:
```
<Axes: title={'center': 'Gaussian Mixture'}>
```


kMeans
4
2
0
−2
−4
−2
0
2
4
6
8


Spectral Clustering
4
2
0
−2
−4
−2
0
2
4
6
8



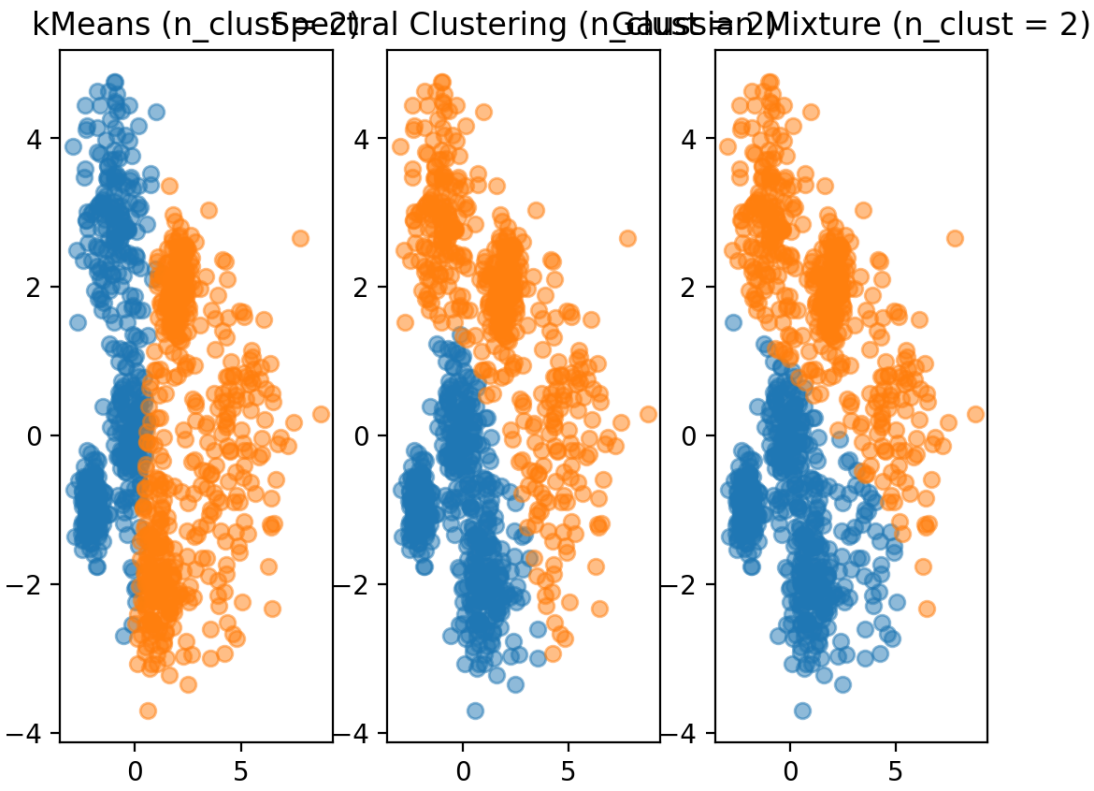
Using the parameters you found for the three models above, run each of the clustering algorithms for `n_clust = [2,3,4,5,6,7,8,9]` and compute the sum of squared distances loss for each case using the provided `compute_loss(x, labels)` function, where labels is the cluster assigned to each point by the algorithm. Plot loss versus number of cluster for each your three models in side-by-side subplots using the provided `plot_pred(x, labels, ax, title)` function.

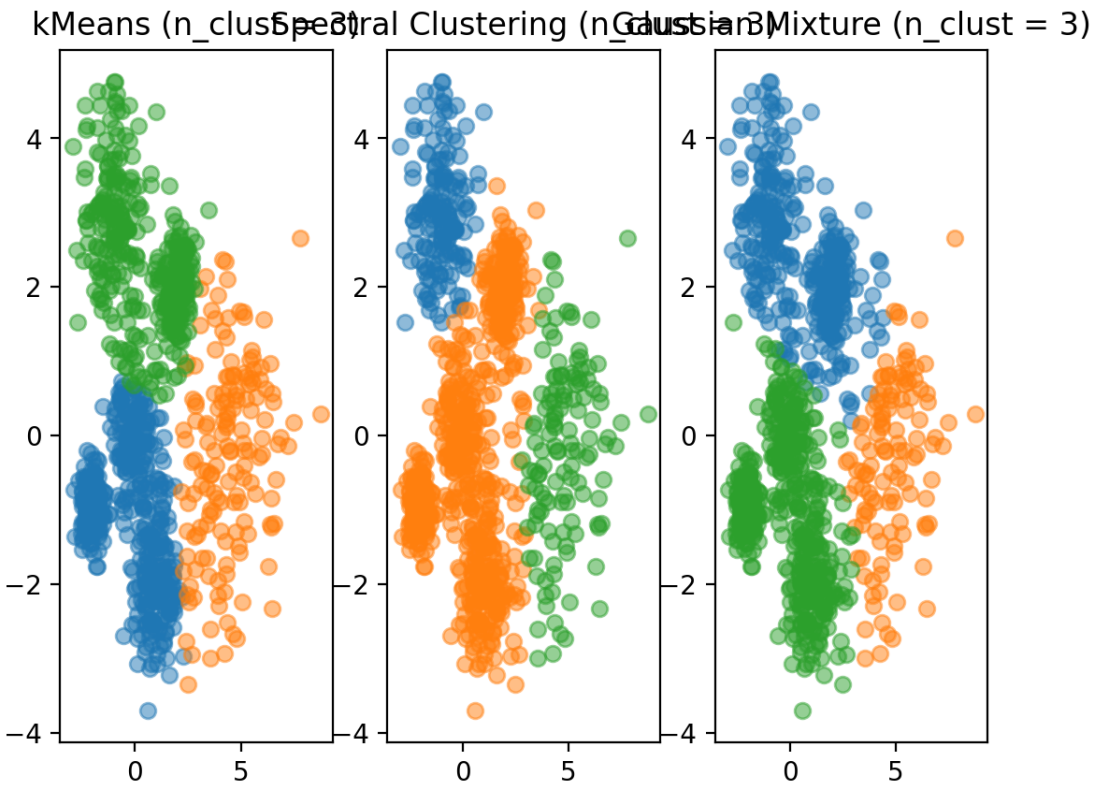
[ ]:
```
model1_sum_square_dist = []
model2_sum_square_dist = []
model3_sum_square_dist = []

for n_clust in [2, 3, 4, 5, 6, 7, 8, 9]:
    #Train models
    model1 = KMeans(n_clusters=n_clust).fit(x)
    model2 = SpectralClustering(n_clusters=n_clust).fit(x)
    model3 = GaussianMixture(n_components=n_clust).fit(x)

    model1_sum_square_dist.append(compute_loss(x, model1.labels_))
    model2_sum_square_dist.append(compute_loss(x, model2.labels_))
    model3_sum_square_dist.append(compute_loss(x, model3.predict(x)))

    fig, ax = plt.subplots(1,3)
```

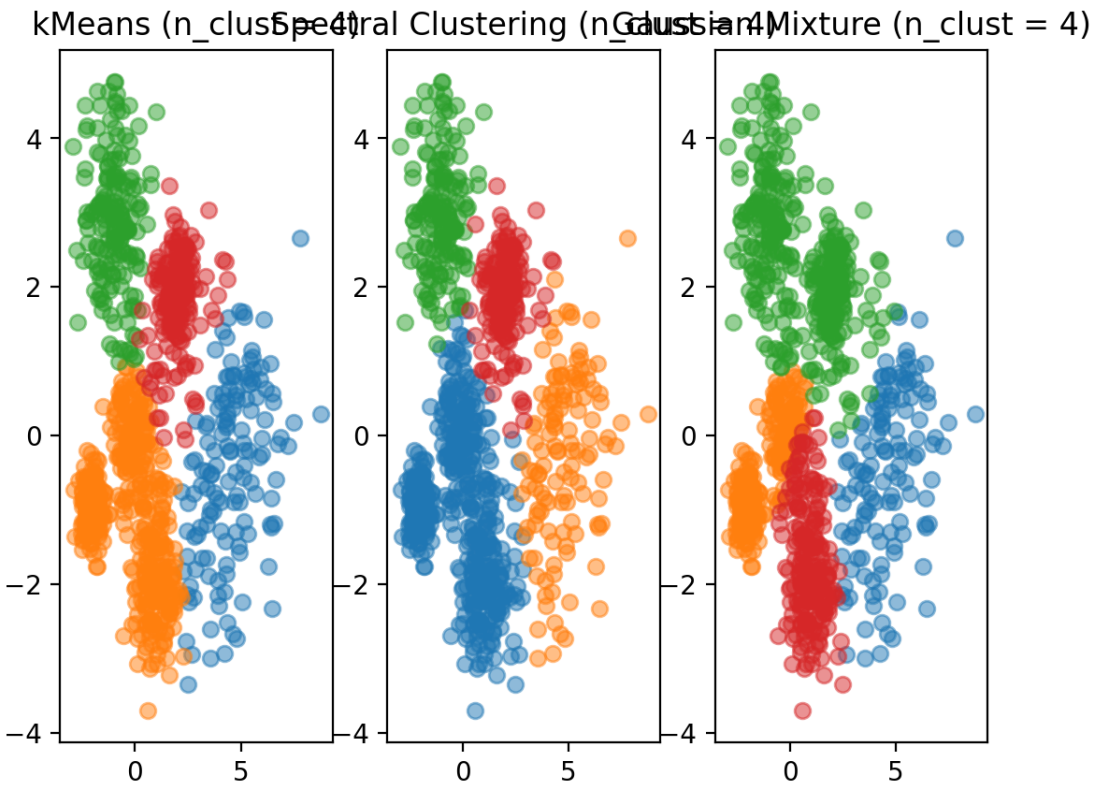
```
    plot_pred(x, model1.labels_, ax[0], f"kMeans (n_clust = {n_clust})")
    plot_pred(x, model2.labels_, ax[1], f"Spectral Clustering (n_clust =␣
 ↪{n_clust})")
    plot_pred(x, model3.predict(x), ax[2], f"Gaussian Mixture (n_clust =␣
 ↪{n_clust})")

fig, ax = plt.subplots(1,3)
plot_loss(model1_sum_square_dist, ax[0], "kMeans Sum Square Dist")
plot_loss(model2_sum_square_dist, ax[1], "Spectral Clustering Sum Square Dist")
plot_loss(model3_sum_square_dist, ax[2], "Gaussian Mixture Sum Square Dist")
```

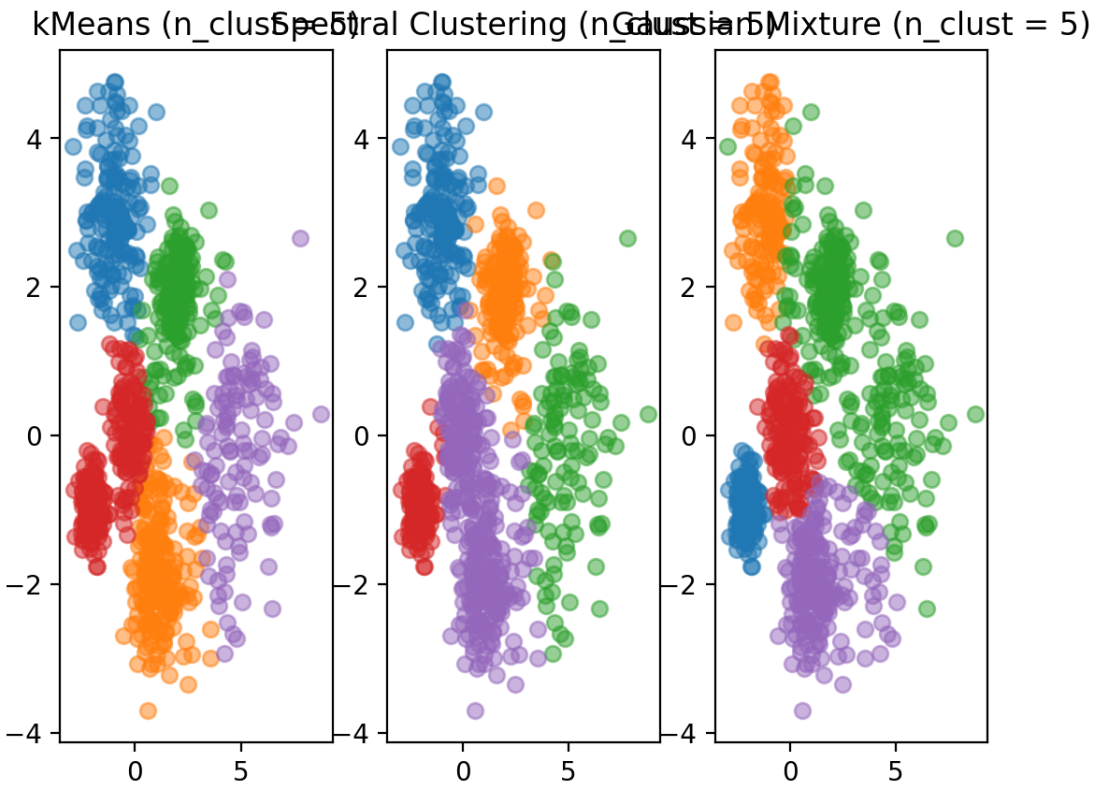
```
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

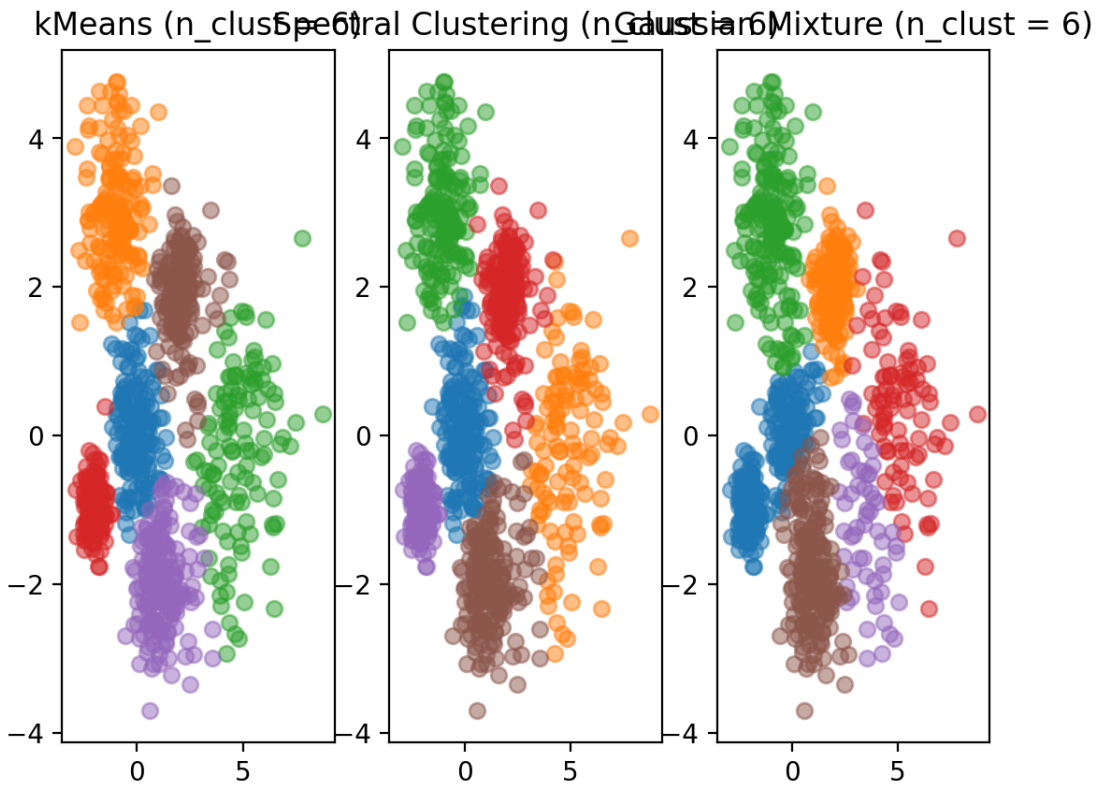
[ ]:
```
<Axes: title={'center': 'Gaussian Mixture Sum Square Dist'}, xlabel='Number of
Clusters', ylabel='Loss'>
```

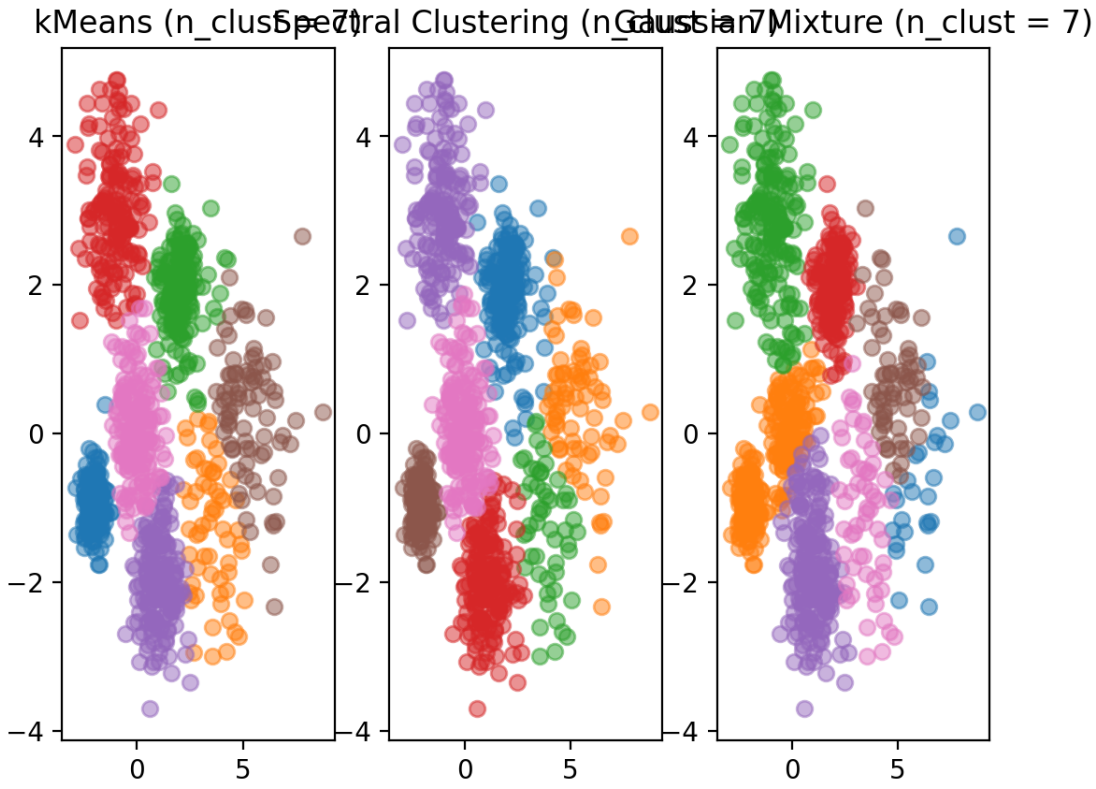

kMeans (n_clust = 2)
Spectral Clustering (n_clust = 2)
Gaussian Mixture (n_clust = 2)
4
2
0
−2
−4
0
5

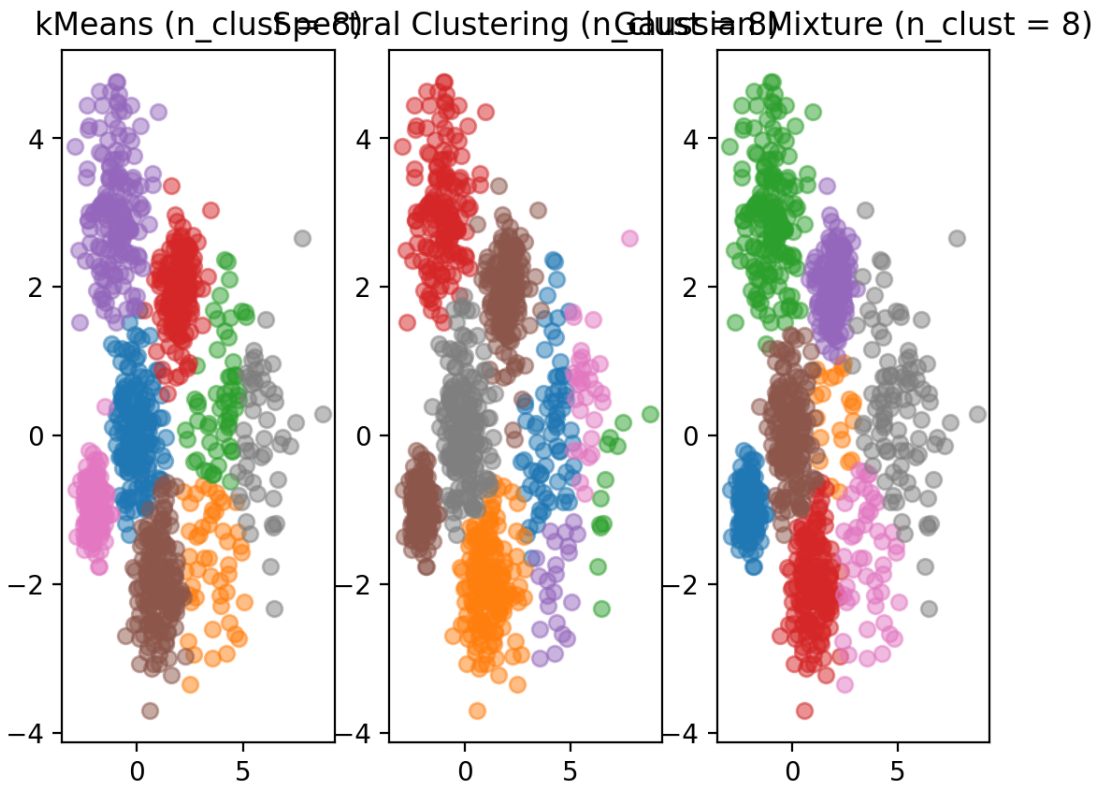

kMeans (n_clust = 3)
Spectral Clustering (n_clust = 3)
Gaussian Mixture (n_clust = 3)
4
2
0
−2
−4
0
5

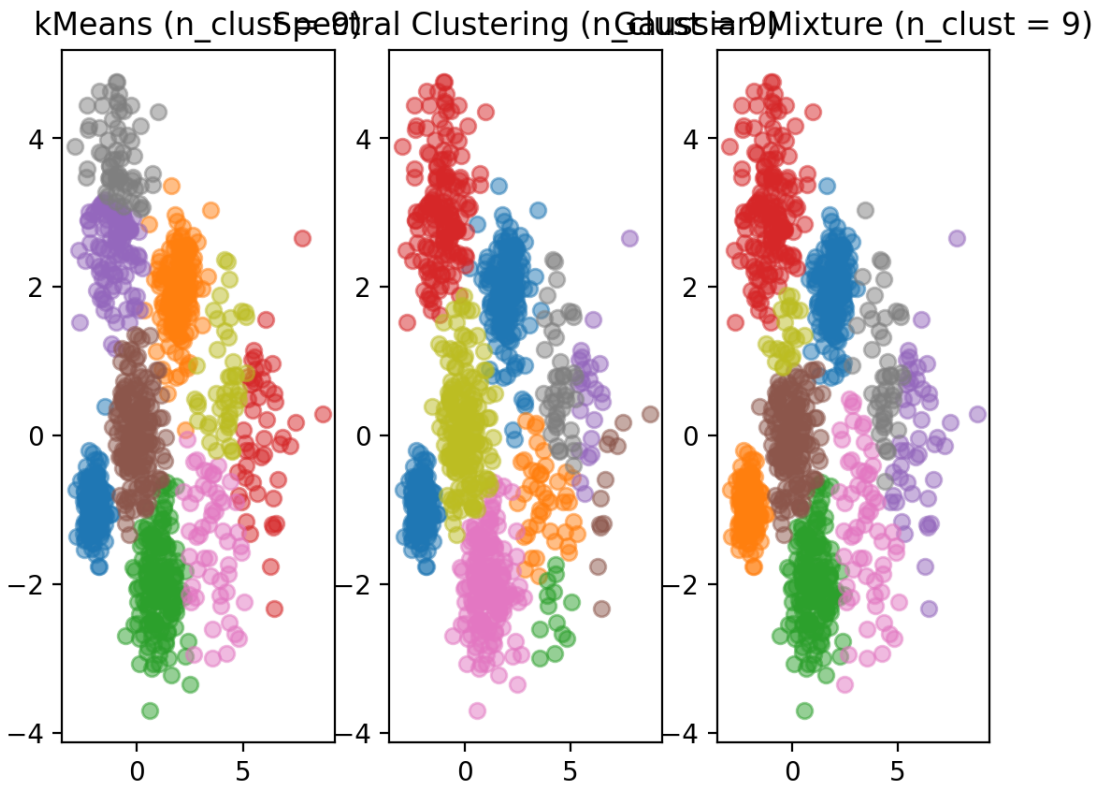

kMeans (n_clust = 4)
Spectral Clustering (n_clust = 4)
Gaussian Mixture (n_clust = 4)
4
2
0
−2
−4
0
5

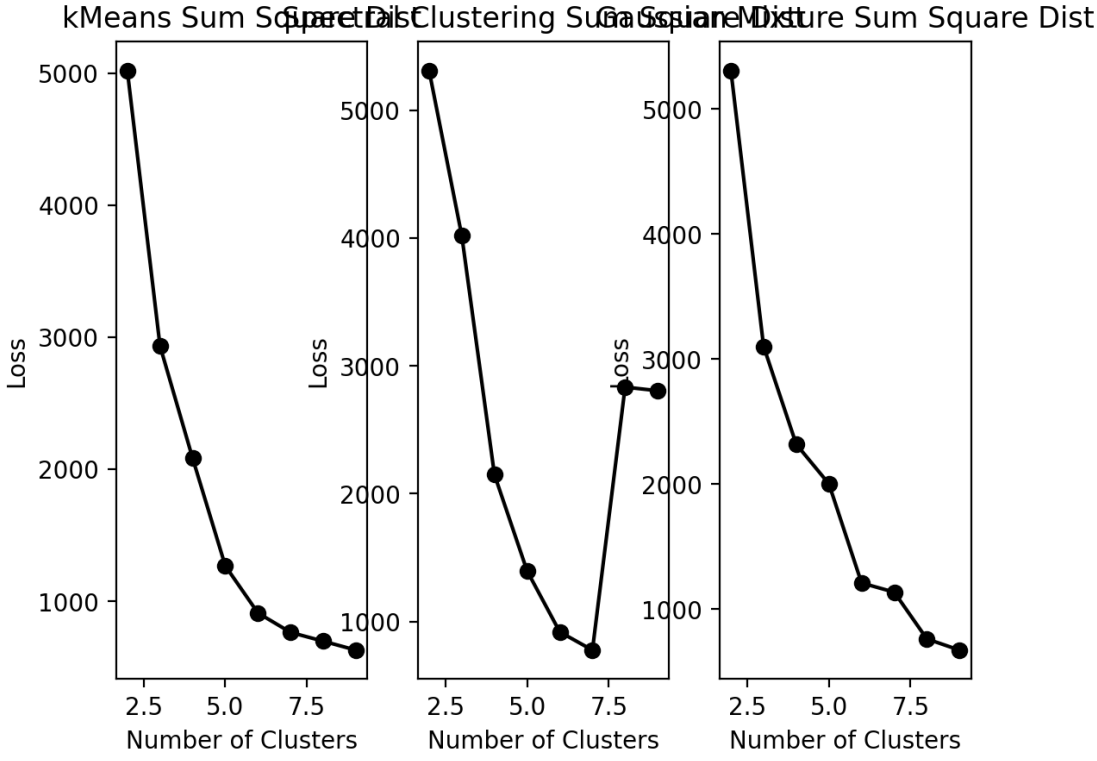

kMeans (n_clust = 5)
Spectral Clustering (n_clust = 5)
Gaussian Mixture (n_clust = 5)
4
2
0
−2
−4
0
5


kMeans (n_clust = 6)
Spectral Clustering (n_clust = 6)
Gaussian Mixture (n_clust = 6)
4
2
0
−2
−4
0
5


kMeans (n_clust = 7)
Spectral Clustering (n_clust = 7)
Gaussian Mixture (n_clust = 7)
4
2
0
−2
−4
0
5


kMeans (n_clust = 8)
Spectral Clustering (n_clust = 8)
Gaussian Mixture (n_clust = 8)
4
2
0
−2
−4
0
5


kMeans (n_clust = 9)
Spectral Clustering (n_clust = 9)
Gaussian Mixture (n_clust = 9)


kMeans Sum Square Dist
Spectral Clustering Sum Square Dist
Gaussian Mixture Sum Square Dist
Loss
Number of Clusters

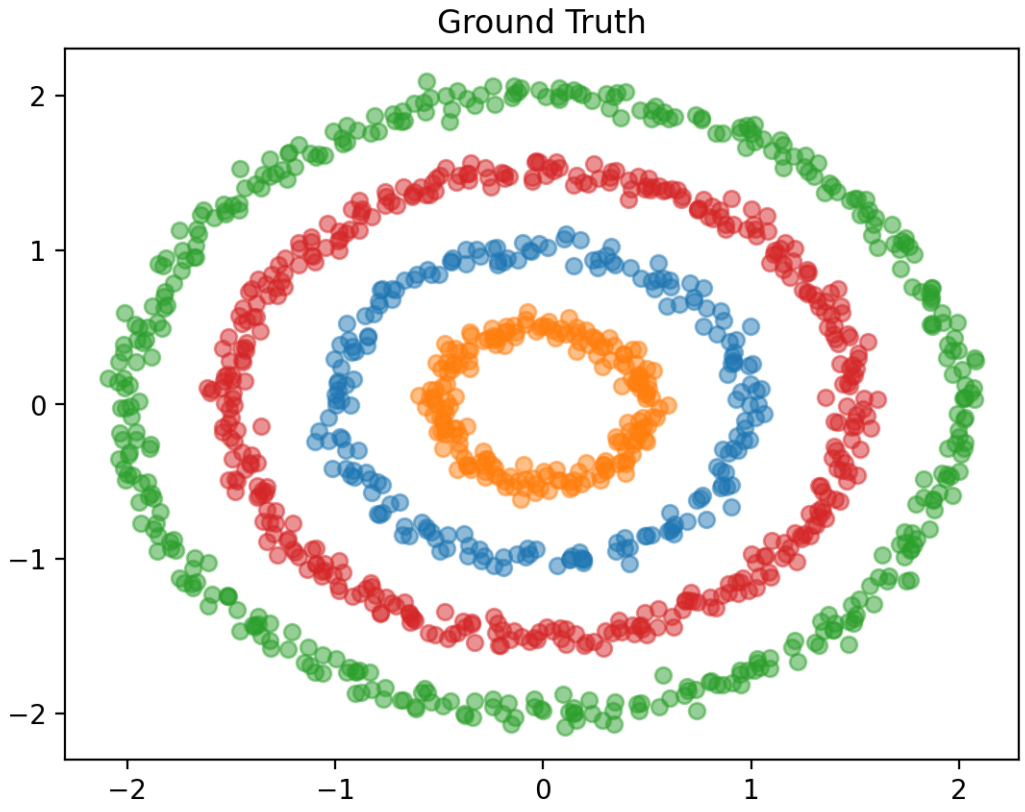
0.3 Concentric circles dataset

