In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
iris = sns.load_dataset('iris')
```

In [2]:

```
#pip install seaborn
```

In [3]:

```
print(iris.head()) #prints first 5 values
```

```
   sepal_length  sepal_width  petal_length  petal_width species
0           5.1          3.5           1.4          0.2  setosa
1           4.9          3.0           1.4          0.2  setosa
2           4.7          3.2           1.3          0.2  setosa
3           4.6          3.1           1.5          0.2  setosa
4           5.0          3.6           1.4          0.2  setosa
```

In [4]:

```
x = iris.iloc[:, [0,1,2,3]].values
```

In [5]:

```
# predicting with k = 5
kmeans5 = KMeans(n_clusters=5)
y_kmeans5 = kmeans5.fit_predict(x)
print(y_kmeans5)

kmeans5.cluster_centers_
```

```
[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 2 2 2 4 2 2 2 4 2 4 4 2 4 2 4 2 2 4 2 4 2 4 2 2
 2 2 2 2 2 4 4 4 4 2 4 2 2 2 4 4 4 2 4 4 4 4 4 2 4 4 0 2 3 0 0 3 4 3 0 3 0
 0 0 2 0 0 0 3 3 2 0 2 3 2 0 3 2 2 0 3 3 3 0 2 2 3 0 0 2 0 0 0 2 0 0 0 2 0
 0 2]
```

Out[5]:

```
array([[6.52916667, 3.05833333, 5.50833333, 2.1625    ],
       [5.006     , 3.428     , 1.462     , 0.246     ],
       [6.20769231, 2.85384615, 4.74615385, 1.56410256],
       [7.475     , 3.125     , 6.3       , 2.05      ],
       [5.508     , 2.6       , 3.908     , 1.204     ]])
```

In [6]:

```
# predicting with k = 4
kmeans4 = KMeans(n_clusters=4)
y_kmeans4 = kmeans4.fit_predict(x)
print(y_kmeans4)

kmeans4.cluster_centers_
```

```
[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 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 3 3 3 2 3 2 3 2 3 2 2 2 2 3 2 3 2 2 3 2 3 2 3 3
 3 3 3 3 3 2 2 2 2 3 2 3 3 3 2 2 2 3 2 2 2 2 2 3 2 2 1 3 1 1 1 1 2 1 1 1 3
 3 1 3 3 1 1 1 1 3 1 3 1 3 1 1 3 3 1 1 1 1 1 3 3 1 1 1 3 1 1 1 3 1 1 1 3 3
 1 3]
```

Out[6]:

```
array([[5.006     , 3.428     , 1.462     , 0.246     ],
       [6.9125    , 3.1       , 5.846875  , 2.13125   ],
```

```
       [5.53214286, 2.63571429, 3.96071429, 1.22857143],
       [6.2525    , 2.855     , 4.815     , 1.625     ]])
```

In [7]:

```python
# predicting with k = 3
kmeans3 = KMeans(n_clusters=3)
y_kmeans3 = kmeans3.fit_predict(x)
print(y_kmeans3)

kmeans3.cluster_centers_
```

```
[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 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]
```

Out[7]:

```
array([[5.9016129 , 2.7483871 , 4.39354839, 1.43387097],
       [5.006     , 3.428     , 1.462     , 0.246     ],
       [6.85      , 3.07368421, 5.74210526, 2.07105263]])
```

In [8]:

```python
# predicting with k = 2
kmeans2 = KMeans(n_clusters=2)
y_kmeans2 = kmeans2.fit_predict(x)
print(y_kmeans2)

kmeans2.cluster_centers_
```

```
[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 0 0 0 0 0 1 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 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 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 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0]
```

Out[8]:

```
array([[6.30103093, 2.88659794, 4.95876289, 1.69587629],
       [5.00566038, 3.36981132, 1.56037736, 0.29056604]])
```

In [9]:

```python
# predicting with k = 1
kmeans1 = KMeans(n_clusters=1)
y_kmeans1 = kmeans1.fit_predict(x)
print(y_kmeans1)

kmeans1.cluster_centers_
```

```
[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 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 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 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 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 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0]
```

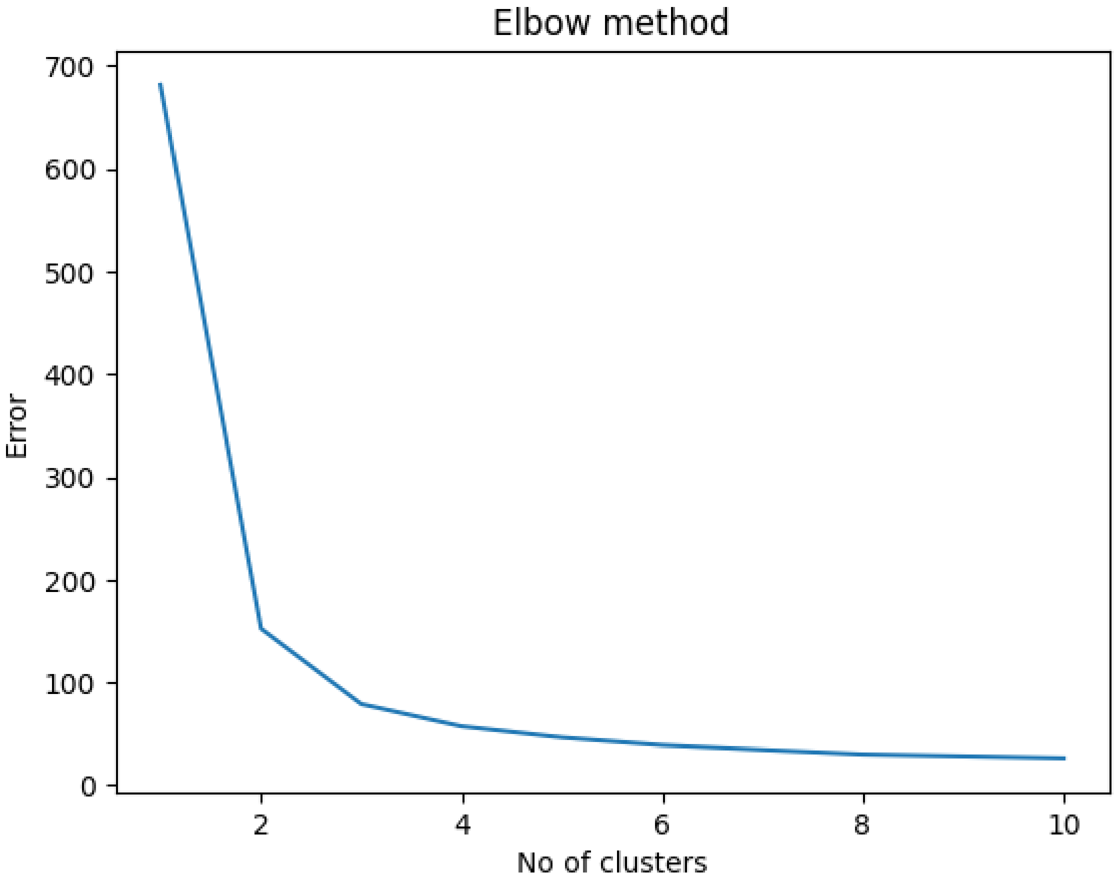
Out[9]:

```
array([[5.84333333, 3.05733333, 3.758     , 1.19933333]])
```

In [10]:

```python
# THis is the elbow method to determine the best value for K
Error =[]
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i).fit(x)
    kmeans.fit(x)
    Error.append(kmeans.inertia_)
plt.plot(range(1, 11), Error)
plt.title('Elbow method')
plt.xlabel('No of clusters')
```

```
plt.ylabel('Error')
plt.show()

# as we can see below the elbow is at 3. It formed an elbow at k = 3
```



In [11]:

```
# Visualizing the K-means = 1 and 2

# setting the size
fig=plt.figure(figsize=(12,4))

# positionining the plots
ax1 = fig.add_subplot(121)
ax2 = fig.add_subplot(122)


# creating the scatter plots
ax1.scatter(x[:,0],x[:,1], c=y_kmeans1,cmap='rainbow')
ax2.scatter(x[:,0],x[:,1], c=y_kmeans2,cmap='rainbow')

# setting the titles
ax1.set_title('Kmeans=1')
ax2.set_title('Kmeans=2')
```

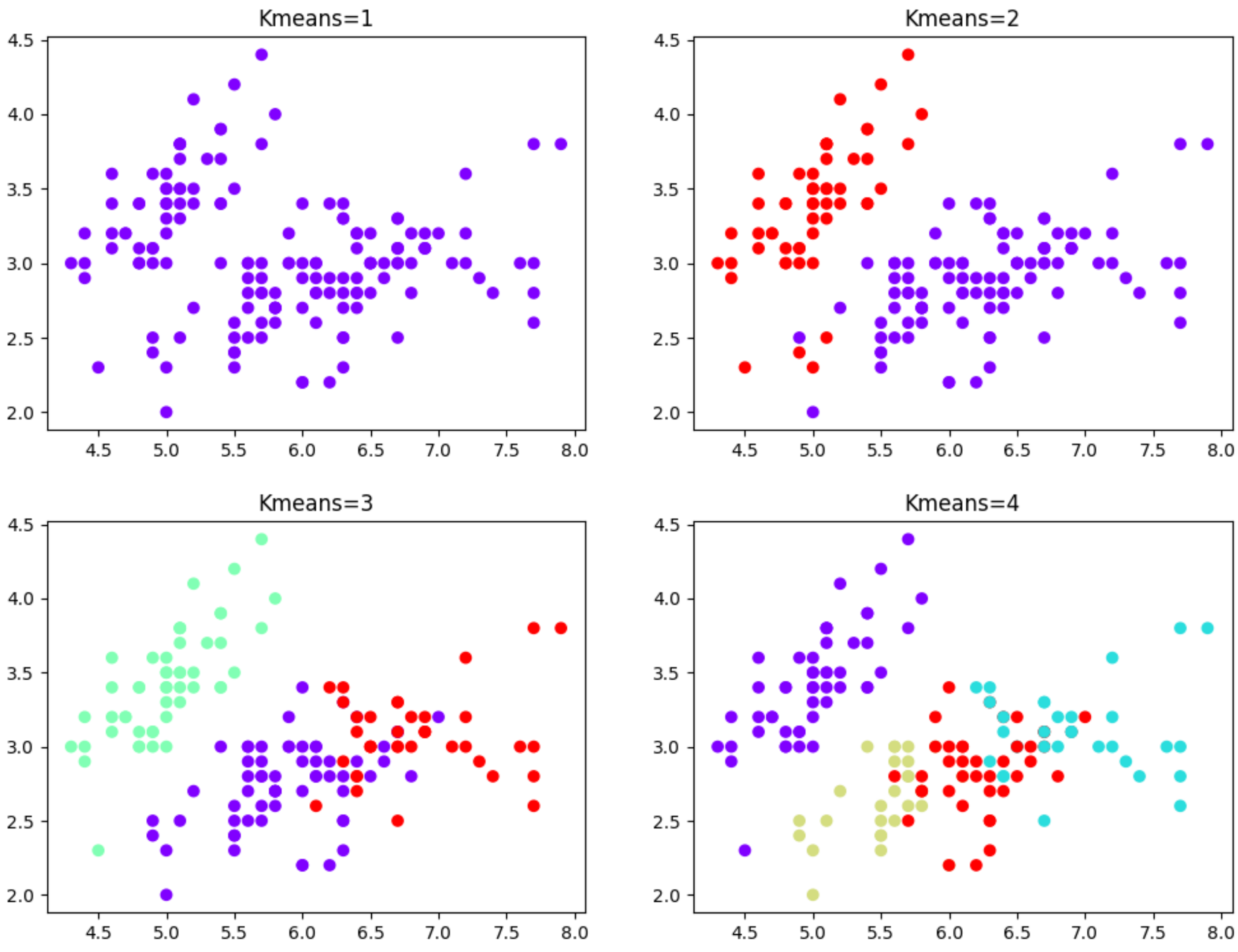
Out[11]:

```
Text(0.5, 1.0, 'Kmeans=2')
```

In [12]:

```
# Visualizing the K-means = 3 and 4

# setting the size
fig=plt.figure(figsize=(12,4))
```

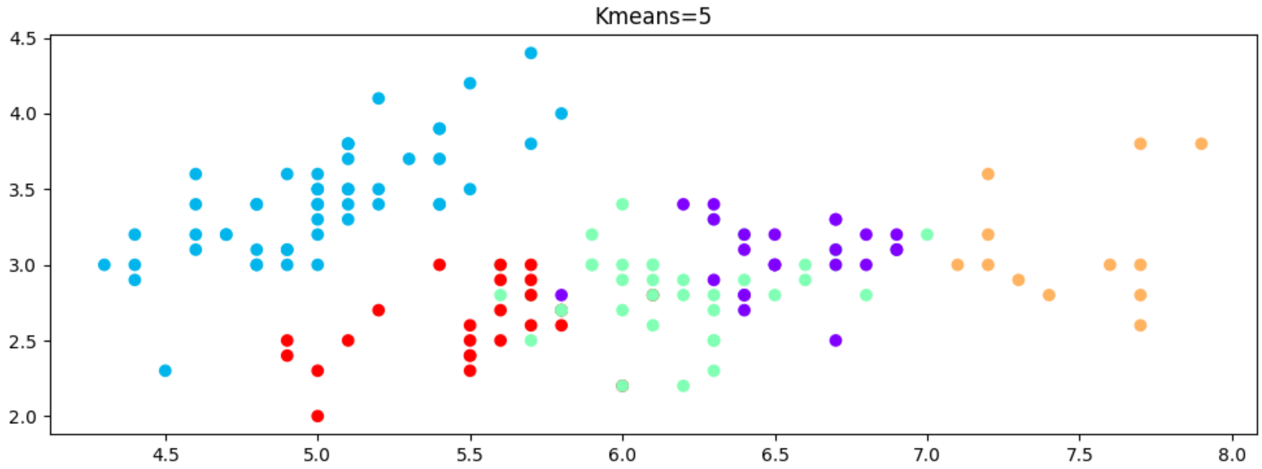
```
# positionining the plots
ax3 = fig.add_subplot(121)
ax4 = fig.add_subplot(122)


# creating the scatter plots
ax3.scatter(x[:,0],x[:,1], c=y_kmeans3,cmap='rainbow')
ax4.scatter(x[:,0],x[:,1], c=y_kmeans4,cmap='rainbow')

# setting the titles
ax3.set_title('Kmeans=3')
ax4.set_title('Kmeans=4')
plt.show()
```



In [13]:

```
# Visualizing the K-means = 5

# setting the size
fig=plt.figure(figsize=(12,4))

# positionining the plots
ax5 = fig.add_subplot()


# creating the scatter plots
ax5.scatter(x[:,0],x[:,1], c=y_kmeans5,cmap='rainbow')

# setting the titles
```

```
ax5.set_title('Kmeans=5')
plt.show()
```



In [14]:

```
species = {"setosa": 0, "versicolor": 1, "virginica": 2}

irisdf = iris.copy()

irisdf["species"] = irisdf["species"].map(species)

irisdf
```

Out[14]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

150 rows × 5 columns

In [15]:

```
# Number of clusters
kmeans = KMeans(n_clusters=3)
# Fitting the input data
kmeans = kmeans.fit(x)
# Getting the cluster labels
labels = kmeans.predict(x)
# Centroid values
centroids = kmeans.cluster_centers_
```

In [16]:

```
from sklearn.metrics import classification_report

target_names = ['setosa', 'versicolor', 'virginica']


print(classification_report(irisdf['species'],kmeans.labels_,target_names=target
```

```
              precision    recall  f1-score   support

      setosa       0.00      0.00      0.00        50
  versicolor       0.00      0.00      0.00        50
   virginica       0.23      0.28      0.25        50

    accuracy                           0.09       150
   macro avg       0.08      0.09      0.08       150
weighted avg       0.08      0.09      0.08       150
```

In [ ]: