kmeans

July 2, 2024

1 K-means Clustering Implementation

1.0.1 Imports

[85]:
```
import numpy as np
import matplotlib.pyplot as plt
import random
import math
```

1.0.2 Generating Test Dataset

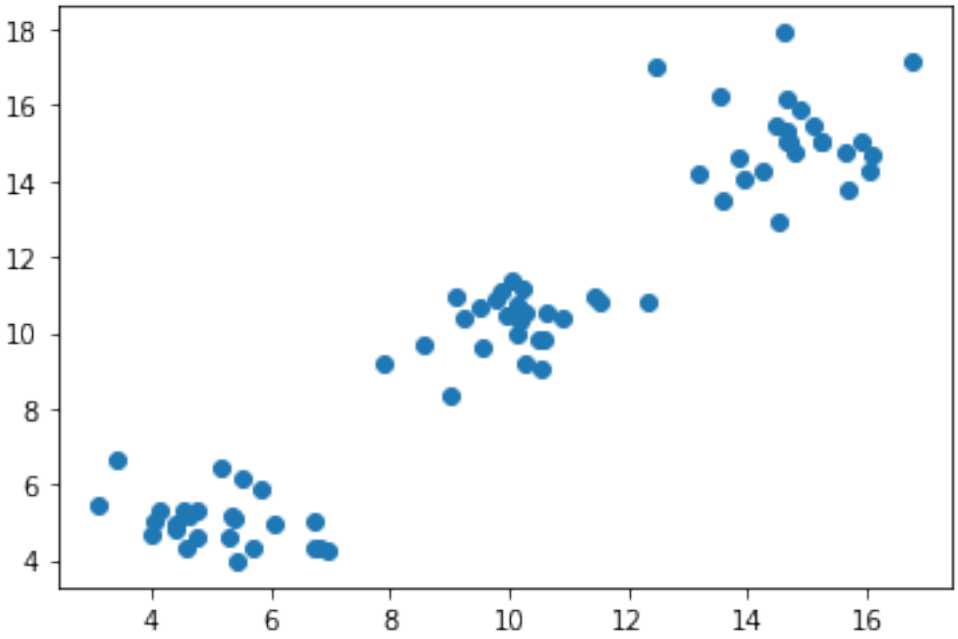
[86]:
```
data = []

data.extend([np.random.normal(5, 1, 2) for x in range(25)])
data.extend([np.random.normal(10, 1, 2) for x in range(25)])
data.extend([np.random.normal(15, 1, 2) for x in range(25)])
```

[87]:
```
x_data = [x[0] for x in data]
y_data = [x[1] for x in data]
```

[88]:
```
plt.scatter(x_data, y_data)
```

[88]:
```
<matplotlib.collections.PathCollection at 0x1c9e4a74820>
```



1.0.3 K-means Algorithm

[89]:
```
def kMeans(data, k, iterations):
    # Step 1: Select initial centroids
    centroids = [random.choice(data) for k in range(k)]
    for iteration in range(iterations):
        # Step 2: Assign points to closest centroid
        closest_centroid = [None] * len(data)
        for i in range(len(data)):
            shortest_dist = np.Inf
            for j in range(k):
                dist = math.dist(data[i], centroids[j])
                if dist < shortest_dist:
                    shortest_dist = dist
                    closest_centroid[i] = j
        # Step 3: Recompute centroids
        for i in range(k):
            centroids[i] = np.mean([data[x] for x in range(len(data)) if i ==␣
↪closest_centroid[x]], axis = 0)
    return centroids, closest_centroid
```

1.0.4 Test the Algorithm

[90]:
```
k = 3
iterations = 20

centroids, closest_centroid = kMeans(data, k, iterations)
```

[91]:
```
plot_setup = [None] * k

for i in range(k):
    plot_setup[i] = [data[x] for x in range(len(data)) if i ==␣
 ↪closest_centroid[x]]
```

[92]:
```
plt.scatter(list(zip(*plot_setup[0]))[0], list(zip(*plot_setup[0]))[1])
plt.scatter(list(zip(*plot_setup[1]))[0], list(zip(*plot_setup[1]))[1])
plt.scatter(list(zip(*plot_setup[2]))[0], list(zip(*plot_setup[2]))[1])
```

[92]:
```
<matplotlib.collections.PathCollection at 0x1c9e4603100>
```