Basic operations using NumPy and Pandas

Importing libraries

In [1]:

```
import numpy as np
import pandas as pd
```

importing the dataset

In [2]:

```
data=pd.read_csv(r"C:\Users\user\Desktop\Ash\fiat500_VehicleSelection_Dataset.csv")
```

Selecting first 10 rows using head()

```
In [3]: data.head(10)
```

Out[3]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price	Unnamed: 9	Unnamed: 10	Unnam
0	1.0	lounge	51.0	882.0	25000.0	1.0	44.907242	8.611560	8900.0	NaN	NaN	CONCATE
1	2.0	pop	51.0	1186.0	32500.0	1.0	45.666359	12.241890	8800.0	NaN	NaN	Length
2	3.0	sport	74.0	4658.0	142228.0	1.0	45.503300	11.417840	4200.0	NaN	NaN	No. of w days btw 23 to 17
3	4.0	lounge	51.0	2739.0	160000.0	1.0	40.633171	17.634609	6000.0	NaN	NaN	
4	5.0	pop	73.0	3074.0	106880.0	1.0	41.903221	12.495650	5700.0	NaN	NaN	Ave
5	6.0	pop	74.0	3623.0	70225.0	1.0	45.000702	7.682270	7900.0	NaN	NaN	c
6	7.0	lounge	51.0	731.0	11600.0	1.0	44.907242	8.611560	10750.0	NaN	NaN	
7	8.0	lounge	51.0	1521.0	49076.0	1.0	41.903221	12.495650	9190.0	NaN	NaN	
8	9.0	sport	73.0	4049.0	76000.0	1.0	45.548000	11.549470	5600.0	NaN	NaN	
9	10.0	sport	51.0	3653.0	89000.0	1.0	45.438301	10.991700	6000.0	NaN	NaN	

Selecting last 10 rows using tail()

```
In [4]: data.tail(10)
```

Out[4]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price	Unnamed: 9
1531	1532.0	sport	73.0	4505.0	127000.0000	1.0	45.528511	9.59323	4.750000e+03	NaN
1532	1533.0	pop	51.0	1917.0	52008.0000	1.0	45.548000	11.54947	9.900000e+03	NaN
1533	1534.0	sport	51.0	3712.0	115280.0000	1.0	45.069679	7.70492	5.200000e+03	NaN
1534	1535.0	lounge	74.0	3835.0	112000.0000	1.0	45.845692	8.66687	4.600000e+03	NaN
1535	1536.0	pop	51.0	2223.0	60457.0000	1.0	45.481541	9.41348	7.500000e+03	NaN
1536	1537.0	lounge	51.0	2557.0	80750.0000	1.0	45.000702	7.68227	5.990000e+03	NaN
1537	1538.0	pop	51.0	1766.0	54276.0000	1.0	40.323410	17.56827	7.900000e+03	NaN
1538	NaN	NaN	NaN	2080506.0	53396.0117	NaN	66966.613720	17784.55279	1.318989e+07	NaN
1539	NaN	NaN	1538.0	NaN	NaN	NaN	NaN	NaN	1.714086e+04	NaN
1540	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.318989e+07	NaN

To get the statistical data of the table

In [5]:
```
data.describe()
```

Out[5]:

	ID	engine_power	age_in_days	km	previous_owners	lat	lon	price	Unnamed: 9
count	1538.000000	1539.000000	1.539000e+03	1539.000000	1538.000000	1539.000000	1539.000000	1.541000e+03	0.0
mean	769.500000	52.870045	3.001763e+03	53396.011704	1.123537	87.026139	23.111829	2.568905e+04	NaN
std	444.126671	38.090731	5.300702e+04	40033.809481	0.416423	1705.913084	453.050800	4.747172e+05	NaN
min	1.000000	51.000000	3.660000e+02	1232.000000	1.000000	36.855839	7.245400	2.500000e+03	NaN
25%	385.250000	51.000000	6.700000e+02	20012.500000	1.000000	41.802990	9.505090	7.190000e+03	NaN
50%	769.500000	51.000000	1.035000e+03	39038.000000	1.000000	44.399971	11.869260	9.000000e+03	NaN
75%	1153.750000	51.000000	2.616000e+03	79535.500000	1.000000	45.467960	12.774520	1.000000e+04	NaN
max	1538.000000	1538.000000	2.080506e+06	235000.000000	4.000000	66966.613720	17784.552790	1.318989e+07	NaN

To find the row and column of the table

In [6]:
```
data.shape
```

Out[6]:
```
(1541, 15)
```

To find the size of the table

In [7]:
```
data.size
```

Out[7]:
```
23115
```

Find missing values

```
In [8]: data.isna()
```

Out[8]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price	Unnamed: 9	Unnamed: 10	Unnamed: 11	Unname
0	False	False	False	False	False	False	False	False	False	True	True	False	Fal
1	False	False	False	False	False	False	False	False	False	True	True	False	Fal
2	False	False	False	False	False	False	False	False	False	True	True	False	Fal
3	False	False	False	False	False	False	False	False	False	True	True	False	Fal
4	False	False	False	False	False	False	False	False	False	True	True	False	Fal
...	...	...	...	...	...	...	...	...	...	...	...	...	
1536	False	False	False	False	False	False	False	False	False	True	True	True	Tr
1537	False	False	False	False	False	False	False	False	False	True	True	True	Tr
1538	True	True	True	False	False	True	False	False	False	True	True	True	Tr
1539	True	True	False	True	True	True	True	True	False	True	True	True	Tr
1540	True	True	True	True	True	True	True	True	False	True	True	True	Tr

1541 rows × 15 columns

filling the missing values as 4

In [9]:

```
data.fillna(value=4)
```

Out[9]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price	Unnamed: 9
0	1.0	lounge	51.0	882.0	25000.0000	1.0	44.907242	8.611560	8.900000e+03	4.0
1	2.0	pop	51.0	1186.0	32500.0000	1.0	45.666359	12.241890	8.800000e+03	4.0
2	3.0	sport	74.0	4658.0	142228.0000	1.0	45.503300	11.417840	4.200000e+03	4.0
3	4.0	lounge	51.0	2739.0	160000.0000	1.0	40.633171	17.634609	6.000000e+03	4.0
4	5.0	pop	73.0	3074.0	106880.0000	1.0	41.903221	12.495650	5.700000e+03	4.0
...	...	...	...	...	...	...	...	...	...	...
1536	1537.0	lounge	51.0	2557.0	80750.0000	1.0	45.000702	7.682270	5.990000e+03	4.0
1537	1538.0	pop	51.0	1766.0	54276.0000	1.0	40.323410	17.568270	7.900000e+03	4.0
1538	4.0	4	4.0	2080506.0	53396.0117	4.0	66966.613720	17784.552790	1.318989e+07	4.0
1539	4.0	4	1538.0	4.0	4.0000	4.0	4.000000	4.000000	1.714086e+04	4.0
1540	4.0	4	4.0	4.0	4.0000	4.0	4.000000	4.000000	1.318989e+07	4.0

