Name: Atharva Suryakant Mali | Class: TY BCA | Roll No.: 04 | DS LAB

Experiment 6

Practical of Clustering

1. Apply K-Means clustering algorithm for given cluster data set.

In [286]:

```
import numpy as np
import pandas as pd
```

In [287]:

```
df=pd.read_csv('cluster.csv')
df.head()
```

Out[287]:

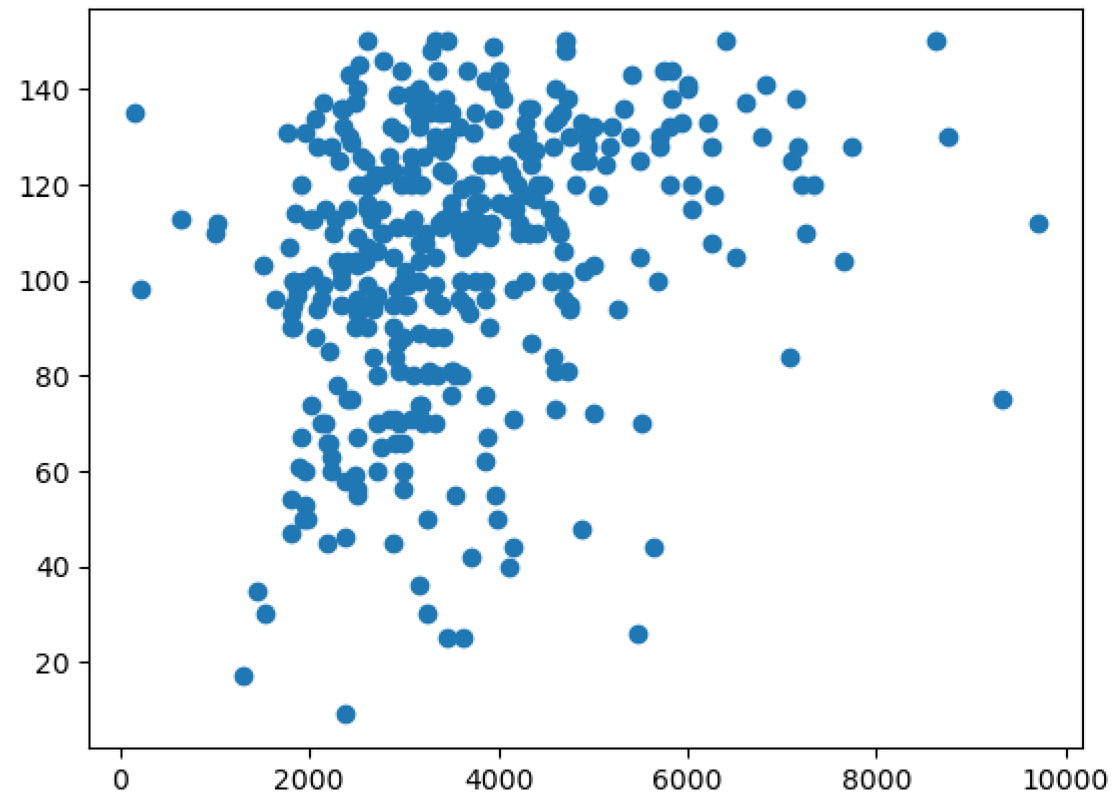
	ApplicantIncome	LoanAmount
0	4583	128
1	3000	66
2	2583	120
3	6000	141
4	2333	95

In [288]:

```
import matplotlib.pyplot as plt
plt.scatter(df['ApplicantIncome'],df['LoanAmount'])
```

Out[288]:

```
<matplotlib.collections.PathCollection at 0x28bdc8a4490>
```



In [289]:

```
from sklearn.cluster import KMeans
```

In [290]:

```
wcss = []
```

In [291]:

```
for i in range(1,11):
    km = KMeans(n_clusters=i)
    km.fit_predict(df)
    wcss.append(km.inertia_)
```

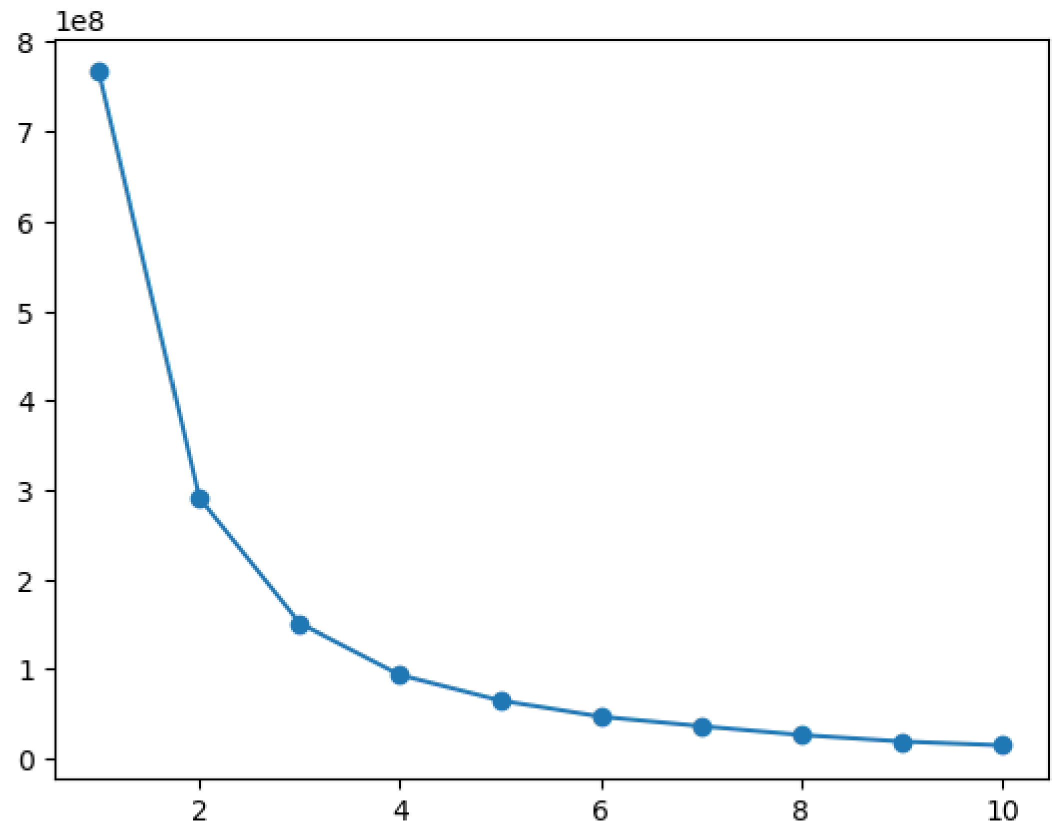
```
C:\Users\surya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:103
6: UserWarning: KMeans is known to have a memory leak on Windows with MKL,
when there are less chunks than available threads. You can avoid it by set
ting the environment variable OMP_NUM_THREADS=2.
  warnings.warn(
```

In [292]:

```
plt.plot(range(1,11),wcss,marker='o')
```

Out[292]:

```
[<matplotlib.lines.Line2D at 0x28bdf46da00>]
```



In [293]:

```
x=df.iloc[:,:].values
```

In [294]:

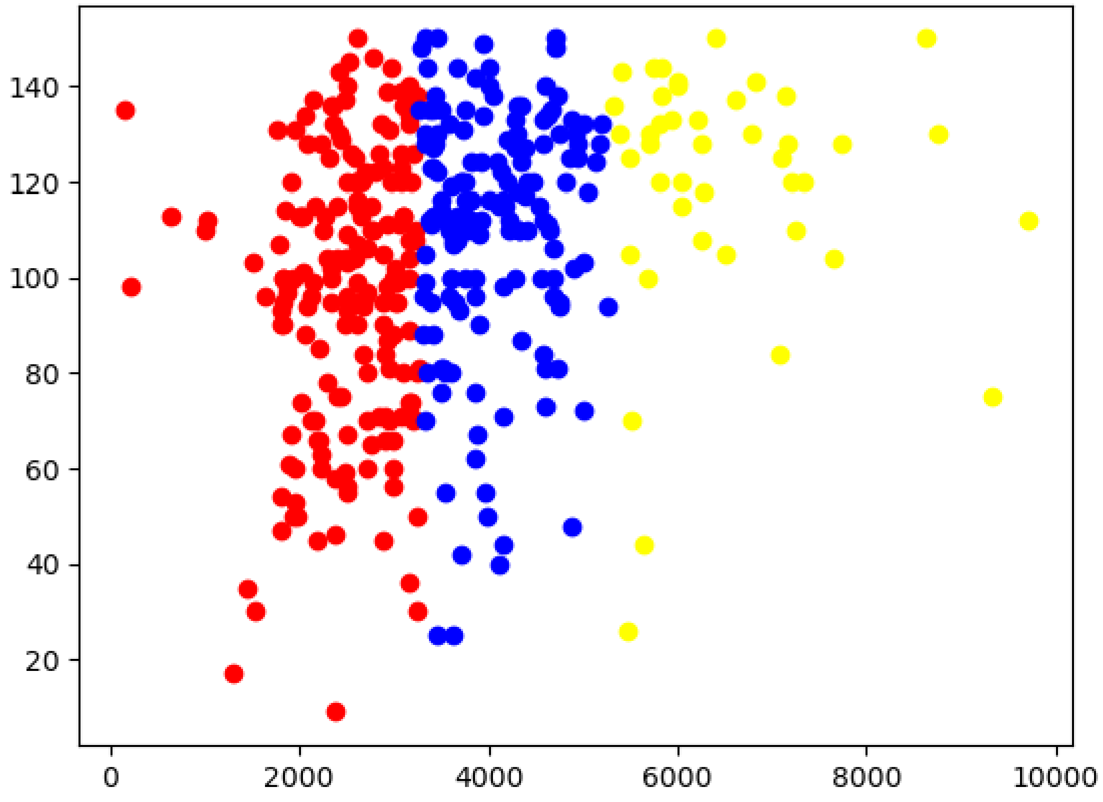
```
km=KMeans(n_clusters=3)
y_means=km.fit_predict(x)
```

In [295]:

```
plt.scatter(x[y_means == 0,0],x[y_means == 0,1],color='red')
plt.scatter(x[y_means == 1,0],x[y_means == 1,1],color='blue')
plt.scatter(x[y_means == 2,0],x[y_means == 2,1],color='yellow')
```

Out[295]:

```
<matplotlib.collections.PathCollection at 0x28bdc86b970>
```



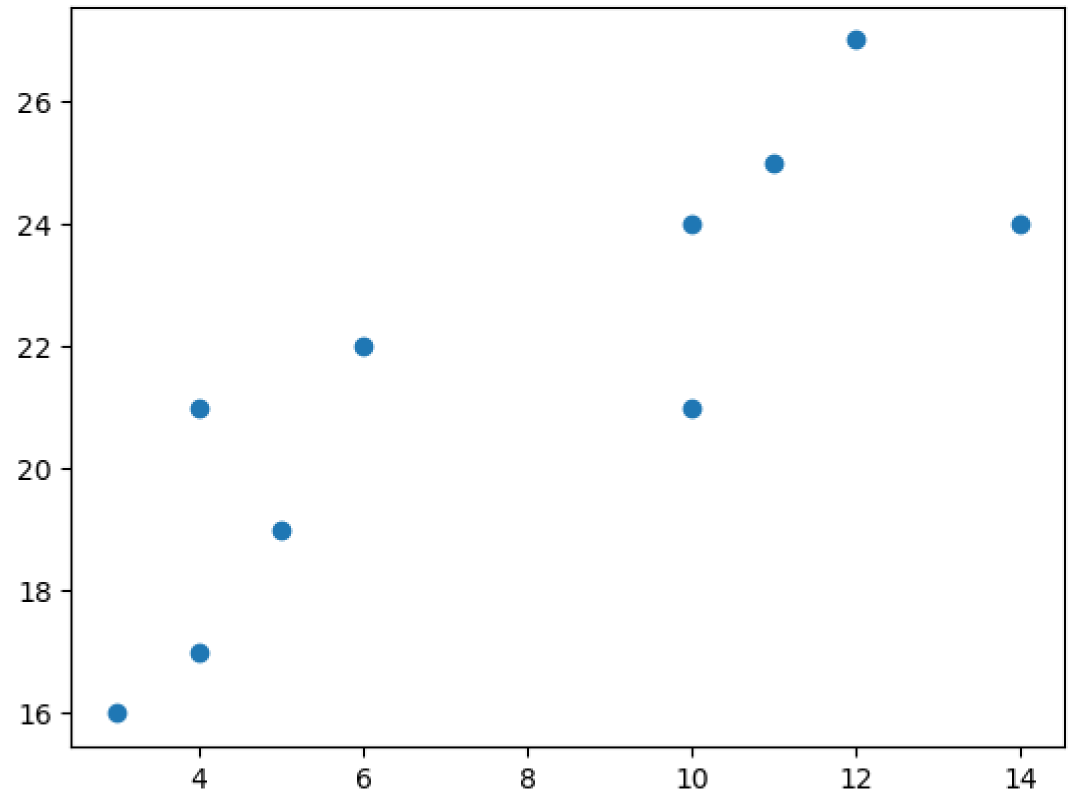
2. Use K-Means clustering to cluster given data set.

In [296]:

```
import matplotlib.pyplot as plt

x=[4,5,10,4,3,11,14,6,10,12]
y=[21,19,24,17,16,25,24,22,21,27]

plt.scatter(x,y)
plt.show()
```



In [297]:

```
from sklearn.cluster import KMeans

data=list(zip(x,y))

print(data)
```

```
[(4, 21), (5, 19), (10, 24), (4, 17), (3, 16), (11, 25), (14, 24), (6, 2
2), (10, 21), (12, 27)]
```

In [298]:

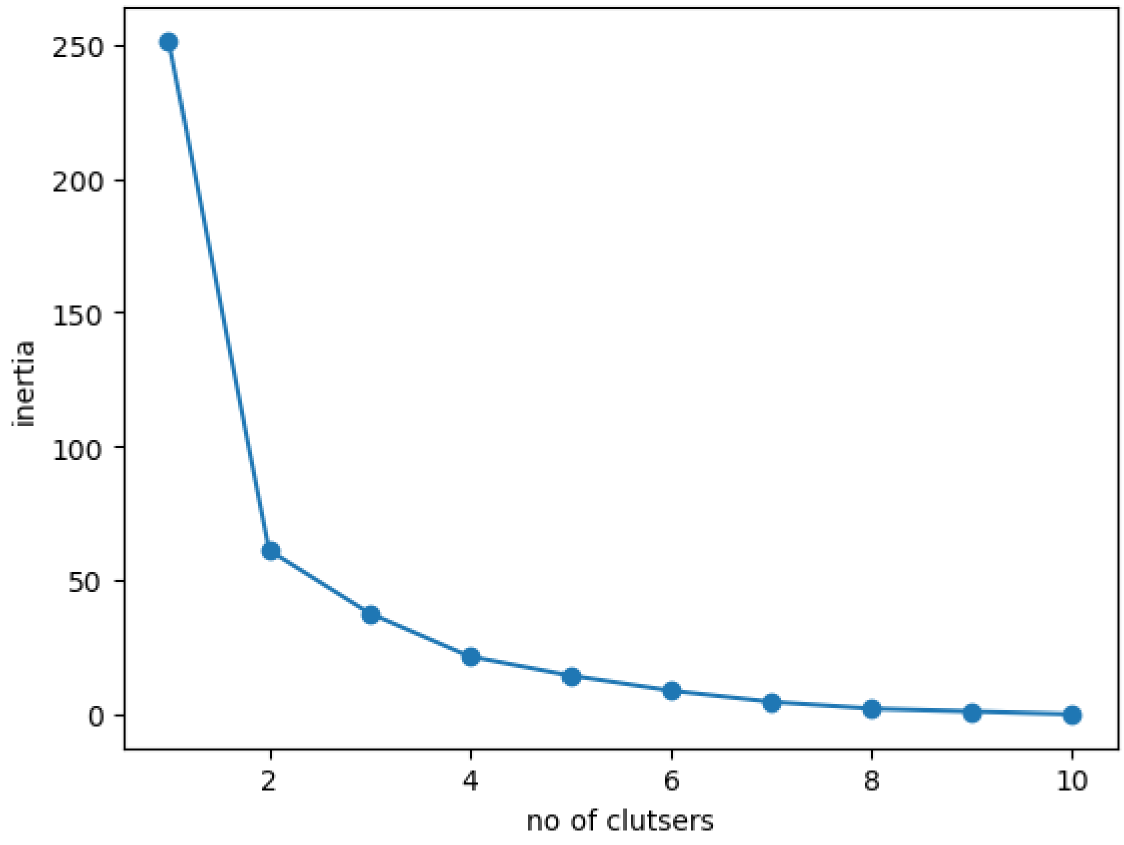
```
inertias=[]

for i in range(1,11):
    kmeans=KMeans(n_clusters=i)
    kmeans.fit(data)
    inertias.append(kmeans.inertia_)
```

```
C:\Users\surya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:103
6: 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 set
ting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

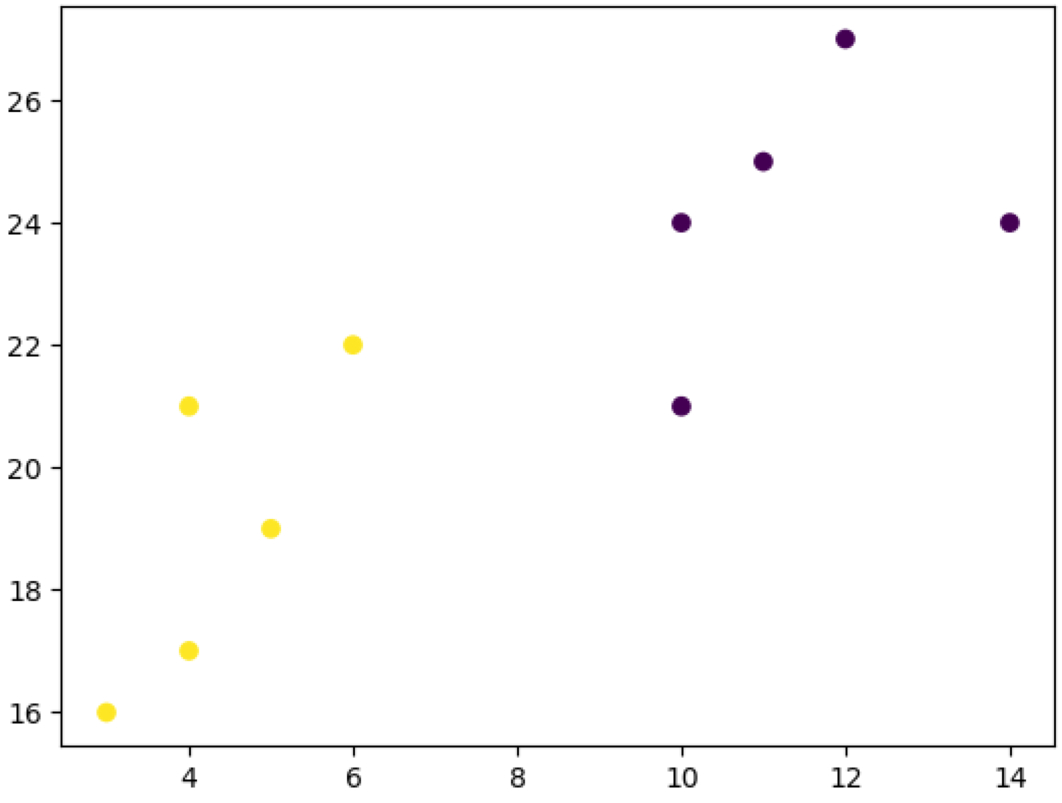
In [299]:

```
plt.plot(range(1,11),inertias,marker='o')
plt.xlabel('no of clutsers')
plt.ylabel('inertia')
plt.show()
```



In [300]:

```
kmeans=KMeans(n_clusters=2)
kmeans.fit(data)
plt.scatter(x,y,c=kmeans.labels_)
plt.show()
```



3. Apply K-Means clustering on Mall_customer data set which is data of customer who visit the mall and spend there. Create clusters for annual income against spending of customer.

In [301]:

```
import numpy as npy
import pandas as pds
```

In [302]:

```
df1=pds.read_csv('Mall_Customers.csv')
df1.head()
```

Out[302]:

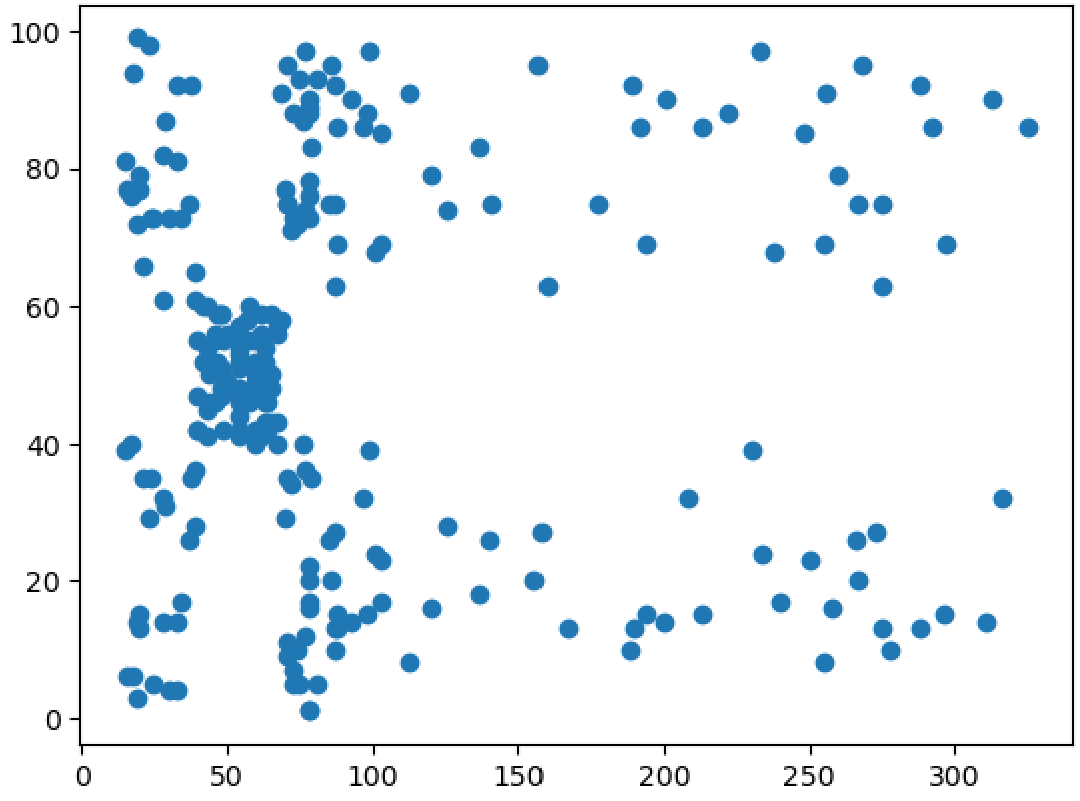
	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

In [303]:

```
import matplotlib.pyplot as plt1
plt1.scatter(df1['Annual Income (k$)'],df1['Spending Score (1-100)'])
```

Out[303]:

```
<matplotlib.collections.PathCollection at 0x28bdf486610>
```



In [304]:

```
from sklearn.cluster import KMeans
```

In [305]:

```
x1=df1['Annual Income (k$)']
y1=df1['Spending Score (1-100)']
data1=list(zip(x1,y1))
```

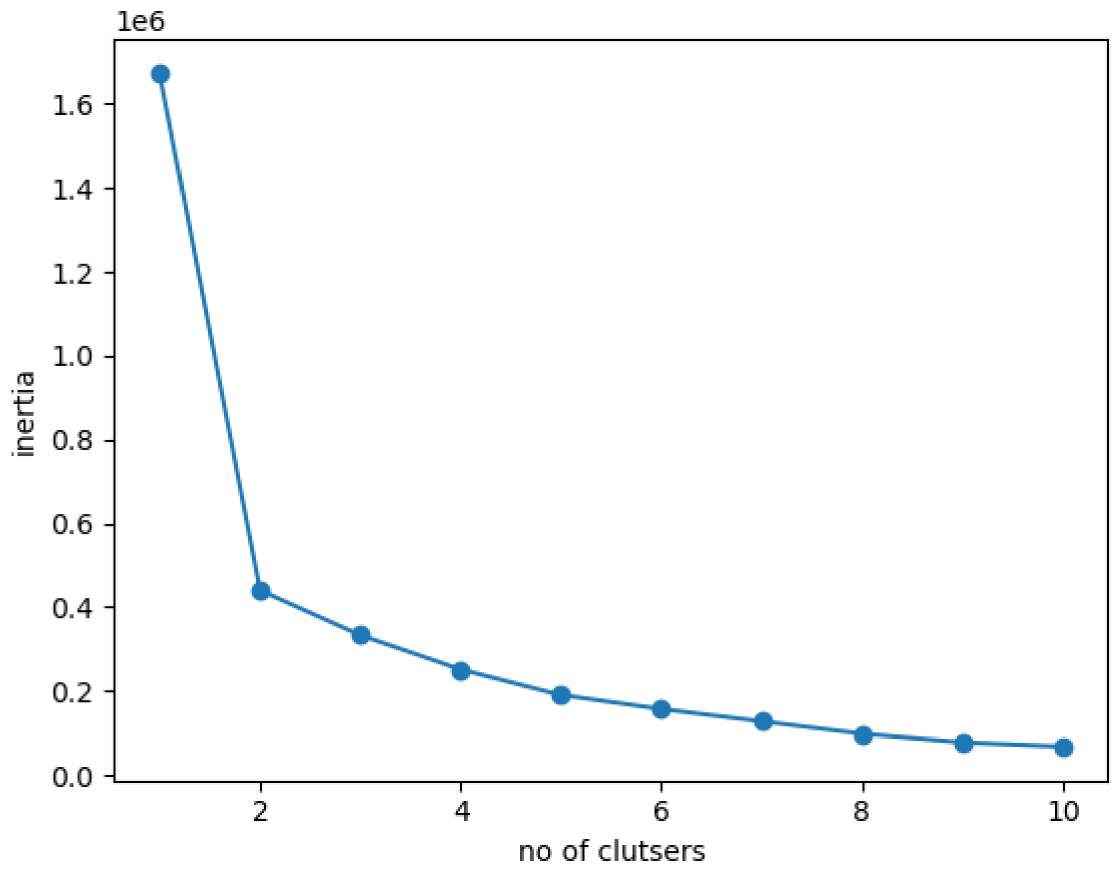
In [306]:

```
wcss = []
for i in range(1,11):
    km1 = KMeans(n_clusters=i)
    km1.fit_predict(data1)
    wcss.append(km1.inertia_)
```

```
C:\Users\surya\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:103
6: 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 set
ting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

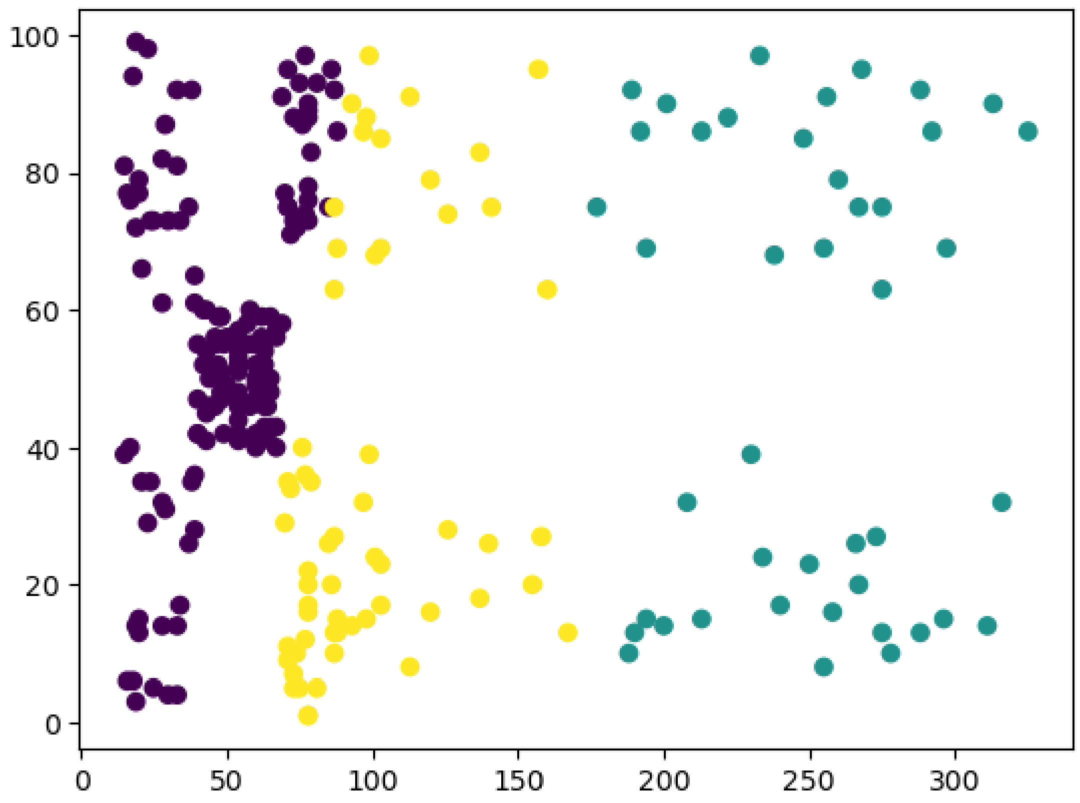
In [307]:

```
plt1.plot(range(1,11),wcss,marker='o')
plt1.xlabel('no of clutsers')
plt1.ylabel('inertia')
plt1.show()
```



In [308]:

```
kmeans1=KMeans(n_clusters=3)
kmeans1.fit(data1)
plt1.scatter(x1,y1,c=kmeans1.labels_)
plt1.show()
```



0
20
40
60
80
100
0
50
100
150
200
250
300