D15

In [4]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [5]:
```
df=pd.read_csv(r"C:\Users\user\Downloads\5_Instagram data.csv")
df
```

Out[5]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	F
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 × 11 columns

In [6]:
```
df.head()
```

Out[6]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Fol
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	

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119 entries, 0 to 118
Data columns (total 11 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Impressions     119 non-null    int64
 1   From Home       119 non-null    int64
 2   From Hashtags   119 non-null    int64
 3   From Explore    119 non-null    int64
 4   From Other      119 non-null    int64
 5   Saves           119 non-null    int64
 6   Comments        119 non-null    int64
 7   Shares          119 non-null    int64
 8   Likes           119 non-null    int64
 9   Profile Visits  119 non-null    int64
 10  Follows         119 non-null    int64
dtypes: int64(11)
memory usage: 10.4 KB
```

In [8]:
```
df.describe()
```

Out[8]:

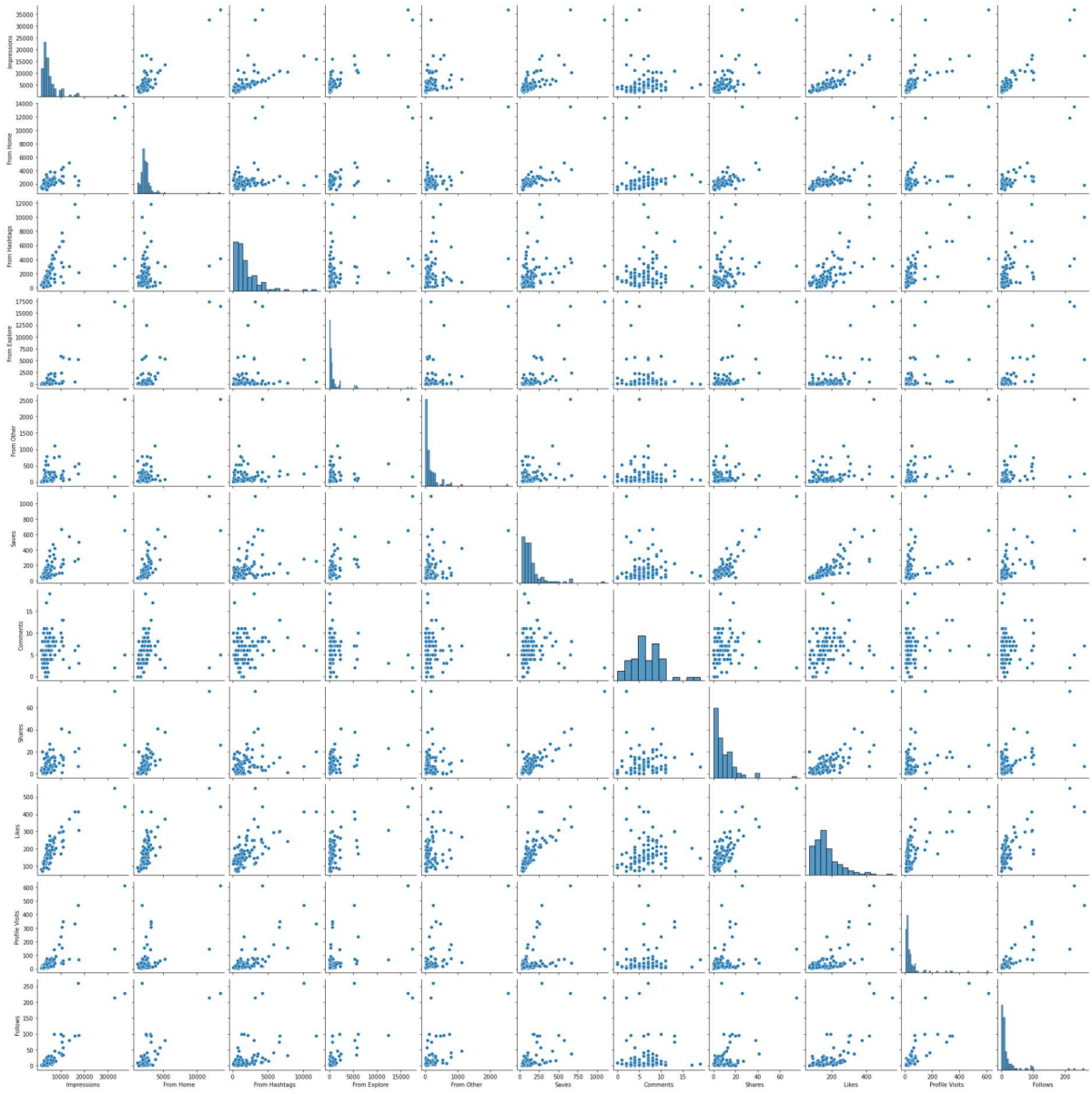
	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comn
count	119.000000	119.000000	119.000000	119.000000	119.000000	119.000000	119.00
mean	5703.991597	2475.789916	1887.512605	1078.100840	171.092437	153.310924	6.66
std	4843.780105	1489.386348	1884.361443	2613.026132	289.431031	156.317731	3.54
min	1941.000000	1133.000000	116.000000	0.000000	9.000000	22.000000	0.00
25%	3467.000000	1945.000000	726.000000	157.500000	38.000000	65.000000	4.00
50%	4289.000000	2207.000000	1278.000000	326.000000	74.000000	109.000000	6.00
75%	6138.000000	2602.500000	2363.500000	689.500000	196.000000	169.000000	8.00
max	36919.000000	13473.000000	11817.000000	17414.000000	2547.000000	1095.000000	19.00

In [9]:
```
df.columns
```

Out[9]:
```
Index(['Impressions', 'From Home', 'From Hashtags', 'From Explore',
       'From Other', 'Saves', 'Comments', 'Shares', 'Likes', 'Profile Visit
s',
       'Follows'],
      dtype='object')
```

In [10]:
```
sns.pairplot(df)
```

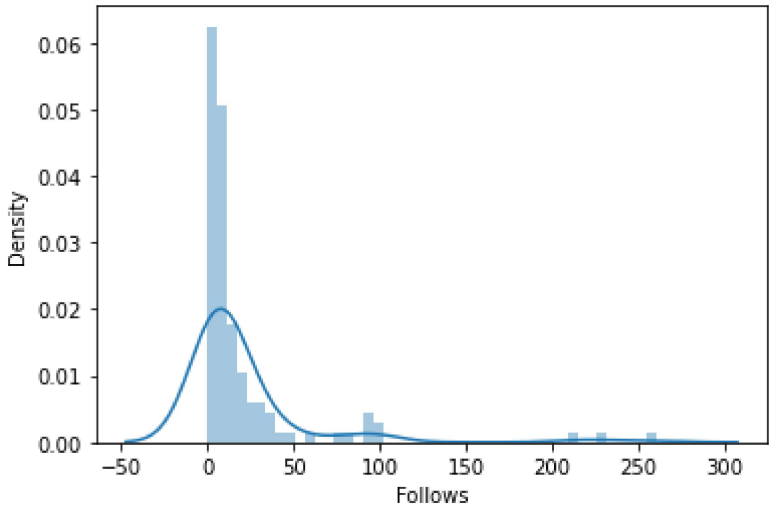
Out[10]:
```
<seaborn.axisgrid.PairGrid at 0x1fd9417c550>
```



In [12]:
```
sns.distplot(df['Follows'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: Fut
ureWarning: `distplot` is a deprecated function and will be removed in a futu
re version. Please adapt your code to use either `displot` (a figure-level fu
nction with similar flexibility) or `histplot` (an axes-level function for hi
stograms).
  warnings.warn(msg, FutureWarning)
```

Out[12]:
```
<AxesSubplot:xlabel='Follows', ylabel='Density'>
```



In [13]:
```
df1=df[['Impressions', 'From Home', 'From Hashtags', 'From Explore',
       'From Other', 'Saves', 'Comments', 'Shares', 'Likes', 'Profile Visits',
       'Follows']]
```

In [ ]:
```
sns.heatmap(df1.corr())
```

In [14]:
```
x=df1[['Impressions', 'From Home', 'From Hashtags', 'From Explore',
       'From Other', 'Saves', 'Comments', 'Shares', 'Likes', 'Profile Visits']]
y=df1['Follows']
```

In [15]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [16]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[16]:
```
LinearRegression()
```

In [17]:
```
print(lr.intercept_)
```

```
8.884351165415511
```

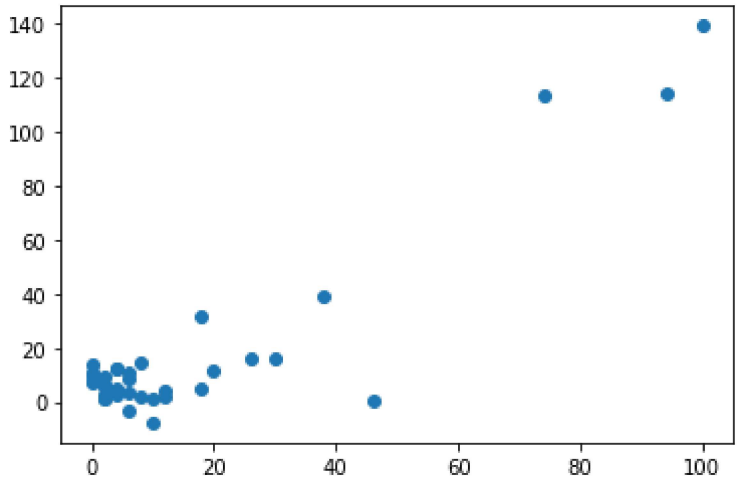
In [18]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[18]:

	Co-efficient
Impressions	0.004457
From Home	-0.010102
From Hashtags	-0.007572
From Explore	0.002257
From Other	-0.036682
Saves	0.013882
Comments	-1.041283
Shares	0.476748
Likes	0.048000
Profile Visits	0.435899

In [19]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[19]:
```
<matplotlib.collections.PathCollection at 0x1fd9f910940>
```



In [20]:
```
print(lr.score(x_test,y_test))
```

```
0.6457381480579063
```

In [21]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [22]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[22]:
```
Ridge(alpha=10)
```

In [23]:
```
rr.score(x_test,y_test)
```

Out[23]:
```
0.646502715764308
```

In [24]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_d
escent.py:530: ConvergenceWarning: Objective did not converge. You might want
to increase the number of iterations. Duality gap: 4162.09276151889, toleranc
e: 17.44706506024096
  model = cd_fast.enet_coordinate_descent(
```

Out[24]:
```
Lasso(alpha=10)
```

In [25]:
```
la.score(x_test,y_test)
```

Out[25]:
```
0.732346401320291
```

In [26]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_d
escent.py:530: ConvergenceWarning: Objective did not converge. You might want
to increase the number of iterations. Duality gap: 5937.571602556707, toleran
ce: 17.44706506024096
  model = cd_fast.enet_coordinate_descent(
```

Out[26]:
```
ElasticNet()
```

In [27]:
```
print(en.coef_)
```

```
[ 0.0020572  -0.00768442 -0.00508426  0.00473964 -0.03393736  0.01608792
 -0.95507691  0.43950735  0.04561138  0.43324727]
```

In [28]:
```
print(en.intercept_)
```

```
8.745384568482883
```

In [29]:
```
print(en.predict(x_train))
```

```
[-5.26438772e+00  5.92258001e+00  6.30779649e+00  4.54457575e+01
 -2.92781348e+00  2.60385626e+01  1.26888394e+02  6.11420540e+00
  5.13388390e+01  3.87543051e+01  3.83045388e+00 -2.69772915e+00
  1.25599934e+00  6.15748852e+01  1.31152797e+01 -2.69772915e+00
  7.20460947e+00  1.02151473e+01  9.95249895e+00  7.29548709e+00
  2.23307702e+01  9.22824320e+00  3.95065667e+01  1.64217837e+01
  1.83811019e+02  8.08318895e+00  4.95120085e+01  3.05655956e+01
  7.31240577e+00  3.87228869e+00  1.14887881e+02  2.18053313e+02
 -2.33185980e+00  1.04180547e+01  1.41210262e+01  8.08318895e+00
 -4.56101303e-01  2.54426833e+01  3.10274038e+01  2.42651638e+01
  9.47923688e+00  1.87504050e+01  5.35005268e+00  2.53508315e+02
 -2.96297186e+00  1.20678467e+01  2.96329610e-01  2.60602820e+01
  1.08050329e+01  1.05248373e+01 -5.26438772e+00  1.63546701e+00
  4.19452845e+00  4.19452845e+00  1.05530540e+01  8.86562638e+00
  6.13358869e+00  4.30826146e+00  1.82961221e+01  1.15025739e+01
  5.28702003e+00  6.62083488e+00  1.07804984e+01  7.70270073e+01
  6.31365613e+00  8.91286086e+00  3.44969091e+00  7.31240577e+00
  1.54756143e+01  2.56432455e+00  6.69634564e+00  5.35005268e+00
  6.16177818e+00  6.57987874e+00  2.47757104e+00  1.63173457e+01
  6.69634564e+00  2.15433671e+01  2.36033575e+01  1.30174923e+01
  3.83045388e+00 -1.42023255e-01  4.00362877e+00]
```

In [30]:
```
print(en.score(x_train,y_train))
```

```
0.9315867296199405
```

In [31]:
```
from sklearn import metrics
```

In [32]:
```
print("Mean Absolytre Error:",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolytre Error: 10.2194882788829
```

In [33]:
```
print("Mean Square Error:",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Square Error: 216.64643007347667
```

In [34]:
```
print("Root Mean Square Error:",np.sqrt(metrics.mean_absolute_error(y_test,pre
```

```
Root Mean Square Error: 3.1967934370057285
```

In [ ]: