```
import numpy as np
import pandas as pd

df=pd.read_csv('/content/Country-data.csv')
df.head()
```

	country	child_mort	exports	health	imports	income	inflation	life_expec	tota
0	Afghanistan	90.2	10.0	7.58	44.9	1610	9.44	56.2	
1	Albania	16.6	28.0	6.55	48.6	9930	4.49	76.3	
2	Algeria	27.3	38.4	4.17	31.4	12900	16.10	76.5	
3	Angola	119.0	62.3	2.85	42.9	5900	22.40	60.1	
4	Antigua and	10.3	45.5	6.03	58.9	19100	1.44	76.8	

```
df.isna().sum()
```

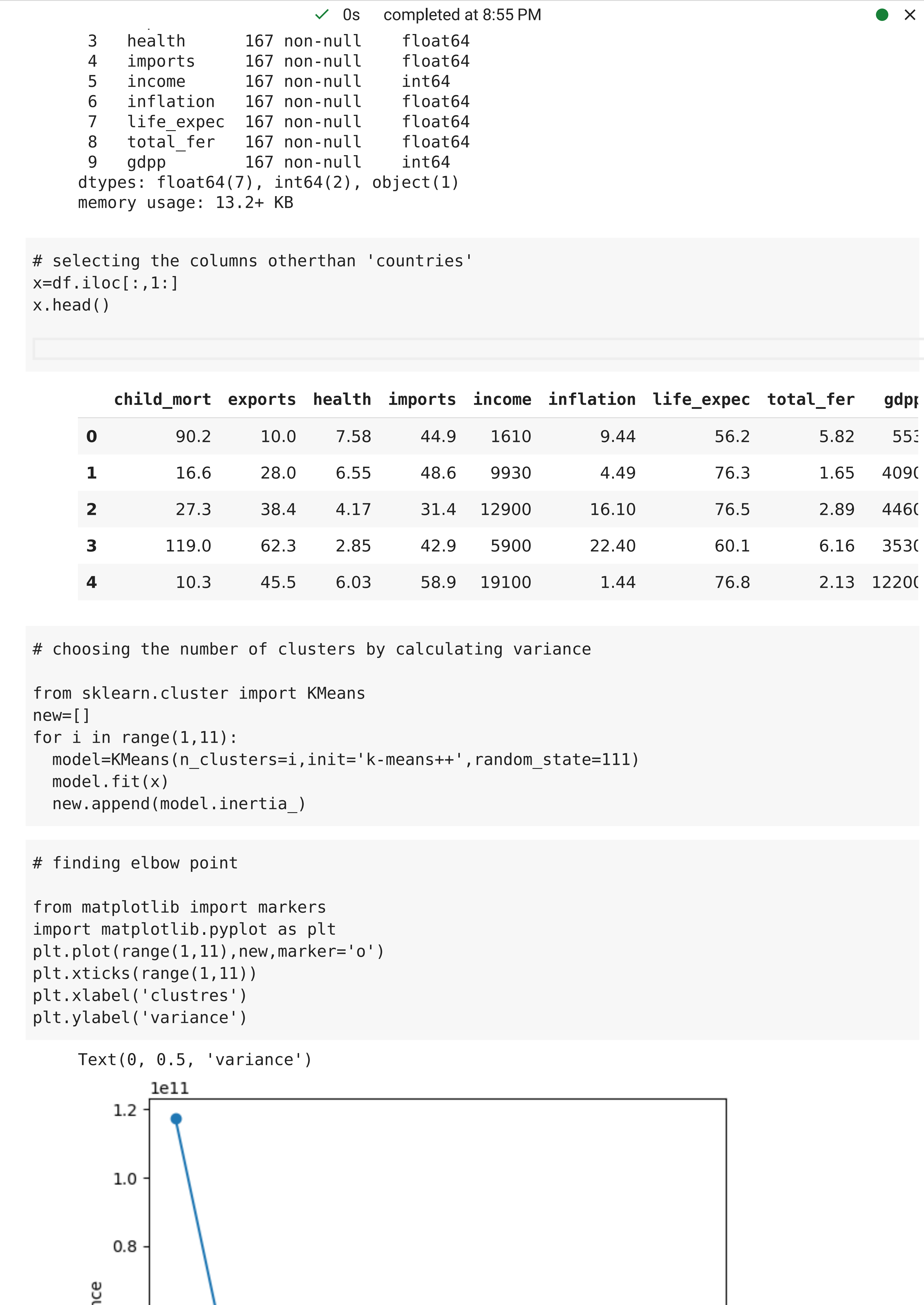
```
country       0
child_mort    0
exports       0
health        0
imports       0
income        0
inflation     0
life_expec    0
total_fer     0
gdpp          0
dtype: int64
```

```
df.describe()
```

	child_mort	exports	health	imports	income	inflation	life_
count	167.000000	167.000000	167.000000	167.000000	167.000000	167.000000	167.0
mean	38.270060	41.108976	6.815689	46.890215	17144.688623	7.781832	70.5
std	40.328931	27.412010	2.746837	24.209589	19278.067698	10.570704	8.8
min	2.600000	0.109000	1.810000	0.065900	609.000000	-4.210000	32.1
25%	8.250000	23.800000	4.920000	30.200000	3355.000000	1.810000	65.3
50%	19.300000	35.000000	6.320000	43.300000	9960.000000	5.390000	73.1
75%	62.100000	51.350000	8.600000	58.750000	22800.000000	10.750000	76.8
max	208.000000	200.000000	17.900000	174.000000	125000.000000	104.000000	82.8

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 167 entries, 0 to 166
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   country     167 non-null    object
 1   child_mort  167 non-null    float64
```



✓ 0s completed at 8:55 PM

```
 3   health      167 non-null    float64
 4   imports     167 non-null    float64
 5   income      167 non-null    int64
 6   inflation   167 non-null    float64
 7   life_expec  167 non-null    float64
 8   total_fer   167 non-null    float64
 9   gdpp        167 non-null    int64
dtypes: float64(7), int64(2), object(1)
memory usage: 13.2+ KB
```

```
# selecting the columns otherthan 'countries'
x=df.iloc[:,1:]
x.head()
```

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
0	90.2	10.0	7.58	44.9	1610	9.44	56.2	5.82	553
1	16.6	28.0	6.55	48.6	9930	4.49	76.3	1.65	4090
2	27.3	38.4	4.17	31.4	12900	16.10	76.5	2.89	4460
3	119.0	62.3	2.85	42.9	5900	22.40	60.1	6.16	3530
4	10.3	45.5	6.03	58.9	19100	1.44	76.8	2.13	12200

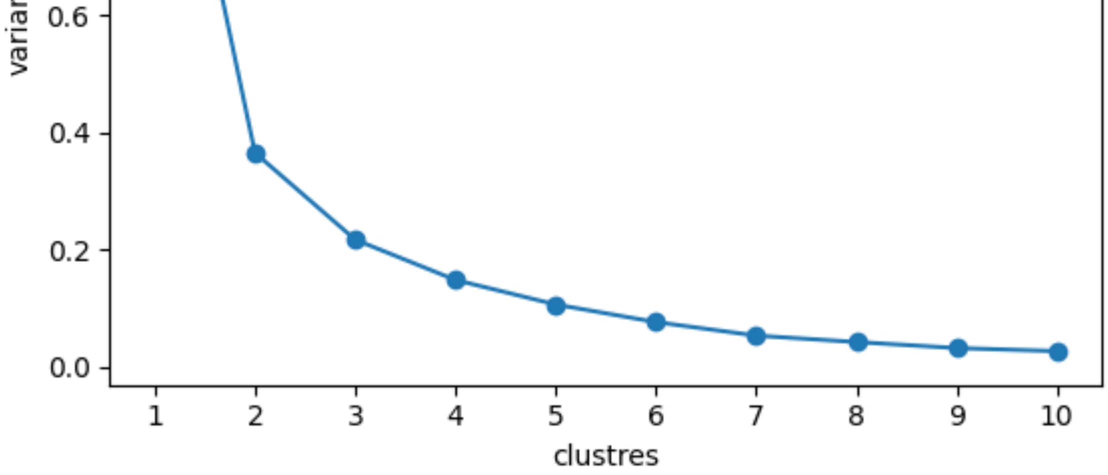
```
# choosing the number of clusters by calculating variance

from sklearn.cluster import KMeans
new=[]
for i in range(1,11):
  model=KMeans(n_clusters=i,init='k-means++',random_state=111)
  model.fit(x)
  new.append(model.inertia_)
```

```
# finding elbow point

from matplotlib import markers
import matplotlib.pyplot as plt
plt.plot(range(1,11),new,marker='o')
plt.xticks(range(1,11))
plt.xlabel('clustres')
plt.ylabel('variance')
```

```
Text(0, 0.5, 'variance')
```



```
# creating the output feature

model1=KMeans(n_clusters=2,init='k-means++',random_state=42)
y=model1.fit_predict(x)
y
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning:
  warnings.warn(
array([0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0], dtype=int32)
```

```
x['cluster']=y
x.head()
```

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdp
0	90.2	10.0	7.58	44.9	1610	9.44	56.2	5.82	553
1	16.6	28.0	6.55	48.6	9930	4.49	76.3	1.65	409
2	27.3	38.4	4.17	31.4	12900	16.10	76.5	2.89	446
3	119.0	62.3	2.85	42.9	5900	22.40	60.1	6.16	353
4	10.3	45.5	6.03	58.9	19100	1.44	76.8	2.13	1220

```
x1=x.iloc[:,:-1].values
y1=x.iloc[:,-1].values
```

```
# standardisation

from sklearn.preprocessing import StandardScaler
std=StandardScaler()
std.fit_transform(x1)
```

```
array([[ 1.29153238, -1.13827979,  0.27908825, ..., -1.61909203,
         1.90288227, -0.67917961],
       [-0.5389489 , -0.47965843, -0.09701618, ...,  0.64786643,
        -0.85997281, -0.48562324],
       [-0.27283273, -0.09912164, -0.96607302, ...,  0.67042323,
        -0.0384044 , -0.46537561],
       ...,
       [-0.37231541,  1.13030491,  0.0088773 , ...,  0.28695762,
        -0.66120626, -0.63775406],
       [ 0.44841668, -0.40647827, -0.59727159, ..., -0.34463279,
         1.14094382, -0.63775406],
       [ 1.11495062, -0.15034774, -0.33801514, ..., -2.09278484,
         1.6246091 , -0.62954556]])
```

```
# model creation

# from sklearn.svm import SVC
# sv=SVC()
# sv.fit(x1,y1)
# ypr=sv.predict(x1)

from sklearn.neighbors import KNeighborsClassifier
knn=KNeighborsClassifier(n_neighbors=5)
knn.fit(x1,y1)
ypr=knn.predict(x1)
```

```
# performance evaluation

from sklearn.metrics import classification_report,confusion_matrix
print(classification_report(y1,ypr))
print(confusion_matrix(y1,ypr))
```

```
              precision    recall  f1-score   support

           0       0.99      1.00      1.00       135
           1       1.00      0.97      0.98        32

    accuracy                           0.99       167
   macro avg       1.00      0.98      0.99       167
weighted avg       0.99      0.99      0.99       167

[[135   0]
 [  1  31]]
```

```
result=pd.DataFrame()
result['country'],result['category']=df['country'],x['cluster']
result.head(10)
```

	country	category
0	Afghanistan	0
1	Albania	0
2	Algeria	0
3	Angola	0
4	Antigua and Barbuda	0
5	Argentina	0
6	Armenia	0
7	Australia	1