

K-Means Clustering Algorithm

January 23, 2020

```
[1]: #Loading Packages
import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import MinMaxScaler
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

```
[2]: iris = pd.read_table("iris.data"
    ↪, sep=",", names=['Sepal_Length', 'Sepal_Width', 'Petal_Length', 'Petal_Width', 'Class'])
iris.head()
```

C:\Users\Trilo\Anaconda3\lib\site-packages\ipykernel_launcher.py:1:

FutureWarning: read_table is deprecated, use read_csv instead.

"""Entry point for launching an IPython kernel.

```
[2]:
```

	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
[3]: print(iris.describe())
```

	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

```
iris['Class'].value_counts()
```

```
Iris-versicolor      50
Iris-setosa          50
Iris-virginica       50
Name: Class, dtype: int64
```

```
x = np.array(iris.drop(['Class'], 1).astype(float))
y = np.array(iris['Class'])
y
```

[illegible]

```

'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
'Iris-virginica', 'Iris-virginica'], dtype=object)

```

```

[6]: kmeans = KMeans(n_clusters = 3, init = 'k-means++', max_iter = 300, n_init = 10,
    random_state = 0)
y_kmeans = kmeans.fit_predict(x)
y_kmeans

```

```

[6]: array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 2, 2, 2, 2, 0, 2, 2, 2,
    2, 2, 2, 0, 0, 2, 2, 2, 2, 0, 2, 0, 2, 0, 2, 2, 0, 0, 2, 2, 2, 2,
    2, 0, 2, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 0])

```

```

[7]: plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c = 'red', label='Iris-setosa')
plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c = 'blue', label='Iris-versicolour')
plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c = 'green', label='Iris-virginica')

#Plotting the centroids of the clusters
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 100, c = 'yellow', label = 'Centroids')

plt.legend()

```

```

[7]: <matplotlib.legend.Legend at 0x23749fb9588>

```