Visualize the "blob" dataset generated below, using a unique color for each cluster of points, where y contains the label of each corresponding point in x.

[ ]:
```
## DO NOT MODIFY
x1, y1 = make_circles(n_samples = 400, noise = 0.05, factor = 0.5, random_state␣
↪= 0)
x2, y2 = make_circles(n_samples = 800, noise = 0.025, factor = 0.75,␣
↪random_state = 1)

x = np.vstack([x1, x2*2])
y = np.hstack([y1, y2+2])
```

[ ]:
```
fig, ax = plt.subplots()
plot_pred(x, y, ax, "Ground Truth")
```

[ ]:
```
<Axes: title={'center': 'Ground Truth'}>
```

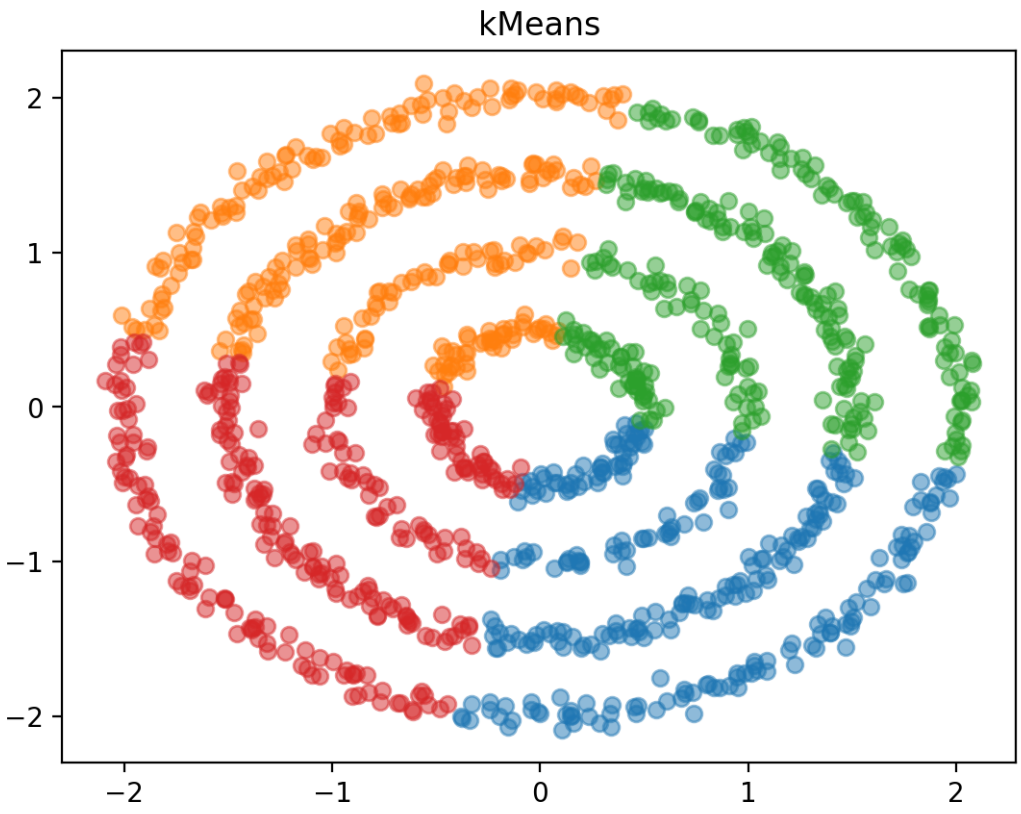


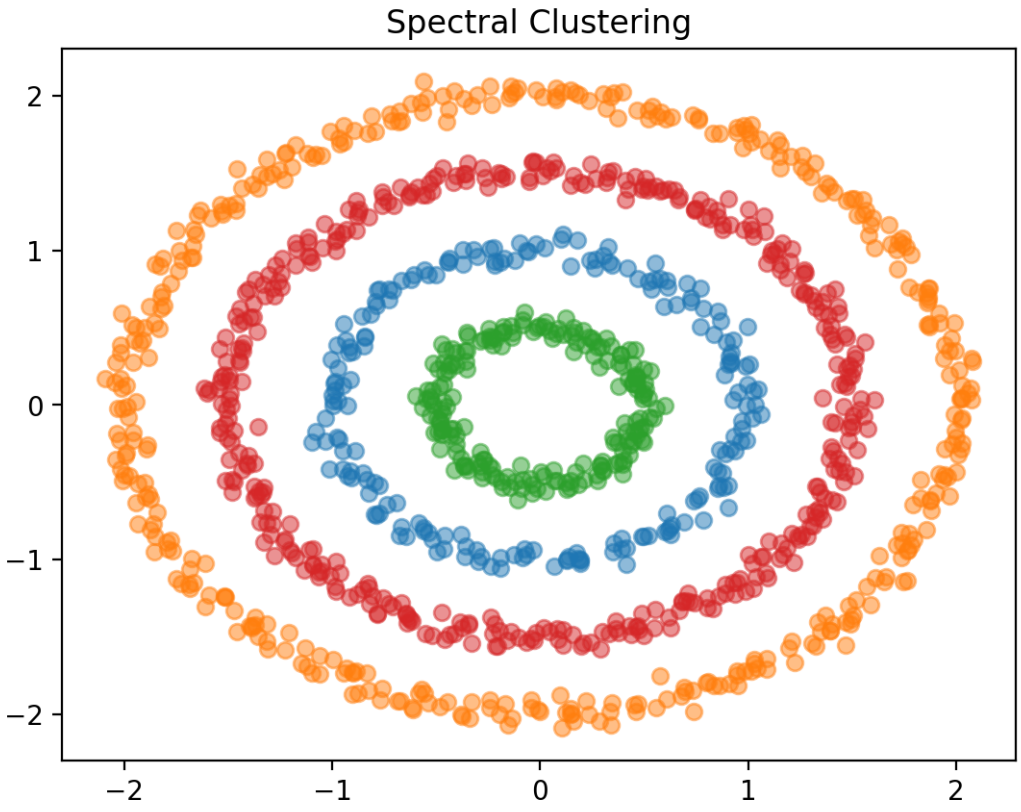
Use the `sklearn` KMeans, Spectral Clustering, and Gaussian Mixture Model functions to cluster the concentric circle data with 4 clusters, and attempt to modify the parameters until you get satisfactory results. Note: you should get good clustering results with Spectral Clustering, but the KMeans and GMM models will struggle to cluster this dataset well. Plot the results of your three models side-by-side using `plt.subplots` and the provided `plot_pred(x, labels, ax, title)` function.

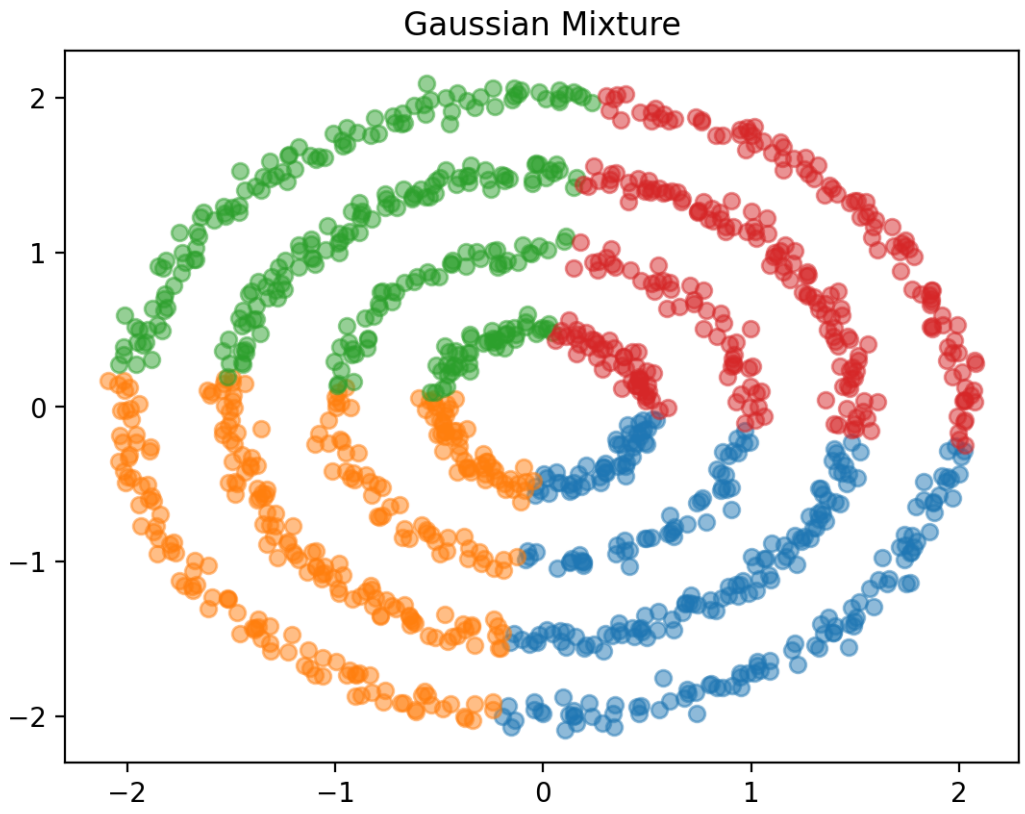
[ ]:
```
fig, ax = plt.subplots()
model1 = KMeans(n_clusters=4).fit(x)
plot_pred(x, model1.labels_, ax, "kMeans")

fig, ax = plt.subplots()
model2 = SpectralClustering(n_clusters=4, affinity='nearest_neighbors',␣
 ↪n_neighbors=15).fit(x)
plot_pred(x, model2.labels_, ax, "Spectral Clustering")

fig, ax = plt.subplots()
model3 = GaussianMixture(n_components=4, covariance_type='spherical').fit(x)
plot_pred(x, model3.predict(x), ax, "Gaussian Mixture")
```

```
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-
packages/sklearn/manifold/_spectral_embedding.py:273: UserWarning: Graph is not
fully connected, spectral embedding may not work as expected.
  warnings.warn(
```

[ ]:
```
<Axes: title={'center': 'Gaussian Mixture'}>
```


kMeans
2
1
0
−1
−2
−2
−1
0
1
2


Spectral Clustering
2
1
0
−1
−2
−2
−1
0
1
2



Using the parameters you found for the three models above, run each of the clustering algorithms for `n_clust = [2,3,4,5,6,7,8,9]` and compute the sum of squared distances loss for each case using the provided `compute_loss(x, labels)` function, where labels is the cluster assigned to each point by the algorithm. Plot loss versus number of cluster for each your three models in side-by-side subplots using the provided `plot_pred(x, labels, ax, title)` function.

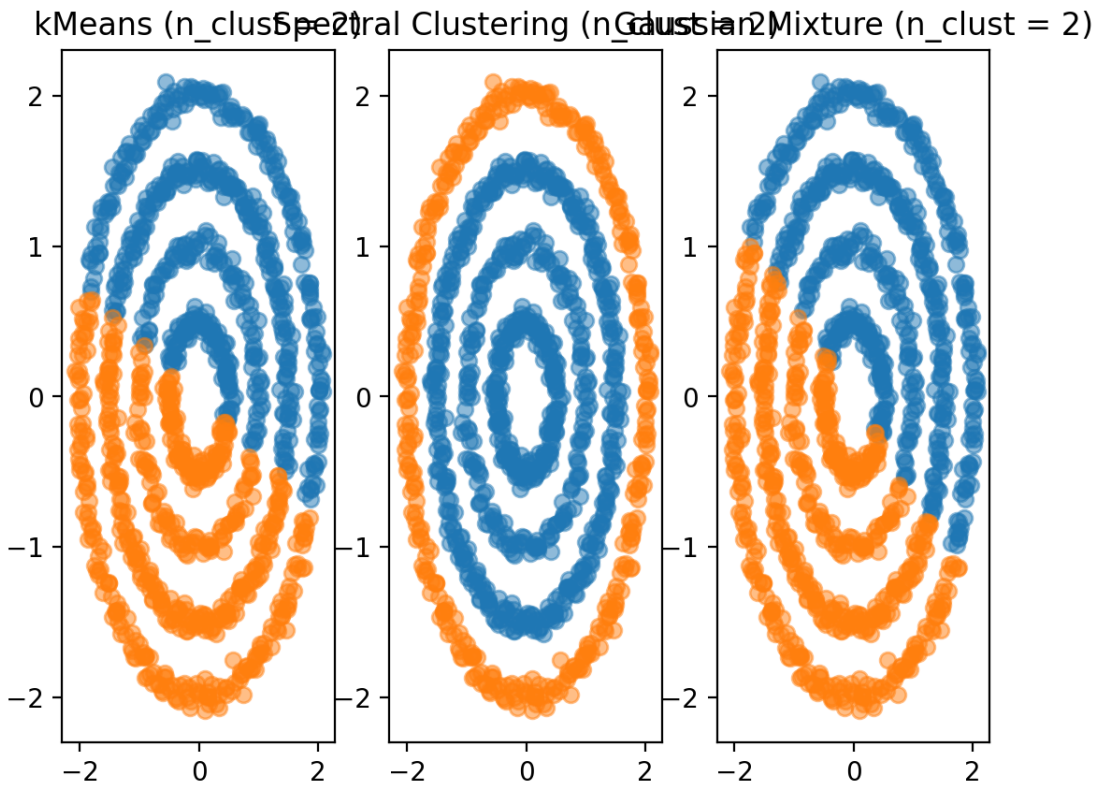
[ ]:
```
model1_sum_square_dist = []
model2_sum_square_dist = []
model3_sum_square_dist = []

for n_clust in [2, 3, 4, 5, 6, 7, 8, 9]:
    #Train models
    model1 = KMeans(n_clusters=n_clust).fit(x)
    model2 = SpectralClustering(n_clusters=n_clust,␣
 ↪affinity='nearest_neighbors', n_neighbors=10).fit(x)
    model3 = GaussianMixture(n_components=n_clust, covariance_type='spherical').
 ↪fit(x)

    model1_sum_square_dist.append(compute_loss(x, model1.labels_))
    model2_sum_square_dist.append(compute_loss(x, model2.labels_))
    model3_sum_square_dist.append(compute_loss(x, model3.predict(x)))
```

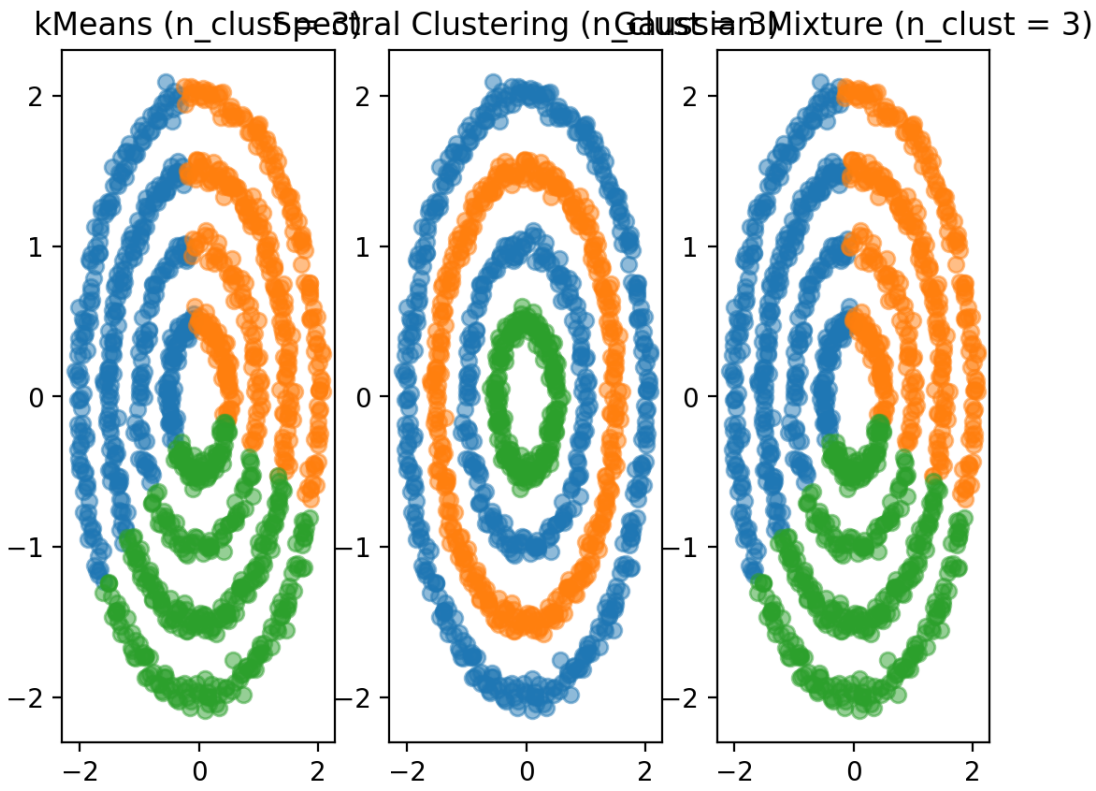
```python
    fig, ax = plt.subplots(1,3)
    plot_pred(x, model1.labels_, ax[0], f"kMeans (n_clust = {n_clust})")
    plot_pred(x, model2.labels_, ax[1], f"Spectral Clustering (n_clust =␣
↪{n_clust})")
    plot_pred(x, model3.predict(x), ax[2], f"Gaussian Mixture (n_clust =␣
↪{n_clust})")

fig, ax = plt.subplots(1,3)
plot_loss(model1_sum_square_dist, ax[0], "kMeans Sum Square Dist")
plot_loss(model2_sum_square_dist, ax[1], "Spectral Clustering Sum Square Dist")
plot_loss(model3_sum_square_dist, ax[2], "Gaussian Mixture Sum Square Dist")
```

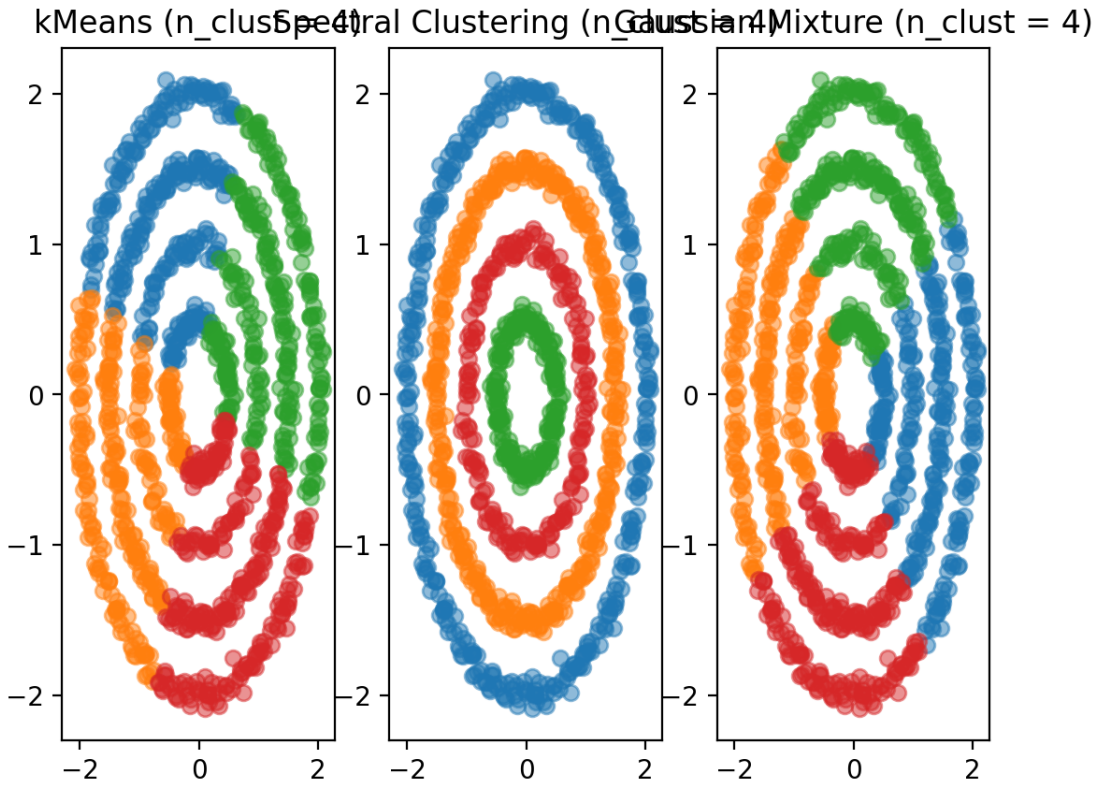
```
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-
packages/sklearn/manifold/_spectral_embedding.py:273: UserWarning: Graph is not
fully connected, spectral embedding may not work as expected.
  warnings.warn(
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-
packages/sklearn/manifold/_spectral_embedding.py:273: UserWarning: Graph is not
fully connected, spectral embedding may not work as expected.
  warnings.warn(
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-
packages/sklearn/manifold/_spectral_embedding.py:273: UserWarning: Graph is not
fully connected, spectral embedding may not work as expected.
  warnings.warn(
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-
packages/sklearn/manifold/_spectral_embedding.py:273: UserWarning: Graph is not
fully connected, spectral embedding may not work as expected.
  warnings.warn(
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
```

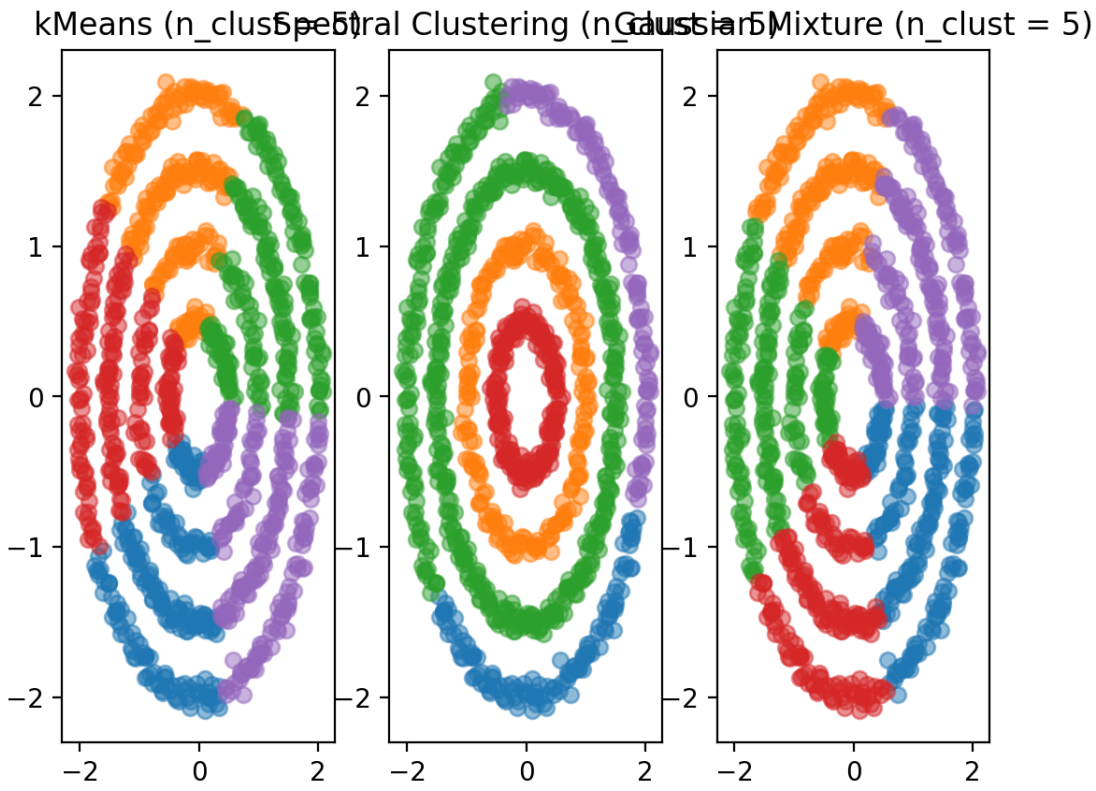
```
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-
packages/sklearn/manifold/_spectral_embedding.py:273: UserWarning: Graph is not
fully connected, spectral embedding may not work as expected.
  warnings.warn(
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-
packages/sklearn/manifold/_spectral_embedding.py:273: UserWarning: Graph is not
fully connected, spectral embedding may not work as expected.
  warnings.warn(
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-
packages/sklearn/manifold/_spectral_embedding.py:273: UserWarning: Graph is not
fully connected, spectral embedding may not work as expected.
  warnings.warn(
/opt/miniconda3/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:1412:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/opt/miniconda3/lib/python3.8/site-
packages/sklearn/manifold/_spectral_embedding.py:273: UserWarning: Graph is not
fully connected, spectral embedding may not work as expected.
  warnings.warn(
```

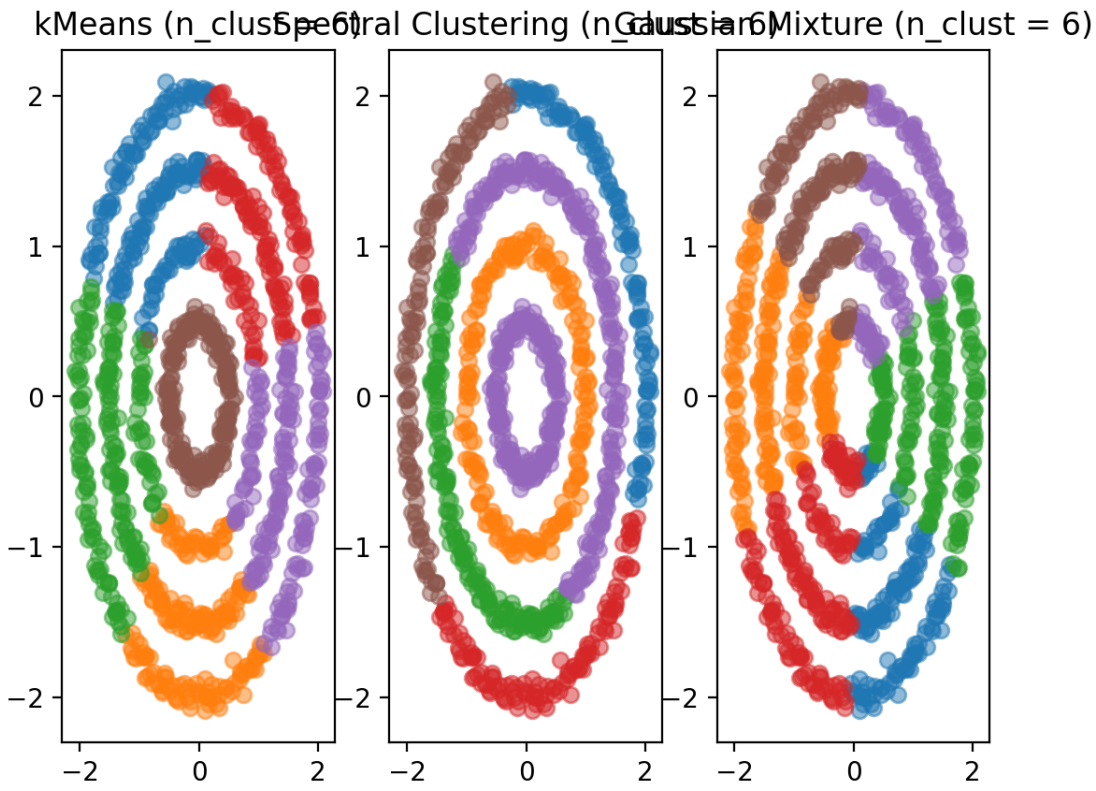
```
[ ]: <Axes: title={'center': 'Gaussian Mixture Sum Square Dist'}, xlabel='Number of
     Clusters', ylabel='Loss'>
```

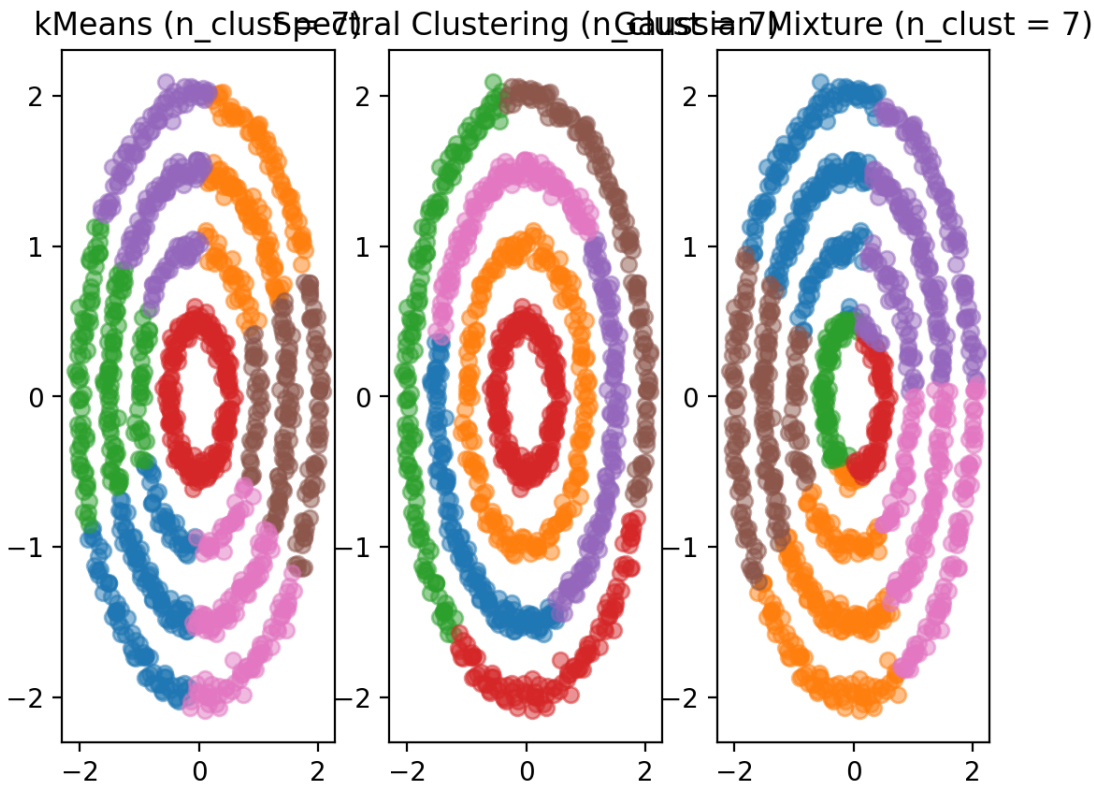

kMeans (n_clust = 2)
Spectral Clustering (n_clust = 2)
Gaussian Mixture (n_clust = 2)
2
1
0
−1
−2
−2
0
2

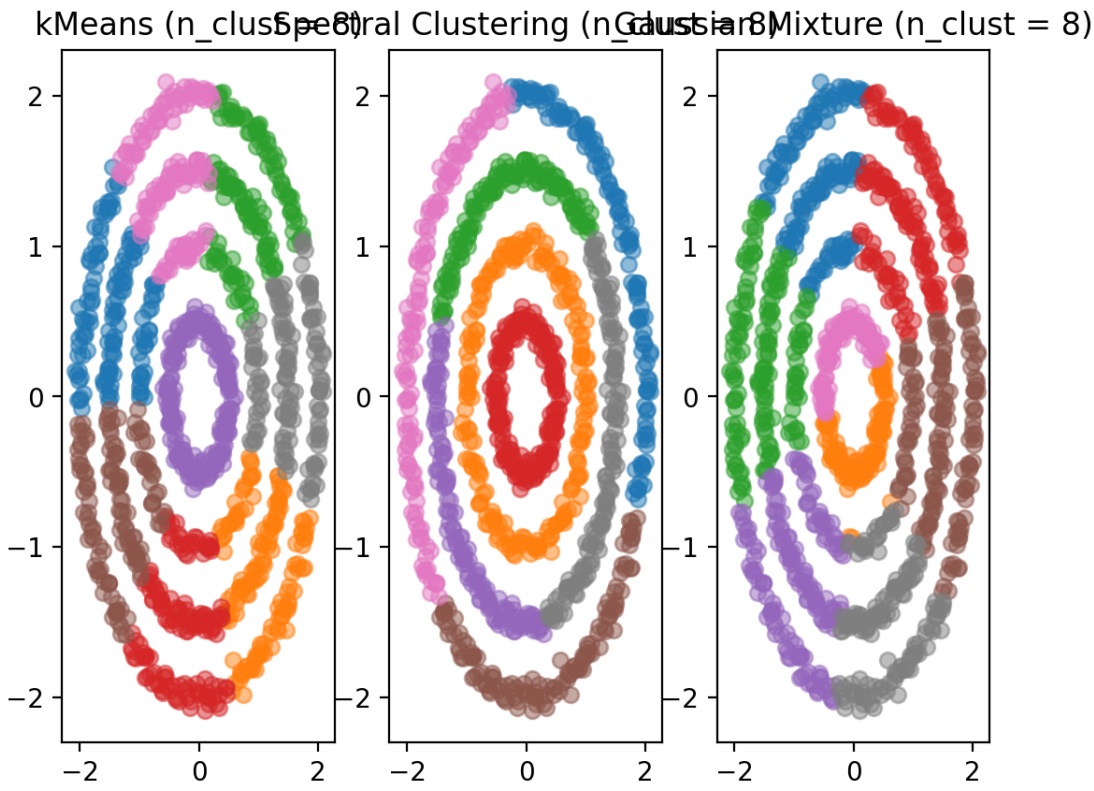

kMeans (n_clust = 3)
Spectral Clustering (n_clust = 3)
Gaussian Mixture (n_clust = 3)
2
1
0
−1
−2
−2
0
2

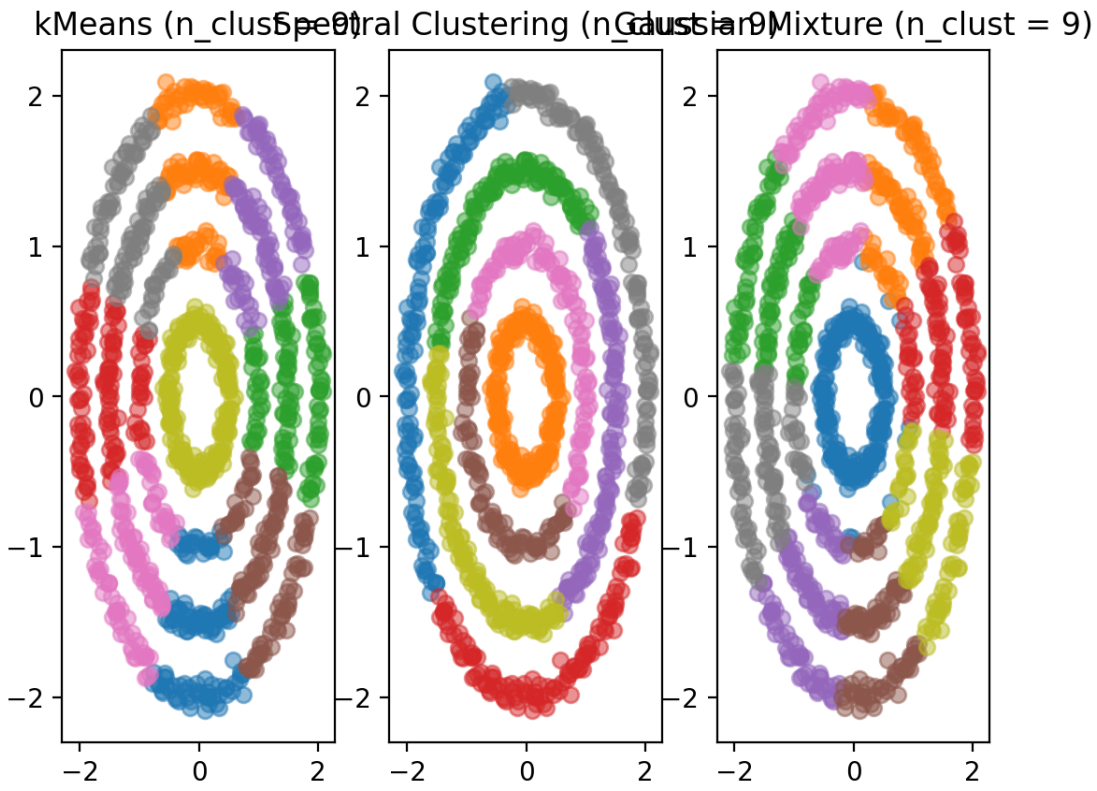

kMeans (n_clust = 4)
Spectral Clustering (n_clust = 4)
Gaussian Mixture (n_clust = 4)
2
1
0
−1
−2
−2
0
2

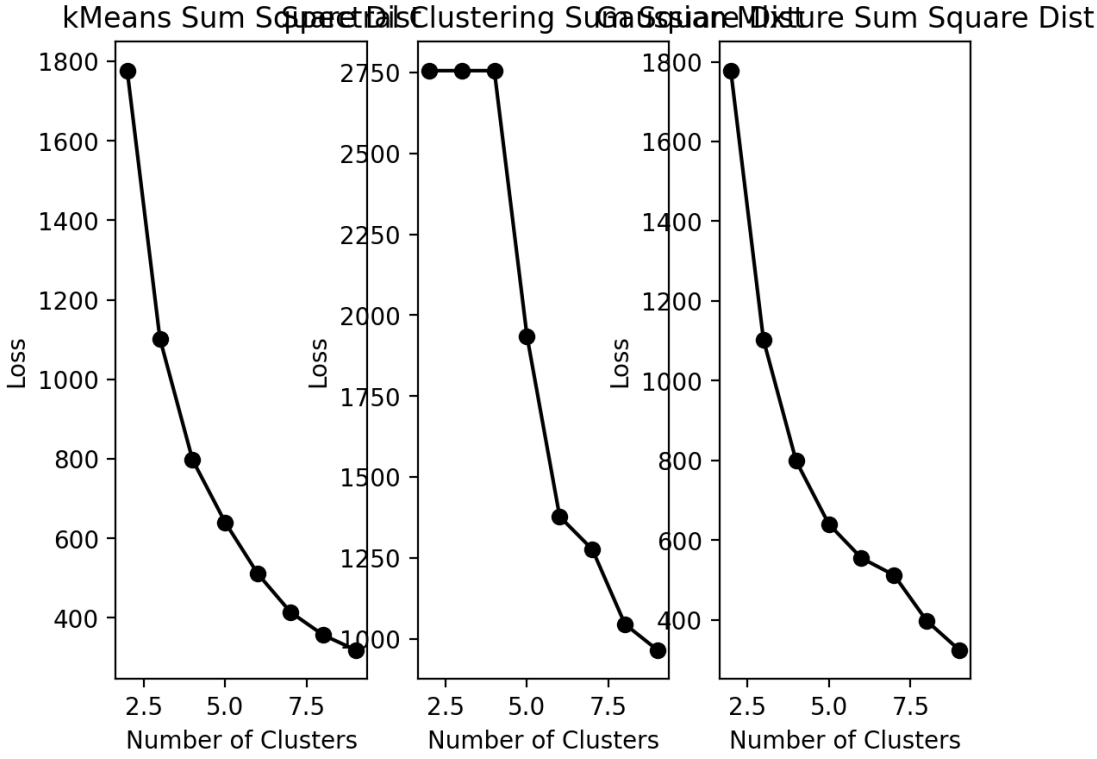

kMeans (n_clust = 5)
Spectral Clustering (n_clust = 5)
Gaussian Mixture (n_clust = 5)
2
1
0
−1
−2
−2
0
2


kMeans (n_clust = 6)
Spectral Clustering (n_clust = 6)
Gaussian Mixture (n_clust = 6)
2
1
0
−1
−2
−2
0
2


kMeans (n_clust = 7)
Spectral Clustering (n_clust = 7)
Gaussian Mixture (n_clust = 7)
2
1
0
−1
−2
−2
0
2


kMeans (n_clust = 8)
Spectral Clustering (n_clust = 8)
Gaussian Mixture (n_clust = 8)
2
1
0
−1
−2
−2
0
2


kMeans (n_clust = 9)
Spectral Clustering (n_clust = 9)
Gaussian Mixture (n_clust = 9)


kMeans Sum Square Dist
Spectral Clustering Sum Square Dist
Gaussian Mixture Sum Square Dist
Loss
Number of Clusters

0.4 Discussion

1. Discuss the performance of the clustering algorithms on the "blob" dataset. Using the elbow method, were you able to identify the number of natural clusters in the dataset for each of the methods? Does the elbow method work better for some algorithms versus others?

The elbow method was only able to be used on the spectral clustering model. For the kMeans and the Gaussian mixture models, the loss kept decreasing as the number of clusters increased because it was just over fitting the data. This is only able to be viewed when looking at the loss of each model.

2. Discuss the performance of the clustering algorithms on the concentric circles dataset. Using the elbow method, were you able to identify the number of natural clusters in the dataset for each of the methods?

The elbow method was not useful at all for determing the number of clusters in the concentric dataset. In terms of performance n the data, the spectral clustering algorithm performed the best, being the only one to accurately fit to the ring shape at the best number of clsuters (4). The other methods weren't able to separate these rings out from one another andn instead blended them together.

3. Does the sum of squared distances work well as a loss function for each of the three clustering algorithms we implemented? Does the sum of squared distance fail on certain types of clusters?

The sum of squares method works well when it comes to the blob datasets but not when it comes to the concentric datasets. In terms of the clustering algorithms, it doesn't seem to have a great advantage for any of the algorithms used.