

```

import pandas as pd
from sklearn.cluster import KMeans
from sklearn.model_selection import train_test_split
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn_extra.cluster import KMedoids
from scipy.cluster.hierarchy import linkage, dendrogram, fcluster

```

```
df=pd.read_csv("sales_data_sample (1).csv", encoding='latin1')
```

```
df.isnull().sum()
```

```

ORDERNUMBER      0
QUANTITYORDERED  0
PRICEEACH         0
ORDERLINENUMBER   0
SALES             0
ORDERDATE         0
STATUS            0
QTR_ID            0
MONTH_ID          0
YEAR_ID           0
PRODUCTLINE       0
MSRP              0
PRODUCTCODE       0
CUSTOMERNAME      0
PHONE             0
ADDRESSLINE1      0
ADDRESSLINE2      2521
CITY              0
STATE             1486
POSTALCODE        76
COUNTRY           0
TERRITORY         1074
CONTACTLASTNAME   0
CONTACTFIRSTNAME  0
DEALSIZE          0
dtype: int64

```

```

df['ADDRESSLINE2']=df['ADDRESSLINE2'].bfill()
df['STATE']=df['STATE'].ffill()
df['POSTALCODE']=df['POSTALCODE'].ffill()
df['TERRITORY']=df['TERRITORY'].bfill()

```

```
df.isnull().sum()
```

```

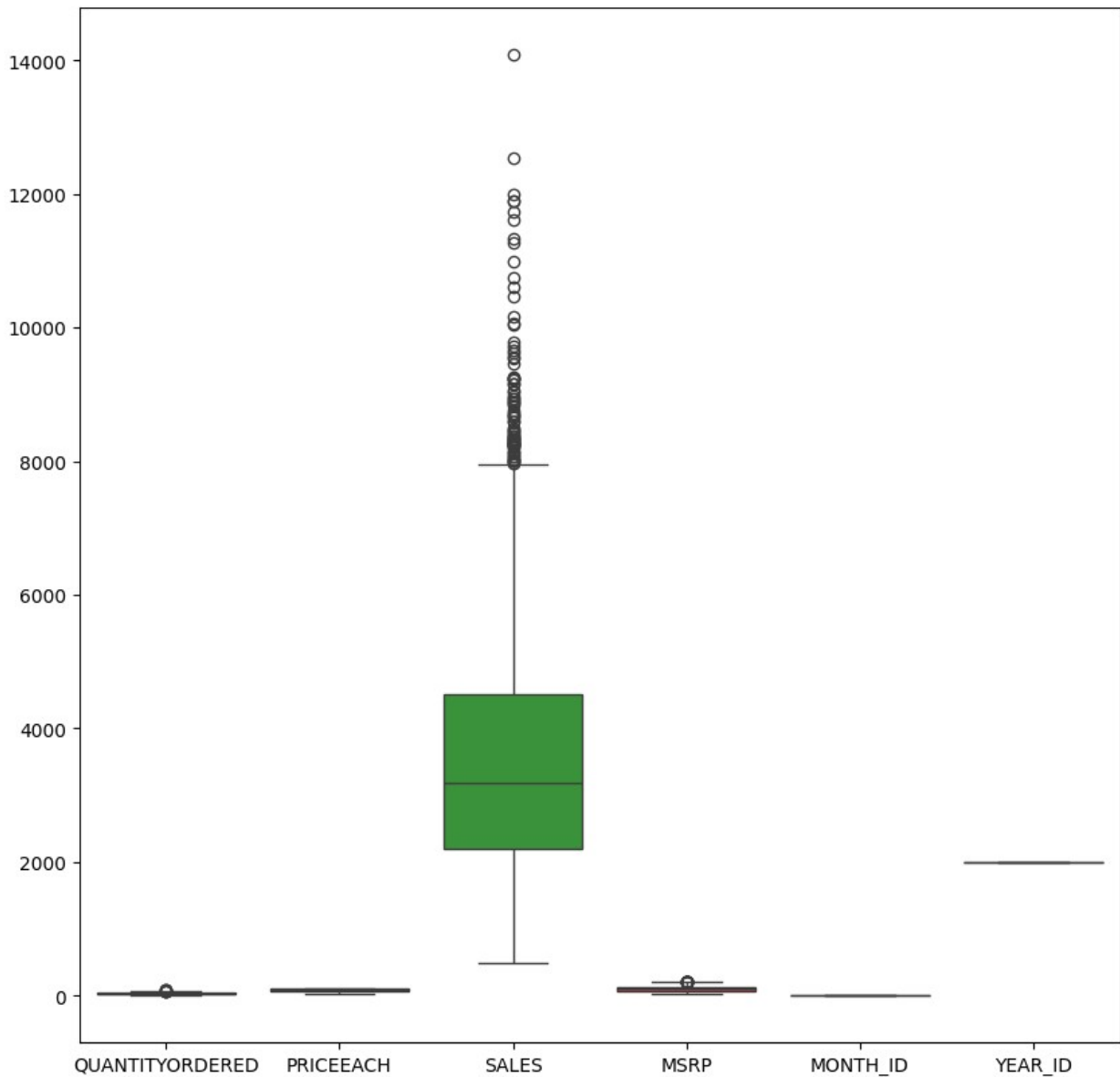
ORDERNUMBER      0
QUANTITYORDERED  0
PRICEEACH         0

```

ORDERLINENUMBER	0
SALES	0
ORDERDATE	0
STATUS	0
QTR_ID	0
MONTH_ID	0
YEAR_ID	0
PRODUCTLINE	0
MSRP	0
PRODUCTCODE	0
CUSTOMERNAME	0
PHONE	0
ADDRESSLINE1	0
ADDRESSLINE2	5
CITY	0
STATE	0
POSTALCODE	0
COUNTRY	0
TERRITORY	1
CONTACTLASTNAME	0
CONTACTFIRSTNAME	0
DEALSIZE	0

dtype: int64

```
num_cols=df[['QUANTITYORDERED','PRICEEACH','SALES','MSRP','MONTH_ID','  
YEAR_ID']]  
plt.figure(figsize=(10,10))  
sns.boxplot(num_cols)  
plt.show()
```



```
X=df[['QUANTITYORDERED', 'PRICEEACH', 'SALES', 'MSRP', 'MONTH_ID', 'YEAR_ID', 'QTR_ID']]
scaler=StandardScaler()
X_scaled=scaler.fit_transform(X)

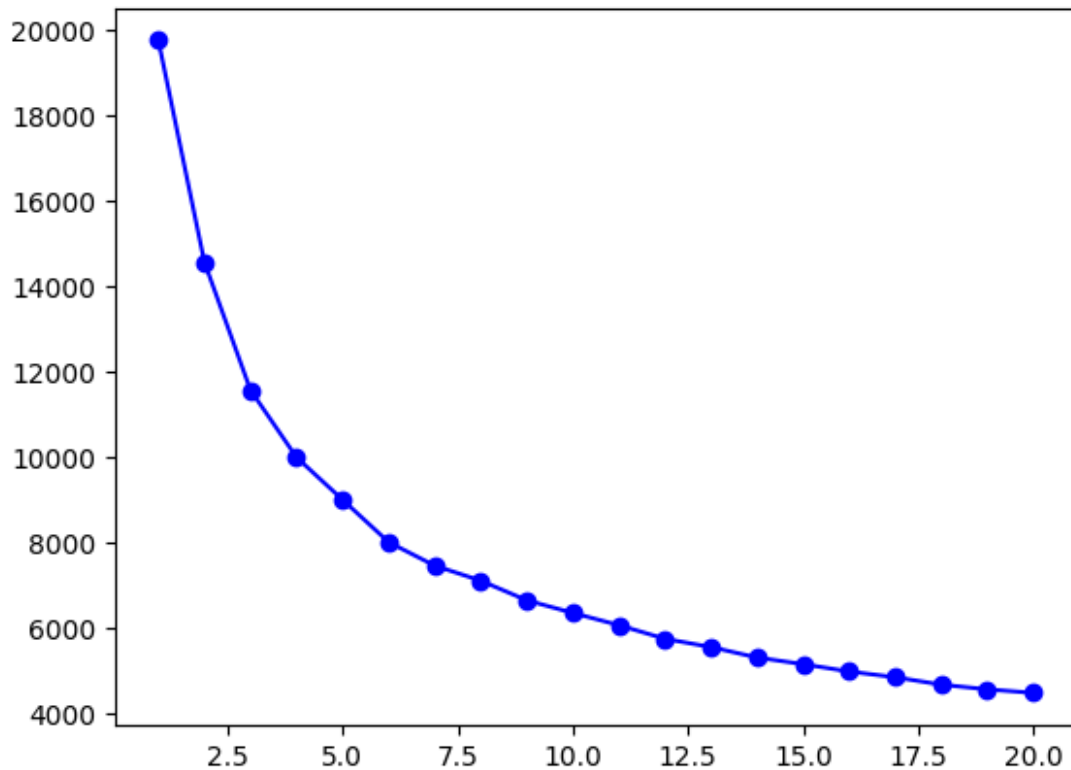
k_range=range(1,21)
inertia_list=[]
for i in k_range:
    kmn=KMeans(n_clusters=i,random_state=42)
    kmn.fit(X_scaled)
    inertia_list.append(kmn.inertia_)
    print(i,kmn.inertia_)
```

```
1 19761.0
2 14564.992533210398
3 11567.803085612331
4 9991.100629855102
5 8997.647088868283
6 8004.04094486497
7 7453.250849412068
8 7100.236497201873
9 6640.72520329007
10 6344.681406931221
11 6057.952307907333
12 5738.196285027825
13 5544.812878462987
14 5299.46041966971
15 5144.065370157961
16 4971.738732958395
17 4838.046621744501
18 4663.902178199858
19 4553.883191689118
20 4470.796022843676
```

```
kmn=KMeans(n_clusters=2,random_state=42)
kmn.fit(X_scaled)
print(kmn.inertia_)
```

```
14564.9925332104
```

```
plt.plot(k_range, inertia_list,'bo-')
plt.show()
```



```
k_range=range(1,21)
k_manhattan=[]
inertia_list=[]
for i in k_range:
    kmn=KMedoids(n_clusters=i, metric='manhattan', random_state=42)
    kmn.fit(X_scaled)
    k_manhattan.append(kmn.inertia_)
    print(i, kmn.inertia_)
```

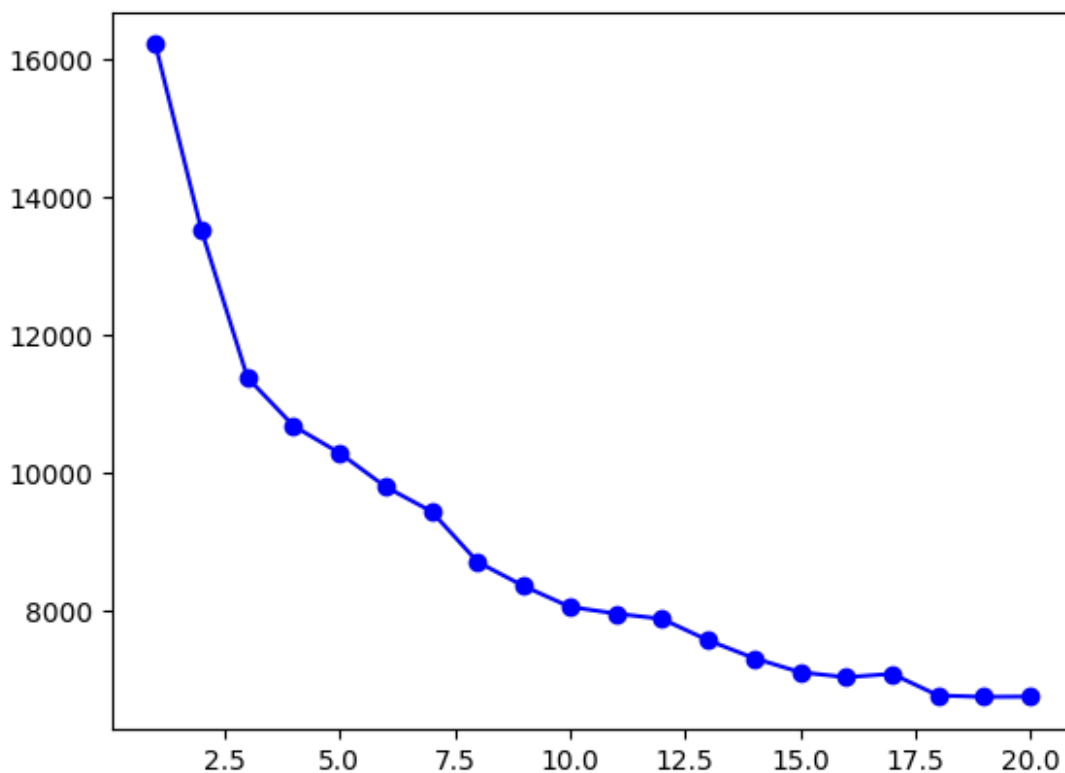
```
1 16205.632724069332
2 13517.399772343855
3 11379.864815045767
4 10680.179899738363
5 10287.872126504182
6 9798.38481747725
7 9440.30841974929
8 8709.219103320404
9 8360.277008558223
10 8059.660547062054
11 7962.354931290544
12 7885.991014058183
13 7575.98303605363
14 7316.724500664352
15 7112.856044961006
16 7040.9010765039675
```

```
17 7089.237793955575
18 6776.295190297531
19 6755.616590466125
20 6760.052022059427
```

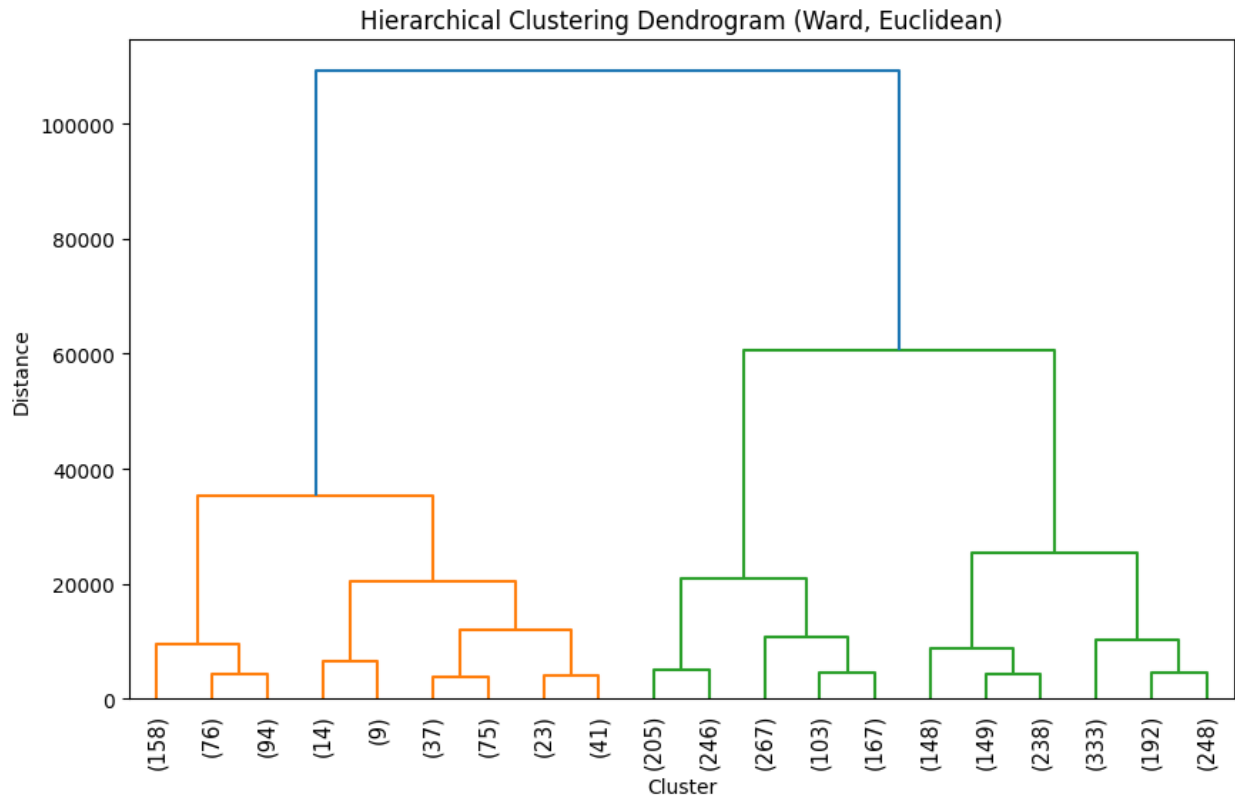
```
print("KMeans using Manhattan")
kmn=KMedoids(n_clusters=18,metric='manhattan', random_state=42)
kmn.fit(X_scaled)
print(kmn.inertia_)
```

```
KMeans using Manhattan
6776.295190297531
```

```
plt.plot(k_range, k_manhattan, 'bo-')
plt.show()
```



```
Z = linkage(X, method="ward", metric="euclidean")
plt.figure(figsize=(10,6))
dendrogram(Z, truncate_mode="lastp", p=20, leaf_rotation=90.,
leaf_font_size=10.5)
plt.title("Hierarchical Clustering Dendrogram (Ward, Euclidean)")
plt.xlabel("Cluster")
plt.ylabel("Distance")
plt.show()
```



```
Z_man = linkage(X, method="average", metric="cityblock")
plt.figure(figsize=(10,6))
dendrogram(Z_man, truncate_mode="lastp", p=20,
leaf_rotation=90.,leaf_font_size=10.)
plt.title("Hierarchical Clustering Dendrogram (Average, Manhattan)")
plt.xlabel("Cluster")
plt.ylabel("Distance")
plt.show()
```

Hierarchical Clustering Dendrogram (Average, Manhattan)

