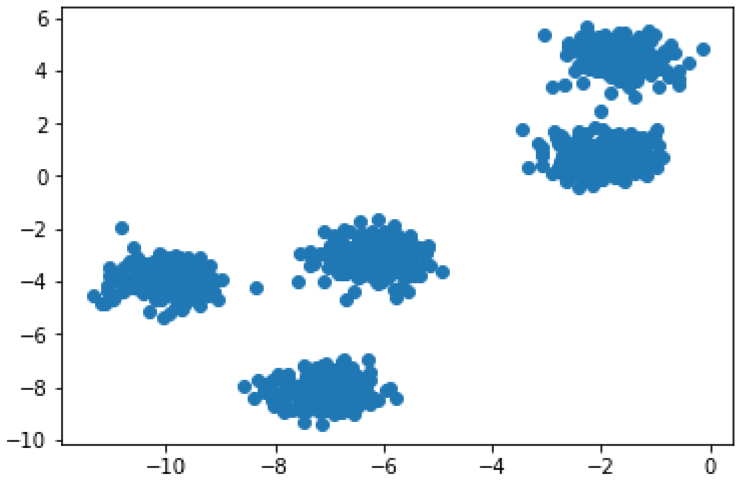
In [1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs
```

In [17]:
```python
X,y = make_blobs(n_samples = 1000,centers = 5,cluster_std=0.5,random_state = 1)
```

In [18]:
```python
plt.scatter(X[:,0],X[:,1])
```

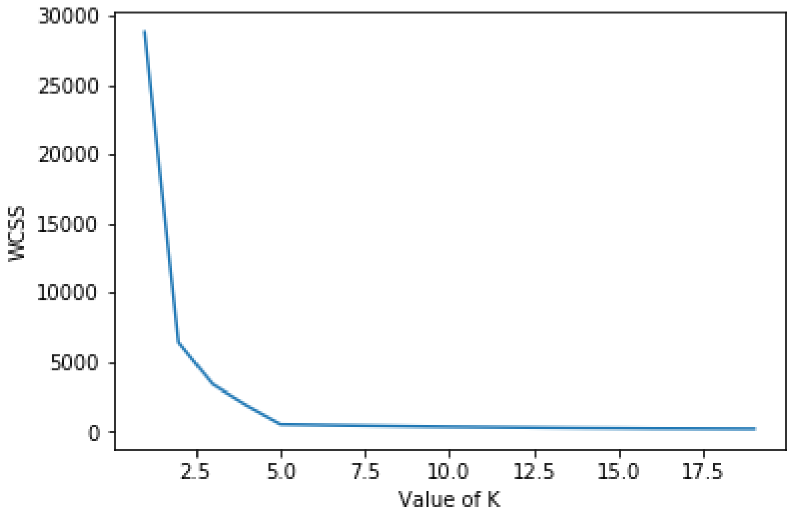
Out[18]:
```
<matplotlib.collections.PathCollection at 0x22068e10988>
```



Elbow Method

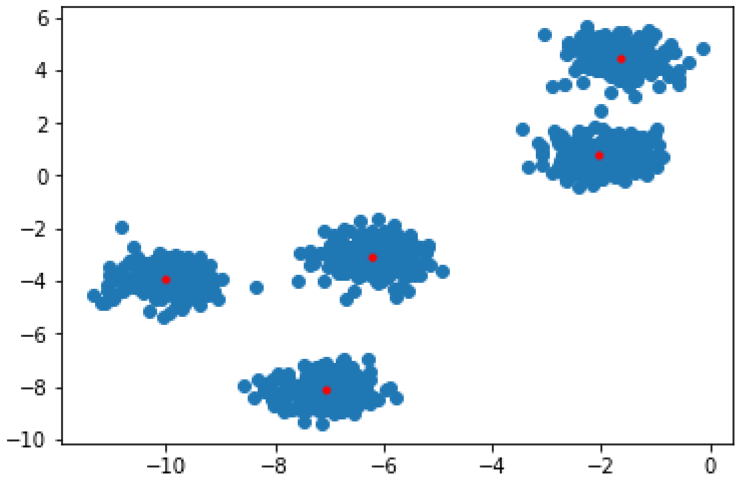
In [19]:
```python
wcss = []

for i in range(1,20):
    kmeans = KMeans(n_clusters = i,init = 'k-means++',n_init = 10,random_state = 1)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)
plt.plot(range(1,20),wcss)
plt.ylabel('WCSS')
plt.xlabel('Value of K')
plt.show()
```

```
C:\Users\yashm\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:882: UserWarning: KMeans is known to have a memor
y leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environm
ent variable OMP_NUM_THREADS=4.
  f"KMeans is known to have a memory leak on Windows "
```



In [20]:
```python
kmeans = KMeans(n_clusters = 5,init = 'k-means++',n_init = 10,random_state = 1)
pred_y = kmeans.fit_predict(X)
```

In [21]:
```python
plt.scatter(X[:,0], X[:,1])
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s=10, c='red')
plt.show()
```

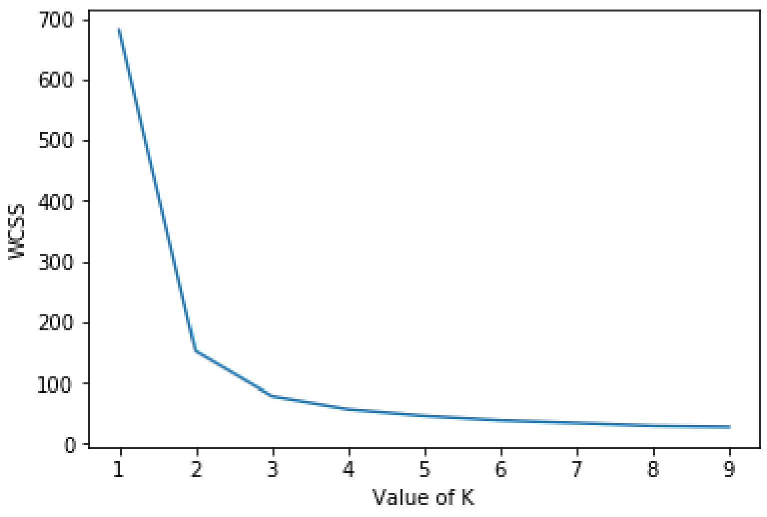


In [22]:
```python
from sklearn.datasets import load_iris
```

In [23]:
```python
iris = load_iris()
```

In [26]:
```python
wcss = []

for i in range(1,10):
    kmeans = KMeans(n_clusters = i,init = 'k-means++',n_init = 10,random_state = 1)
    kmeans.fit(iris.data)
    wcss.append(kmeans.inertia_)
plt.plot(range(1,10),wcss)
plt.ylabel('WCSS')
plt.xlabel('Value of K')
plt.show()
```

```
C:\Users\yashm\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:882: UserWarning: KMeans is known to have a memor
y leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environm
ent variable OMP_NUM_THREADS=1.
  f"KMeans is known to have a memory leak on Windows "
```



In [22]:
```
iris.data.shape
```
Out[22]:
```
(150, 4)
```

In [23]:
```
kmeans = KMeans(n_clusters = 3,init = 'k-means++',n_init = 10,random_state = 1)
pred_y = kmeans.fit_predict(iris.data)
```

In [24]:
```
pred_y
```
Out[24]:
```
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, 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])
```

In [25]:
```
from sklearn.metrics import confusion_matrix
```

In [26]:
```
confusion_matrix(pred_y,iris.target)
```
Out[26]:
```
array([[ 0, 48, 14],
       [50,  0,  0],
       [ 0,  2, 36]], dtype=int64)
```

In [82]:
```
data=pd.read_csv('gapminder (1) (2).csv')
```

In [83]:
```
data.head()
```

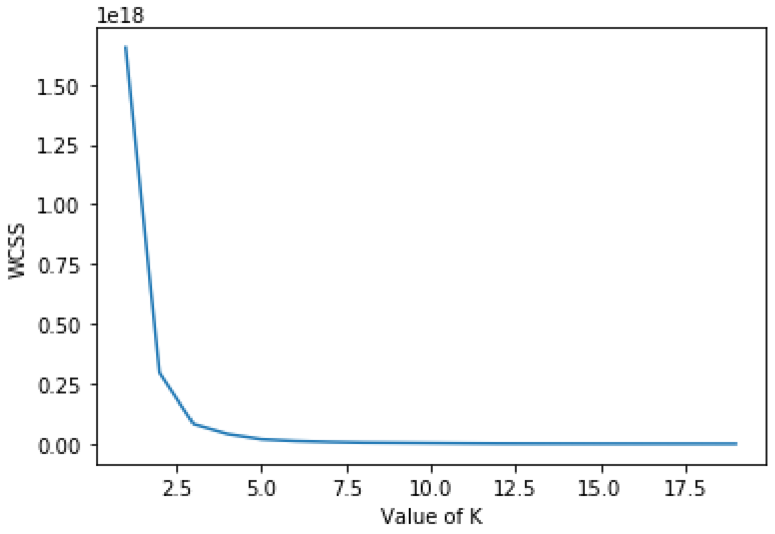
Out[83]:

	population	fertility	HIV	CO2	BMI_male	GDP	BMI_female	life	child_mortality	Region
0	34811059.0	2.73	0.1	3.328945	24.59620	12314.0	129.9049	75.3	29.5	Middle East & North Africa
1	19842251.0	6.43	2.0	1.474353	22.25083	7103.0	130.1247	58.3	192.0	Sub-Saharan Africa
2	40381860.0	2.24	0.5	4.785170	27.50170	14646.0	118.8915	75.5	15.4	America
3	2975029.0	1.40	0.1	1.804106	25.35542	7383.0	132.8108	72.5	20.0	Europe & Central Asia
4	21370348.0	1.96	0.1	18.016313	27.56373	41312.0	117.3755	81.5	5.2	East Asia & Pacific

In [84]:
```
x = data.drop('Region',1)
```

In [85]:
```python
wcss = []

for i in range(1,20):
    kmeans = KMeans(n_clusters = i,init = 'k-means++',n_init = 10,random_state = 1)
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)
plt.plot(range(1,20),wcss)
plt.ylabel('WCSS')
plt.xlabel('Value of K')
plt.show()
```

```
C:\Users\yashm\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:882: UserWarning: KMeans is known to have a memor
y leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environm
ent variable OMP_NUM_THREADS=1.
  f"KMeans is known to have a memory leak on Windows "
```



In [86]:
```python
kmeans = KMeans(n_clusters = 3,init = 'k-means++',n_init = 10,random_state = 1)
kmeans.fit(x)
```

Out[86]:
```
KMeans(n_clusters=3, random_state=1)
```

In [87]:
```
x
```

Out[87]:

	population	fertility	HIV	CO2	BMI_male	GDP	BMI_female	life	child_mortality
0	34811059.0	2.73	0.1	3.328945	24.59620	12314.0	129.9049	75.3	29.5
1	19842251.0	6.43	2.0	1.474353	22.25083	7103.0	130.1247	58.3	192.0
2	40381860.0	2.24	0.5	4.785170	27.50170	14646.0	118.8915	75.5	15.4
3	2975029.0	1.40	0.1	1.804106	25.35542	7383.0	132.8108	72.5	20.0
4	21370348.0	1.96	0.1	18.016313	27.56373	41312.0	117.3755	81.5	5.2
...	...	...	...	...	...	...	...	...	...
134	3350832.0	2.11	0.5	2.489764	26.39123	15317.0	124.2604	76.0	13.0
135	26952719.0	2.46	0.1	4.476669	25.32054	3733.0	124.3462	68.7	49.2
136	86589342.0	1.86	0.4	1.479347	20.91630	4085.0	121.9367	75.4	26.2
137	13114579.0	5.88	13.6	0.148982	20.68321	3039.0	132.4493	52.0	94.9
138	13495462.0	3.85	15.1	0.654323	22.02660	1286.0	131.9745	49.0	98.3

139 rows × 9 columns

In [88]:
```
x['cluster'] = kmeans.predict(x)
```

In [90]:
```
for i in range(3):
    print(x[x['cluster']==i]['GDP'].mean())
```

```
16685.86046511628
3901.0
17379.333333333332
```

In [94]:
```
x[x['cluster']==2]
```

Out[94]:

	population	fertility	HIV	CO2	BMI_male	GDP	BMI_female	life	child_mortality	cluster
8	148252473.0	2.38	0.06	0.319161	20.39742	2265.0	125.0307	68.4	55.9	2
17	194769696.0	1.90	0.45	2.023773	25.78623	13906.0	124.8745	73.9	18.6	2
59	235360765.0	2.48	0.20	1.755044	21.85576	7856.0	126.4216	69.5	36.2	2
65	127317900.0	1.34	0.06	9.536606	23.50004	34800.0	121.9651	82.6	3.4	2
82	114972821.0	2.35	0.30	4.261172	27.42468	15826.0	122.1216	75.7	17.9	2
93	151115683.0	6.02	3.60	0.614690	23.03322	4684.0	135.4920	58.0	140.9	2
96	163096985.0	3.58	0.10	0.935618	22.29914	4187.0	126.5196	64.1	95.5	2
106	143123163.0	1.49	1.00	11.982718	26.01131	22506.0	128.4903	67.6	13.5	2
133	304473143.0	2.07	0.60	18.545992	28.45698	50384.0	118.4777	78.2	7.7	2

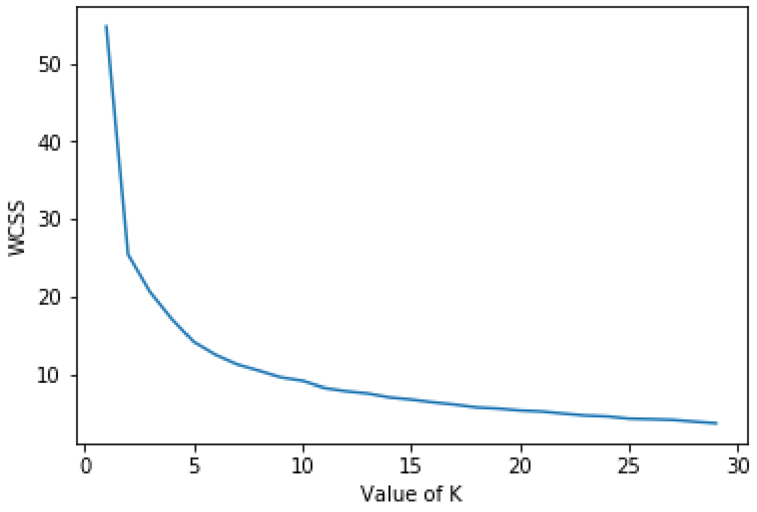
In [74]:
```
len(data['Region'].unique())
```

Out[74]:
```
6
```

In [75]:
```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
x = scaler.fit_transform(x)
```

In [76]:
```python
wcss = []

for i in range(1,30):
    kmeans = KMeans(n_clusters = i,init = 'k-means++',n_init = 10,random_state = 1)
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)
plt.plot(range(1,30),wcss)
plt.ylabel('WCSS')
plt.xlabel('Value of K')
plt.show()
```

```
C:\Users\yashm\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:882: UserWarning: KMeans is known to have a memor
y leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environm
ent variable OMP_NUM_THREADS=1.
  f"KMeans is known to have a memory leak on Windows "
```



In [77]:
```python
data['Region'].unique()
```

Out[77]:
```
array(['Middle East & North Africa', 'Sub-Saharan Africa', 'America',
       'Europe & Central Asia', 'East Asia & Pacific', 'South Asia'],
      dtype=object)
```

In [ ]:
```python

```