[HW9] Problem 5: Regularized and Kernel k-Means

Part (e)

For this part, we will try to gain more intuition on kernel k-means by running it on real examples and visualizing it. There is no code needed to be written, and you simply have to run the demo to find the "best" kernel and hyperparameters. Of course, you are welcome to add some code if it helps you automate the process.

In [1]:
```
%matplotlib inline

import numpy as np
import matplotlib.pyplot as plt
import sklearn

from sklearn.cluster import KMeans
from kernel_k_means import KernelKMeans, plot_clusters, load_dataset
```

We want to see how some kernels are more or less suitable for different datasets and structures we want to discover. In other words, given some initial guess of the data structure, we want to find the "best" kernel for k-means to clusters the data based on this structure. Typically, we use an unsupervised algorithm such as k-means to discover some underlying structures of the data which we do not have the ground truth information for. So "best" here is not clearly defined and is open to interpretation. Just use your own judgement based on the visualization.

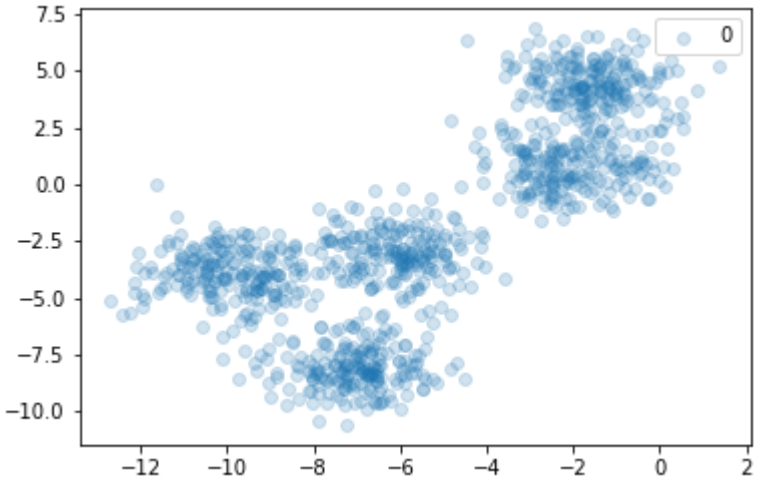
Follow the following steps and answer the questions (in bold). Then, repeat them for each of the three datasets (`'gaussian', 'circles', 'digits'`):

1. Load and plot without ground truth labels.
2. Guess the number of clusters. Run k-means with different kernels. Try different hyperparameters for each kernel.
3. **Report the number of clusters you choose, the best kernel and its hyperparameters.**
4. Now plot the data again with ground truth labels.
5. **Does your conclusion in 3 change when you see the ground truth labels? If it does, what is your new guess on the best kernel and hyperparameters?**

- Try `kernel` between `['linear', 'rbf', 'poly']` .
- Linear kernel has no hyperparameter.
- For RBF kernel, adjust `gamma` between 0.001 - 0.1.
- For polynomial kernel, you can fix `gamma` and `coef0` and only change `degree` between 2 - 4.
- Digits dataset is a dataset of grayscale images of digits from 0 to 9 where we reduce the dimension from 64 (8 by 8 pixels) to 2 by PCA for visualization and filter out some of the digits.

(1) Load and plot the data without ground truth labels.

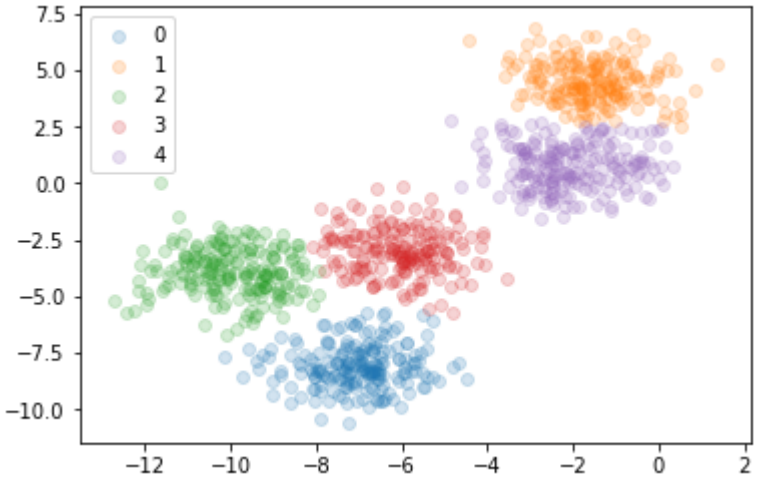
In [9]:
```
X, y = load_dataset('gaussian')
# X, y = load_dataset('circles')
# X, y = load_dataset('digits')

# Plot ground truth without ground truth label
plot_clusters(X, np.zeros(len(X)))
```



(2) Guess the number of clusters. Run k-means with different kernels. Try different hyperparameters for each kernel.

In [39]:
```
n_clusters = 5
kernel = 'polynomial'
gamma = .1   # for RBF kernel
degree = 1    # for polynomial kernel
```

In [40]:
```
kkm = KernelKMeans(n_clusters=n_clusters, max_iter=100, random_state=0,
                   kernel=kernel, gamma=gamma, degree=degree)
labels = kkm.fit_predict(X)
plot_clusters(X, labels)
```

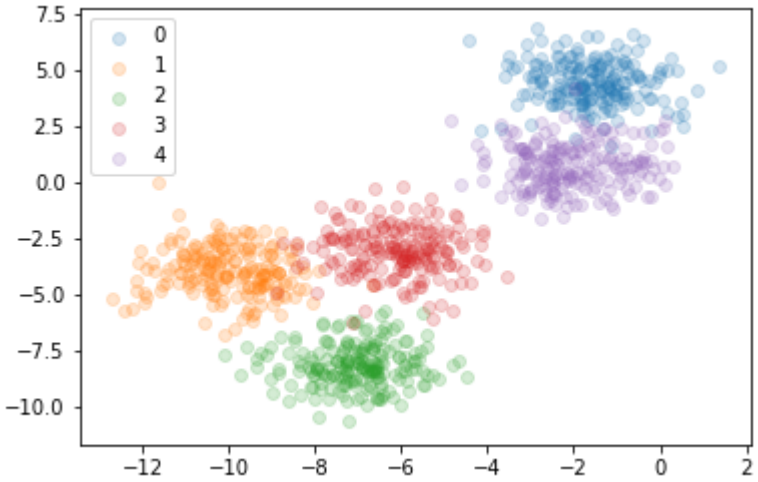


(3) **Report the number of clusters you choose (`n_cluster`), the best kernel (`kernel`) and its hyperparameters (`gamma` or `degree`).**

For 5 clusters, the RBF kernel performs best with gamma = 0.01, and the polynomial kernel performs best with degree 3

(4) Plot the data again with ground truth labels.

In [41]:
```
plot_clusters(X, y)
```



(5) **Does your conclusion in 3 change when you see the ground truth labels? If it does, what is your new guess on the best kernel and hyperparameters?**

Nope it doesn't change. Don't forget to repeat for all three datasets.

In [ ]:
```
```