In [3]:
```
#import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [18]:
```
data=pd.read_csv(r"C:\Users\user\Desktop\Vicky\5_Instagram data.csv")
```

In [49]:
```
data.head()
```

Out[49]:

	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 [20]:
```
data.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 [21]:
```
data.describe()
```

Out[21]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comn
count	119.000000	119.000000	119.000000	119.000000	119.000000	119.000000	119.0(
mean	5703.991597	2475.789916	1887.512605	1078.100840	171.092437	153.310924	6.6(
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	0.0(
25%	3467.000000	1945.000000	726.000000	157.500000	38.000000	65.000000	4.0(
50%	4289.000000	2207.000000	1278.000000	326.000000	74.000000	109.000000	6.0(
75%	6138.000000	2602.500000	2363.500000	689.500000	196.000000	169.000000	8.0(
max	36919.000000	13473.000000	11817.000000	17414.000000	2547.000000	1095.000000	19.0(

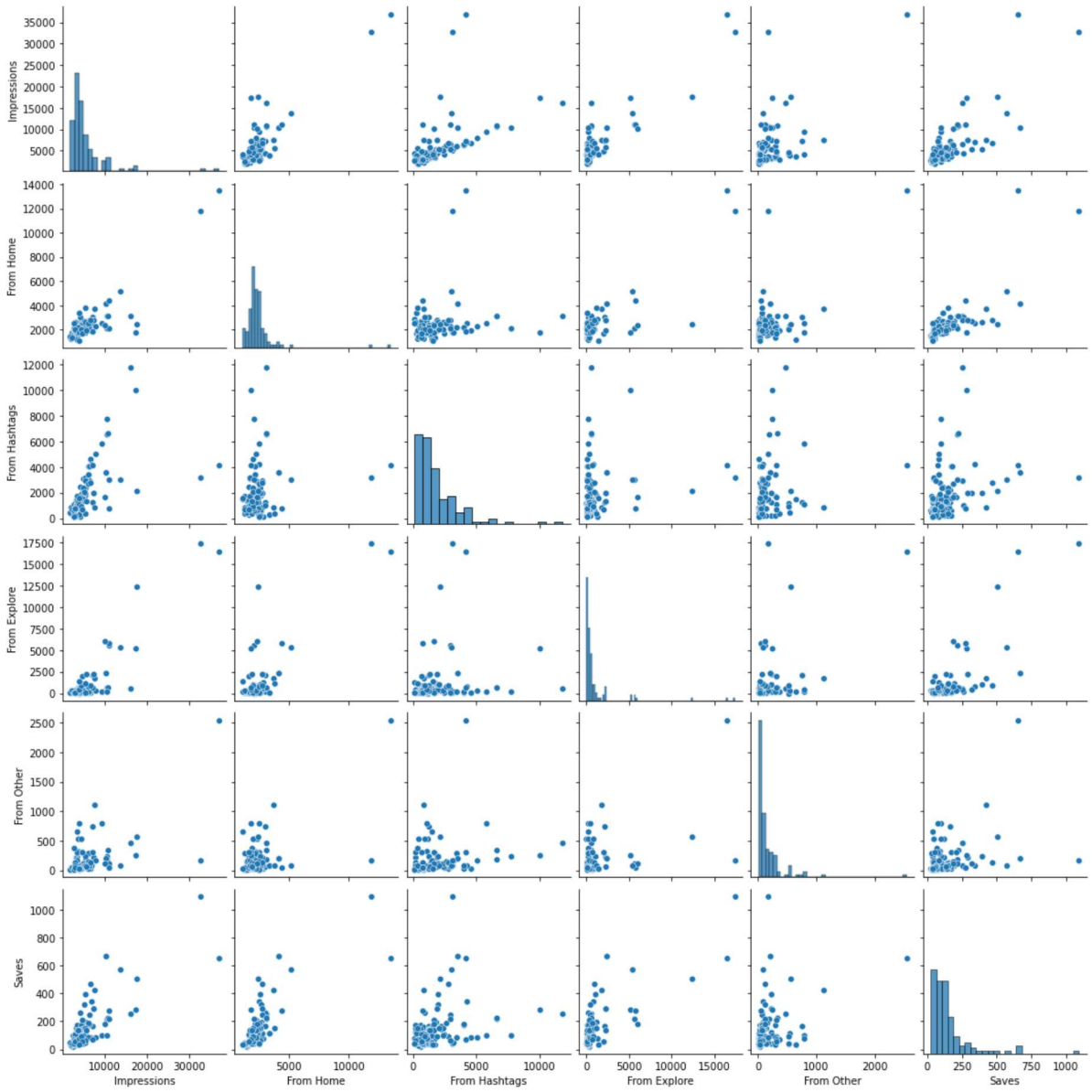
In [22]:
```
data.columns
```

Out[22]:
```
Index(['Impressions', 'From Home', 'From Hashtags', 'From Explore',
       'From Other', 'Saves', 'Comments', 'Shares', 'Likes', 'Profile Visit
s',
       'Follows', 'Caption', 'Hashtags'],
      dtype='object')
```

In [25]:
```
data1=data[['Impressions', 'From Home', 'From Hashtags', 'From Explore',
       'From Other', 'Saves']]
```

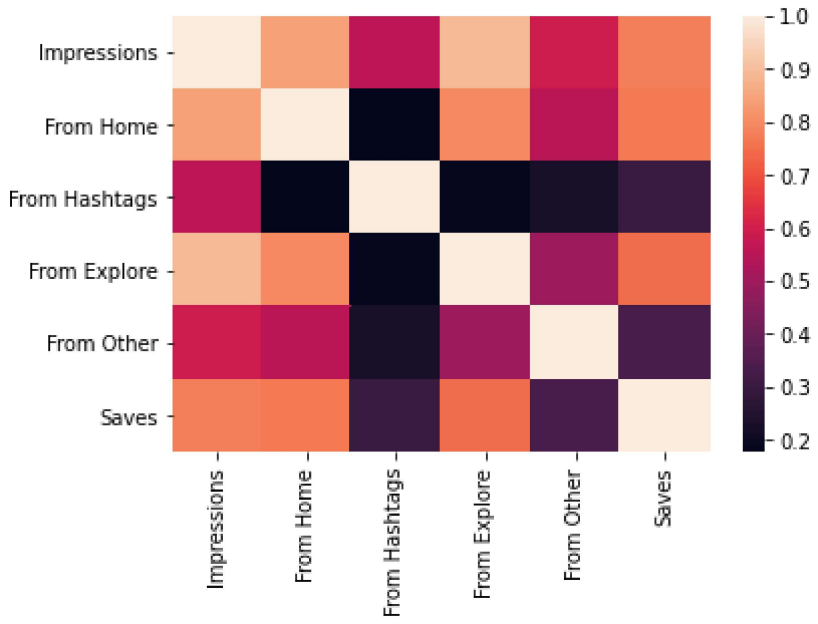
In [26]:
```
sns.pairplot(data1)
```

Out[26]:
```
<seaborn.axisgrid.PairGrid at 0x1e322fb9460>
```



In [27]:
```
sns.heatmap(data1.corr())
```

Out[27]: <AxesSubplot:>



In [50]:
```
x=data[['Follows','Likes']]
y=data1['Saves']
```

In [51]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.1)
```

In [52]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[52]: LinearRegression()

In [53]:
```
lr.intercept_
```

Out[53]: -132.07670138242148

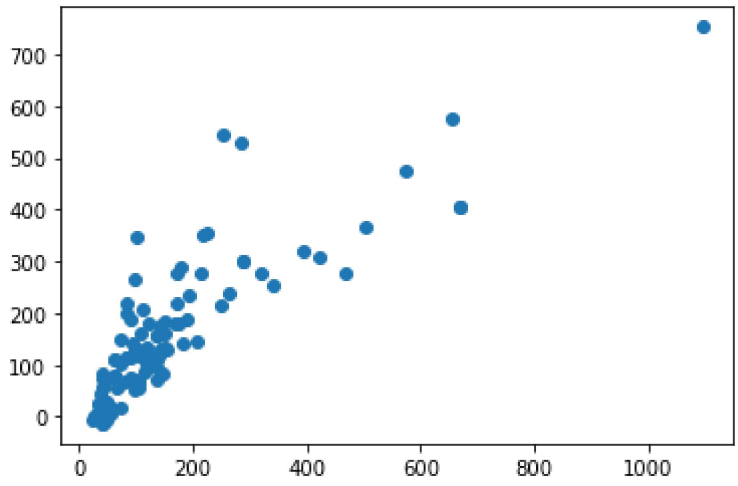
In [54]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=["Co-efficient"])
coeff
```

Out[54]:

	Co-efficient
Follows	-0.087813
Likes	1.647898

In [55]:
```
prediction = lr.predict(x_train)
plt.scatter(y_train,prediction)
```

Out[55]:
```
<matplotlib.collections.PathCollection at 0x1e325e95d90>
```



In [56]:
```
lr.score(x_test,y_test)
```

Out[56]:
```
0.465928808365188
```

In [57]:
```
lr.score(x_train,y_train)
```

Out[57]:
```
0.7178672143491022
```

In [58]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [59]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
rr.score(x_test,y_test)
```

Out[59]:
```
0.46595510504047044
```

In [60]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
la.score(x_test,y_test)
```

Out[60]:
```
0.4703656188565337
```

In [61]:
```
from sklearn.linear_model import ElasticNet
en= ElasticNet()
en.fit(x_train,y_train)
```

Out[61]:
```
ElasticNet()
```

In [62]:
```
print(en.coef_)
```

```
[-0.08649398  1.64722308]
```

In [63]:
```
print(en.intercept_)
```

```
-131.98670068332322
```

In [64]:
```
prediction = en.predict(x_test)
prediction
```

Out[64]:
```
array([-13.38663908,  55.27776632, 142.7535774 , 116.57099611,
       205.52104231,  55.45075427,  93.33688506, 105.04043456,
        94.29215632, 163.8215015 , 114.92377303, 208.91349495])
```

In [65]:
```
print(en.score(x_test,y_train))
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-65-48d1f0543252> in <module>
----> 1 print(en.score(x_test,y_train))

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py in score(self, X,
y, sample_weight)
    552         from .metrics import r2_score
    553         y_pred = self.predict(X)
--> 554         return r2_score(y, y_pred, sample_weight=sample_weight)
    555
    556     def _more_tags(self):

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inn
er_f(*args, **kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64
     65             # extra_args > 0

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\_regression.py in
r2_score(y_true, y_pred, sample_weight, multioutput)
    674     -3.0
    675     """
--> 676     y_type, y_true, y_pred, multioutput = _check_reg_targets(
    677         y_true, y_pred, multioutput)
    678     check_consistent_length(y_true, y_pred, sample_weight)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\_regression.py in
_check_reg_targets(y_true, y_pred, multioutput, dtype)
     86         the dtype argument passed to check_array.
     87     """
---> 88     check_consistent_length(y_true, y_pred)
     89     y_true = check_array(y_true, ensure_2d=False, dtype=dtype)
     90     y_pred = check_array(y_pred, ensure_2d=False, dtype=dtype)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in che
ck_consistent_length(*arrays)
    260     uniques = np.unique(lengths)
    261     if len(uniques) > 1:
--> 262         raise ValueError("Found input variables with inconsistent num
bers of"
    263                          " samples: %r" % [int(l) for l in lengths])
    264

ValueError: Found input variables with inconsistent numbers of samples: [107,
12]
```

In [66]:
```
from sklearn import metrics
```

In [67]:
```
print("Mean Absolute error:",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute error: 37.41752414066705
```

In [68]:
```
print("Mean Absolute Square error:",metrics.mean_squared_error(y_test,predicti
```

```
Mean Absolute Square error: 2219.8520976747755
```

In [69]:
```
print("Root mean Square error:",np.sqrt(metrics.mean_squared_error(y_test,pred
```

```
Root mean Square error: 47.115306405400524
```

In [ ]:

In [ ]: