kaviyadevi 20106064

In [1]:
```
#to import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
#to import dataset
data1=pd.read_csv(r"C:\Users\user\Downloads\5_Instagram data - 5_Instagram data.c
data1
```

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

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits
118	36919	13473	4176	16444	2547	653	5	26	443	611

119 rows × 13 columns

In [3]:
```
#to display top 5 rows
data=data1.head()
data
```

Out[3]:

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

DATA CLEANING AND PREPROCESSING

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Impressions     5 non-null      int64 
 1   From Home       5 non-null      int64 
 2   From Hashtags   5 non-null      int64 
 3   From Explore    5 non-null      int64 
 4   From Other      5 non-null      int64 
 5   Saves           5 non-null      int64 
 6   Comments        5 non-null      int64 
 7   Shares          5 non-null      int64 
 8   Likes           5 non-null      int64 
 9   Profile Visits  5 non-null      int64 
 10  Follows         5 non-null      int64 
 11  Caption         5 non-null      object
 12  Hashtags        5 non-null      object
dtypes: int64(11), object(2)
memory usage: 648.0+ bytes
```

In [5]:
```
#to display summary of statistics
data.describe()
```

Out[5]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	
count	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000	5
mean	4076.200000	2360.400000	986.000000	600.800000	155.400000	120.200000	8.400000	6
std	1048.350228	449.256386	599.178187	475.157553	211.696245	62.210932	2.408319	4
min	2518.000000	1704.000000	255.000000	0.000000	37.000000	41.000000	5.000000	1
25%	3920.000000	2085.000000	621.000000	279.000000	56.000000	96.000000	7.000000	4
50%	4021.000000	2586.000000	1028.000000	619.000000	73.000000	98.000000	9.000000	5
75%	4528.000000	2700.000000	1188.000000	932.000000	78.000000	172.000000	10.000000	7
max	5394.000000	2727.000000	1838.000000	1174.000000	533.000000	194.000000	11.000000	14

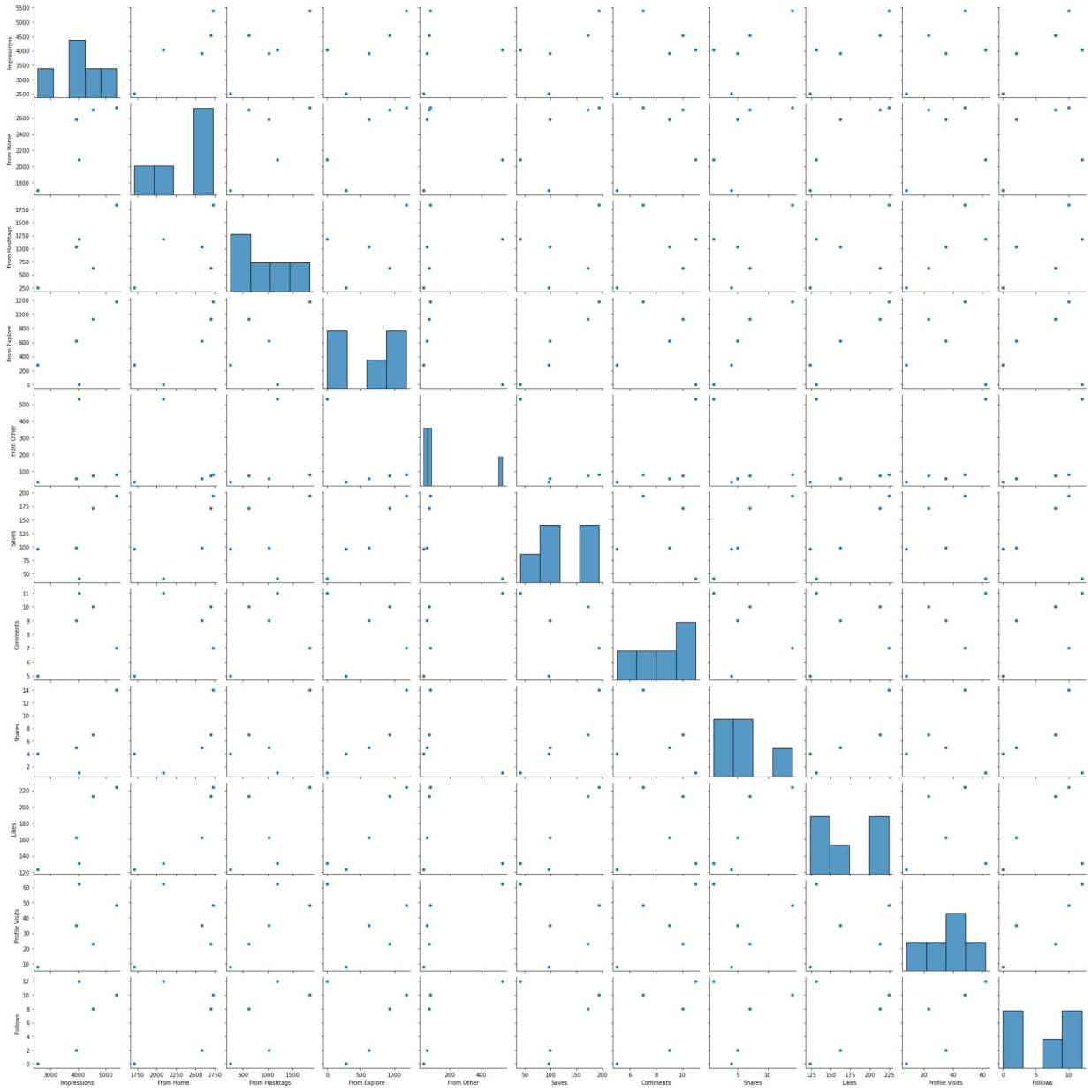
In [6]:
```
#to display the column heading
data.columns
```

Out[6]:
```
Index(['Impressions', 'From Home', 'From Hashtags', 'From Explore',
       'From Other', 'Saves', 'Comments', 'Shares', 'Likes', 'Profile Visits',
       'Follows', 'Caption', 'Hashtags'],
      dtype='object')
```

EDA and DATA VISUALIZATION

In [7]:
```
sns.pairplot(data)
```

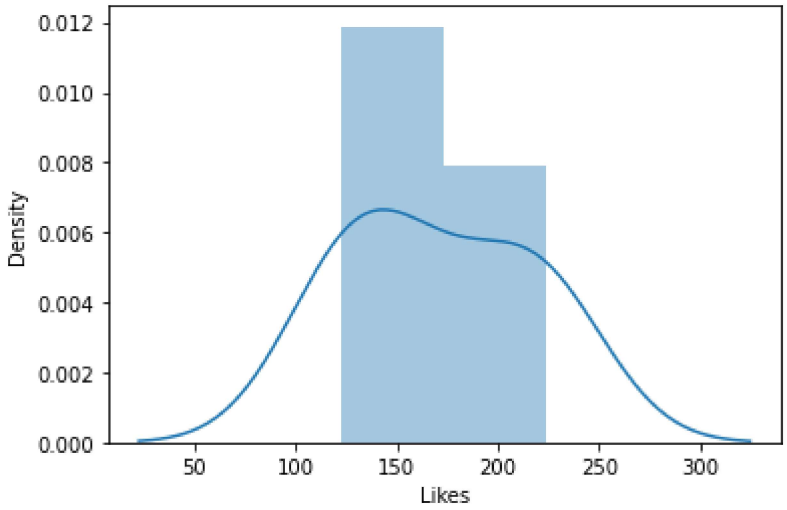
Out[7]:
```
<seaborn.axisgrid.PairGrid at 0x178d67a8130>
```



In [8]:
```
sns.distplot(data['Likes'])
```

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

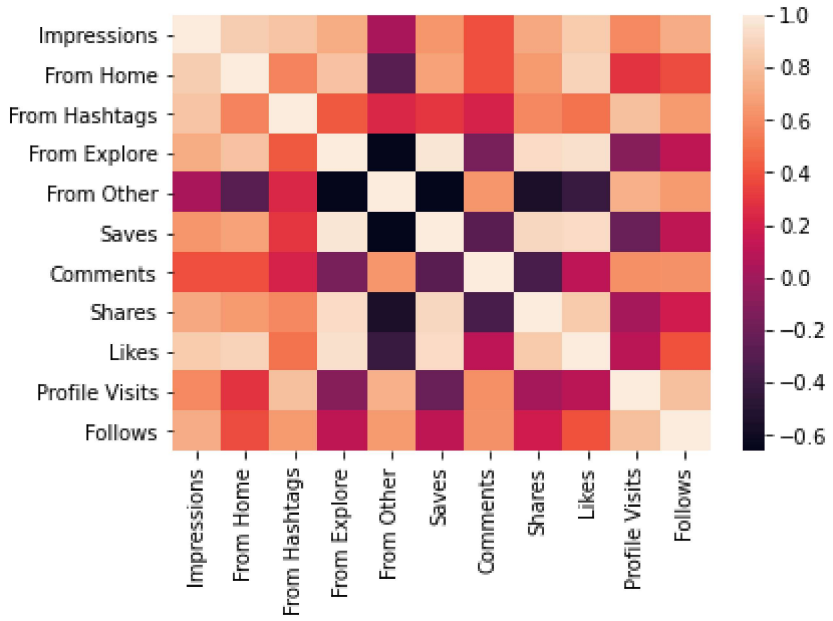
Out[8]: <AxesSubplot:xlabel='Likes', ylabel='Density'>



In [9]:
```
df=data[['Impressions', 'From Home', 'From Hashtags', 'From Explore',
       'From Other', 'Saves', 'Comments', 'Shares', 'Likes', 'Profile Visits',
       'Follows', 'Caption', 'Hashtags']]
```

In [10]:
```
sns.heatmap(df.corr())
```

Out[10]: <AxesSubplot:>



TRAINING MODEL

In [11]:
```
x=df[['Impressions', 'From Home', 'From Hashtags', 'From Explore','From Other',
y=df[['Likes']]
```

In [12]:
```
#to split my dataset into trainning and test
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 [13]:
```
from sklearn.linear_model import LinearRegression

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

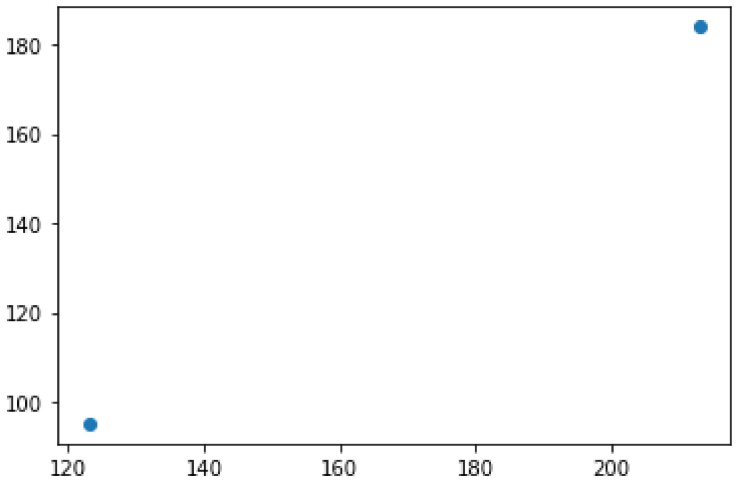
Out[13]: LinearRegression()

In [14]:
```
#to find intercept
print(lr.intercept_)
```

```
[-6.72381213]
```

In [15]:
```
prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[15]:
```
<matplotlib.collections.PathCollection at 0x178dcc72e20>
```



In [16]:
```
print(lr.score(x_test,y_test))
```

```
0.6004389468939054
```

RIDGE AND LASSO REGRESSION

In [17]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [18]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[18]:
```
Ridge(alpha=10)
```

In [19]:
```
rr.score(x_test,y_test)
```

Out[19]:
```
0.6004357921164281
```

In [20]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_des
cent.py:530: ConvergenceWarning: Objective did not converge. You might want to
increase the number of iterations. Duality gap: 0.77651300311621, tolerance: 0.
4484666666666666
  model = cd_fast.enet_coordinate_descent(
```

Out[20]:
```
Lasso(alpha=10)
```

In [21]:
```
la.score(x_test,y_test)
```

Out[21]:
```
0.2688269522950242
```

In [22]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[22]:
```
ElasticNet()
```

In [23]:
```
print(en.coef_)
```

```
[ 0.04802523  0.04680051 -0.02360282  0.00708096 -0.00888274  0.
 -0.          0.          0.         -0.         -0.        ]
```

In [24]:
```
print(en.predict(x_test))
```

```
[208.20256294  69.3930897 ]
```

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

```
0.28476142359310297
```

In [26]:
```
from sklearn import metrics
```

In [27]:
```
print("Mean Absolute error",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute error 28.438987770511233
```

In [28]:
```
print("Mean Squared error",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Squared error 809.1111325398415
```

In [29]:
```
print("Root Mean Absolute error",np.sqrt(metrics.mean_squared_error(y_test,predic
```

```
Root Mean Absolute error 28.444878845582053
```