# EDA with Data Collection, Data Cleaning and Pre-processing with

# Instagram dataset

# Data cleaning and Pre-procrossing

To import library

In [1]:

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

To import dataset

In [2]:

```
d=pd.read_csv(r"c:\Users\user\Downloads\5_instagram data.csv")
d
```

Out[2]:

| | Impressions | From Home | From Hashtags | From Explore | From Other | Saves | Comments | Shares | Likes | Profile Visits |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 3920 | 2586 | 1028 | 619 | 56 | 98 | 9 | 5 | 162 | 35 |
| **1** | 5394 | 2727 | 1838 | 1174 | 78 | 194 | 7 | 14 | 224 | 48 |
| **2** | 4021 | 2085 | 1188 | 0 | 533 | 41 | 11 | 1 | 131 | 62 |
| **3** | 4528 | 2700 | 621 | 932 | 73 | 172 | 10 | 7 | 213 | 23 |
| **4** | 2518 | 1704 | 255 | 279 | 37 | 96 | 5 | 4 | 123 | 8 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | .. |
| **114** | 13700 | 5185 | 3041 | 5352 | 77 | 573 | 2 | 38 | 373 | 73 |
| **115** | 5731 | 1923 | 1368 | 2266 | 65 | 135 | 4 | 1 | 148 | 20 |
| **116** | 4139 | 1133 | 1538 | 1367 | 33 | 36 | 0 | 1 | 92 | 34 |
| **117** | 32695 | 11815 | 3147 | 17414 | 170 | 1095 | 2 | 75 | 549 | 148 |
| **118** | 36919 | 13473 | 4176 | 16444 | 2547 | 653 | 5 | 26 | 443 | 611 |

119 rows × 13 columns

To get top 10 record

In [3]:

```
d.head(10)
```

Out[3]:

| | Impressions | From Home | From Hashtags | From Explore | From Other | Saves | Comments | Shares | Likes | Profile Visits |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 3920 | 2586 | 1028 | 619 | 56 | 98 | 9 | 5 | 162 | 35 |
| **1** | 5394 | 2727 | 1838 | 1174 | 78 | 194 | 7 | 14 | 224 | 48 |
| **2** | 4021 | 2085 | 1188 | 0 | 533 | 41 | 11 | 1 | 131 | 62 |
| **3** | 4528 | 2700 | 621 | 932 | 73 | 172 | 10 | 7 | 213 | 23 |
| **4** | 2518 | 1704 | 255 | 279 | 37 | 96 | 5 | 4 | 123 | 8 |
| **5** | 3884 | 2046 | 1214 | 329 | 43 | 74 | 7 | 10 | 144 | 9 |
| **6** | 2621 | 1543 | 599 | 333 | 25 | 22 | 5 | 1 | 76 | 26 |
| **7** | 3541 | 2071 | 628 | 500 | 60 | 135 | 4 | 9 | 124 | 12 |
| **8** | 3749 | 2384 | 857 | 248 | 49 | 155 | 6 | 8 | 159 | 36 |
| **9** | 4115 | 2609 | 1104 | 178 | 46 | 122 | 6 | 3 | 191 | 31 |

To get last 10

In [4]:

```
d.tail(10)
```

Out[4]:

| | Impressions | From Home | From Hashtags | From Explore | From Other | Saves | Comments | Shares | Likes | Profile Visits |
|---|---|---|---|---|---|---|---|---|---|---|
| **109** | 17713 | 2449 | 2141 | 12389 | 561 | 504 | 3 | 23 | 308 | 70 |
| **110** | 5563 | 3813 | 362 | 1135 | 76 | 149 | 5 | 8 | 163 | 22 |
| **111** | 4842 | 1658 | 694 | 2036 | 310 | 55 | 6 | 4 | 86 | 46 |
| **112** | 11149 | 4439 | 747 | 5762 | 53 | 273 | 4 | 13 | 210 | 61 |
| **113** | 10206 | 2371 | 1624 | 6000 | 117 | 182 | 10 | 17 | 172 | 237 |
| **114** | 13700 | 5185 | 3041 | 5352 | 77 | 573 | 2 | 38 | 373 | 73 |
| **115** | 5731 | 1923 | 1368 | 2266 | 65 | 135 | 4 | 1 | 148 | 20 |
| **116** | 4139 | 1133 | 1538 | 1367 | 33 | 36 | 0 | 1 | 92 | 34 |
| **117** | 32695 | 11815 | 3147 | 17414 | 170 | 1095 | 2 | 75 | 549 | 148 |
| **118** | 36919 | 13473 | 4176 | 16444 | 2547 | 653 | 5 | 26 | 443 | 611 |

To describe statistics Analysis

In [5]:

```
d.describe()
```

Out[5]:

| | Impressions | From Home | From Hashtags | From Explore | From Other | Saves | C |
|---|---|---|---|---|---|---|---|
| **count** | 119.000000 | 119.000000 | 119.000000 | 119.000000 | 119.000000 | 119.000000 | 11 |
| **mean** | 5703.991597 | 2475.789916 | 1887.512605 | 1078.100840 | 171.092437 | 153.310924 | |
| **std** | 4843.780105 | 1489.386348 | 1884.361443 | 2613.026132 | 289.431031 | 156.317731 | |
| **min** | 1941.000000 | 1133.000000 | 116.000000 | 0.000000 | 9.000000 | 22.000000 | |
| **25%** | 3467.000000 | 1945.000000 | 726.000000 | 157.500000 | 38.000000 | 65.000000 | |
| **50%** | 4289.000000 | 2207.000000 | 1278.000000 | 326.000000 | 74.000000 | 109.000000 | |
| **75%** | 6138.000000 | 2602.500000 | 2363.500000 | 689.500000 | 196.000000 | 169.000000 | |
| **max** | 36919.000000 | 13473.000000 | 11817.000000 | 17414.000000 | 2547.000000 | 1095.000000 | 1 |

To get rows and columns

In [6]:

```
np.shape(d)
```

Out[6]:

```
(119, 13)
```

To get number of elements

In [7]:

```
np.size(d)
```

Out[7]:

```
1547
```

To get the missing value

In [8]:

```
d.isna()
```

Out[8]:

| | Impressions | From Home | From Hashtags | From Explore | From Other | Saves | Comments | Shares | Likes | Profile Visits |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | False | False | False | False | False | False | False | False | False | False |
| **1** | False | False | False | False | False | False | False | False | False | False |
| **2** | False | False | False | False | False | False | False | False | False | False |
| **3** | False | False | False | False | False | False | False | False | False | False |
| **4** | False | False | False | False | False | False | False | False | False | False |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | .. |
| **114** | False | False | False | False | False | False | False | False | False | False |
| **115** | False | False | False | False | False | False | False | False | False | False |
| **116** | False | False | False | False | False | False | False | False | False | False |
| **117** | False | False | False | False | False | False | False | False | False | False |
| **118** | False | False | False | False | False | False | False | False | False | False |

119 rows × 13 columns

To drop the missing elements

In [9]:

```
d.dropna(axis=1,how='any')
```

Out[9]:

| | Impressions | From Home | From Hashtags | From Explore | From Other | Saves | Comments | Shares | Likes | Profile Visits |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 3920 | 2586 | 1028 | 619 | 56 | 98 | 9 | 5 | 162 | 35 |
| **1** | 5394 | 2727 | 1838 | 1174 | 78 | 194 | 7 | 14 | 224 | 48 |
| **2** | 4021 | 2085 | 1188 | 0 | 533 | 41 | 11 | 1 | 131 | 62 |
| **3** | 4528 | 2700 | 621 | 932 | 73 | 172 | 10 | 7 | 213 | 23 |
| **4** | 2518 | 1704 | 255 | 279 | 37 | 96 | 5 | 4 | 123 | 8 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | .. |
| **114** | 13700 | 5185 | 3041 | 5352 | 77 | 573 | 2 | 38 | 373 | 73 |
| **115** | 5731 | 1923 | 1368 | 2266 | 65 | 135 | 4 | 1 | 148 | 20 |
| **116** | 4139 | 1133 | 1538 | 1367 | 33 | 36 | 0 | 1 | 92 | 34 |
| **117** | 32695 | 11815 | 3147 | 17414 | 170 | 1095 | 2 | 75 | 549 | 148 |
| **118** | 36919 | 13473 | 4176 | 16444 | 2547 | 653 | 5 | 26 | 443 | 61 |

119 rows × 13 columns

In [10]:

```
d["Saves"]
```

Out[10]:

```
0        98
1       194
2        41
3       172
4        96
       ... 
114     573
115     135
116      36
117    1095
118     653
Name: Saves, Length: 119, dtype: int64
```

# Visualization

In [11]:

```
data=pd.DataFrame(d[['Saves','Shares']][0:500])
data
```

Out[11]:

|  | Saves | Shares |
|---|---|---|
| **0** | 98 | 5 |
| **1** | 194 | 14 |
| **2** | 41 | 1 |
| **3** | 172 | 7 |
| **4** | 96 | 4 |
| **...** | ... | ... |
| **114** | 573 | 38 |
| **115** | 135 | 1 |
| **116** | 36 | 1 |
| **117** | 1095 | 75 |
| **118** | 653 | 26 |

119 rows × 2 columns

In [12]:

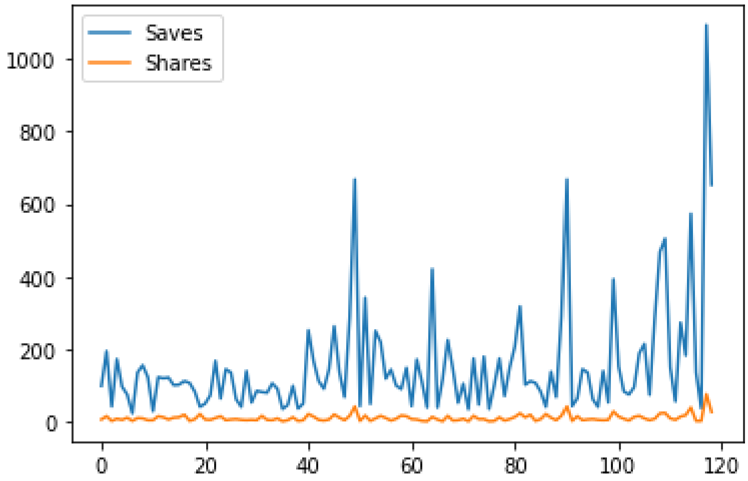
```
import matplotlib.pyplot as pp
```

In [13]:

```
data.plot.line()
```

Out[13]:

```
<AxesSubplot:>
```

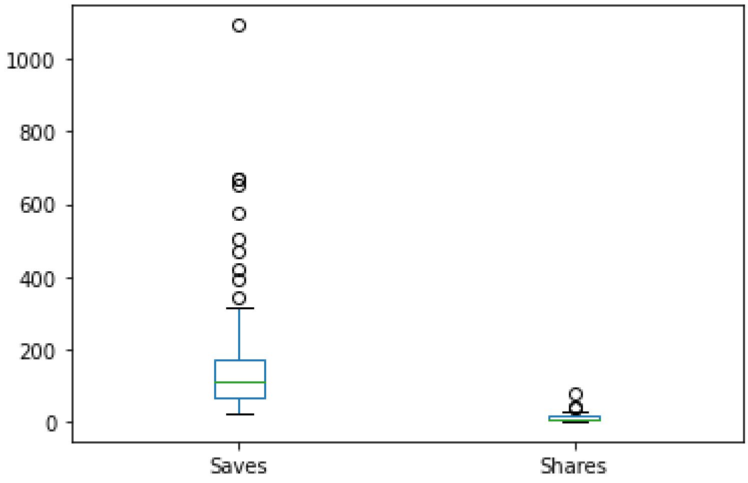


In [14]:

```
data.plot.box()
```

Out[14]:

```
<AxesSubplot:>
```

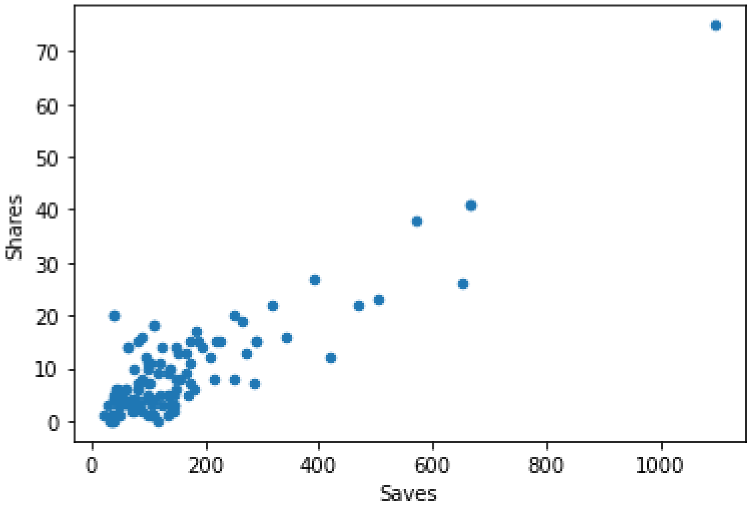


In [15]:

```
data.plot.scatter(x="Saves",y="Shares")
```

Out[15]:

```
<AxesSubplot:xlabel='Saves', ylabel='Shares'>
```

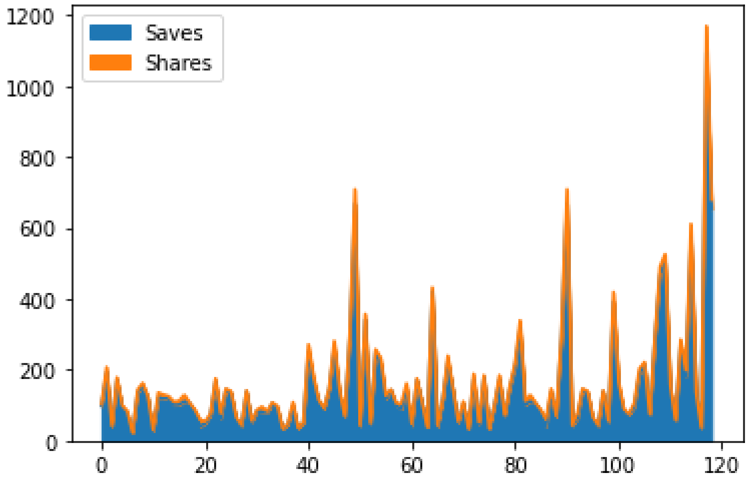


In [16]:

```
data.plot.area()
```

Out[16]:

```
<AxesSubplot:>
```

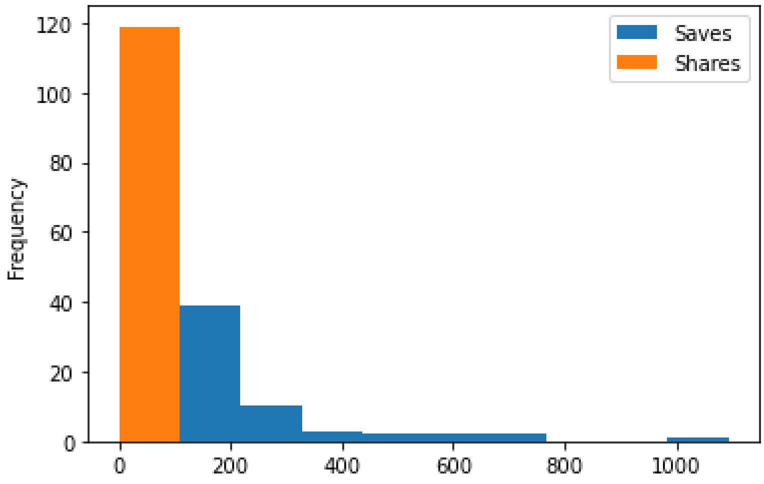


In [17]:

```
data.plot.hist()
```

Out[17]:

```
<AxesSubplot:ylabel='Frequency'>
```

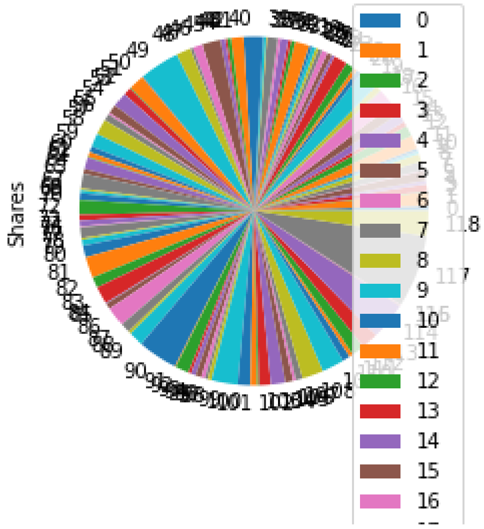


In [18]:

```
data=pd.DataFrame(d[['Saves','Shares']][0:200])
data.plot.pie(y="Shares")
```

Out[18]:

```
<AxesSubplot:ylabel='Shares'>
```



# Statistics

# Mean,median,mode,describe

In [19]:

```
data=pd.DataFrame(d[['Saves','Shares']][0:500])
data
```

Out[19]:

| | Saves | Shares |
|---|---|---|
| **0** | 98 | 5 |
| **1** | 194 | 14 |
| **2** | 41 | 1 |
| **3** | 172 | 7 |
| **4** | 96 | 4 |
| **...** | ... | ... |
| **114** | 573 | 38 |
| **115** | 135 | 1 |
| **116** | 36 | 1 |
| **117** | 1095 | 75 |
| **118** | 653 | 26 |

119 rows × 2 columns

In [20]:

```
print(data.mean())
```

```
Saves     153.310924
Shares      9.361345
dtype: float64
```

In [21]:

```
print(data.median())
```

```
Saves     109.0
Shares      6.0
dtype: float64
```

In [22]:

```
print(data.mode())
```

```
   Saves  Shares
0     40     3.0
1    135     NaN
2    144     NaN
```

In [23]:

```
data.fillna(value=1)
```

Out[23]:

| | Saves | Shares |
|---|---|---|
| **0** | 98 | 5 |
| **1** | 194 | 14 |
| **2** | 41 | 1 |
| **3** | 172 | 7 |
| **4** | 96 | 4 |
| **...** | ... | ... |
| **114** | 573 | 38 |
| **115** | 135 | 1 |
| **116** | 36 | 1 |
| **117** | 1095 | 75 |
| **118** | 653 | 26 |

119 rows × 2 columns

In [24]:

```
print(data.describe())
```

```
             Saves      Shares
count   119.000000  119.000000
mean    153.310924    9.361345
std     156.317731   10.089205
min      22.000000    0.000000
25%      65.000000    3.000000
50%     109.000000    6.000000
75%     169.000000   13.500000
max    1095.000000   75.000000
```

# Sum,cumsum,count,min,max

In [25]:

```
print(data.sum())
```

```
Saves     18244
Shares     1114
dtype: int64
```

In [26]:

```
print(data.cumsum())
```

```
     Saves  Shares
0       98       5
1      292      19
2      333      20
3      505      27
4      601      31
..     ...     ...
114  16325    1011
115  16460    1012
116  16496    1013
117  17591    1088
118  18244    1114

[119 rows x 2 columns]
```

In [27]:

```
print(data.count())
```

```
Saves     119
Shares    119
dtype: int64
```

In [28]:

```
print(data.min())
```

```
Saves     22
Shares     0
dtype: int64
```

In [29]:

```
print(data.max())
```

```
Saves     1095
Shares      75
dtype: int64
```

# covariance and correlation (spearman and pearsons)

In [30]:

```
data1=data['Saves'][0:10]
data1
```

Out[30]:

```
0     98
1    194
2     41
3    172
4     96
5     74
6     22
7    135
8    155
9    122
Name: Saves, dtype: int64
```

In [31]:

```
data2=data['Shares'][0:10]
data2
```

Out[31]:

```
0     5
1    14
2     1
3     7
4     4
5    10
6     1
7     9
8     8
9     3
Name: Shares, dtype: int64
```

In [32]:

```
from numpy import cov
print(cov(data1,data2))
```

```
[[3091.87777778  171.35555556]
 [ 171.35555556   17.51111111]]
```

In [33]:

```
from scipy.stats import pearsonr
print(pearsonr(data1,data2))
```

```
(0.7364278065804685, 0.015141913073821655)
```

In [34]:

```
from scipy.stats import spearmanr
print(spearmanr(data1,data2))
```

```
SpearmanrResult(correlation=0.6565379871744699, pvalue=0.03920438633255675
4)
```

In [ ]:

```

```