1541 rows × 15 columns

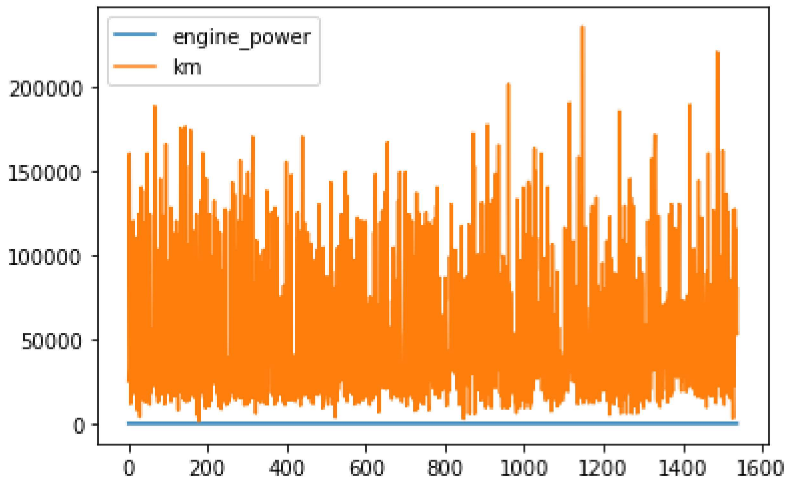
Visualization

In [24]:

```
df=data[['engine_power','km']]
```

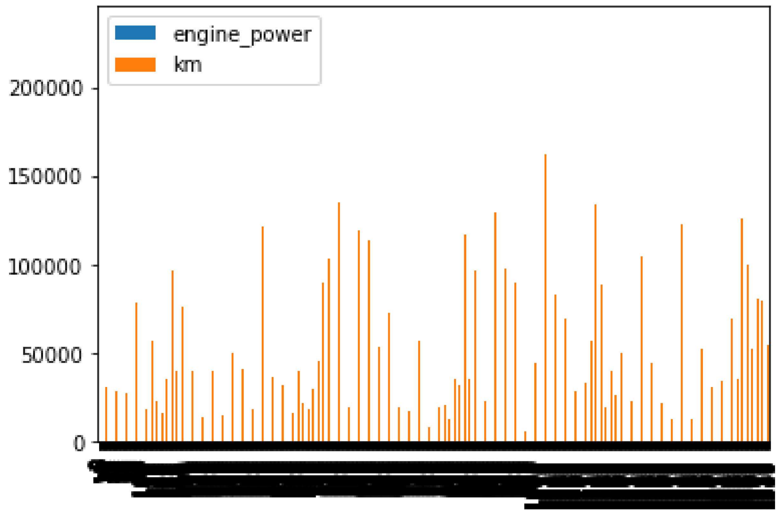
In [25]:
```
df.plot.line()
```

Out[25]:
```
<AxesSubplot:>
```



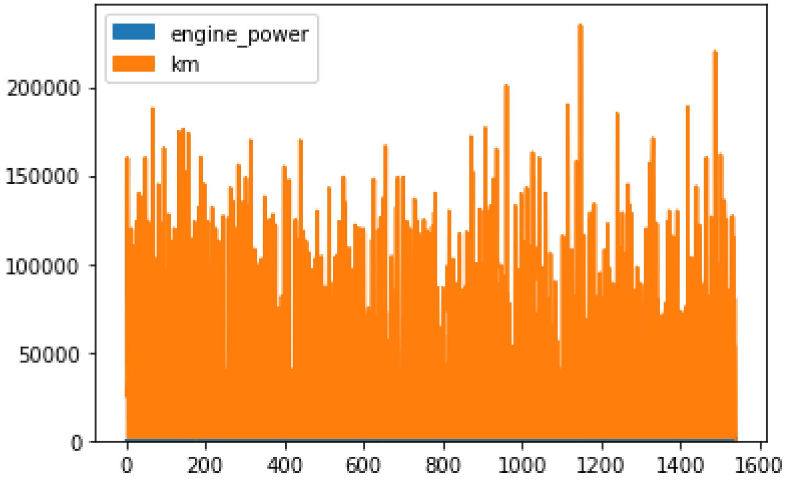
In [26]:
```
df.plot.bar()
```

Out[26]:
```
<AxesSubplot:>
```



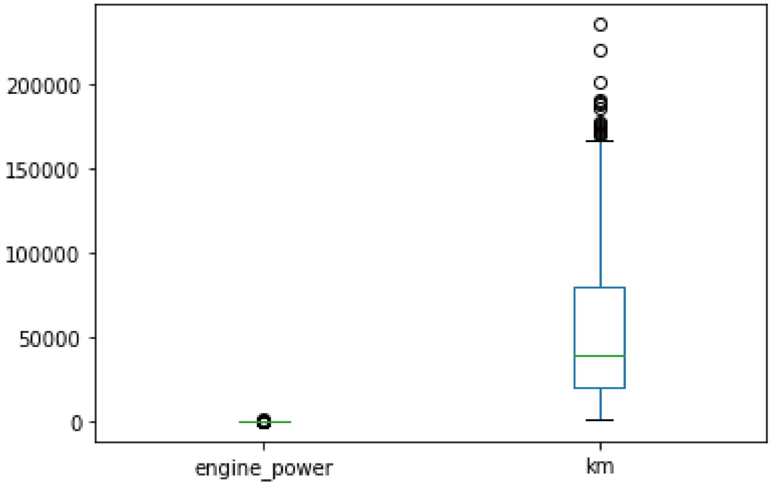
```
In [27]: df.plot.area()
```

```
Out[27]: <AxesSubplot:>
```



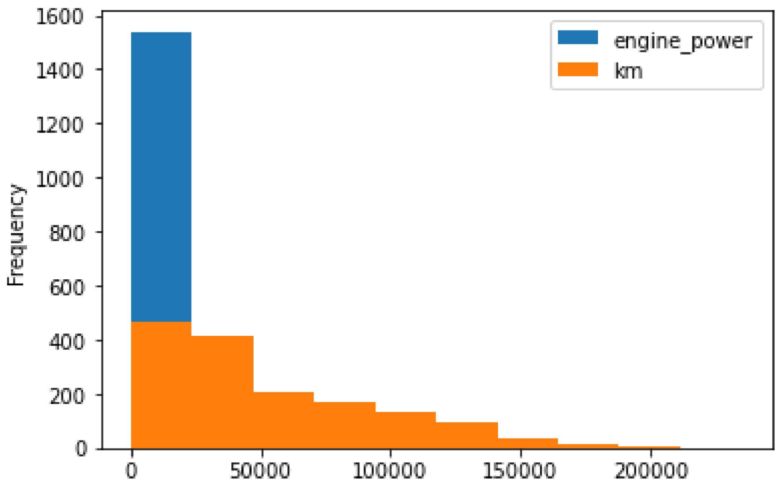
```
In [28]: df.plot.box()
```

```
Out[28]: <AxesSubplot:>
```



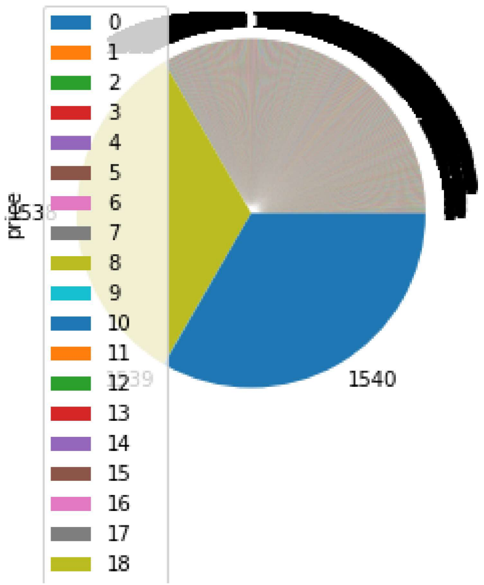
```
In [29]: df.plot.hist()
```

```
Out[29]: <AxesSubplot:ylabel='Frequency'>
```



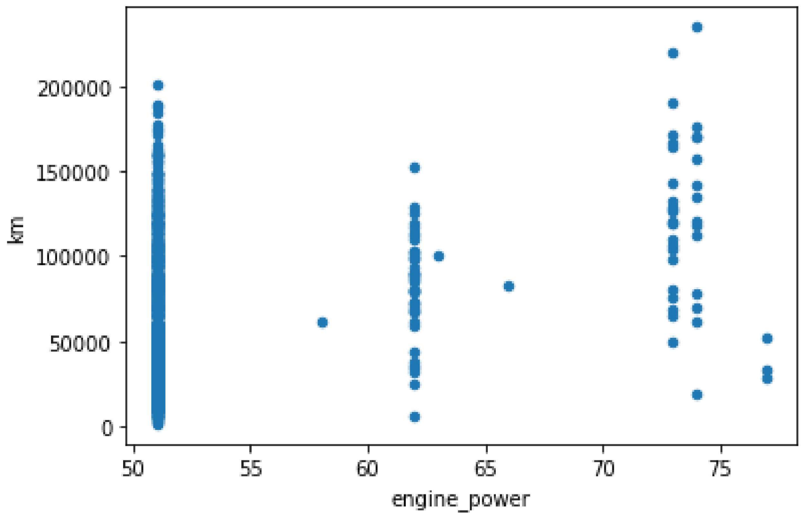
```
In [30]: data.plot.pie(y='price')
```

```
Out[30]: <AxesSubplot:ylabel='price'>
```



```
In [34]: data.plot.scatter(x='engine_power',y='km')
```

```
Out[34]: <AxesSubplot:xlabel='engine_power', ylabel='km'>
```



Data Set 2 [2015]

Importing the dataset

```
In [11]: data1=pd.read_csv(r"C:\Users\user\Desktop\Ash\2015.csv")
```

Select first 10 rows

In [12]:

```
data1.head(10)
```

Out[12]:

	Country	Region	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dystopia Residua
0	Switzerland	Western Europe	1	7.587	0.03411	1.39651	1.34951	0.94143	0.66557	0.41978	0.29678	2.51738
1	Iceland	Western Europe	2	7.561	0.04884	1.30232	1.40223	0.94784	0.62877	0.14145	0.43630	2.70201
2	Denmark	Western Europe	3	7.527	0.03328	1.32548	1.36058	0.87464	0.64938	0.48357	0.34139	2.49204
3	Norway	Western Europe	4	7.522	0.03880	1.45900	1.33095	0.88521	0.66973	0.36503	0.34699	2.46531
4	Canada	North America	5	7.427	0.03553	1.32629	1.32261	0.90563	0.63297	0.32957	0.45811	2.45176
5	Finland	Western Europe	6	7.406	0.03140	1.29025	1.31826	0.88911	0.64169	0.41372	0.23351	2.61955
6	Netherlands	Western Europe	7	7.378	0.02799	1.32944	1.28017	0.89284	0.61576	0.31814	0.47610	2.46570
7	Sweden	Western Europe	8	7.364	0.03157	1.33171	1.28907	0.91087	0.65980	0.43844	0.36262	2.37119
8	New Zealand	Australia and New Zealand	9	7.286	0.03371	1.25018	1.31967	0.90837	0.63938	0.42922	0.47501	2.26425
9	Australia	Australia and New Zealand	10	7.284	0.04083	1.33358	1.30923	0.93156	0.65124	0.35637	0.43562	2.26646

Select the last 10 rows

In [13]:

```
data1.tail(10)
```

Out[13]:

	Country	Region	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dysto Resid
148	Chad	Sub-Saharan Africa	149	3.667	0.03830	0.34193	0.76062	0.15010	0.23501	0.05269	0.18386	1.94
149	Guinea	Sub-Saharan Africa	150	3.656	0.03590	0.17417	0.46475	0.24009	0.37725	0.12139	0.28657	1.99
150	Ivory Coast	Sub-Saharan Africa	151	3.655	0.05141	0.46534	0.77115	0.15185	0.46866	0.17922	0.20165	1.41
151	Burkina Faso	Sub-Saharan Africa	152	3.587	0.04324	0.25812	0.85188	0.27125	0.39493	0.12832	0.21747	1.46
152	Afghanistan	Southern Asia	153	3.575	0.03084	0.31982	0.30285	0.30335	0.23414	0.09719	0.36510	1.95
153	Rwanda	Sub-Saharan Africa	154	3.465	0.03464	0.22208	0.77370	0.42864	0.59201	0.55191	0.22628	0.67
154	Benin	Sub-Saharan Africa	155	3.340	0.03656	0.28665	0.35386	0.31910	0.48450	0.08010	0.18260	1.63
155	Syria	Middle East and Northern Africa	156	3.006	0.05015	0.66320	0.47489	0.72193	0.15684	0.18906	0.47179	0.32
156	Burundi	Sub-Saharan Africa	157	2.905	0.08658	0.01530	0.41587	0.22396	0.11850	0.10062	0.19727	1.83
157	Togo	Sub-Saharan Africa	158	2.839	0.06727	0.20868	0.13995	0.28443	0.36453	0.10731	0.16681	1.56

To get the statistical data of the table

In [14]:
```
data1.describe()
```

Out[14]:

	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dystopia Residual
count	158.000000	158.000000	158.000000	158.000000	158.000000	158.000000	158.000000	158.000000	158.000000	158.000000
mean	79.493671	5.375734	0.047885	0.846137	0.991046	0.630259	0.428615	0.143422	0.237296	2.098977
std	45.754363	1.145010	0.017146	0.403121	0.272369	0.247078	0.150693	0.120034	0.126685	0.553550
min	1.000000	2.839000	0.018480	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.328580
25%	40.250000	4.526000	0.037268	0.545808	0.856823	0.439185	0.328330	0.061675	0.150553	1.759410
50%	79.500000	5.232500	0.043940	0.910245	1.029510	0.696705	0.435515	0.107220	0.216130	2.095415
75%	118.750000	6.243750	0.052300	1.158448	1.214405	0.811013	0.549092	0.180255	0.309883	2.462415
max	158.000000	7.587000	0.136930	1.690420	1.402230	1.025250	0.669730	0.551910	0.795880	3.602140

To find the row and columns of the table

In [15]:
```
data1.shape
```

Out[15]:
```
(158, 12)
```

Find the size of the table

In [16]:
```
data1.size
```

Out[16]:
```
1896
```

Find the missing values of the table

In [17]:

```
data1.isna()
```

Out[17]:

	Country	Region	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dystopia Residual
0	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...
153	False	False	False	False	False	False	False	False	False	False	False	False
154	False	False	False	False	False	False	False	False	False	False	False	False
155	False	False	False	False	False	False	False	False	False	False	False	False
156	False	False	False	False	False	False	False	False	False	False	False	False
157	False	False	False	False	False	False	False	False	False	False	False	False

158 rows × 12 columns

To delete to rows containing empty cells

In [18]:
```
data1.dropna()
```

Out[18]:

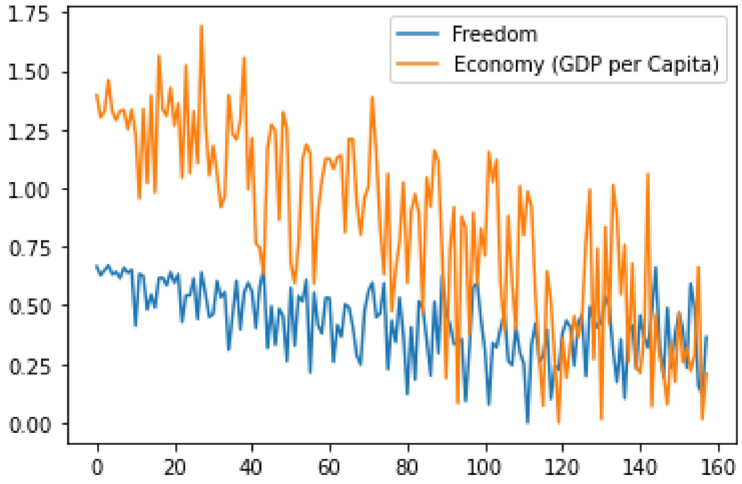
	Country	Region	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity	Dystop Residu
0	Switzerland	Western Europe	1	7.587	0.03411	1.39651	1.34951	0.94143	0.66557	0.41978	0.29678	2.517
1	Iceland	Western Europe	2	7.561	0.04884	1.30232	1.40223	0.94784	0.62877	0.14145	0.43630	2.702
2	Denmark	Western Europe	3	7.527	0.03328	1.32548	1.36058	0.87464	0.64938	0.48357	0.34139	2.492
3	Norway	Western Europe	4	7.522	0.03880	1.45900	1.33095	0.88521	0.66973	0.36503	0.34699	2.465
4	Canada	North America	5	7.427	0.03553	1.32629	1.32261	0.90563	0.63297	0.32957	0.45811	2.451
...	...	...	...	...	...	...	...	...	...	...	...	
153	Rwanda	Sub-Saharan Africa	154	3.465	0.03464	0.22208	0.77370	0.42864	0.59201	0.55191	0.22628	0.670
154	Benin	Sub-Saharan Africa	155	3.340	0.03656	0.28665	0.35386	0.31910	0.48450	0.08010	0.18260	1.633
155	Syria	Middle East and Northern Africa	156	3.006	0.05015	0.66320	0.47489	0.72193	0.15684	0.18906	0.47179	0.328
156	Burundi	Sub-Saharan Africa	157	2.905	0.08658	0.01530	0.41587	0.22396	0.11850	0.10062	0.19727	1.833
157	Togo	Sub-Saharan Africa	158	2.839	0.06727	0.20868	0.13995	0.28443	0.36453	0.10731	0.16681	1.567

158 rows × 12 columns

```
In [37]: df1= data1[['Freedom','Economy (GDP per Capita)']]
```

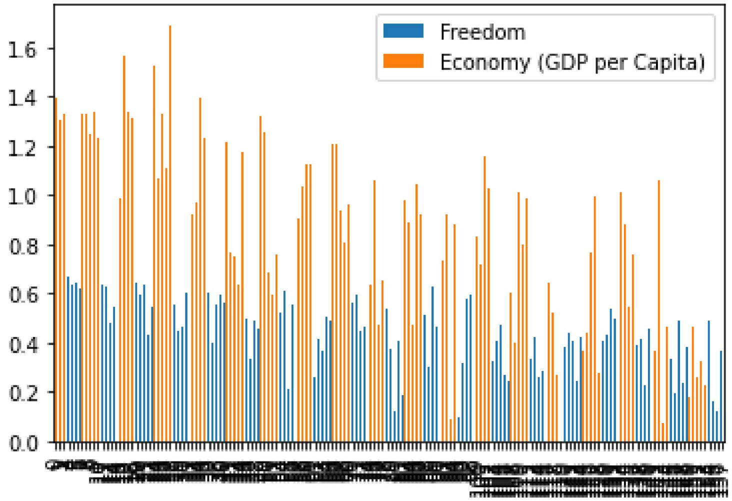
```
In [38]: df1.plot.line()
```

```
Out[38]: <AxesSubplot:>
```



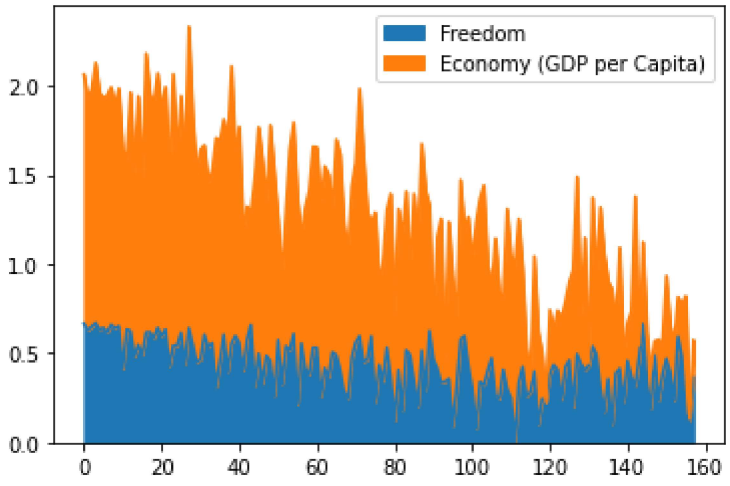
```
In [39]: df1.plot.bar()
```

```
Out[39]: <AxesSubplot:>
```



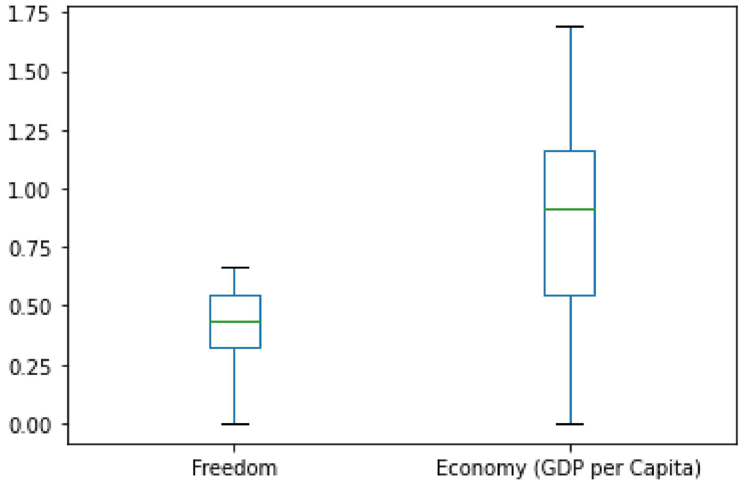
```
In [40]: df1.plot.area()
```

```
Out[40]: <AxesSubplot:>
```



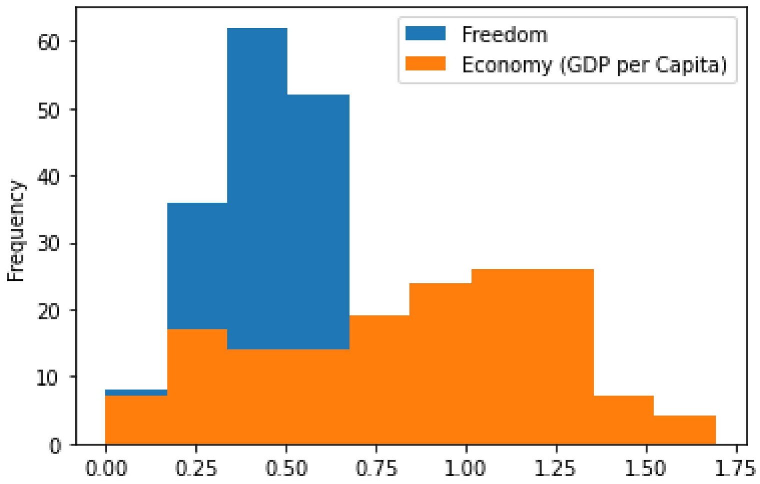
```
In [41]: df1.plot.box()
```

```
Out[41]: <AxesSubplot:>
```



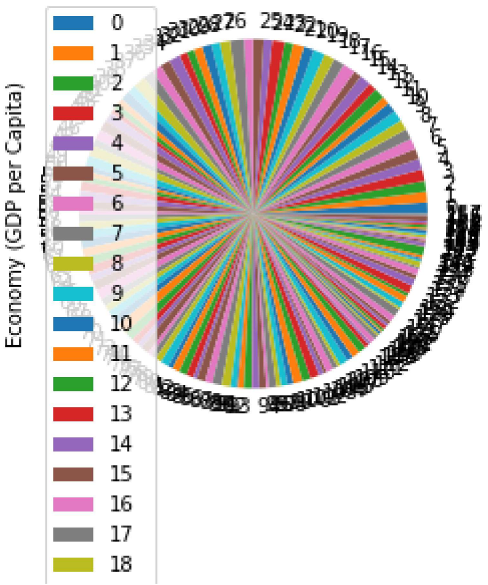
```
In [42]: df1.plot.hist()
```

```
Out[42]: <AxesSubplot:ylabel='Frequency'>
```



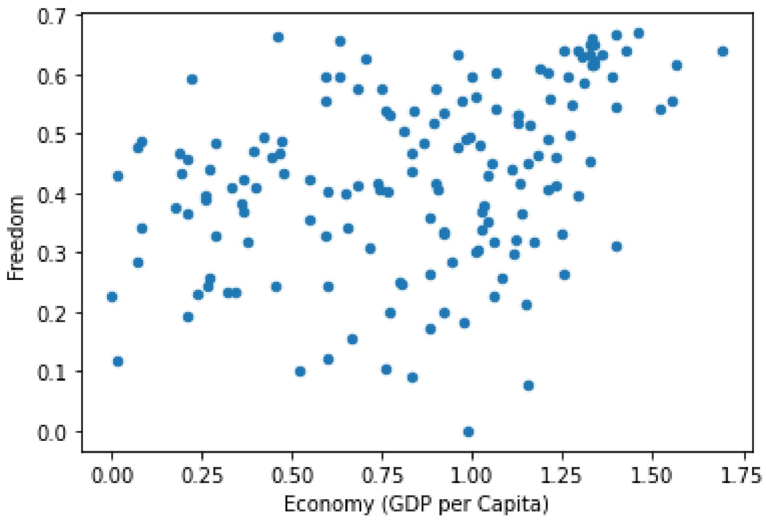
```
In [43]: data1.plot.pie(y='Economy (GDP per Capita)')
```

```
Out[43]: <AxesSubplot:ylabel='Economy (GDP per Capita)'>
```



```
In [44]: data1.plot.scatter(x='Economy (GDP per Capita)',y='Freedom')
```

```
Out[44]: <AxesSubplot:xlabel='Economy (GDP per Capita)', ylabel='Freedom'>
```


Freedom
Economy (GDP per Capita)