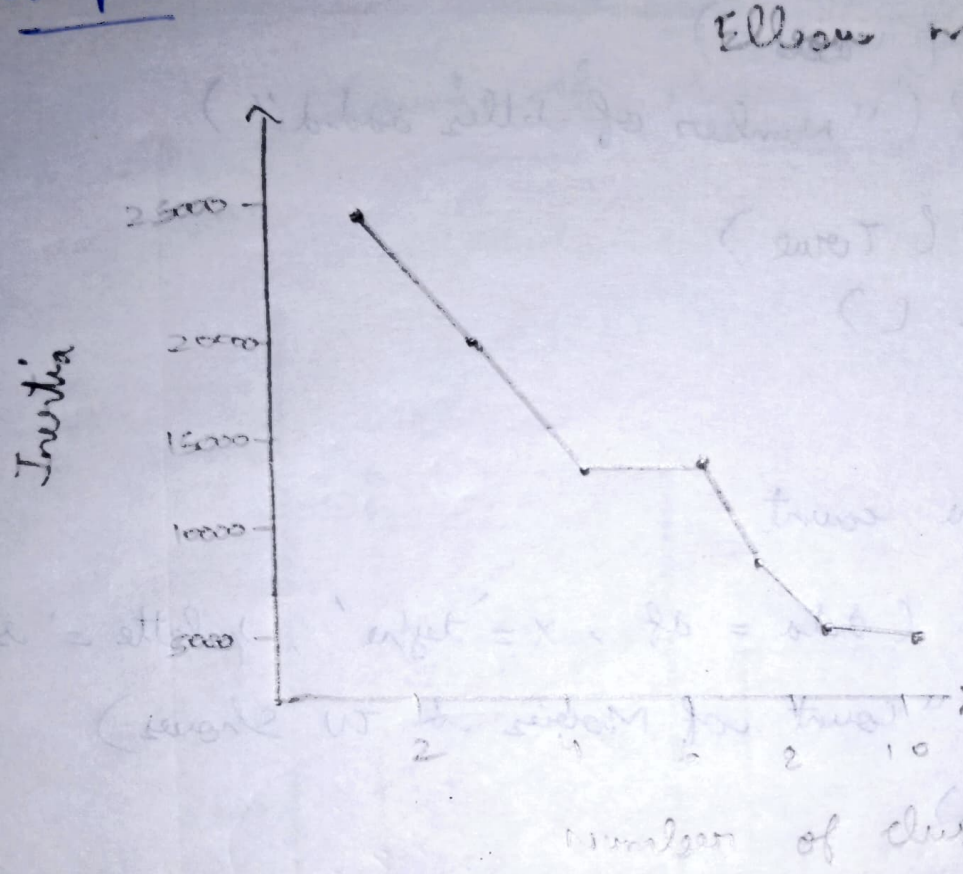
Output:


Elbow method
Inertia
25000
20000
15000
10000
5000
2
4
6
8
10
Number of clusters

EXP. NO : 7

16/9/25

Clustering

Aim :

To write a python program for clustering using python and import necessary dataset.

Code :

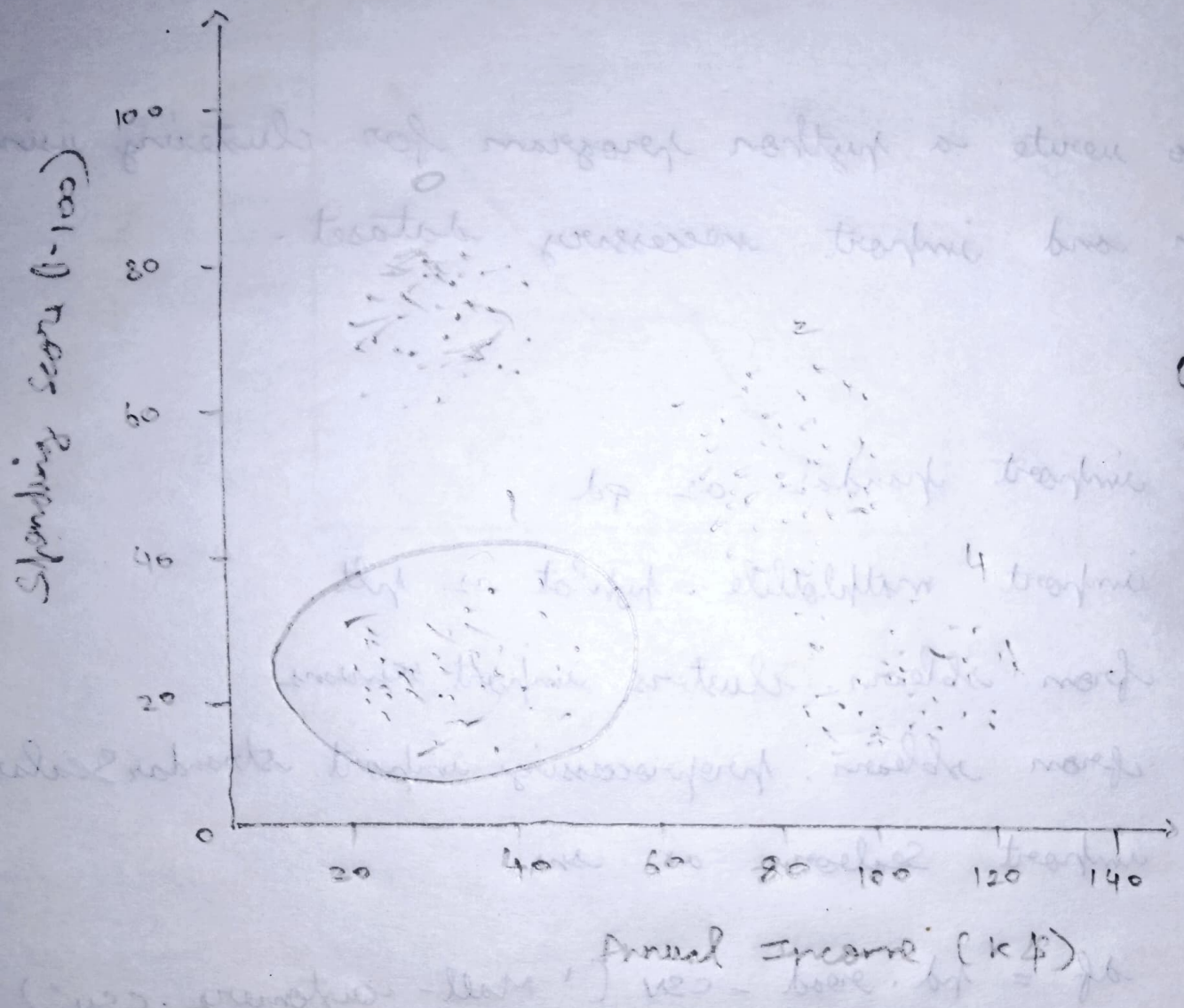
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.clusters import Kmeans
from sklearn.preprocessing import standardScalar
import Seaborn as sns

df = pd.read_csv('Mall_customers.csv')

kmeans = Kmeans(n_clusters = 5, random_state = 42
df['clusters'] = kmeans.fit_predicised(df['
                                Annual Income (k$)'])

distortions = []

for i in range(1, 11):
    km = kmeans(n_clusters = i)
    km.fit(df['annual income (k$)', 'Spending Score
                                (1-100)']])
    distortion.append(km.inertia_)
plt.plot(range(1, 11), distortion, marker = 'o')
plt.title('Elbow method')
plt.xlabel('inertia')
plt.ylabel('No. of Clusters')
```


Spending score (1-100)
100
80
60
40
20
0
20
40
60
80
100
120
140
Annual Income (k$)
Cluster
0 -> •
1 -> *
2 -> X
3 -> 0
4 -> 1

```
import numpy as np
from sklearn.metrics import silhouette_score
from sklearn.clusters import spectral_clustering

# Load data
wine = load_wine()
X = pd.DataFrame(wine.data, columns = wine.feature_names)
X_scaled = StandardScale().fit_transform(X)

# Generate base clustering
base_clustering = []
for k in [3, 4, 5]:
    km = kmeans(n_cluster = k, random_state = 42)
    base_clustering.append(km.fit_predicted(X_scaled))

# Apply ensemble
ensemble_labels = cspa_ensemble(base_clustering)

# Evaluate:
print("Silhoutte Score:" silhouette_score())

# plot clusters:
plt.figure(fig_size = (10, 6))
plt.scatter(X_pca[:, 0], X_pca[:, 1],
            C = ensemble_labels, cmap = 'viridis',
                                    S = 30)
plt.title("CSPA ensembling cluster on Wine")
plt.show()
```

Result: Therefore, the required program for clustering has been executed successfully.