EDA On Stock Market Dataset

Importing Libraries

In [2]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

Importing Dataset

In [3]:

```
df = pd.read_csv("nifty_500.csv")
# To Display first 5 records
df.head()
```

Out[3]:

	Company Name	Symbol	Industry	Series	Open	High	Low	Previous Close	Last Traded Price	Cl
0	3M India Ltd.	3MINDIA	Diversified	EQ	21950.00	21999.00	21126.05	21854.05	21575.00	-
1	Aarti Drugs Ltd.	AARTIDRUGS	Healthcare	EQ	400.50	401.80	394.10	403.85	400.00	
2	Aavas Financiers Ltd.	AAVAS	Financial Services	EQ	1997.10	2004.05	1894.50	2015.45	1943.15	
3	ABB India Ltd.	ABB	Capital Goods	EQ	2260.35	2311.50	2260.35	2300.90	2280.00	
4	Abbott India Ltd.	ABBOTINDIA	Healthcare	EQ	18700.40	19200.00	18605.00	18760.40	19199.80	-

In [4]:

```
# To display last 5 records
df.tail()
```

Out[4]:

	Company Name	Symbol	Industry	Series	Open	High	Low	Previous Close	Last Traded Price
496	Zensar Technolgies Ltd.	ZENSARTECH	Information Technology	EQ	273.15	273.55	268.40	272.10	270.0
497	ZF Commercial Vehicle Control Systems India Ltd.	ZFCVINDIA	Automobile and Auto Components	EQ	7748.00	7900.00	7525.30	7716.60	7680.0
498	Zomato Ltd.	ZOMATO	Consumer Services	EQ	54.15	56.70	52.55	53.85	56.0
499	Zydus Lifesciences Ltd.	ZYDUSLIFE	Healthcare	EQ	356.90	364.05	354.30	357.00	364.0
500	Zydus Wellness Ltd.	ZYDUSWELL	Fast Moving Consumer Goods	EQ	1635.00	1635.00	1605.00	1636.85	1627.0

In [5]:

```
#To know the type of data
df.dtypes
```

Out[5]:

```
Company Name                   object
Symbol                         object
Industry                       object
Series                         object
Open                          float64
High                          float64
Low                           float64
Previous Close                float64
Last Traded Price             float64
Change                         object
Percentage Change              object
Share Volume                    int64
Value (Indian Rupee)          float64
52 Week High                  float64
52 Week Low                   float64
365 Day Percentage Change      object
30 Day Percentage Change       object
dtype: object
```

Info about the Dataset

In [6]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 501 entries, 0 to 500
Data columns (total 17 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   Company Name               501 non-null    object 
 1   Symbol                     501 non-null    object 
 2   Industry                   501 non-null    object 
 3   Series                     501 non-null    object 
 4   Open                       501 non-null    float64
 5   High                       501 non-null    float64
 6   Low                        501 non-null    float64
 7   Previous Close             501 non-null    float64
 8   Last Traded Price          501 non-null    float64
 9   Change                     501 non-null    object 
 10  Percentage Change          501 non-null    object 
 11  Share Volume               501 non-null    int64  
 12  Value (Indian Rupee)       501 non-null    float64
 13  52 Week High               501 non-null    float64
 14  52 Week Low                501 non-null    float64
 15  365 Day Percentage Change  501 non-null    object 
 16  30 Day Percentage Change   501 non-null    object 
dtypes: float64(8), int64(1), object(8)
memory usage: 66.7+ KB
```

In [7]:

```
#To Display the columns of the Dataset
df.columns
```

Out[7]:

```
Index(['Company Name', 'Symbol', 'Industry', 'Series', 'Open', 'High', 'Low',
       'Previous Close', 'Last Traded Price', 'Change', 'Percentage Change',
       'Share Volume', 'Value (Indian Rupee)', '52 Week High', '52 Week Low',
       '365 Day Percentage Change', '30 Day Percentage Change'],
      dtype='object')
```

Getting Shapes

In [8]:

```
# To enables us to obtain the shape of a DataFrame.
df.shape
```

Out[8]:
```
(501, 17)
```

Descriptive Statictics

In [9]:
```
# It returns description of the data in the DataFrame.
df.describe()
```

Out[9]:

	Open	High	Low	Previous Close	Last Traded Price	Share Volume	Value
count	501.000000	501.000000	501.000000	501.000000	501.000000	5.010000e+02	5.010
mean	1525.904491	1553.804990	1504.042415	1528.061277	1536.925449	2.580350e+06	8.635
std	4466.627117	4576.377692	4435.492332	4477.209376	4532.004734	9.407021e+06	4.335
min	6.750000	6.950000	6.700000	6.850000	6.800000	1.507000e+03	2.587
25%	215.300000	221.550000	210.600000	217.200000	214.650000	7.740500e+04	4.502
50%	551.100000	569.100000	547.000000	554.750000	563.000000	3.296100e+05	1.533
75%	1404.500000	1421.250000	1396.850000	1411.700000	1410.000000	1.235612e+06	6.644
max	70300.000000	72500.000000	70300.000000	70800.900000	71900.000000	1.257883e+08	9.211

Finding Duplicate Values

In [10]:
```
# It returns a Series with True and False values that describe which rows in the D
df.duplicated().sum()
```

Out[10]:
```
0
```

Data Cleaning

In [11]:
```
# It Detects missing values.
df.isna().any()
```

Out[11]:
```
Company Name                False
Symbol                      False
Industry                    False
Series                      False
Open                        False
High                        False
Low                         False
Previous Close              False
Last Traded Price           False
Change                      False
Percentage Change           False
Share Volume                False
Value (Indian Rupee)        False
52 Week High                False
52 Week Low                 False
365 Day Percentage Change   False
30 Day Percentage Change    False
dtype: bool
```

Observation

- There are no missing values.

Getting unique values

In [12]:

```
#Returns the count of unqiue values in column.
df.nunique()
```

Out[12]:

```
Company Name                 501
Symbol                       501
Industry                      21
Series                         2
Open                         492
High                         495
Low                          493
Previous Close               495
Last Traded Price            493
Change                       372
Percentage Change            354
Share Volume                 501
Value (Indian Rupee)         501
52 Week High                 497
52 Week Low                  494
365 Day Percentage Change    441
30 Day Percentage Change     458
dtype: int64
```

Observation

- There are 21 unique industries which are listed in NIFTY-500

Dropping Symbols and Series Columns

In [13]:

```
df.drop(columns=['Symbol','Series'],inplace=True)
```

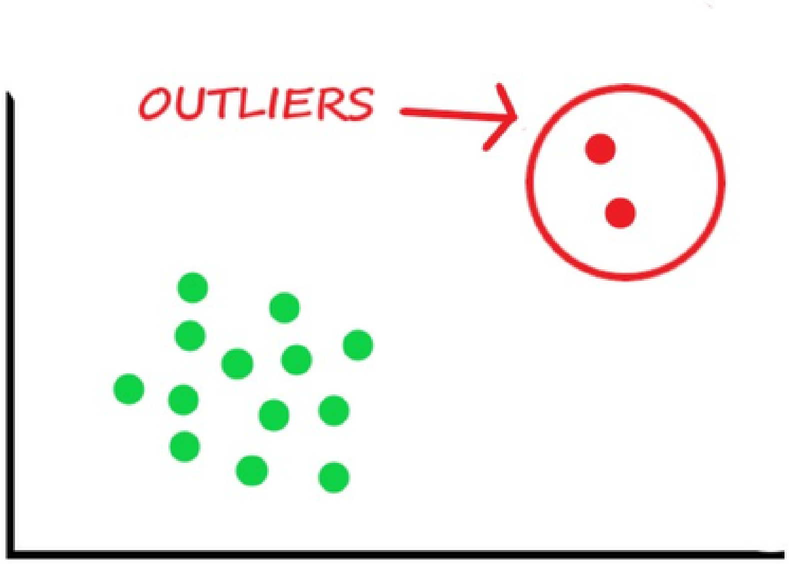
In [14]:

```
df.columns
```

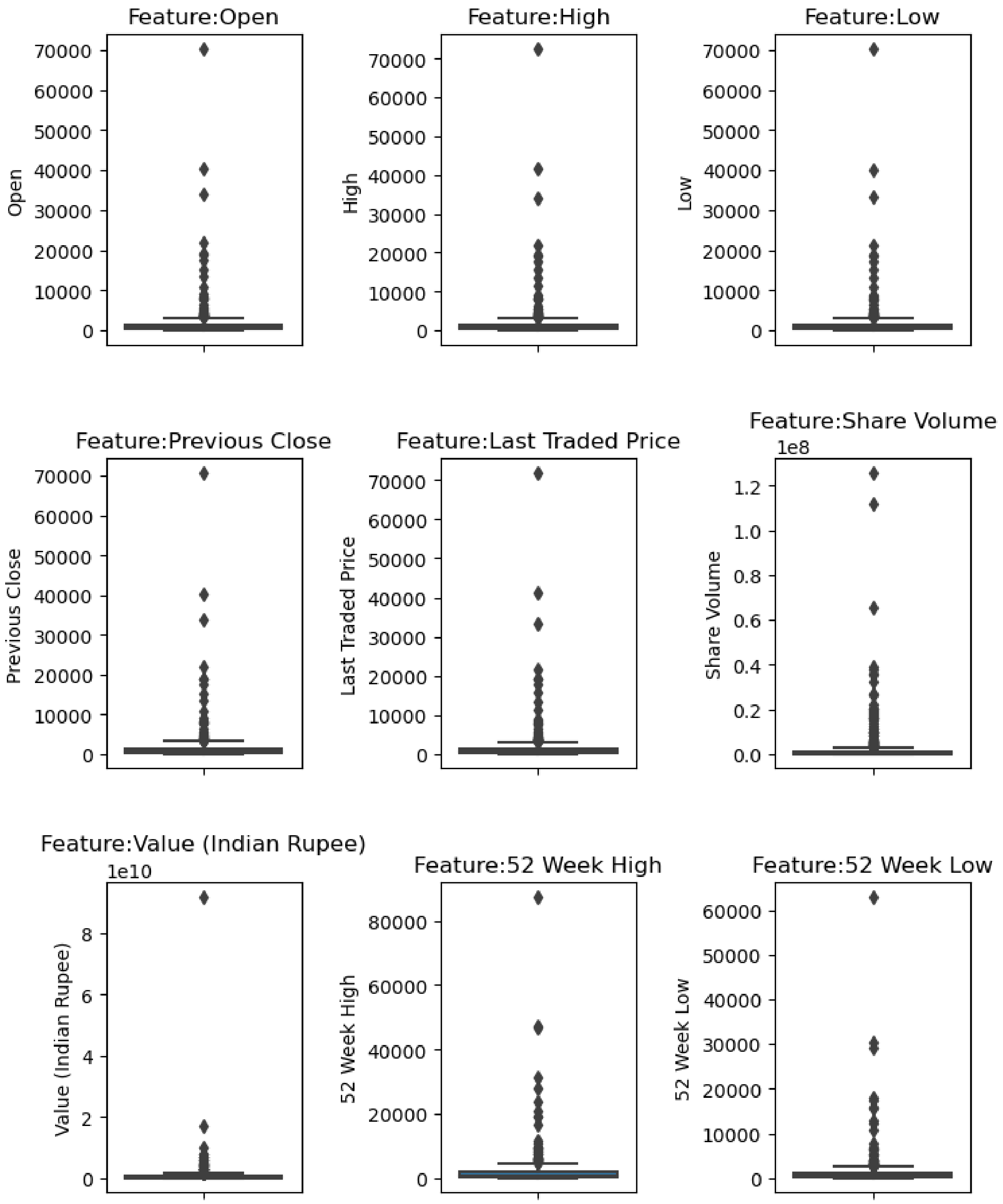
Out[14]:

```
Index(['Company Name', 'Industry', 'Open', 'High', 'Low', 'Previous Close',
       'Last Traded Price', 'Change', 'Percentage Change', 'Share Volume',
       'Value (Indian Rupee)', '52 Week High', '52 Week Low',
       '365 Day Percentage Change', '30 Day Percentage Change'],
      dtype='object')
```

Detecting Outliers



In [15]:
```
fig, axs = plt.subplots(3, 3, figsize=(8,10))
fig.tight_layout(pad=4.0)
features = ['Open', 'High', 'Low', 'Previous Close', 'Last Traded Price','Share Vo
for f,ax in zip(features,axs.ravel()):
    ax=sns.boxplot(ax=ax,data=df,y=df[f])
    ax.set_title('Feature:'+ f)
```

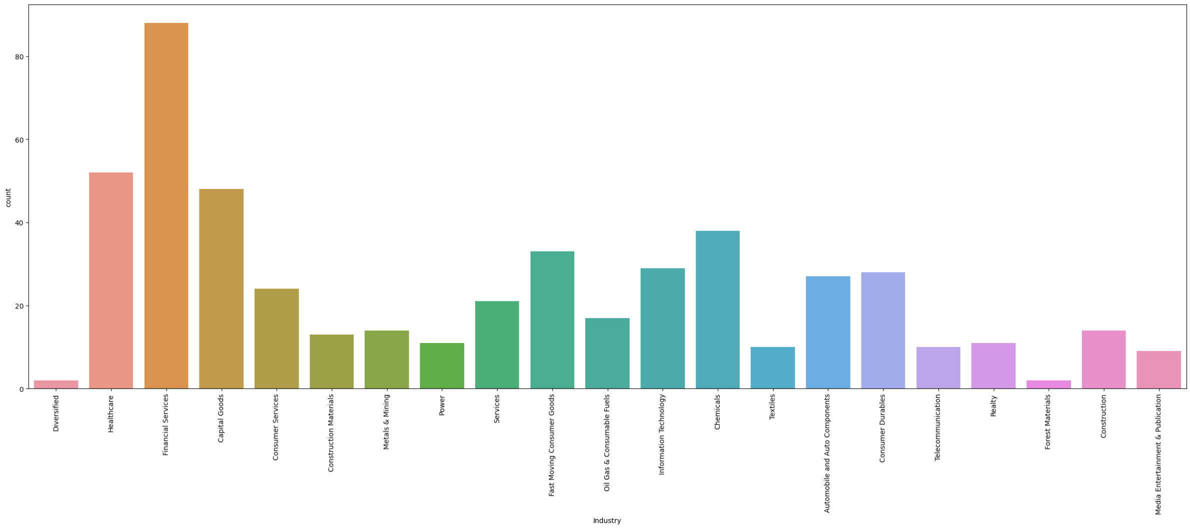


Data Visualization

Distributions of Industries

In [16]:
```
plt.figure(figsize=(30,10))
sns.countplot(x ='Industry', data = df)
plt.xticks(rotation=90)
```

```
Out[16]: (array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
                 17, 18, 19, 20]),
          [Text(0, 0, 'Diversified'),
           Text(1, 0, 'Healthcare'),
           Text(2, 0, 'Financial Services'),
           Text(3, 0, 'Capital Goods'),
           Text(4, 0, 'Consumer Services'),
           Text(5, 0, 'Construction Materials'),
           Text(6, 0, 'Metals & Mining'),
           Text(7, 0, 'Power'),
           Text(8, 0, 'Services'),
           Text(9, 0, 'Fast Moving Consumer Goods'),
           Text(10, 0, 'Oil Gas & Consumable Fuels'),
           Text(11, 0, 'Information Technology'),
           Text(12, 0, 'Chemicals'),
           Text(13, 0, 'Textiles'),
           Text(14, 0, 'Automobile and Auto Components'),
           Text(15, 0, 'Consumer Durables'),
           Text(16, 0, 'Telecommunication'),
           Text(17, 0, 'Realty'),
           Text(18, 0, 'Forest Materials'),
           Text(19, 0, 'Construction'),
           Text(20, 0, 'Media Entertainment & Publication')])
```

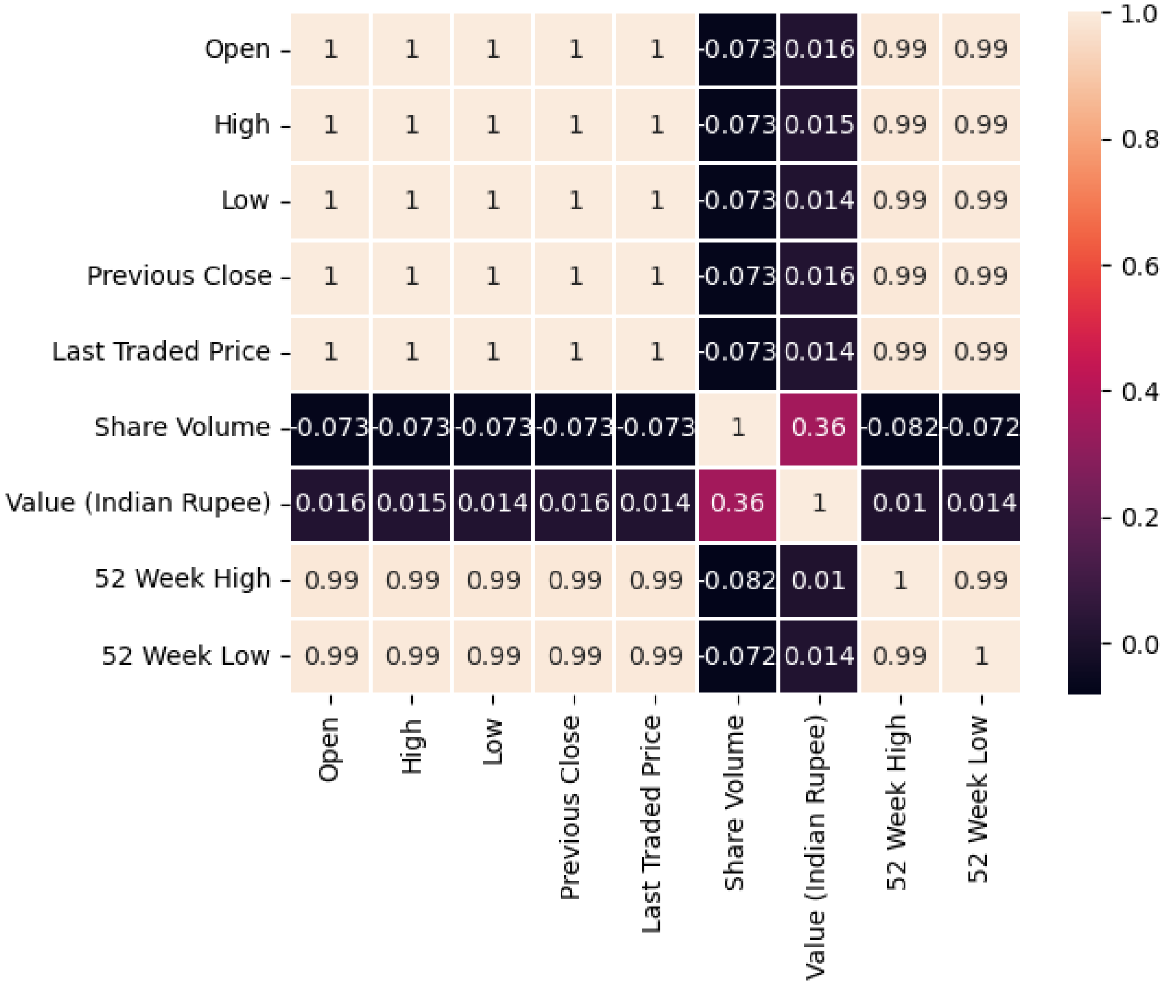


Observations

- Financial Services Industry contributes more value counts
- Forest Materials and Diversified industries have low value counts

```
In [17]: # a graphical representation of data using colors to visualize the value of the mat
         sns.heatmap(df.corr(),annot=True,linewidths=0.2) #Brighter color represent less co
```

```
Out[17]: <Axes: >
```



	Open	High	Low	Previous Close	Last Traded Price	Share Volume	Value (Indian Rupee)	52 Week High	52 Week Low
Open	1	1	1	1	1	-0.073	0.016	0.99	0.99
High	1	1	1	1	1	-0.073	0.015	0.99	0.99
Low	1	1	1	1	1	-0.073	0.014	0.99	0.99
Previous Close	1	1	1	1	1	-0.073	0.016	0.99	0.99
Last Traded Price	1	1	1	1	1	-0.073	0.014	0.99	0.99
Share Volume	-0.073	-0.073	-0.073	-0.073	-0.073	1	0.36	-0.082	-0.072
Value (Indian Rupee)	0.016	0.015	0.014	0.016	0.014	0.36	1	0.01	0.014
52 Week High	0.99	0.99	0.99	0.99	0.99	-0.082	0.01	1	0.99
52 Week Low	0.99	0.99	0.99	0.99	0.99	-0.072	0.014	0.99	1

Mean and median of (Open, High, Low, Previous Close, Last Traded Price) for each Industry

In [18]:
```
# Using groupby() method we can split the dataset into subsets to make computations
df.groupby(['Industry']).agg({ 'Open':[np.mean, np.median],'High':[np.mean, np.medi
```

Out[18]:

	Open		High		Low		
	mean	median	mean	median	mean	median	
Industry							
Automobile and Auto Components	4686.357407	925.000	4805.753704	947.000	4656.305556	900.000	4
Capital Goods	2023.683333	779.450	2043.246875	788.275	1982.681250	767.175	2
Chemicals	1653.782895	916.900	1677.843421	925.750	1622.730263	891.575	1
Construction	330.100000	216.750	332.957143	220.525	324.575000	213.375	
Construction Materials	2640.884615	874.900	2683.861538	877.100	2611.669231	826.000	2
Consumer Durables	1233.205357	925.350	1254.112500	934.375	1216.071429	911.500	1
Consumer Services	968.502083	505.125	980.156250	526.500	949.131250	499.275	
Diversified	11445.750000	11445.750	11476.675000	11476.675	11031.075000	11031.075	11
Fast Moving Consumer Goods	1674.628788	525.500	1710.254545	532.550	1661.478788	513.000	1
Financial Services	799.367045	413.575	820.278977	418.150	788.560227	406.000	
Forest Materials	553.000000	553.000	563.850000	563.850	547.025000	547.025	
Healthcare	1565.975962	628.500	1591.922115	637.450	1548.542308	622.350	1
Information Technology	1637.731034	997.000	1655.579310	1004.000	1606.179310	980.300	1
Media Entertainment & Publication	461.472222	362.900	470.227778	368.950	451.922222	353.650	
Metals & Mining	399.625000	233.550	407.957143	236.300	392.725000	225.675	
Oil Gas & Consumable Fuels	509.114706	218.000	517.217647	222.400	489.620588	214.800	
Power	530.095455	202.700	544.522727	207.650	501.059091	199.000	
Realty	592.168182	498.300	608.472727	514.000	585.318182	491.550	
Services	1035.950000	474.000	1049.730952	474.000	1019.664286	455.000	1
Telecommunication	357.950000	134.950	365.585000	136.400	353.430000	131.775	
Textiles	4662.745000	390.725	4806.150000	394.825	4637.955000	384.300	4

Average Valuation of each Industry

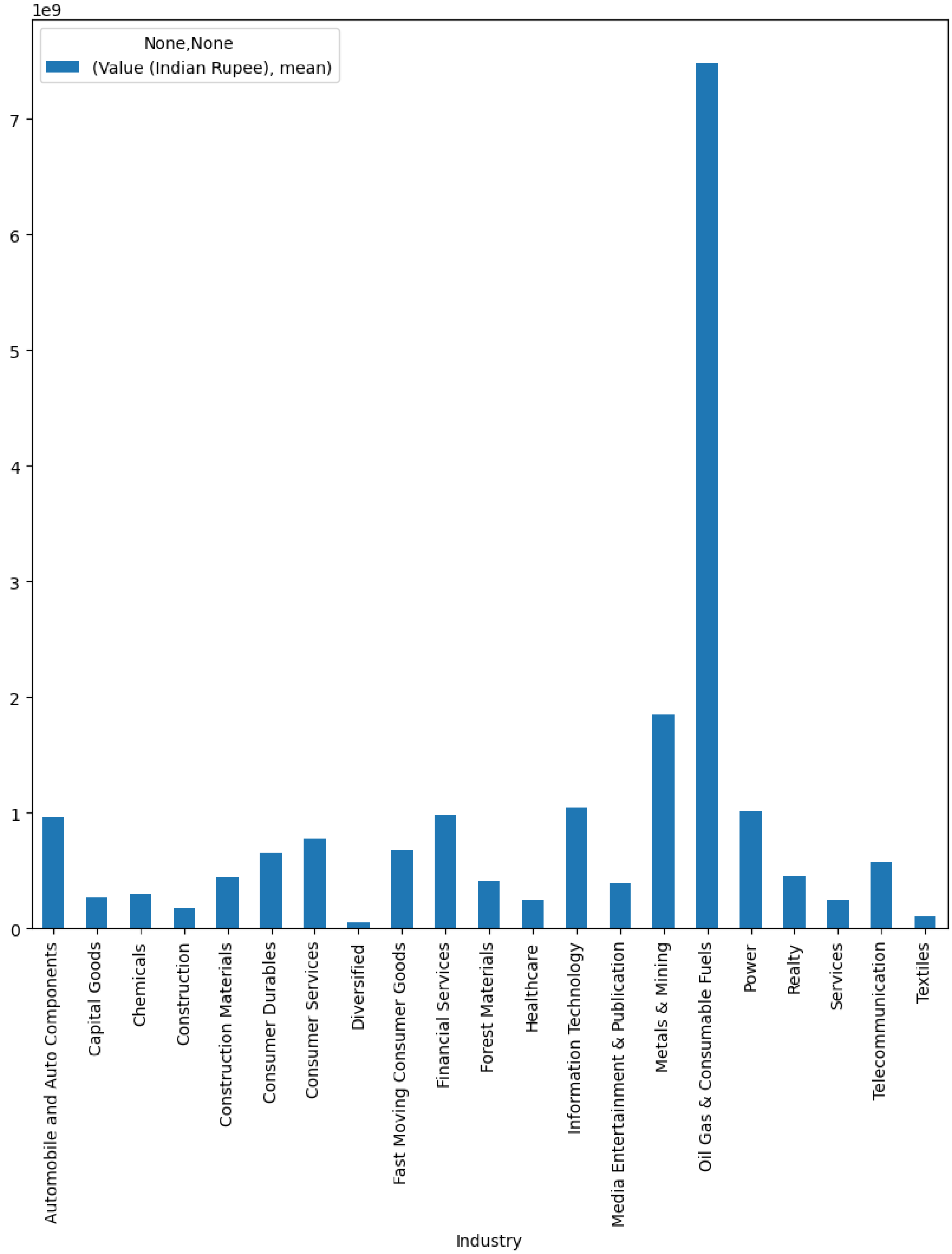
In [19]:
```
df.groupby(['Industry']).agg({ 'Value (Indian Rupee)':[np.mean]})
```

Out[19]:

	Value (Indian Rupee)
	mean
Industry	
Automobile and Auto Components	9.584557e+08
Capital Goods	2.629504e+08
Chemicals	3.017017e+08
Construction	1.801224e+08
Construction Materials	4.451913e+08
Consumer Durables	6.549041e+08
Consumer Services	7.753057e+08
Diversified	5.731713e+07
Fast Moving Consumer Goods	6.800218e+08
Financial Services	9.786358e+08
Forest Materials	4.092066e+08
Healthcare	2.479522e+08
Information Technology	1.047738e+09
Media Entertainment & Publication	3.897143e+08
Metals & Mining	1.851775e+09
Oil Gas & Consumable Fuels	7.484906e+09
Power	1.015213e+09
Realty	4.495636e+08
Services	2.436190e+08
Telecommunication	5.706232e+08
Textiles	1.020310e+08

In [20]:
```
df.groupby(['Industry']).agg({ 'Value (Indian Rupee)':[np.mean]}).plot(kind="bar",
```

Out[20]:
```
<Axes: xlabel='Industry'>
```



Observations

- Oil Gas and Consumable Fules Industry has the highest valuation followed by Metals & Publication Industry.
- Diversified Industry has lowest valuations followed by Textile Industry.

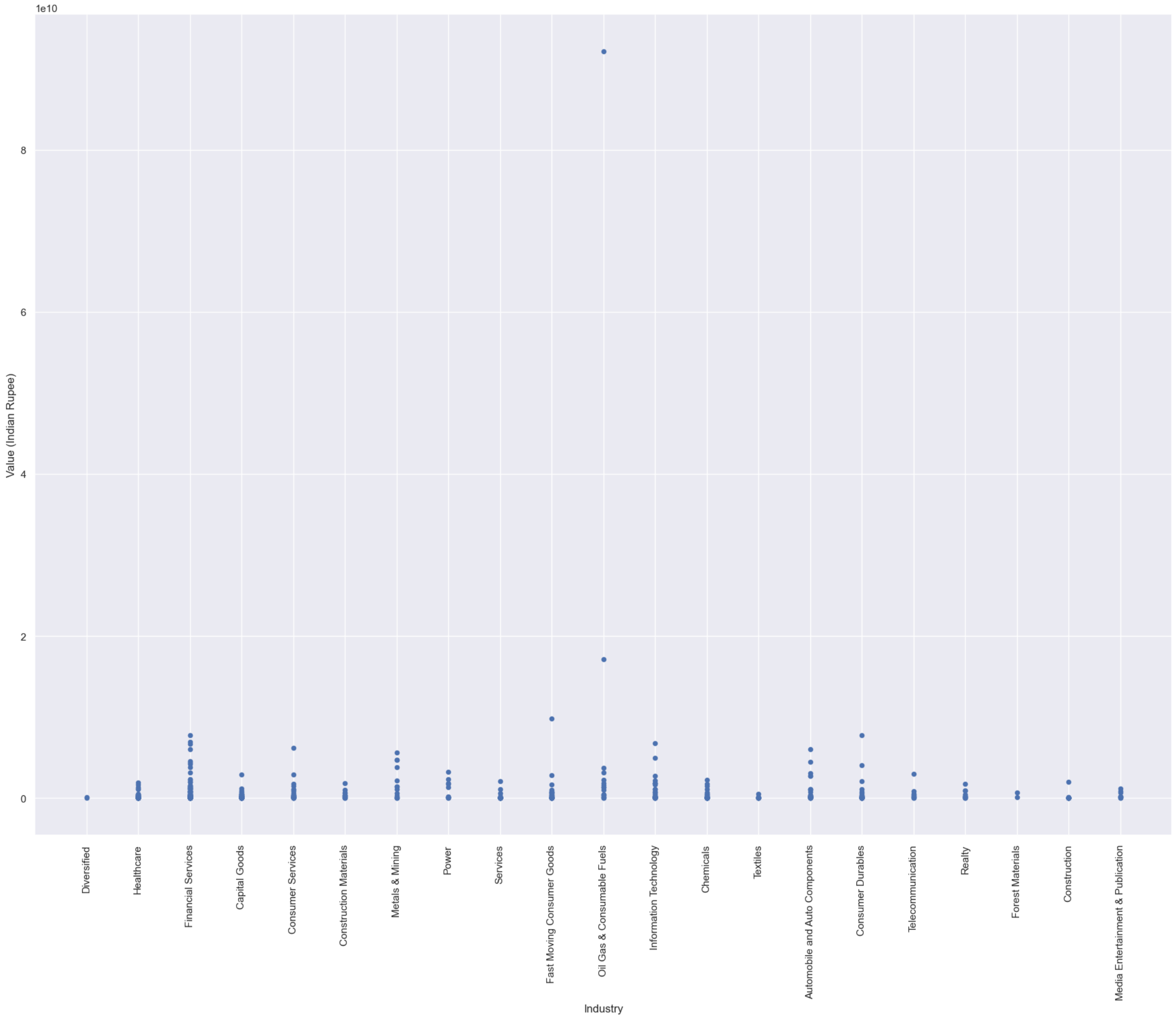
Scatter plot for Valuations

In [43]:
```
df.plot.scatter(x='Industry', y='Value (Indian Rupee)')
```

```
plt.xticks(rotation=90)          # xticks function is used to get or set the curr
```

Out[43]:

```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20],
 [Text(0, 0, 'Diversified'),
  Text(1, 0, 'Healthcare'),
  Text(2, 0, 'Financial Services'),
  Text(3, 0, 'Capital Goods'),
  Text(4, 0, 'Consumer Services'),
  Text(5, 0, 'Construction Materials'),
  Text(6, 0, 'Metals & Mining'),
  Text(7, 0, 'Power'),
  Text(8, 0, 'Services'),
  Text(9, 0, 'Fast Moving Consumer Goods'),
  Text(10, 0, 'Oil Gas & Consumable Fuels'),
  Text(11, 0, 'Information Technology'),
  Text(12, 0, 'Chemicals'),
  Text(13, 0, 'Textiles'),
  Text(14, 0, 'Automobile and Auto Components'),
  Text(15, 0, 'Consumer Durables'),
  Text(16, 0, 'Telecommunication'),
  Text(17, 0, 'Realty'),
  Text(18, 0, 'Forest Materials'),
  Text(19, 0, 'Construction'),
  Text(20, 0, 'Media Entertainment & Publication')])
```

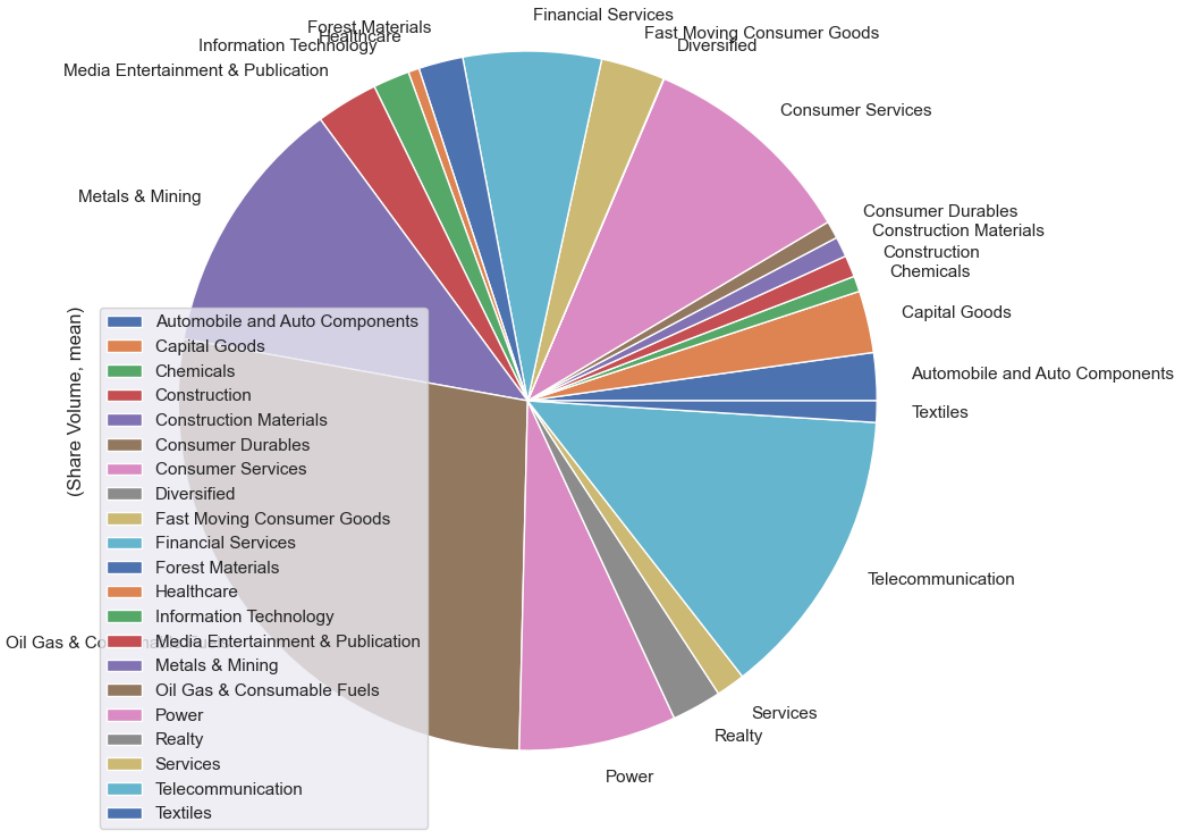


In [40]:

```
df.groupby(['Industry']).agg({ 'Share Volume':[np.mean]}).plot(kind="pie",subplots=
```

Out[40]:

```
(array([], dtype=float64), [])
```



Observations

- Oil Gas & Consumable Fuels industries has the highest share volume.