iris cluster sse

May 11, 2023

[39]:
```
from sklearn.cluster import KMeans
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from matplotlib import pyplot as plt
%matplotlib inline
```

[40]:
```
from sklearn.datasets import load_iris
```

[41]:
```
iris=load_iris()
```

[42]:
```
dir(iris)
```

[42]:
```
['DESCR',
 'data',
 'data_module',
 'feature_names',
 'filename',
 'frame',
 'target',
 'target_names']
```

[43]:
```
df=pd.DataFrame(iris.data,columns=iris.feature_names)
df.head()
```

[43]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

[44]:
```
df1=df.drop(columns=['sepal length (cm)','sepal width (cm)'])

df.drop(['sepal length (cm)', 'sepal width (cm)'],axis='columns',inplace=True)
#x=df.drop(columns=['target','species'])
```

[45]:
```
df1.head(4)
```

[45]:
```
   petal length (cm)  petal width (cm)
0                1.4               0.2
1                1.4               0.2
2                1.3               0.2
3                1.5               0.2
```

[ ]:
```
plt.scatter(petal length (cm),)
```

[48]:
```
km=KMeans(n_clusters=3)
yp=km.fit_predict(df1)
yp
```

[48]:
```
array([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, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```

[52]:
```
df1['cluster']=yp
df1.head(3)
```

[52]:
```
   petal length (cm)  petal width (cm)  cluster
0                1.4               0.2        0
1                1.4               0.2        0
2                1.3               0.2        0
```

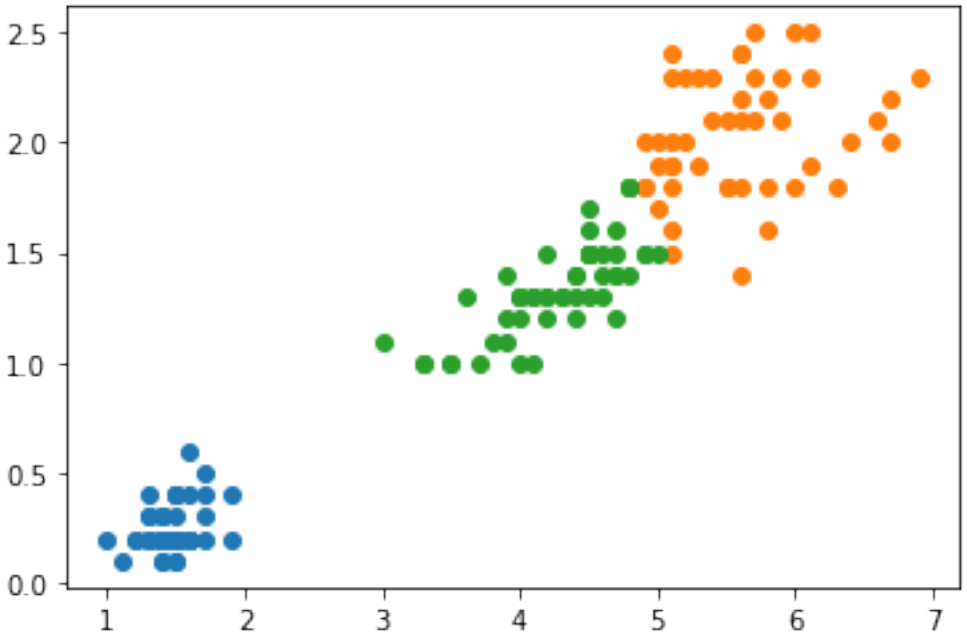
[51]:
```
df1.cluster.unique()
```

[51]:
```
array([0, 2, 1])
```

[55]:
```
d1=df1[df1.cluster==0]
d2=df1[df1.cluster==1]
d3=df1[df1.cluster==2]
```

[56]:
```
plt.scatter(d1['petal length (cm)'],d1['petal width (cm)'])
plt.scatter(d2['petal length (cm)'],d2['petal width (cm)'])
plt.scatter(d3['petal length (cm)'],d3['petal width (cm)'])
```

[56]:
```
<matplotlib.collections.PathCollection at 0x297f72b1940>
```

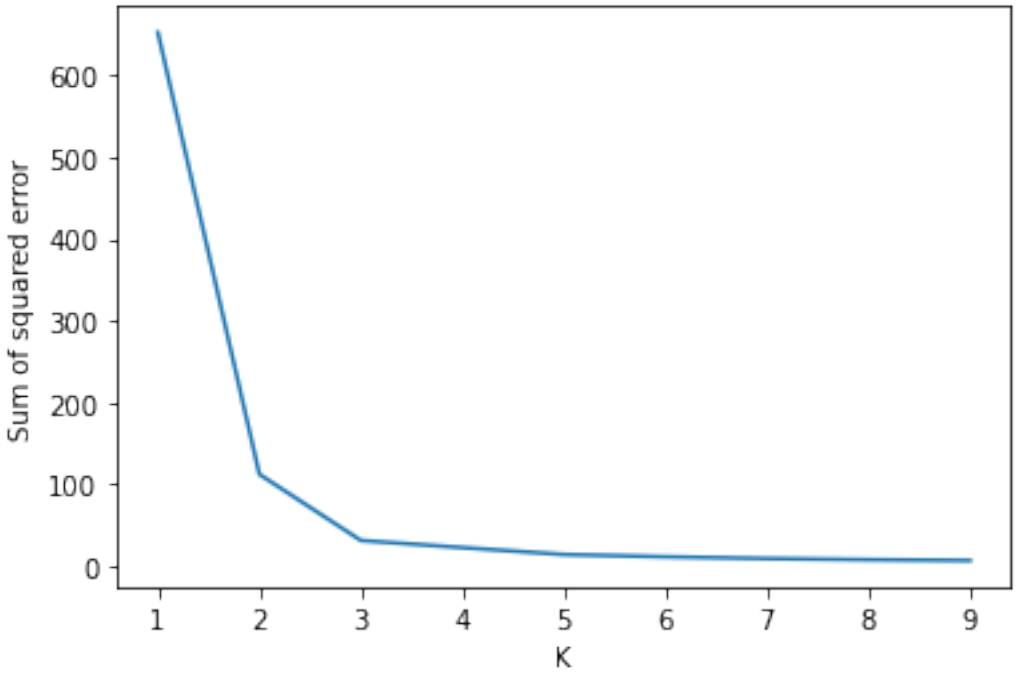


[59]:
```
sse=[]
krange=range(1,10)
for k in krange:
    km = KMeans(n_clusters=k)
    k=km.fit(df1)
    sse.append(km.inertia_)
```

```
C:\Users\Rakesh\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1036:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

[61]:
```
plt.xlabel('K')
plt.ylabel('Sum of squared error')
plt.plot(krange,sse)
```

[61]:
```
[<matplotlib.lines.Line2D at 0x297f7867730>]
```


Sum of squared error
600
500
400
300
200
100
0
1
2
3
4
5
6
7
8
9
K

[ ]: