

## Problem statement

predicting the house price in USA.To create a model to help him estimate of what the house would sell for.

In [1]:

```
1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
4 import seaborn as sns
```

In [2]:

```
1 df=pd.read_csv("instagram csv")
```

## To display top 10 rows

In [3]: 1 df.head(10)

Out[3]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Follows	Caption
0	3920	2586	1028	619	56	98	9	5	162	35	2	Here are some of the most important data visua... #finance💎#money💎#bu
1	5394	2727	1838	1174	78	194	7	14	224	48	10	Here are some of the best data science project... #healthcare💎#health💎#
2	4021	2085	1188	0	533	41	11	1	131	62	12	Learn how to train a machine learning model an... #data💎#datascience
3	4528	2700	621	932	73	172	10	7	213	23	8	Here💎s how you can write a Python program to d... #python💎#pythonprogr:
4	2518	1704	255	279	37	96	5	4	123	8	0	Plotting annotations while visualizing your da... #datavisualization💎#
5	3884	2046	1214	329	43	74	7	10	144	9	2	Here are some of the most important soft skill... #data💎#datascience
6	2621	1543	599	333	25	22	5	1	76	26	0	Learn how to analyze a candlestick chart as a ... #stockmarket💎#investii

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Follows	Caption
7	3541	2071	628	500	60	135	4	9	124	12	6	Here are some of the best books that you can f... #python💎#pythonprogr
8	3749	2384	857	248	49	155	6	8	159	36	4	Here are some of the best data analysis projec... #dataanalytics💎#data
9	4115	2609	1104	178	46	122	6	3	191	31	6	Here are two best ways to count the number of ... #python💎#pythonprogr

## Data Cleaning And Pre-Processing

In [4]:

```
1 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]:

```
1 # Display the statistical summary
2 df.describe()
```

Out[5]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profi Visi
count	119.000000	119.000000	119.000000	119.000000	119.000000	119.000000	119.000000	119.000000	119.000000	119.000000
mean	5703.991597	2475.789916	1887.512605	1078.100840	171.092437	153.310924	6.663866	9.361345	173.781513	50.621840
std	4843.780105	1489.386348	1884.361443	2613.026132	289.431031	156.317731	3.544576	10.089205	82.378947	87.088400
min	1941.000000	1133.000000	116.000000	0.000000	9.000000	22.000000	0.000000	0.000000	72.000000	4.000000
25%	3467.000000	1945.000000	726.000000	157.500000	38.000000	65.000000	4.000000	3.000000	121.500000	15.000000
50%	4289.000000	2207.000000	1278.000000	326.000000	74.000000	109.000000	6.000000	6.000000	151.000000	23.000000
75%	6138.000000	2602.500000	2363.500000	689.500000	196.000000	169.000000	8.000000	13.500000	204.000000	42.000000
max	36919.000000	13473.000000	11817.000000	17414.000000	2547.000000	1095.000000	19.000000	75.000000	549.000000	611.000000

```
In [6]: 1 # To display the col headings
        2 df.columns
```

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

```
In [7]: 1 cols=df.dropna(axis=1)
```

```
In [8]: 1 cols.columns
```

```
Out[8]: Index(['Impressions', 'From Home', 'From Hashtags', 'From Explore',
              'From Other', 'Saves', 'Comments', 'Shares', 'Likes', 'Profile Visits',
              'Follows', 'Caption', 'Hashtags'],
              dtype='object')
```

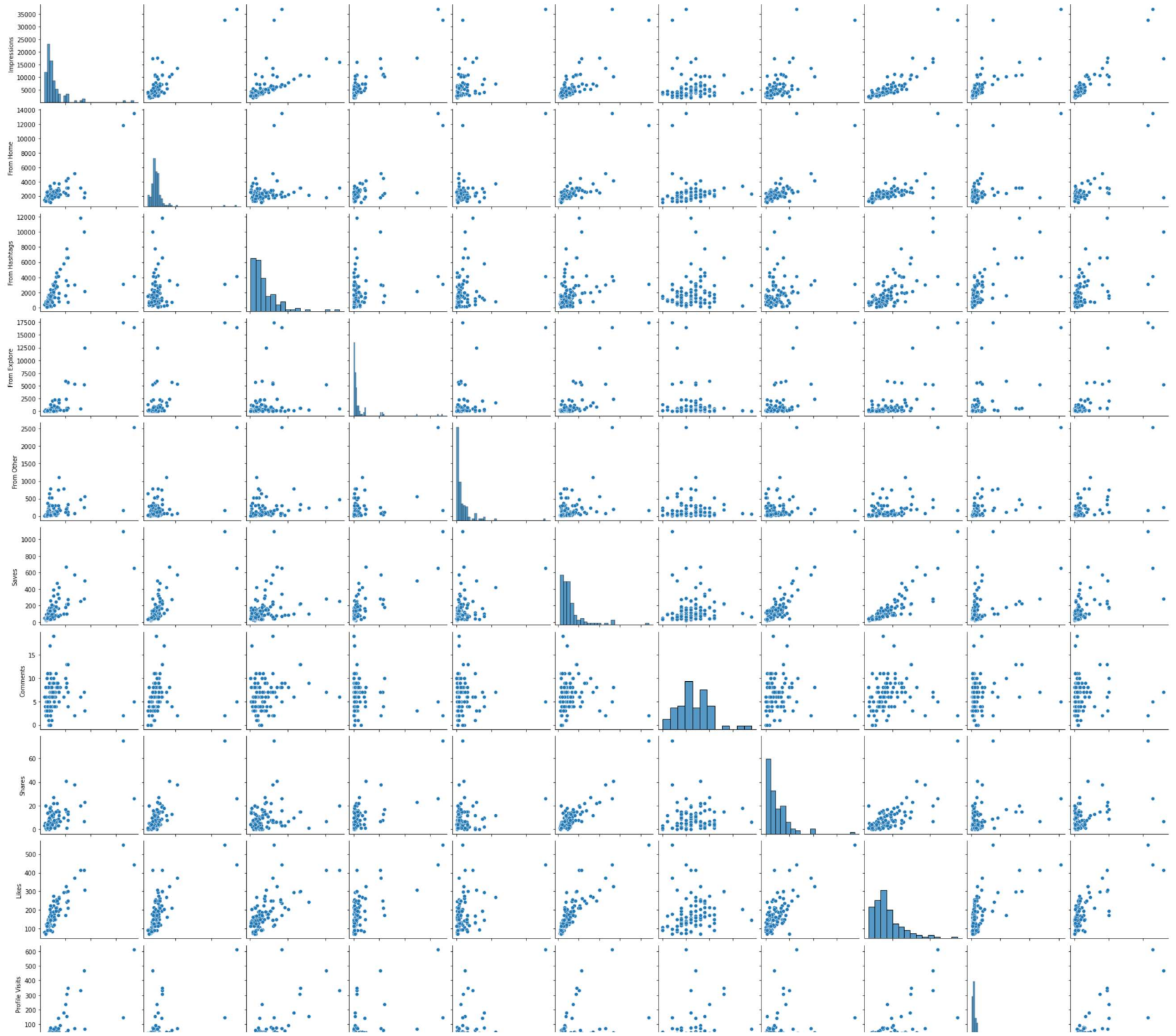
## EDA and Visualization

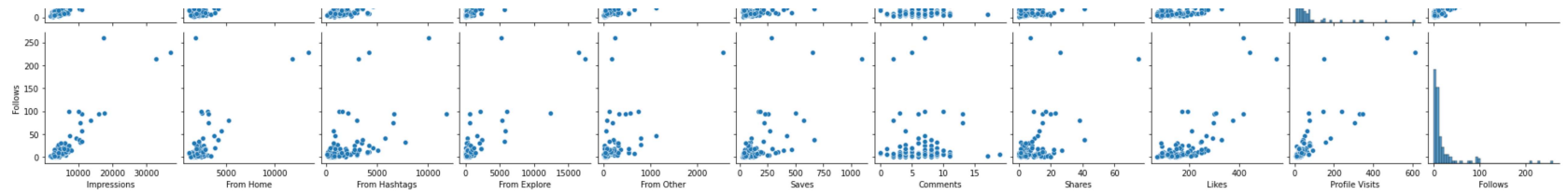
In [9]: 1 sns.pairplot(cols)

Out[9]: <seaborn.axisgrid.PairGrid at 0x25a93afa340>



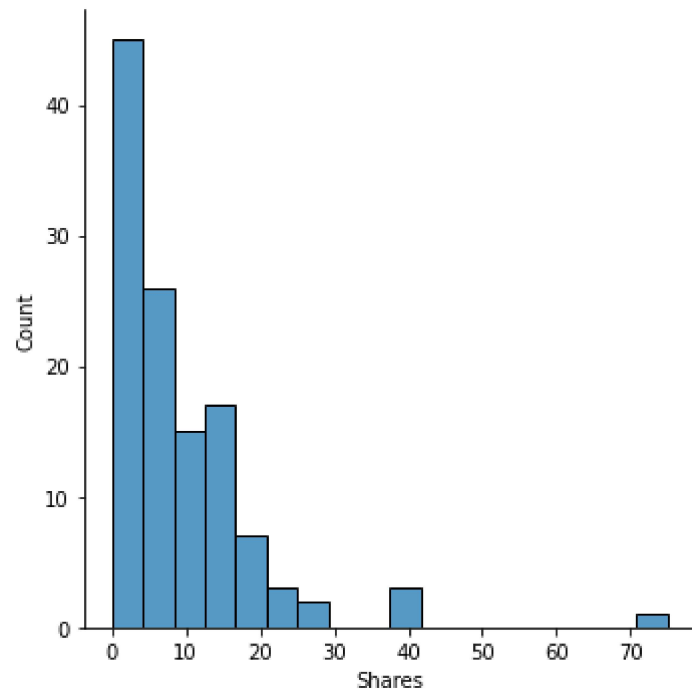






```
In [10]: 1 sns.displot(df['Shares'])
```

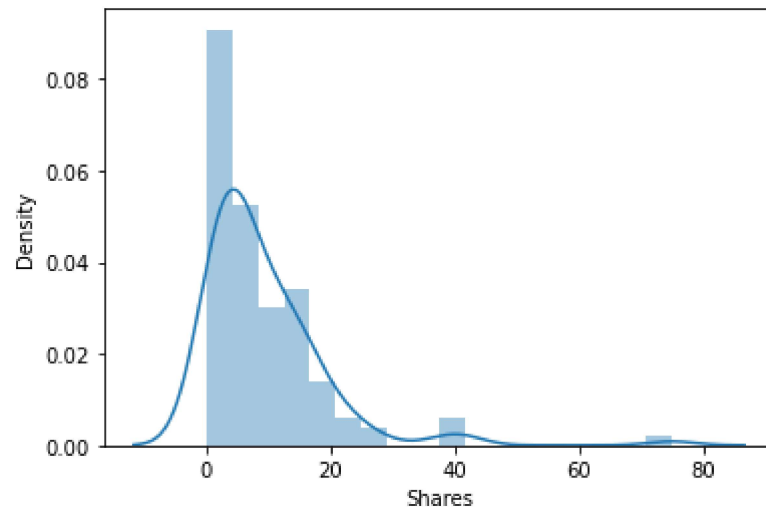
```
Out[10]: <seaborn.axisgrid.FacetGrid at 0x25a96c50d60>
```



```
In [11]: 1 # We use displot in older version we get distplot use displot
        2 sns.distplot(df['Shares'])
```

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[11]: <AxesSubplot:xlabel='Shares', ylabel='Density'>
```



```
In [12]: 1 df1=cols[['Saves', 'Comments', 'Shares', 'Likes', 'Profile Visits',
2           'Follows', 'Caption', 'Hashtags']]
3 df1
```

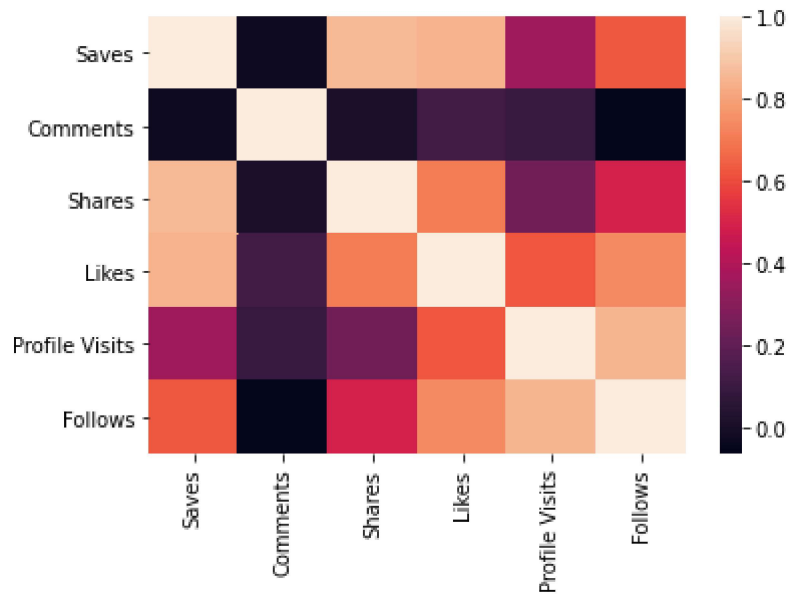
Out[12]:

	Saves	Comments	Shares	Likes	Profile Visits	Follows	Caption	Hashtags
0	98	9	5	162	35	2	Here are some of the most important data visua...	#finance💎#money💎#business💎#investing💎#investme...
1	194	7	14	224	48	10	Here are some of the best data science project...	#healthcare💎#health💎#covid💎#data💎#datascience💎...
2	41	11	1	131	62	12	Learn how to train a machine learning model an...	#data💎#datascience💎#dataanalysis💎#dataanalytic...
3	172	10	7	213	23	8	Here💎s how you can write a Python program to d...	#python💎#pythonprogramming💎#pythonprojects💎#py...
4	96	5	4	123	8	0	Plotting annotations while visualizing your da...	#datavisualization💎#datascience💎#data💎#dataana...
...	...	...	...	...	...	...	...	...
114	573	2	38	373	73	80	Here are some of the best data science certifi...	#datascience💎#datasciencejobs💎#datasciencetrai...
115	135	4	1	148	20	18	Clustering is a machine learning technique use...	#machinelearning💎#machinelearningalgorithms💎#d...
116	36	0	1	92	34	10	Clustering music genres is a task of grouping ...	#machinelearning💎#machinelearningalgorithms💎#d...
117	1095	2	75	549	148	214	Here are some of the best data science certifi...	#datascience💎#datasciencejobs💎#datasciencetrai...
118	653	5	26	443	611	228	175 Python Projects with Source Code solved an...	#python💎#pythonprogramming💎#pythonprojects💎#py...

119 rows × 8 columns

```
In [13]: 1 sns.heatmap(df1.corr())
```

```
Out[13]: <AxesSubplot:>
```



## To train the model - MODEL BUILD

Going to train linear regression model; We split our data into 2 variables x and y where x is independent var(input) and y is dependent on x(output), we could ignore address col as it is not required for our model

```
In [14]: 1 x=df1[['Saves', 'Comments', 'Shares', 'Likes', 'Profile Visits',]]  
2 y=df1[['Shares']]
```

## To split the dataset into test data

```
In [15]: 1 # importing lib for splitting test data  
2 from sklearn.model_selection import train_test_split
```

```
In [16]: 1 x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

```
In [17]: 1 from sklearn.linear_model import LinearRegression
2
3 lr=LinearRegression()
4 lr.fit(x_train,y_train)
```

Out[17]: LinearRegression()

```
In [18]: 1 print(lr.intercept_)
```

[5.32907052e-15]

```
In [19]: 1 print(lr.score(x_test,y_test))
```

1.0

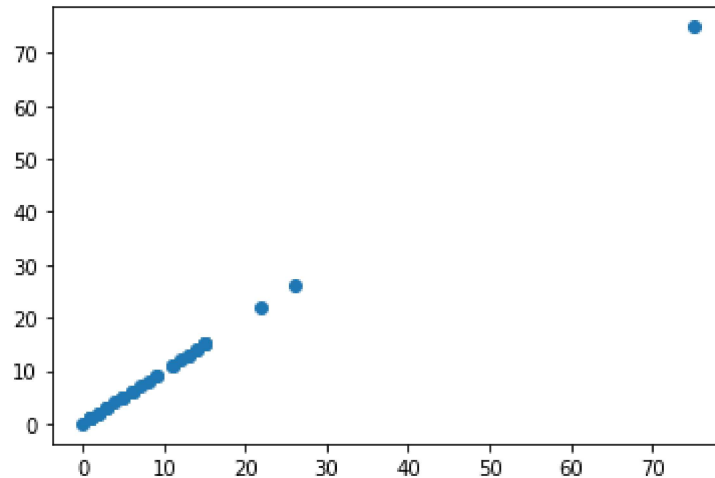
```
In [20]: 1 coeff=pd.DataFrame(lr.coef_)
2 coeff
```

Out[20]:

	0	1	2	3	4
0	-5.907229e-18	-7.346613e-16	1.0	2.601377e-17	2.764061e-17

```
In [21]: 1 pred = lr.predict(x_test)
         2 plt.scatter(y_test,pred)
```

Out[21]: <matplotlib.collections.PathCollection at 0x25a9b38dbb0>



```
In [22]: 1 from sklearn.linear_model import Ridge,Lasso
```

```
In [23]: 1 rr=Ridge(alpha=20)
         2 rr.fit(x_train,y_train)
```

Out[23]: Ridge(alpha=20)

```
In [24]: 1 rr.score(x_test,y_test)
```

Out[24]: 0.9999896534670728

```
In [26]: 1 la=Lasso(alpha=20)
         2 la.fit(x_train,y_train)
```

Out[26]: Lasso(alpha=20)

```
In [27]: 1 la.score(x_test,y_test)
```

Out[27]: 0.9365836117472807

In [ ]: 

1	
---	--