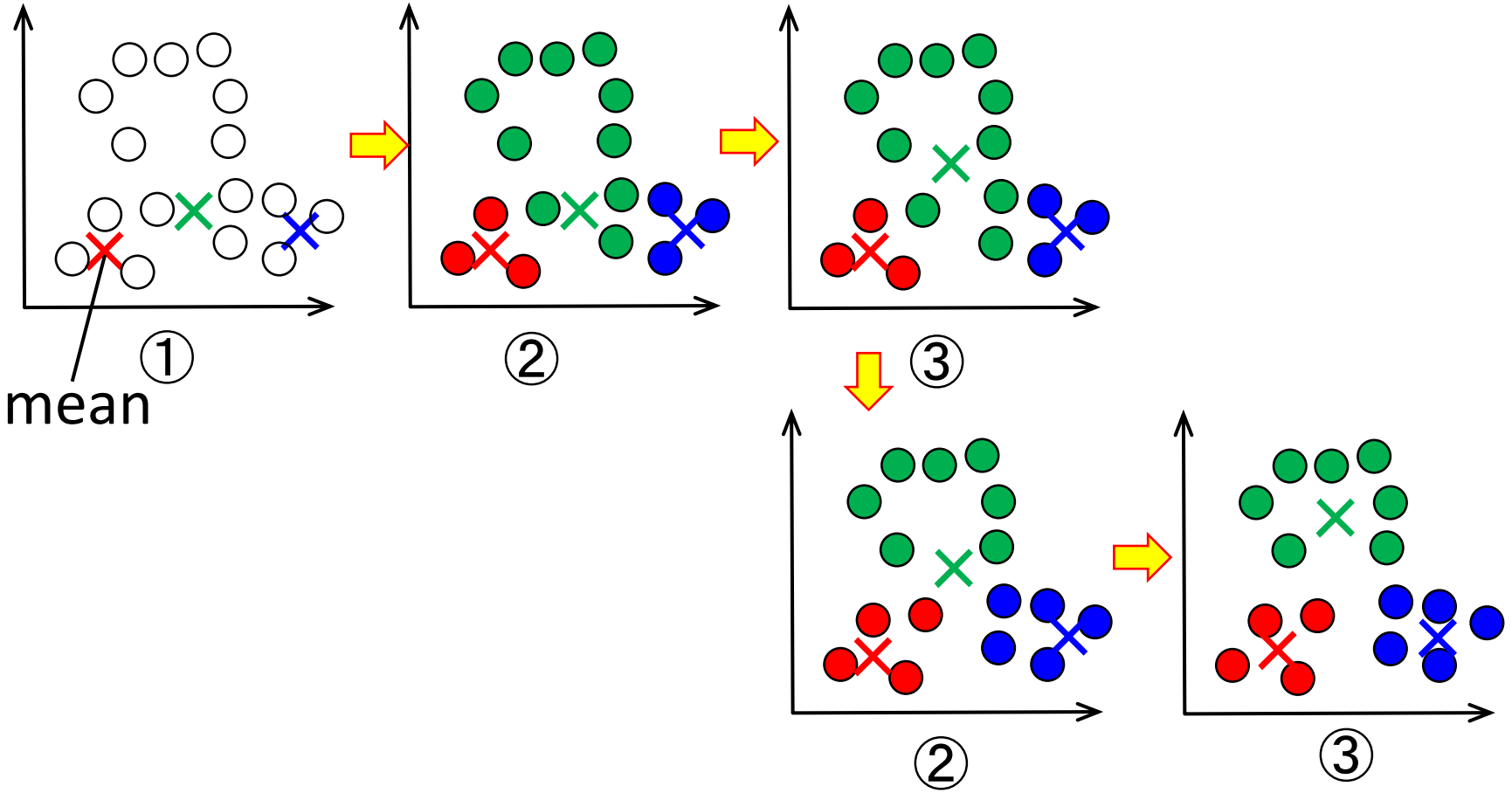
# k-means（k-平均法）

①データの中で適当のk個の”mean“を選択
②最も近いmeanに基づいてk個に分割
③k個のクラスタの重心を算出して、新たな”means”とする
④上記②、③を繰り返して、変わらなくなるまで繰り返す



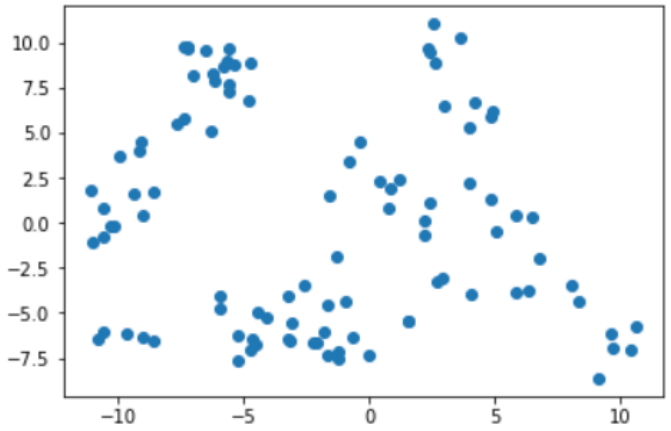
In [5]:
```
%matplotlib inline
from sklearn import datasets
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
X,y = datasets.make_blobs(centers=20,random_state=5)
plt.scatter(X[:,0],X[:,1])
plt.show()
```

データセットを読込

KMeansを読込

塊の数

xに0列目、yに1列目の値を入れた散布図

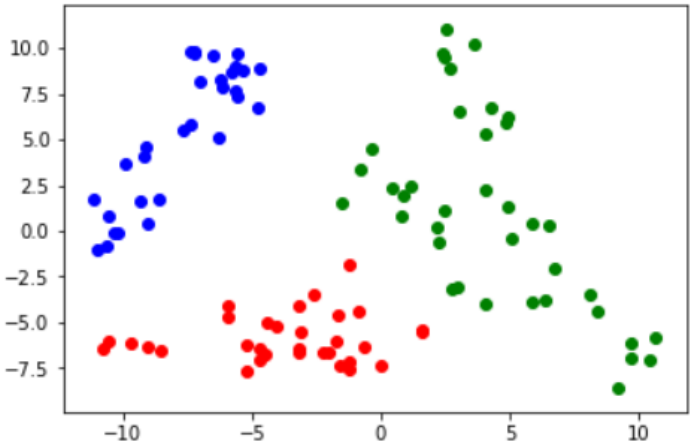



kmeansでクラスタリング実行

クラスタは3種類

In [7]:
```
kmeans = KMeans(n_clusters=3)
kmeans.fit(X)
plt.scatter(X[kmeans.labels_==0,0],X[kmeans.labels_==0,1],c="red")
plt.scatter(X[kmeans.labels_==1,0],X[kmeans.labels_==1,1],c="blue")
plt.scatter(X[kmeans.labels_==2,0],X[kmeans.labels_==2,1],c="green")
plt.show()
```

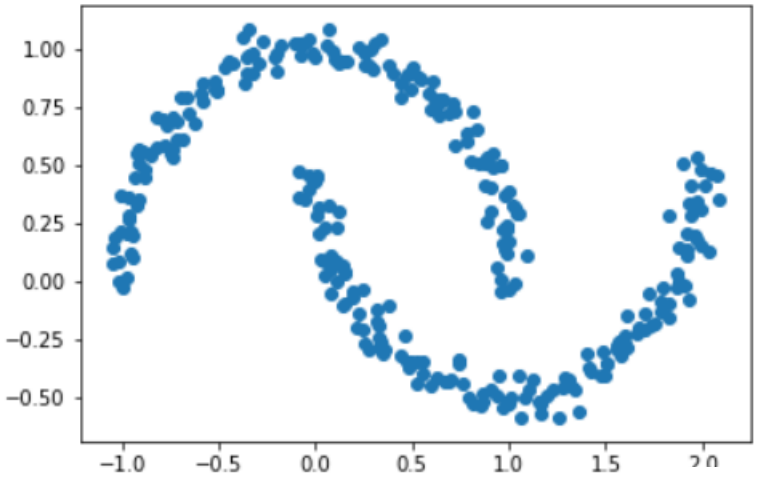
クラスタ0の0列目をx、1列目をyにして
赤色で散布図



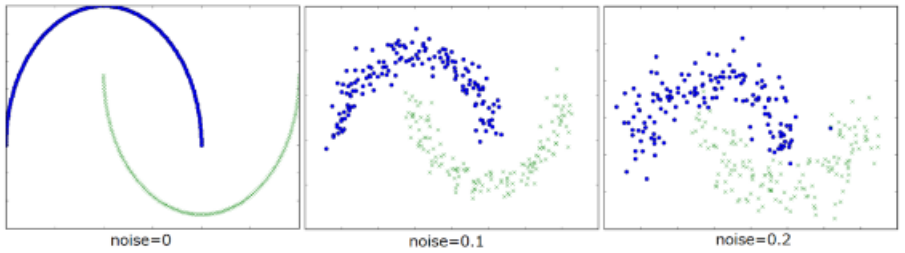
```
In [11]: X,y = datasets.make_moons(n_samples=300,noise=.05)
         plt.scatter(X[:,0],X[:,1])
         plt.show()
```

月形状のデータセットを読込

xに0列目、yに1列目の値を入れた散布図




noiseの数値で以下のようにﾌﾟﾛｯﾄ




kmeansでクラスタリング実行

クラスタは2種類

```
In [12]: kmeans = KMeans(n_clusters=2)
         kmeans.fit(X)
         plt.scatter(X[kmeans.labels_==0,0],X[kmeans.labels_==0,1],c="red")
         plt.scatter(X[kmeans.labels_==1,0],X[kmeans.labels_==1,1],c="blue")
         plt.show()
```

クラスタ0の0列目をx、1列目をyにして赤色で散布図

kmeans法では、クラスタリング失敗