In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
df=pd.read_csv(r'C:\Users\user\Downloads\5_Instagram data.csv')
df
```

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	61

119 rows × 13 columns

In [3]:

```
df.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

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119 entries, 0 to 118
Data columns (total 13 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 
 11  Caption         119 non-null    object
 12  Hashtags        119 non-null    object
dtypes: int64(11), object(2)
memory usage: 12.2+ KB
```

In [5]:

```
df.describe()
```

Out[5]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Co
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

In [6]:

```
df.columns
```

Out[6]:

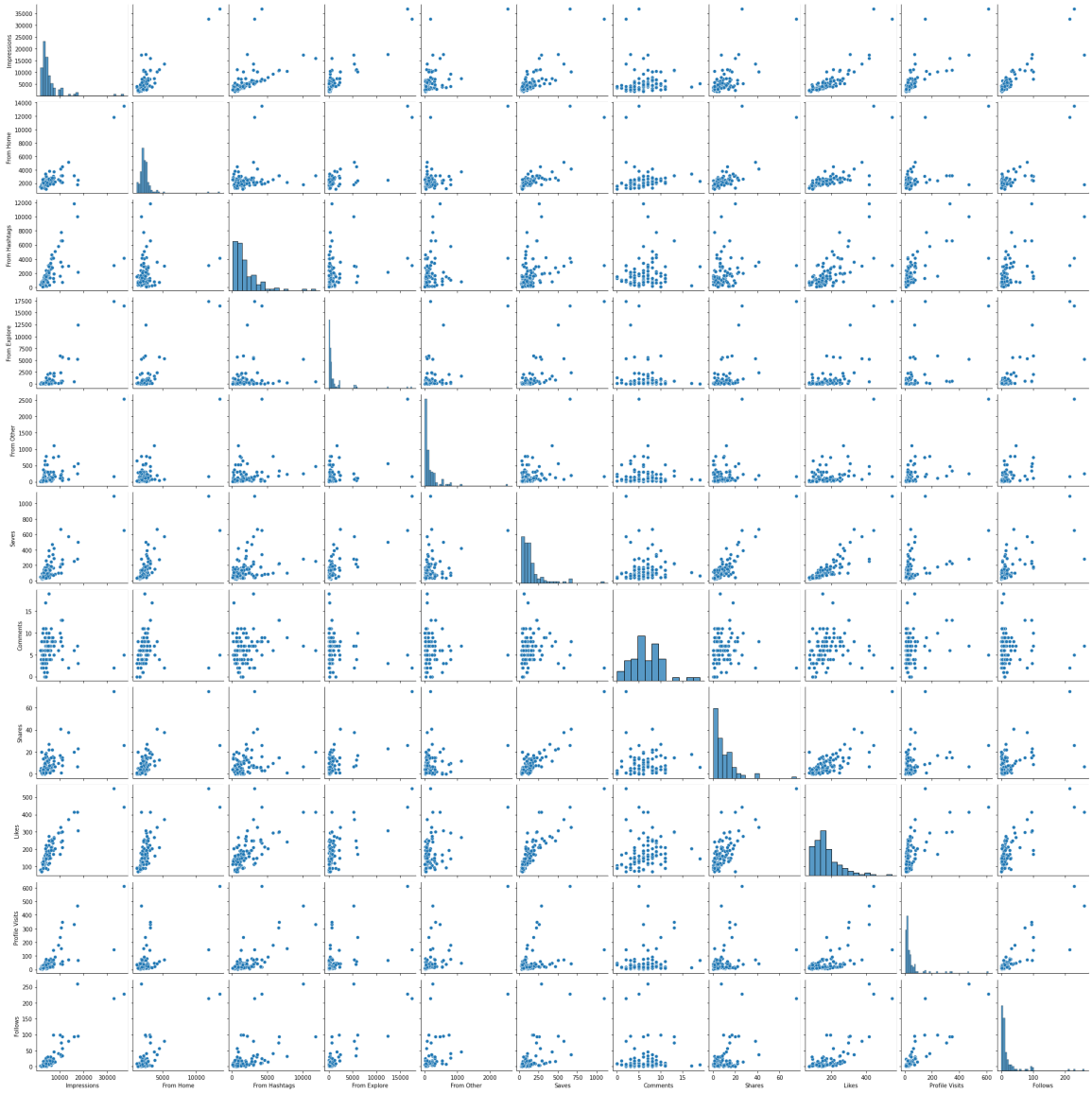
```
Index(['Impressions', 'From Home', 'From Hashtags', 'From Explore',
       'From Other', 'Saves', 'Comments', 'Shares', 'Likes', 'Profile Visi
ts',
       'Follows', 'Caption', 'Hashtags'],
      dtype='object')
```

In [7]:

```
sns.pairplot(df)
```

Out[7]:

```
<seaborn.axisgrid.PairGrid at 0x256ccd127f0>
```



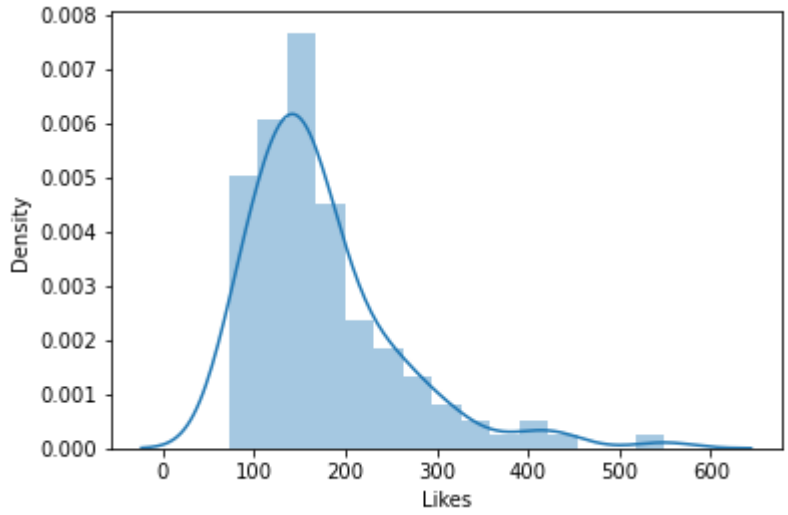
In [8]:

```
sns.distplot(df['Likes'])
```

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

Out[8]:

```
<AxesSubplot:xlabel='Likes', ylabel='Density'>
```

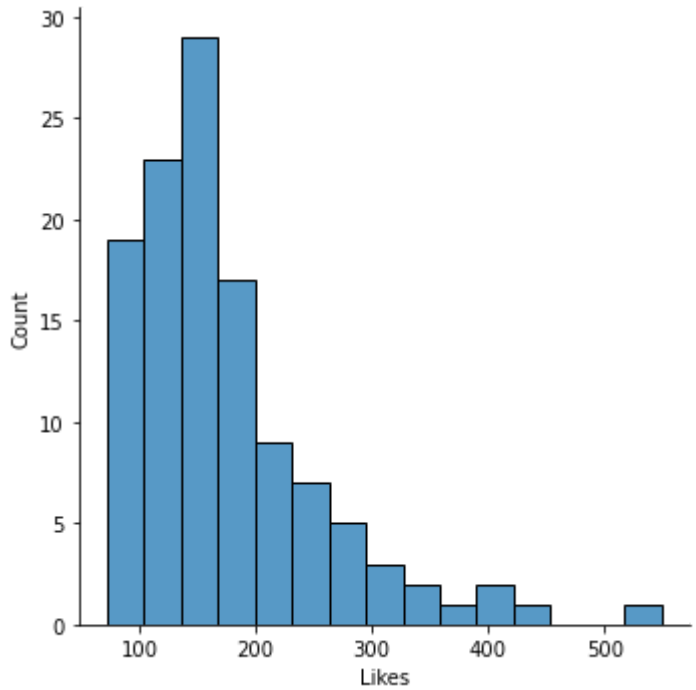


In [9]:

```
sns.displot(df["Likes"])
```

Out[9]:

```
<seaborn.axisgrid.FacetGrid at 0x256d1897e20>
```



In [10]:

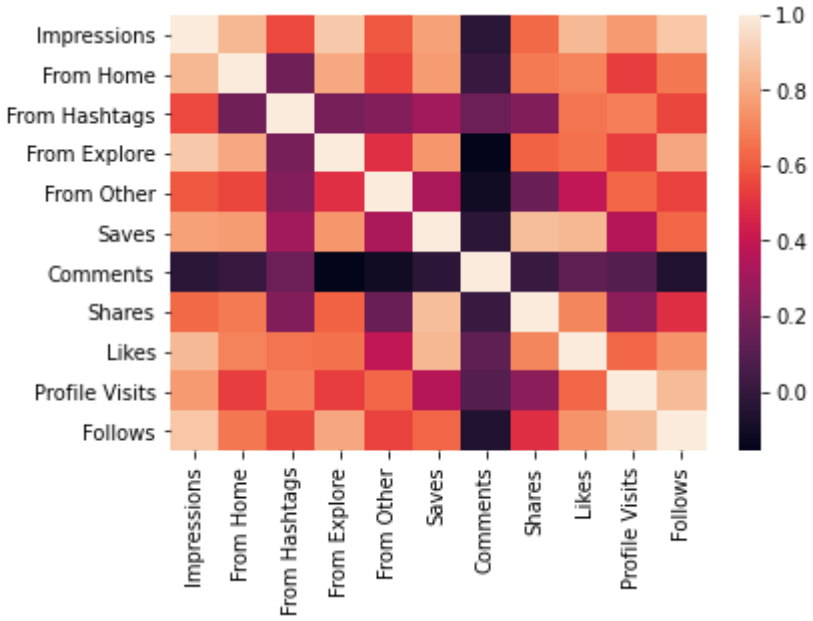
```
df1=df[['Impressions', 'From Home', 'From Hashtags', 'From Explore',
       'From Other', 'Saves', 'Comments', 'Shares', 'Likes', 'Profile Visits',
       'Follows', 'Caption', 'Hashtags']]
```

In [11]:

```
sns.heatmap(df1.corr())
```

Out[11]:

```
<AxesSubplot:>
```



In [12]:

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

In [13]:

```
from sklearn.model_selection import train_test_split
```

In [14]:

```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [15]:

```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)#ValueError: Input contains NaN, infinity or a value too large for
```

Out[15]:

```
LinearRegression()
```

In [16]:

```
print(lr.intercept_)
```

```
[52.1549699]
```

In [17]:

```
coef= pd.DataFrame(lr.coef_)
coef
```

Out[17]:

	0	1	2	3	4	5	6	7	8
0	-0.031871	0.044938	0.049805	0.031865	0.011158	0.303383	2.141816	-0.462822	-0.152376

In [18]:

```
print(lr.score(x_test,y_test))
```

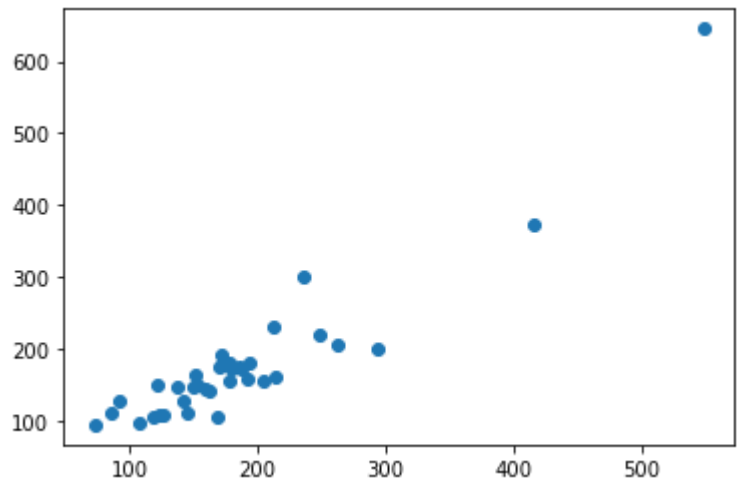
```
0.8343518667436152
```

In [19]:

```
prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[19]:

```
<matplotlib.collections.PathCollection at 0x256d45be970>
```



In [20]:

```
lr.score(x_test,y_test)
```

Out[20]:

```
0.8343518667436152
```

In [21]:

```
lr.score(x_train,y_train)
```

Out[21]:

```
0.9264445510602879
```

In [22]:

```
from sklearn.linear_model import Ridge,Lasso
```

In [23]:

```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[23]:

```
Ridge(alpha=10)
```

In [24]:

```
rr.score(x_test,y_test)
```

Out[24]:

```
0.8345223724242389
```

In [25]:

```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinat
e_descent.py:530: ConvergenceWarning: Objective did not converge. You migh
t want to increase the number of iterations. Duality gap: 20274.1448588656
68, tolerance: 51.745889156626504
  model = cd_fast.enet_coordinate_descent(
```

Out[25]:

```
Lasso(alpha=10)
```

In [26]:

```
la.score(x_test,y_test)
```

Out[26]:

```
0.852951762072607
```

Elastic Net

In [27]:

```
from sklearn.linear_model import ElasticNet
en = ElasticNet()
en.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinat
e_descent.py:530: ConvergenceWarning: Objective did not converge. You migh
t want to increase the number of iterations. Duality gap: 19393.8926398587
06, tolerance: 51.745889156626504
  model = cd_fast.enet_coordinate_descent(
```

Out[27]:

```
ElasticNet()
```

In [28]:

```
print(en.coef_)
```

```
[-0.01503018  0.02810439  0.03276695  0.01495072 -0.00793482  0.3046822
  2.03927163 -0.42691811 -0.13942724  0.49835745]
```

In [29]:

```
print(en.intercept_)
```

```
[51.42498414]
```

In [30]:

```
prediction=en.predict(x_test)
print(prediction)
```

```
[176.14787025 218.78395063 172.446132   192.03076019  94.09863838
 149.1726717  645.88725036 163.54296558 126.24031335 155.26450757
 201.2627035  286.61067295 179.02825785 208.51055721 156.86960435
 174.96067037 151.76093978 126.33883897 160.2111225   98.23674708
 106.39915542 109.51229286 371.60803316 174.12009333 109.37103194
 144.27103208 231.76819413 108.38558223 109.30000111 163.84279142
 147.50312985 110.32487392 179.96241064 146.49804073 145.84425408
 185.09772745]
```

In [31]:

```
print(en.score(x_test,y_test))
```

```
0.8430529053764912
```

Evaluation Metrics

In [34]:

```
from sklearn import metrics
```

In [35]:

```
print("Mean Absolute Error:",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute Error: 26.321478475007027
```

In [36]:

```
print("Mean Squared Error:",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Squared Error: 1207.4269384523398
```

In [37]:

```
print("Root Mean Squared Error:",np.sqrt(metrics.mean_squared_error(y_test,prediction)))
```

```
Root Mean Squared Error: 34.74804941938957
```