

In [1]:

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

DATA COLLECTION

In [2]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\5_Instagram data - 5_Instagram data.csv")  
a
```

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	36
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	26
4	2518	1704	255	279	37	96	5	4	123	8
...
114	13700	5185	3041	5352	77	573	2	38	373	76
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

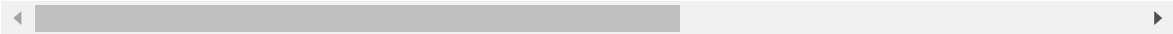
119 rows × 13 columns

In [3]:

```
b=a.head(10)  
b
```

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



DATA CLEANING AND PRE-PROCESSING

In [4]:

```
b.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 13 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Impressions            10 non-null     int64
1   From Home              10 non-null     int64
2   From Hashtags          10 non-null     int64
3   From Explore           10 non-null     int64
4   From Other             10 non-null     int64
5   Saves                  10 non-null     int64
6   Comments               10 non-null     int64
7   Shares                 10 non-null     int64
8   Likes                  10 non-null     int64
9   Profile Visits         10 non-null     int64
10  Follows                10 non-null     int64
11  Caption                10 non-null     object
12  Hashtags               10 non-null     object
dtypes: int64(11), object(2)
memory usage: 1.1+ KB
```

In [5]:

```
b.describe()
```

Out[5]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments
count	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000
mean	3829.100000	2245.500000	933.200000	459.200000	100.000000	110.900000	7.000000
std	838.988869	420.106666	443.303458	359.254413	152.969859	55.604656	2.309400
min	2518.000000	1543.000000	255.000000	0.000000	25.000000	22.000000	4.000000
25%	3593.000000	2052.250000	622.750000	255.750000	43.750000	79.500000	5.250000
50%	3902.000000	2234.500000	942.500000	331.000000	52.500000	110.000000	6.500000
75%	4091.500000	2603.250000	1167.000000	589.250000	69.750000	150.000000	8.500000
max	5394.000000	2727.000000	1838.000000	1174.000000	533.000000	194.000000	11.000000

In [6]:

```
b.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')
```

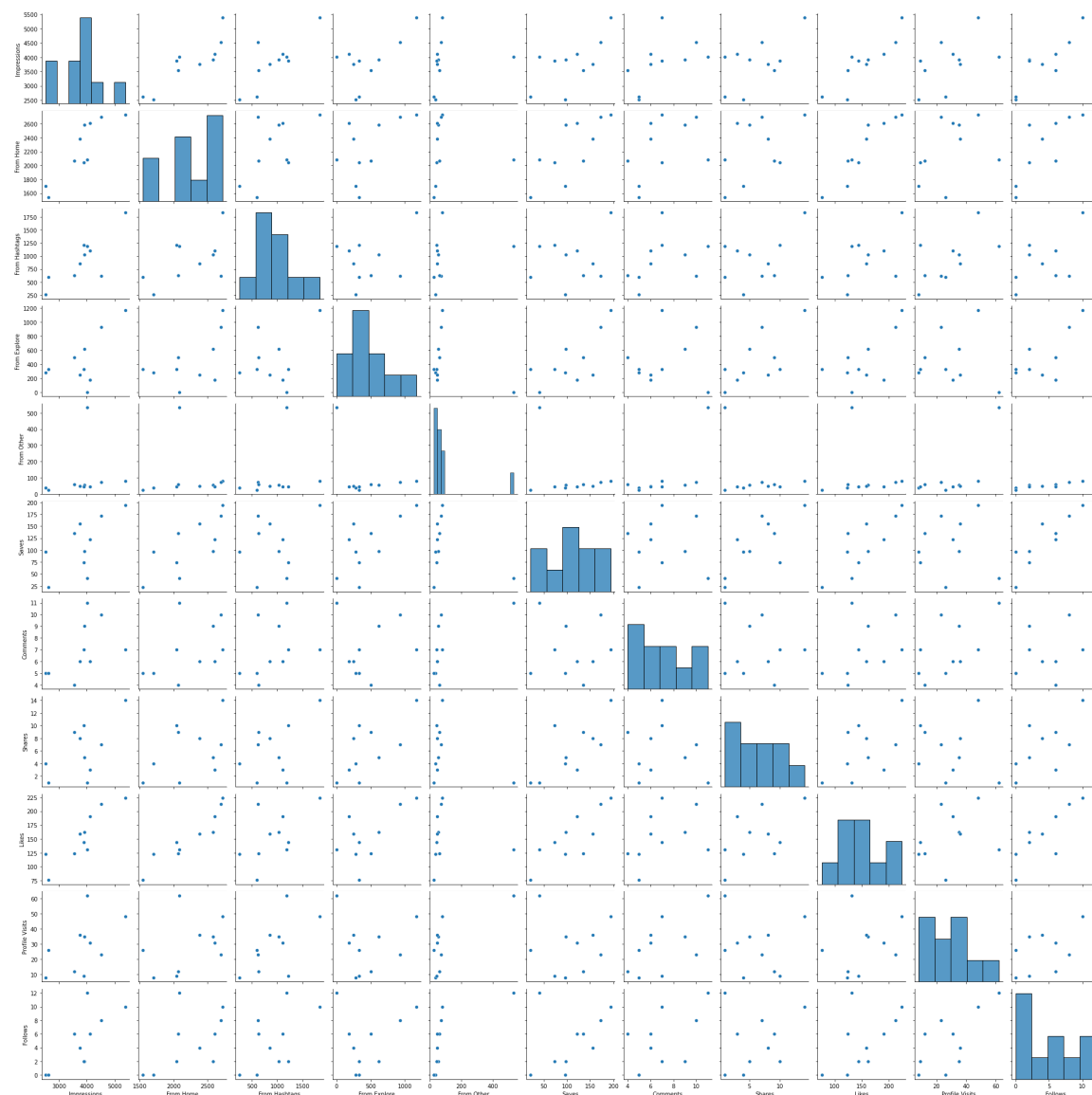
EDA AND VISUALIZATION

In [7]:

```
sns.pairplot(b)
```

Out[7]:

<seaborn.axisgrid.PairGrid at 0x15e63efacd0>



In [8]:

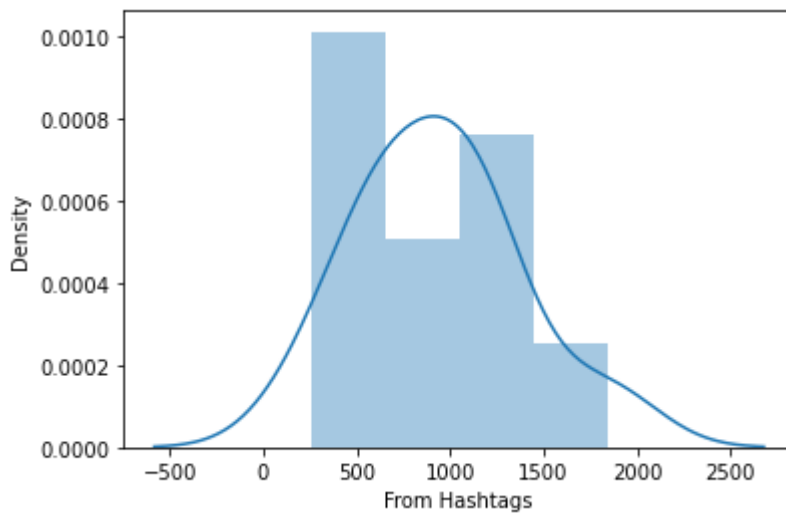
```
sns.distplot(b['From Hashtags'])
```

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 function for histograms).

warnings.warn(msg, FutureWarning)

Out[8]:

<AxesSubplot:xlabel='From Hashtags', ylabel='Density'>



In [9]:

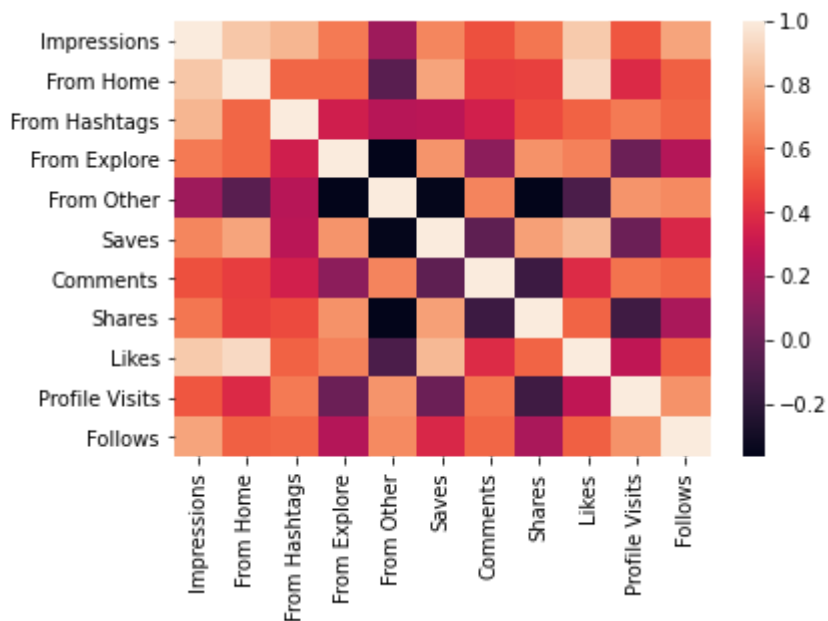
```
f=b[['Impressions', 'From Home', 'From Hashtags', 'From Explore',  
     'From Other', 'Saves', 'Comments', 'Shares', 'Likes', 'Profile Visits',  
     'Follows', 'Caption', 'Hashtags']]
```

In [10]:

```
sns.heatmap(f.corr())
```

Out[10]:

<AxesSubplot:>



In [11]:

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

In [12]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.5)
```

In [13]:

```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[13]:

LinearRegression()

In [14]:

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

-783.813607552309

In [15]:

```
r=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])  
r
```

Out[15]:

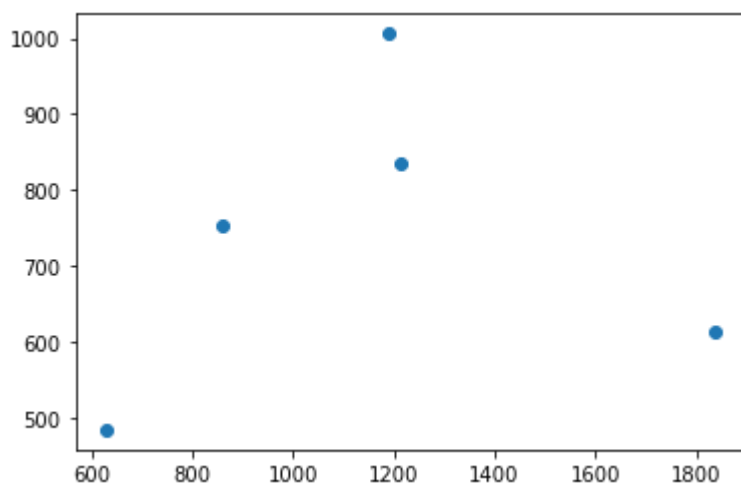
	Co-efficient
Impressions	0.207631
From Home	0.821720
From Explore	-0.449513
From Other	-0.573891
Saves	-4.427113
Comments	0.032891
Shares	-0.125017
Likes	-2.615661
Profile Visits	1.187803
Follows	-0.164287

In [16]:

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

Out[16]:

<matplotlib.collections.PathCollection at 0x15e6a421f40>



In [17]:

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

-1.0383891817704867

In [18]:

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

Out[18]:

1.0

RIDGE REGRESSION

In [19]:

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

In [20]:

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

Out[20]:

Ridge(alpha=10)

In [21]:

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

Out[21]:

-1.0369589034940931

LASSO REGRESSION

In [22]:

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

Out[22]:

Lasso(alpha=10)

In [23]:

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

Out[23]:

-1.2560870952280059

In []: