```
from sklearn.cluster import DBSCAN
from sklearn.preprocessing import StandardScaler
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

# Clustering with outliers

## Dataset

```
id = "1dr93lHQUchIii1lwsGoS40VcUj-rW4H1"
print("https://drive.google.com/uc?export=download&id=" + id)
```

```
https://drive.google.com/uc?export=download&id=1dr93lHQUchIii1lwsGoS40VcUj-rW4
```

```
!wget "https://drive.google.com/uc?export=download&id=1dr93lHQUchIii1lwsGoS40VcUj-r
```

```
--2022-06-08 12:37:53--  https://drive.google.com/uc?export=download&id=1dr93l
Resolving drive.google.com (drive.google.com)... 172.217.204.101, 172.217.204.
Connecting to drive.google.com (drive.google.com)|172.217.204.101|:443... conn
HTTP request sent, awaiting response... 303 See Other
Location: https://doc-04-ag-docs.googleusercontent.com/docs/securesc/ha0ro937g
Warning: wildcards not supported in HTTP.
--2022-06-08 12:37:54--  https://doc-04-ag-docs.googleusercontent.com/docs/sec
Resolving doc-04-ag-docs.googleusercontent.com (doc-04-ag-docs.googleusercont
Connecting to doc-04-ag-docs.googleusercontent.com (doc-04-ag-docs.googleuserc
HTTP request sent, awaiting response... 200 OK
Length: 15021 (15K) [text/csv]
Saving to: ‘wholesaledata.csv’

wholesaledata.csv   100%[===================>]  14.67K  --.-KB/s    in 0s

2022-06-08 12:37:54 (96.1 MB/s) - ‘wholesaledata.csv’ saved [15021/15021]
```

```
df = pd.read_csv('./wholesaledata.csv')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 440 entries, 0 to 439
Data columns (total 8 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Channel           440 non-null    int64
 1   Region            440 non-null    int64
 2   Fresh             440 non-null    int64
 3   Milk              440 non-null    int64
```

```
 4   Grocery           440 non-null    int64
 5   Frozen            440 non-null    int64
 6   Detergents_Paper  440 non-null    int64
 7   Delicassen        440 non-null    int64
dtypes: int64(8)
memory usage: 27.6 KB
```

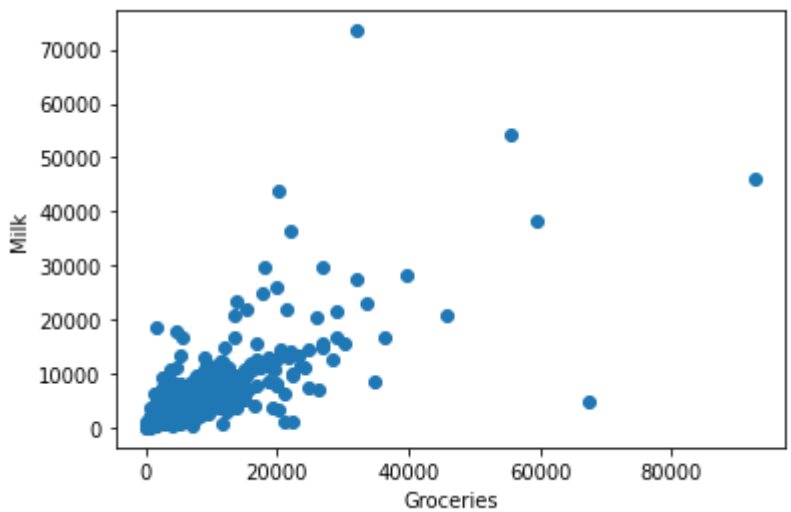
```
df.head()
```

| | Channel | Region | Fresh | Milk | Grocery | Frozen | Detergents_Paper | Delicassen |
|---|---|---|---|---|---|---|---|---|
| **0** | 2 | 3 | 12669 | 9656 | 7561 | 214 | 2674 | 1338 |
| **1** | 2 | 3 | 7057 | 9810 | 9568 | 1762 | 3293 | 1776 |
| **2** | 2 | 3 | 6353 | 8808 | 7684 | 2405 | 3516 | 7844 |
| **3** | 1 | 3 | 13265 | 1196 | 4221 | 6404 | 507 | 1788 |
| **4** | 2 | 3 | 22615 | 5410 | 7198 | 3915 | 1777 | 5185 |

```
# Dropping categorical variables for simplicity

df.drop(["Channel", "Region"], axis = 1, inplace = True)
```

## Simple Visualization

```
# Let's plot two features data now
x = df['Grocery']
y = df['Milk']

plt.scatter(x,y)
plt.xlabel("Groceries")
plt.ylabel("Milk")
plt.show()
```




```
## Outliers even in 2 features
```

```
## DBSCAN preferred over KMeans
```

# DBSCAN

```
df = df[['Grocery', 'Milk']]
std_scaler = StandardScaler().fit(df)
std_df = std_scaler.transform(df)
```

```
# https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html
dbsc = DBSCAN(eps = .5, min_samples = 15).fit(std_df)
```

```
#"Noisy samples are given the label -1." --> Reference
labels = dbsc.labels_
labels
```

```
array([ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0, -1,  0,  0,  0,  0, -1,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, -1,  0, -1,  0, -1,  0,
        0,  0,  0,  0,  0, -1,  0,  0,  0,  0, -1,  0,  0,  0, -1,  0,  0,
        0,  0,  0, -1,  0,  0,  0,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0,
       -1, -1,  0,  0,  0,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0,  0,  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,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  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, -1,  0,  0,  0,  0,  0,  0,
        0, -1,  0,  0,  0,  0,  0,  0,  0,  0,  0, -1,  0, -1,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0, -1,  0,  0,  0,  0, -1,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, -1,  0,  0, -1,
        0,  0,  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,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, -1,  0,  0,
        0,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0,  0,  0, -1,  0,  0,  0,
        0,  0, -1,  0,  0,  0,  0,  0, -1,  0, -1,  0,  0,  0,  0,  0,  0,
        0,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
       -1, -1,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  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,  0,  0,  0,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0, -1,  0,  0,  0,  0,  0,  0,  0,  0,  0, -1,  0,  0])
```

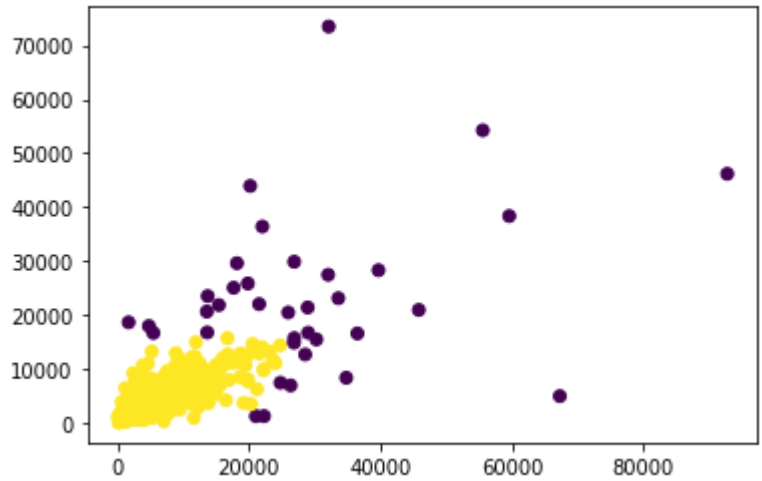
```
out_df = df.copy()
out_df['label'] = dbsc.labels_
```

```
out_df['label'].value_counts()
```

```
 0    404
-1     36
Name: label, dtype: int64
```

```
plt.scatter(out_df['Grocery'], out_df['Milk'], c=out_df['label'])
```

```
<matplotlib.collections.PathCollection at 0x7f36beee4f50>
```




## KMeans

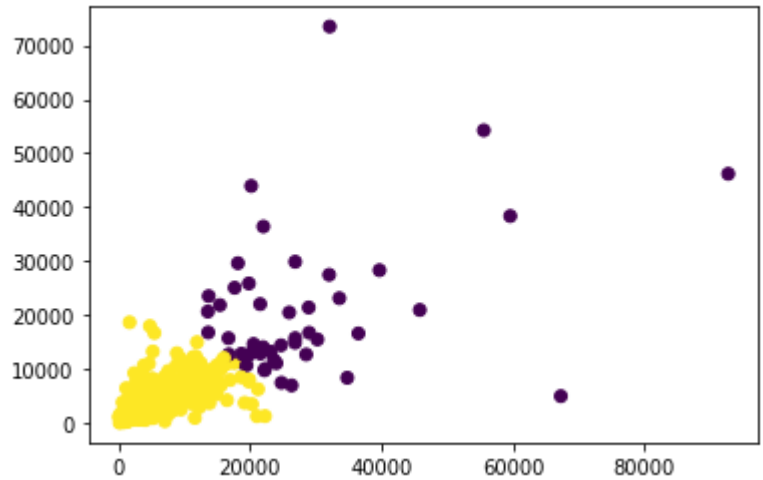
```
from sklearn.cluster import KMeans

k = 2 ## arbitrary value
kmeans = KMeans(n_clusters=k)
y_pred = kmeans.fit_predict(std_df)

clusters = df.copy()
clusters['label'] = kmeans.labels_
clusters
```

| | Grocery | Milk | label |
|---|---|---|---|
| **0** | 7561 | 9656 | 1 |
| **1** | 9568 | 9810 | 1 |
| **2** | 7684 | 8808 | 1 |
| **3** | 4221 | 1196 | 1 |
| **4** | 7198 | 5410 | 1 |
| **...** | ... | ... | ... |
| **435** | 16027 | 12051 | 1 |
| **436** | 764 | 1431 | 1 |
| **437** | 30243 | 15488 | 0 |
| **438** | 2232 | 1981 | 1 |
| **439** | 2510 | 1698 | 1 |

440 rows × 3 columns

```
plt.scatter(clusters['Grocery'], clusters['Milk'], c=clusters['label'])
```

```
<matplotlib.collections.PathCollection at 0x7f36be948fd0>
```




Double-click (or enter) to edit

# Finance Data clustering

## Dataset

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
```

```
id = "1giO5bbp3l0INVvTQIGJ7s_Ai5_TWNuIb"
print("https://drive.google.com/uc?export=download&id=" + id)
```

```
https://drive.google.com/uc?export=download&id=1giO5bbp3l0INVvTQIGJ7s_Ai5_TWNu
```

```
!wget "https://drive.google.com/uc?export=download&id=1giO5bbp3l0INVvTQIGJ7s_Ai5_TW
```

```
--2022-06-08 16:12:43--  https://drive.google.com/uc?export=download&id=1giO5b
Resolving drive.google.com (drive.google.com)... 142.250.101.101, 142.250.101.
Connecting to drive.google.com (drive.google.com)|142.250.101.101|:443... conn
HTTP request sent, awaiting response... 303 See Other
Location: https://doc-08-64-docs.googleusercontent.com/docs/securesc/ha0ro937g
Warning: wildcards not supported in HTTP.
--2022-06-08 16:12:44--  https://doc-08-64-docs.googleusercontent.com/docs/sec
Resolving doc-08-64-docs.googleusercontent.com (doc-08-64-docs.googleusercont
Connecting to doc-08-64-docs.googleusercontent.com (doc-08-64-docs.googleuser
```

```
HTTP request sent, awaiting response... 200 OK
Length: 3053 (3.0K) [text/csv]
Saving to: ‘ind_nifty50list.csv’

ind_nifty50list.csv 100%[===================>]   2.98K  --.-KB/s    in 0s

2022-06-08 16:12:44 (152 MB/s) - ‘ind_nifty50list.csv’ saved [3053/3053]
```

```
stocks_df = pd.read_csv("./ind_nifty50list.csv")
list_of_symobols = list(stocks_df['Symbol'])
stocks_df.head()
```

| | Company Name | Industry | Symbol | Series |
|---|---|---|---|---|
| 0 | Adani Ports and Special Economic Zone Ltd. | SERVICES | ADANIPORTS | EQ |
| 1 | Asian Paints Ltd. | CONSUMER GOODS | ASIANPAINT | EQ |
| 2 | Axis Bank Ltd. | FINANCIAL SERVICES | AXISBANK | EQ |
| 3 | Bajaj Auto Ltd. | AUTOMOBILE | BAJAJ-AUTO | EQ |
| 4 | Bajaj Finance Ltd. | FINANCIAL SERVICES | BAJFINANCE | EQ |

```
!pip install yfinance
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-w
Collecting yfinance
  Downloading yfinance-0.1.70-py2.py3-none-any.whl (26 kB)
Requirement already satisfied: multitasking>=0.0.7 in /usr/local/lib/python3.7
Collecting requests>=2.26
  Downloading requests-2.27.1-py2.py3-none-any.whl (63 kB)
     |████████████████████████████████| 63 kB 1.0 MB/s
Requirement already satisfied: numpy>=1.15 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: pandas>=0.24.0 in /usr/local/lib/python3.7/dist
Collecting lxml>=4.5.1
  Downloading lxml-4.9.0-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64
     |████████████████████████████████| 6.4 MB 7.7 MB/s
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/pyt
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-p
Installing collected packages: requests, lxml, yfinance
  Attempting uninstall: requests
    Found existing installation: requests 2.23.0
    Uninstalling requests-2.23.0:
      Successfully uninstalled requests-2.23.0
  Attempting uninstall: lxml
    Found existing installation: lxml 4.2.6
    Uninstalling lxml-4.2.6:
      Successfully uninstalled lxml-4.2.6
ERROR: pip's dependency resolver does not currently take into account all the
google-colab 1.0.0 requires requests~=2.23.0, but you have requests 2.27.1 whi
```

```
datascience 0.10.6 requires folium==0.2.1, but you have folium 0.8.3 which is
Successfully installed lxml-4.9.0 requests-2.27.1 yfinance-0.1.70
```

```
!pip install fix-yahoo-finance
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-w
Requirement already satisfied: fix-yahoo-finance in /usr/local/lib/python3.7/d
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: multitasking in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/pyt
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3
```

```
yf_symbols = list(map(lambda x: x + '.NS', list_of_symobols))
yf_symbols
```

```
['ADANIPORTS.NS',
 'ASIANPAINT.NS',
 'AXISBANK.NS',
 'BAJAJ-AUTO.NS',
 'BAJFINANCE.NS',
 'BAJAJFINSV.NS',
 'BPCL.NS',
 'BHARTIARTL.NS',
 'BRITANNIA.NS',
 'CIPLA.NS',
 'COALINDIA.NS',
 'DIVISLAB.NS',
 'DRREDDY.NS',
 'EICHERMOT.NS',
 'GRASIM.NS',
 'HCLTECH.NS',
 'HDFCBANK.NS',
 'HDFCLIFE.NS',
 'HEROMOTOCO.NS',
 'HINDALCO.NS',
 'HINDUNILVR.NS',
 'HDFC.NS',
 'ICICIBANK.NS',
 'ITC.NS',
 'IOC.NS',
 'INDUSINDBK.NS',
 'INFY.NS',
 'JSWSTEEL.NS',
 'KOTAKBANK.NS',
 'LT.NS',
 'M&M.NS',
 'MARUTI.NS',
 'NTPC.NS',
 'NESTLEIND.NS',
```

```
       'ONGC.NS',
       'POWERGRID.NS',
       'RELIANCE.NS',
       'SBILIFE.NS',
       'SHREECEM.NS',
       'SBIN.NS',
       'SUNPHARMA.NS',
       'TCS.NS',
       'TATACONSUM.NS',
       'TATAMOTORS.NS',
       'TATASTEEL.NS',
       'TECHM.NS',
       'TITAN.NS',
       'UPL.NS',
       'ULTRACEMCO.NS',
       'WIPRO.NS']
```

```
import yfinance as yf
```

```
stock_financials = {
    'marketCap': [],
    'regularMarketVolume': [],
    'earningsQuarterlyGrowth': [],
    'bookValue': [],
    'totalRevenue': [],
    'returnOnAssets': [],
    'profitMargins': [],
    'earningsGrowth': []
}

for ticker in yf_symbols:
    stock_info = yf.Ticker(ticker).info
    stock_financials['marketCap'].append(stock_info['marketCap'])
    stock_financials['regularMarketVolume'].append(stock_info['regularMarketVolume'
    stock_financials['earningsQuarterlyGrowth'].append(stock_info['earningsQuarterl
    stock_financials['bookValue'].append(stock_info['bookValue'])
    stock_financials['totalRevenue'].append(stock_info['totalRevenue'])
    stock_financials['returnOnAssets'].append(stock_info['returnOnAssets'])
    stock_financials['profitMargins'].append(stock_info['profitMargins'])
    stock_financials['earningsGrowth'].append(stock_info['earningsGrowth'])
```

```
df = pd.DataFrame(stock_financials)
df.head
```

| | marketCap | regularMarketVolume | earningsQuarterlyGrowth | bookValue | tota |
|---|---|---|---|---|---|
| **0** | 1539072720896 | 2839851 | -0.205 | 181.165 | 1.59 |

```
df.shape
```

```
(50, 8)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 8 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   marketCap                50 non-null     int64
 1   regularMarketVolume      50 non-null     int64
 2   earningsQuarterlyGrowth  48 non-null     float64
 3   bookValue                49 non-null     float64
 4   totalRevenue             49 non-null     float64
 5   returnOnAssets           22 non-null     float64
 6   profitMargins            50 non-null     float64
 7   earningsGrowth           48 non-null     float64
dtypes: float64(6), int64(2)
memory usage: 3.2 KB
```

```
import yfinance as yf

stock_prices = yf.download(yf_symbols, start='2020-01-01')['Adj Close']
stock_prices.columns = list_of_symobols
```

```
[*********************100%***********************]  50 of 50 completed
```

```
stock_prices.shape
```

```
(606, 50)
```

```
stock_prices.head()
```

| Date | ADANIPORTS | ASIANPAINT | AXISBANK | BAJAJ-AUTO | BAJFINANCE | BAJAJFINSV | |
|---|---|---|---|---|---|---|---|
| 2020-01-01 | 370.923370 | 1770.856567 | 748.700012 | 2913.946289 | 9370.917969 | 4214.786133 | 45 |
| 2020- | 376.325409 | 1768.338379 | 756.950012 | 2887.027588 | 9497.965820 | 4229.478516 | 45 |

```
##2020 returns
price_2020 = stock_prices.loc["2020-01-02 00:00:00":"2020-12-31 00:00:00"]
stock_prices.loc['returns_2020'] = (price_2020.loc['2020-12-31 00:00:00'] / price_2

stock_prices
```

| Date | ADANIPORTS | ASIANPAINT | AXISBANK | BAJAJ-AUTO | BAJFINANCE | BAJAJFI |
|---|---|---|---|---|---|---|
| 2020-01-01 00:00:00 | 370.923370 | 1770.856567 | 748.700012 | 2913.946289 | 9370.917969 | 4214.786 |
| 2020-01-02 00:00:00 | 376.325409 | 1768.338379 | 756.950012 | 2887.027588 | 9497.965820 | 4229.478 |
| 2020-01-03 00:00:00 | 375.687012 | 1729.577393 | 742.950012 | 2841.747314 | 9338.342773 | 4177.084 |
| 2020-01-06 00:00:00 | 373.427948 | 1685.878662 | 723.250000 | 2809.926270 | 9035.437500 | 3981.102 |
| 2020-01-07 00:00:00 | 377.946014 | 1702.913940 | 725.750000 | 2810.203613 | 9088.344727 | 3992.009 |
| ... | ... | ... | ... | ... | ... | |
| 2022-06-03 00:00:00 | 739.900024 | 2886.899902 | 677.299988 | 3672.699951 | 12691.599609 | 6028.200 |
| 2022-06-06 00:00:00 | 741.250000 | 2817.449951 | 672.200012 | 3817.000000 | 12516.400391 | 6021.000 |
| 2022-06-07 00:00:00 | 734.549988 | 2744.699951 | 665.849976 | 3834.100098 | 12338.700195 | 5878.600 |
| 2022-06-08 00:00:00 | 728.599976 | 2705.199951 | 658.599976 | 3794.149902 | 12465.000000 | 5954.299 |
| returns_2020 | 27.651373 | 55.379620 | -18.032895 | 15.306363 | -6.251606 | 24.995 |

607 rows × 50 columns

```
stock_prices = stock_prices.transpose()
stock_prices.head()
```

| Date | 2020-01-01 00:00:00 | 2020-01-02 00:00:00 | 2020-01-03 00:00:00 | 2020-01-06 00:00:00 | 2020-01-07 00:00:00 | 2020- 00:00 |
|---|---|---|---|---|---|---|
| **ADANIPORTS** | 370.923370 | 376.325409 | 375.687012 | 373.427948 | 377.946014 | 378.780 |
| **ASIANPAINT** | 1770.856567 | 1768.338379 | 1729.577393 | 1685.878662 | 1702.913940 | 1707.259 |
| **AXISBANK** | 748.700012 | 756.950012 | 742.950012 | 723.250000 | 725.750000 | 724.500 |
| **BAJAJ-AUTO** | 2913.946289 | 2887.027588 | 2841.747314 | 2809.926270 | 2810.203613 | 2829.906 |
| **BAJFINANCE** | 9370.917969 | 9497.965820 | 9338.342773 | 9035.437500 | 9088.344727 | 9138.154 |

5 rows × 607 columns

```
prices = stock_prices.iloc[:, -1]
df.index = stock_prices.index
df['return_2020'] = prices
df.head()
```

| | marketCap | regularMarketVolume | earningsQuarterlyGrowth | bookV |
|---|---|---|---|---|
| **ADANIPORTS** | 1539072720896 | 2839851 | -0.205 | 18 |
| **ASIANPAINT** | 2596018061312 | 1742929 | -0.002 | 14 |
| **AXISBANK** | 2041133006848 | 9280420 | 0.502 | 38 |
| **BAJAJ-AUTO** | 1097901801472 | 346513 | -0.016 | 103 |
| **BAJFINANCE** | 3702931324928 | 1334050 | 0.797 | 72 |

```
# drop null values
df.isna().sum()
```

```
marketCap                  0
regularMarketVolume        0
earningsQuarterlyGrowth    2
bookValue                  1
totalRevenue               1
returnOnAssets            28
profitMargins              0
earningsGrowth             2
return_2020                0
dtype: int64
```

```
df['returnOnAssets'] = df['returnOnAssets'].replace(np.nan, 0)
```

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

```
marketCap                  0
regularMarketVolume        0
earningsQuarterlyGrowth    2
bookValue                  1
totalRevenue               1
returnOnAssets             0
profitMargins              0
earningsGrowth             2
return_2020                0
dtype: int64
```

```
df.dropna(axis=0, inplace=True)
df.shape
```

```
(47, 9)
```

## Hiearchical Clustering

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaler.fit(df)
X = scaler.transform(df)
```

```
scaled_df = pd.DataFrame(X, columns=df.columns, index=df.index)
```

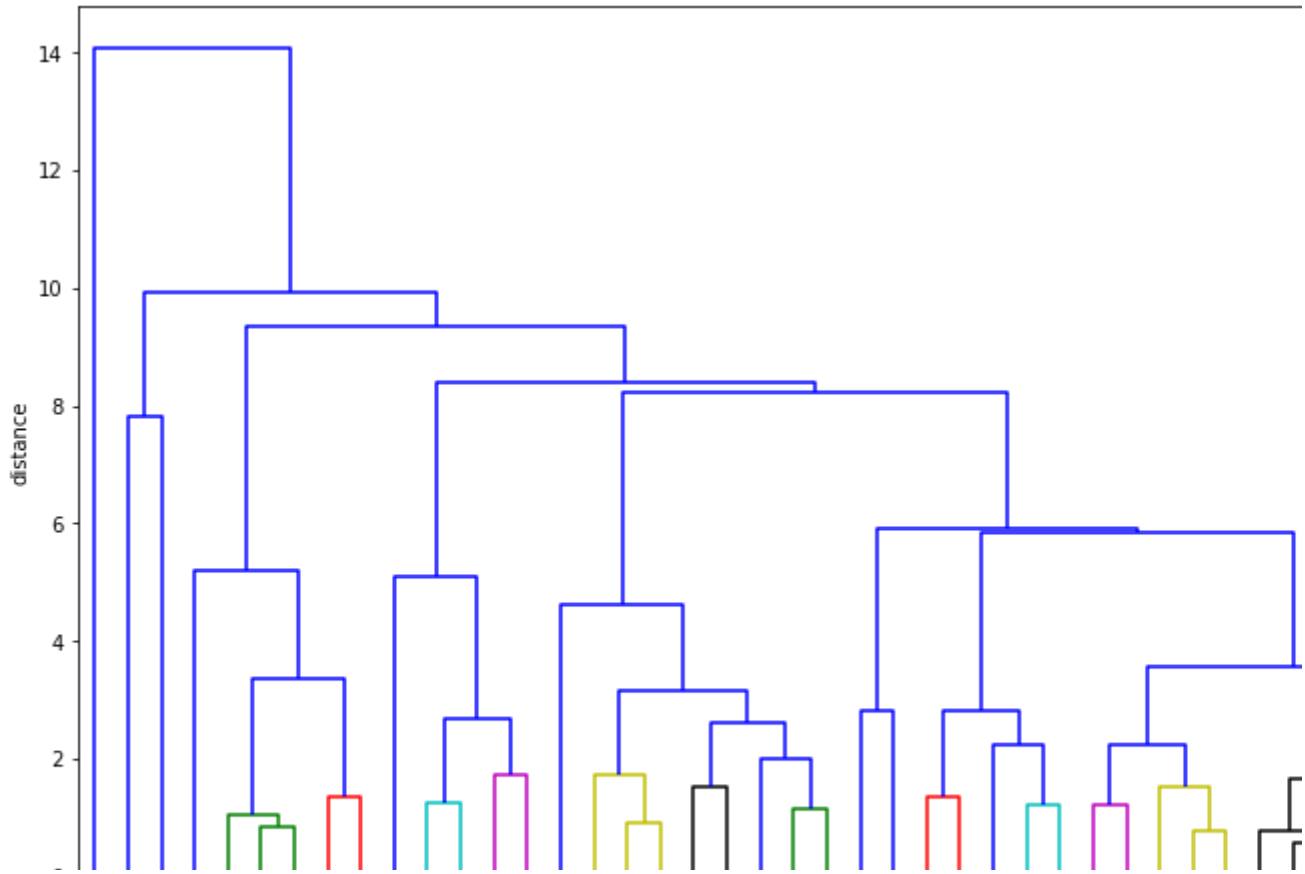
```
# import hierarchical clustering libraries
import scipy.cluster.hierarchy as sch

# Refer https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarch
Z = sch.linkage(scaled_df, method='ward') #linkage = ward
```

```
Z.shape
```

```
(46, 4)
```

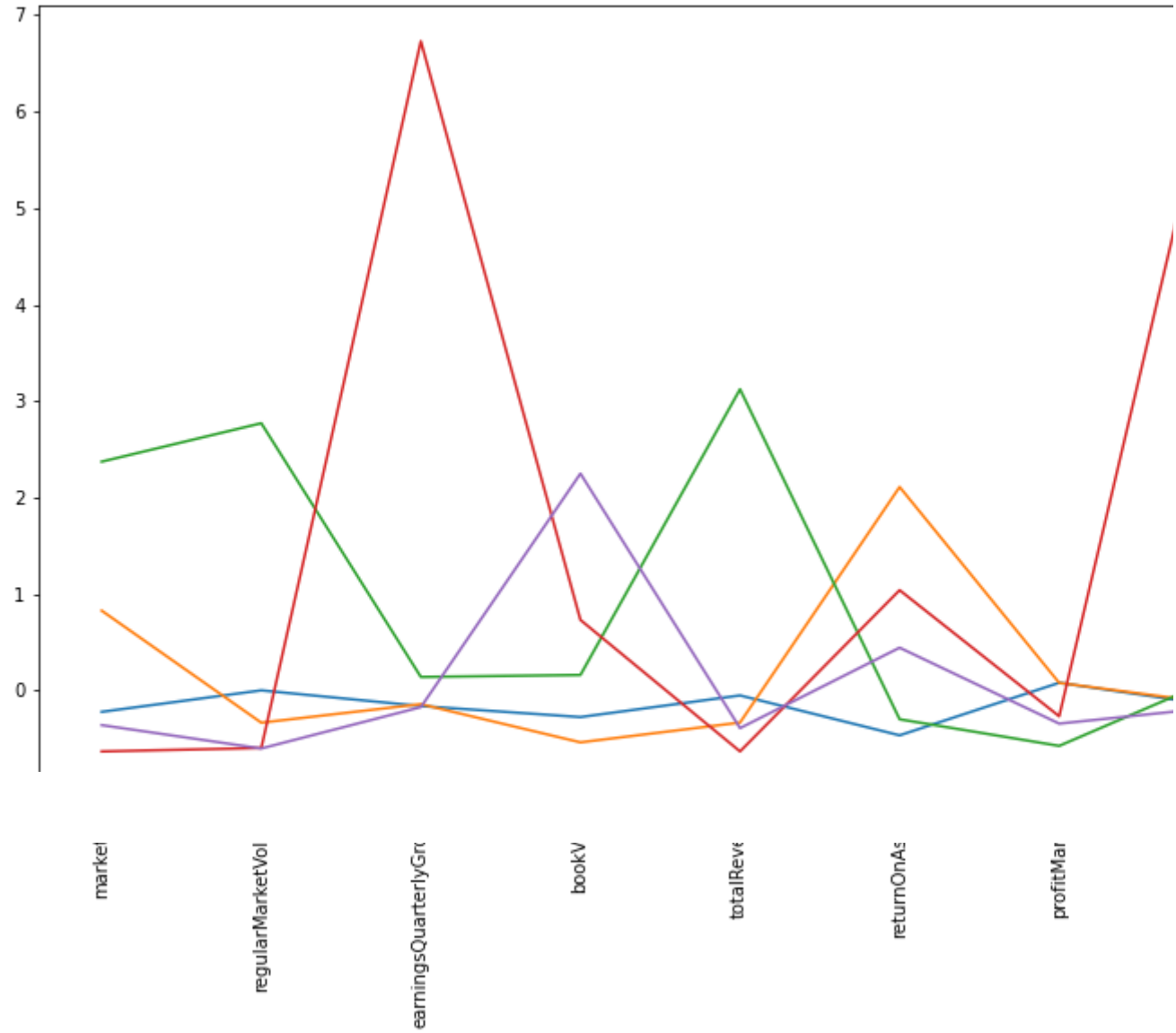
```
fig, ax = plt.subplots(figsize=(14, 8))
sch.dendrogram(Z, labels=scaled_df.index, ax=ax, color_threshold=2)
plt.xticks(rotation=90)
ax.set_ylabel('distance')
```

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




```
# import hierarchical clustering libraries
from sklearn.cluster import AgglomerativeClustering


# create clusters
hc_2020 = AgglomerativeClustering(n_clusters=5, affinity = 'euclidean', linkage = '

y_pred = hc_2020.fit_predict(scaled_df)


#Plot a line graph to see the characteristics of the clusters
scaled_df['label'] = pd.Series(y_pred, index=scaled_df.index)

clustered_df = scaled_df.groupby('label').mean()

labels = ['Cluster 1', 'Cluster 2', 'Cluster 3', 'Cluster 4', 'Cluster 5']

plt.figure(figsize=(14,8))
plt.plot(clustered_df.T, label=labels)
plt.xticks(rotation=90)
plt.legend(labels)
```

```
<matplotlib.legend.Legend at 0x7f36d94d4490>
```




✓ 0s completed at 21:43