

LinearRegression

```
In [1]: import numpy as np
import pandas as pd
```

data collection

```
In [2]: import numpy as np
import pandas as pd
import matplotlib.pyplot as pp
import seaborn as sb
```

```
In [3]: df = pd.read_csv(r"C:\Users\user\Desktop\10_USA_Housing.csv")
df
```

Out[3]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferry Ap 674\nLaurabury, N 3701
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson View Suite 079\nLak Kathleen, CA
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Elizabet Stravenue\nDanieltow WI 06482
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFPO A 4482
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nFNP AE 0938
...	
4995	60567.944140	7.830362	6.137356	3.46	22837.361035	1.060194e+06	USNS Williams\nFNP AP 30153-765
4996	78491.275435	6.999135	6.576763	4.02	25616.115489	1.482618e+06	PSC 9258, Bc 8489\nAPO AA 4299 335
4997	63390.686886	7.250591	4.805081	2.13	33266.145490	1.030730e+06	4215 Tracy Garde Suite 076\nJoshualan VA 01
4998	68001.331235	5.534388	7.130144	5.44	42625.620156	1.198657e+06	USS Wallace\nFPO A 7331
4999	65510.581804	5.992305	6.792336	4.07	46501.283803	1.298950e+06	37778 George Ridge Apt. 509\nEast Holl NV 2

5000 rows × 7 columns

first 10 rows

In [4]:

```
df.head(10)
```

Out[4]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferry Apt. 674\nLaurabury, NE 3701...
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson Views Suite 079\nLake Kathleen, CA...
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Elizabeth Stravenue\nDanielstown, WI 06482...
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFPO AP 44820
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nFPO AE 09386
5	80175.754159	4.988408	6.104512	4.04	26748.428425	1.068138e+06	06039 Jennifer Islands Apt. 443\nTracyport, KS...
6	64698.463428	6.025336	8.147760	3.41	60828.249085	1.502056e+06	4759 Daniel Shoals Suite 442\nNguyenburgh, CO ...
7	78394.339278	6.989780	6.620478	2.42	36516.358972	1.573937e+06	972 Joyce Viaduct\nLake William, TN 17778-6483
8	59927.660813	5.362126	6.393121	2.30	29387.396003	7.988695e+05	USS Gilbert\nFPO AA 20957
9	81885.927184	4.423672	8.167688	6.10	40149.965749	1.545155e+06	Unit 9446 Box 0958\nDPO AE 97025

data cleaning

In [6]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 7 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Avg. Area Income                     5000 non-null  float64
1   Avg. Area House Age                  5000 non-null  float64
```

```
2 Avg. Area Number of Rooms 5000 non-null float64
3 Avg. Area Number of Bedrooms 5000 non-null float64
4 Area Population 5000 non-null float64
5 Price 5000 non-null float64
6 Address 5000 non-null object
dtypes: float64(6), object(1)
memory usage: 273.6+ KB
```

In [7]:

```
df.describe()
```

Out[7]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5.000000e+03
mean	68583.108984	5.977222	6.987792	3.981330	36163.516039	1.232073e+06
std	10657.991214	0.991456	1.005833	1.234137	9925.650114	3.531176e+05
min	17796.631190	2.644304	3.236194	2.000000	172.610686	1.593866e+04
25%	61480.562388	5.322283	6.299250	3.140000	29403.928702	9.975771e+05
50%	68804.286404	5.970429	7.002902	4.050000	36199.406689	1.232669e+06
75%	75783.338666	6.650808	7.665871	4.490000	42861.290769	1.471210e+06
max	107701.748378	9.519088	10.759588	6.500000	69621.713378	2.469066e+06

In [9]:

```
df.columns
```

Out[9]:

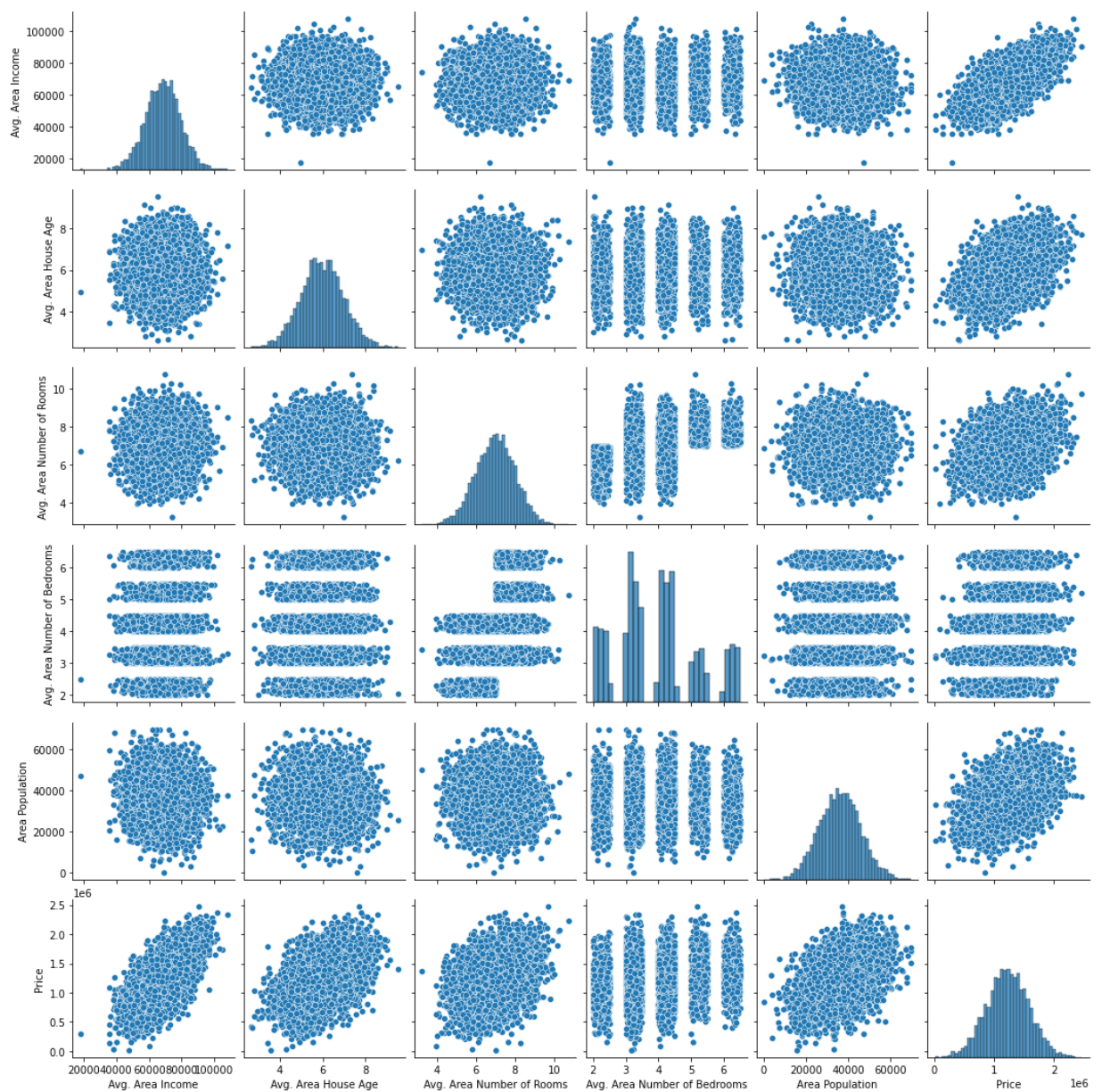
```
Index(['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',  
      'Avg. Area Number of Bedrooms', 'Area Population', 'Price', 'Address'],  
      dtype='object')
```

In [10]:

```
sb.pairplot(df)
```

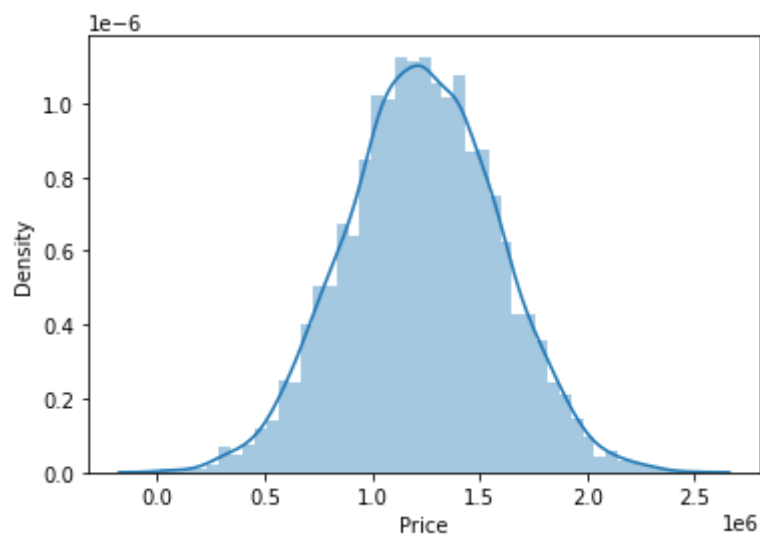
Out[10]:

```
<seaborn.axisgrid.PairGrid at 0x2590edcc580>
```



```
In [16]: sb.distplot(df["Price"])
```

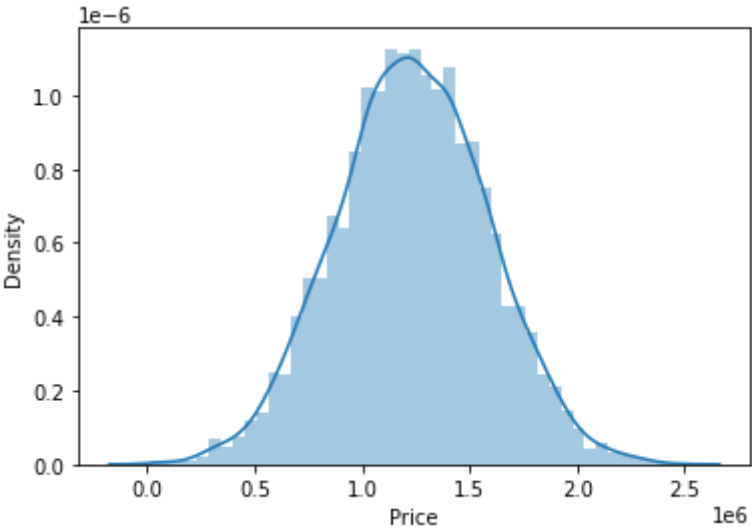
```
Out[16]: <AxesSubplot:xlabel='Price', ylabel='Density'>
```



```
In [17]: sb.distplot(df["Price"])
```

```
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[17]: <AxesSubplot:xlabel='Price', ylabel='Density'>
```



```
In [20]: df1=df[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
              'Avg. Area Number of Bedrooms', 'Area Population', 'Price', 'Address']]
df1
```

Out[20]:

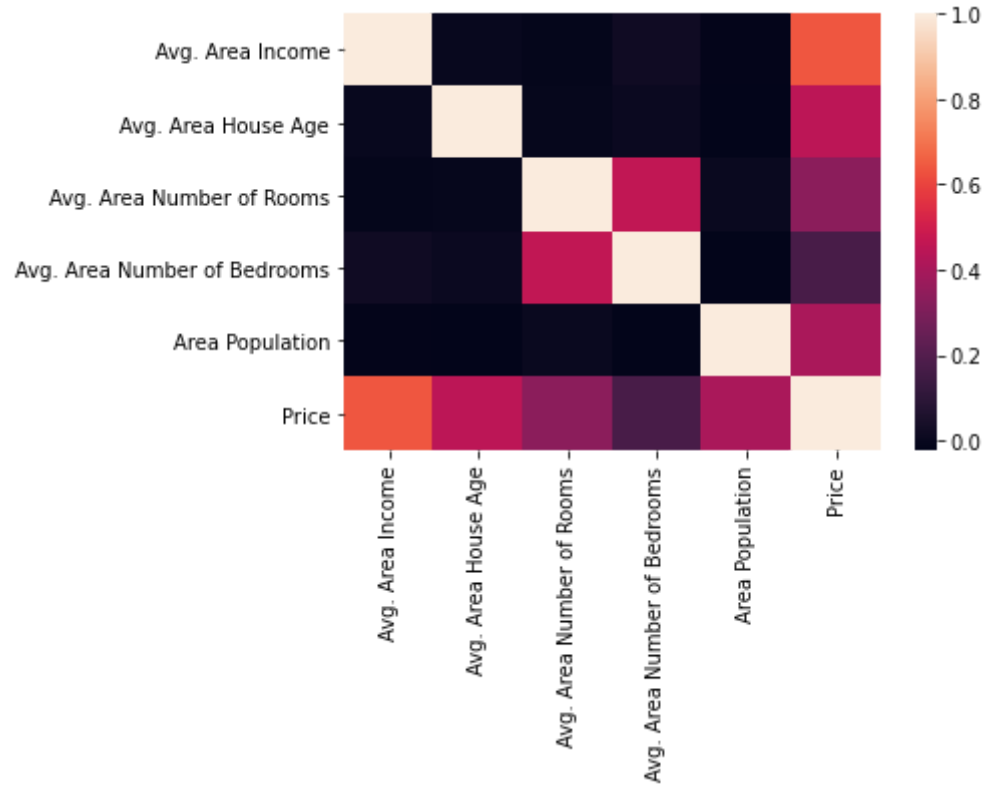
	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferry Ap 674\nLaurabury, N 3701
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson View Suite 079\nLak Kathleen, CA
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Elizabet Stravenue\nDanieltow WI 06482
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFPO A 4482
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nnFP AE 0938
...
4995	60567.944140	7.830362	6.137356	3.46	22837.361035	1.060194e+06	USNS Williams\nnFP AP 30153-765
4996	78491.275435	6.999135	6.576763	4.02	25616.115489	1.482618e+06	PSC 9258, Bc 8489\nAPO AA 4299 335
4997	63390.686886	7.250591	4.805081	2.13	33266.145490	1.030730e+06	4215 Tracy Garde Suite 076\nJoshualan VA 01

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
4998	68001.331235	5.534388	7.130144	5.44	42625.620156	1.198657e+06	USS Wallace\nFPO A 7331
4999	65510.581804	5.992305	6.792336	4.07	46501.283803	1.298950e+06	37778 George Ridge Apt. 509\nEast Holl NV 2

5000 rows × 7 columns

```
In [21]: sb.heatmap(df1.corr())
```

Out[21]: <AxesSubplot:>



model building

```
In [34]: x = df1[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
                'Avg. Area Number of Bedrooms', 'Area Population', 'Price']]
         y = df1['Price']
```

```
In [35]: 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 [36]: from sklearn.linear_model import LinearRegression
```

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

Out[36]: LinearRegression()

```
In [38]: print(lr.intercept_)
```

-2.3283064365386963e-10

```
In [42]: coef = pd.DataFrame(lr.coef_,x.columns,columns=['Co_efficient'])  
coef
```

Out[42]:

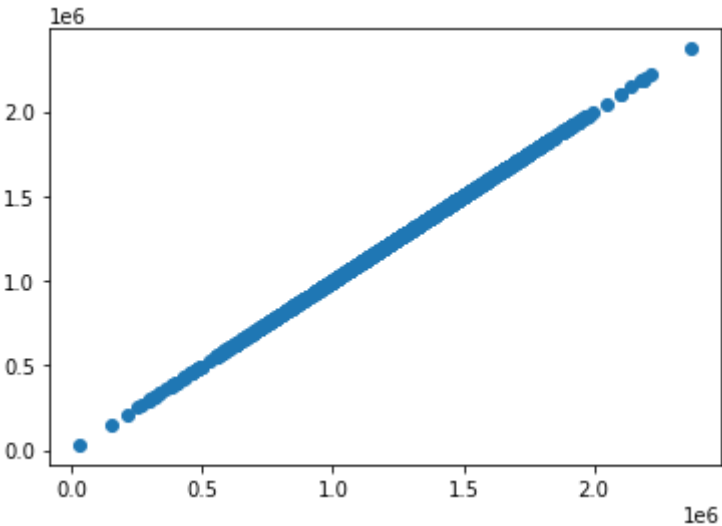
	Co_efficient
Avg. Area Income	3.184052e-15
Avg. Area House Age	-5.061788e-11
Avg. Area Number of Rooms	7.996921e-11
Avg. Area Number of Bedrooms	1.311991e-12
Area Population	9.261990e-15
Price	1.000000e+00

```
In [43]: print(lr.score(x_test,y_test))
```

1.0

```
In [44]: prediction = lr.predict(x_test)  
pp.scatter(y_test,prediction)
```

Out[44]: <matplotlib.collections.PathCollection at 0x25913950f40>



```
In [ ]:
```

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as pp

In [9]: import seaborn as sb

In [10]: df = pd.read_csv(r"C:\Users\user\Desktop\9_bottle.csv")
df
```

C:\ProgramData\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3165: DtypeWarning: Columns (47,73) have mixed types.Specify dtype option on import or set low_memory=False.

```
has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

Out[10]:

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2Sat	..
0	1	1	054.0 056.0	19-4903CR-HY-060-0930-05400560-0000A-3	0	10.500	33.4400	NaN	25.64900	NaN	..
1	1	2	054.0 056.0	19-4903CR-HY-060-0930-05400560-0008A-3	8	10.460	33.4400	NaN	25.65600	NaN	..
2	1	3	054.0 056.0	19-4903CR-HY-060-0930-05400560-0010A-7	10	10.460	33.4370	NaN	25.65400	NaN	..
3	1	4	054.0 056.0	19-4903CR-HY-060-0930-05400560-0019A-3	19	10.450	33.4200	NaN	25.64300	NaN	..
4	1	5	054.0 056.0	19-4903CR-HY-060-0930-05400560-0020A-7	20	10.450	33.4210	NaN	25.64300	NaN	..
...
864858	34404	864859	093.4 026.4	20-1611SR-MX-310-2239-09340264-0000A-7	0	18.744	33.4083	5.805	23.87055	108.74	..

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2Sat	..
864859	34404	864860	093.4 026.4	20- 1611SR- MX-310- 2239- 09340264- 0002A-3	2	18.744	33.4083	5.805	23.87072	108.74	..
864860	34404	864861	093.4 026.4	20- 1611SR- MX-310- 2239- 09340264- 0005A-3	5	18.692	33.4150	5.796	23.88911	108.46	..
864861	34404	864862	093.4 026.4	20- 1611SR- MX-310- 2239- 09340264- 0010A-3	10	18.161	33.4062	5.816	24.01426	107.74	..
864862	34404	864863	093.4 026.4	20- 1611SR- MX-310- 2239- 09340264- 0015A-3	15	17.533	33.3880	5.774	24.15297	105.66	..

864863 rows × 74 columns

In [11]:

df.head(10)

Out[11]:

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2Sat	...	R_PHA
0	1	1	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0000A-3	0	10.50	33.440	NaN	25.649	NaN	...	N
1	1	2	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0008A-3	8	10.46	33.440	NaN	25.656	NaN	...	N
2	1	3	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0010A-7	10	10.46	33.437	NaN	25.654	NaN	...	N
3	1	4	054.0 056.0	19- 4903CR-	19	10.45	33.420	NaN	25.643	NaN	...	N

Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2Sat	...	R_PHA
			HY-060-0930-05400560-0019A-3								
4	1	5	054.0 056.0 19-4903CR-HY-060-0930-05400560-0020A-7	20	10.45	33.421	NaN	25.643	NaN	...	N
5	1	6	054.0 056.0 19-4903CR-HY-060-0930-05400560-0030A-7	30	10.45	33.431	NaN	25.651	NaN	...	N
6	1	7	054.0 056.0 19-4903CR-HY-060-0930-05400560-0039A-3	39	10.45	33.440	NaN	25.658	NaN	...	N
7	1	8	054.0 056.0 19-4903CR-HY-060-0930-05400560-0050A-7	50	10.24	33.424	NaN	25.682	NaN	...	N
8	1	9	054.0 056.0 19-4903CR-HY-060-0930-05400560-0058A-3	58	10.06	33.420	NaN	25.710	NaN	...	N
9	1	10	054.0 056.0 19-4903CR-HY-060-0930-05400560-0075A-7	75	9.86	33.494	NaN	25.801	NaN	...	N

10 rows × 74 columns

In [12]:

df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 864863 entries, 0 to 864862
Data columns (total 74 columns):
Column Non-Null Count Dtype
--- -
0 Cst_Cnt 864863 non-null int64
1 Btl_Cnt 864863 non-null int64
2 Sta_ID 864863 non-null object
3 Depth_ID 864863 non-null object

4	Depthm	864863	non-null	int64
5	T_degC	853900	non-null	float64
6	Salnty	817509	non-null	float64
7	O2ml_L	696201	non-null	float64
8	STheta	812174	non-null	float64
9	O2Sat	661274	non-null	float64
10	Oxy_μmol/Kg	661268	non-null	float64
11	BtlNum	118667	non-null	float64
12	RecInd	864863	non-null	int64
13	T_prec	853900	non-null	float64
14	T_qual	23127	non-null	float64
15	S_prec	817509	non-null	float64
16	S_qual	74914	non-null	float64
17	P_qual	673755	non-null	float64
18	O_qual	184676	non-null	float64
19	SThtaq	65823	non-null	float64
20	O2Satq	217797	non-null	float64
21	ChlorA	225272	non-null	float64
22	Chlqua	639166	non-null	float64
23	Phaeop	225271	non-null	float64
24	Phaqua	639170	non-null	float64
25	PO4uM	413317	non-null	float64
26	PO4q	451786	non-null	float64
27	SiO3uM	354091	non-null	float64
28	SiO3qu	510866	non-null	float64
29	NO2uM	337576	non-null	float64
30	NO2q	529474	non-null	float64
31	NO3uM	337403	non-null	float64
32	NO3q	529933	non-null	float64
33	NH3uM	64962	non-null	float64
34	NH3q	808299	non-null	float64
35	C14As1	14432	non-null	float64
36	C14A1p	12760	non-null	float64
37	C14A1q	848605	non-null	float64
38	C14As2	14414	non-null	float64
39	C14A2p	12742	non-null	float64
40	C14A2q	848623	non-null	float64
41	DarkAs	22649	non-null	float64
42	DarkAp	20457	non-null	float64
43	DarkAq	840440	non-null	float64
44	MeanAs	22650	non-null	float64
45	MeanAp	20457	non-null	float64
46	MeanAq	840439	non-null	float64
47	IncTim	14437	non-null	object
48	LightP	18651	non-null	float64
49	R_Depth	864863	non-null	float64
50	R_TEMP	853900	non-null	float64
51	R_POTEMP	818816	non-null	float64
52	R_SALINITY	817509	non-null	float64
53	R_SIGMA	812007	non-null	float64
54	R_SVA	812092	non-null	float64
55	R_DYNHT	818206	non-null	float64
56	R_O2	696201	non-null	float64
57	R_O2Sat	666448	non-null	float64
58	R_SIO3	354099	non-null	float64
59	R_PO4	413325	non-null	float64
60	R_NO3	337411	non-null	float64
61	R_NO2	337584	non-null	float64
62	R_NH4	64982	non-null	float64
63	R_CHLA	225276	non-null	float64
64	R_PHAEO	225275	non-null	float64
65	R_PRES	864863	non-null	int64
66	R_SAMP	122006	non-null	float64
67	DIC1	1999	non-null	float64
68	DIC2	224	non-null	float64
69	TA1	2084	non-null	float64
70	TA2	234	non-null	float64
71	pH2	10	non-null	float64
72	pH1	84	non-null	float64

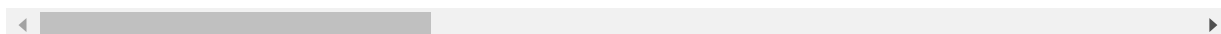
73 DIC Quality Comment 55 non-null object
 dtypes: float64(65), int64(5), object(4)
 memory usage: 488.3+ MB

In [13]: `df.describe()`

Out[13]:

	Cst_Cnt	Btl_Cnt	Depthm	T_degC	Salnty	O2ml_L	
count	864863.000000	864863.000000	864863.000000	853900.000000	817509.000000	696201.000000	8
mean	17138.790958	432432.000000	226.831951	10.799677	33.840350	3.392468	
std	10240.949817	249664.587267	316.050259	4.243825	0.461843	2.073256	
min	1.000000	1.000000	0.000000	1.440000	28.431000	-0.010000	
25%	8269.000000	216216.500000	46.000000	7.680000	33.488000	1.360000	
50%	16848.000000	432432.000000	125.000000	10.060000	33.863000	3.440000	
75%	26557.000000	648647.500000	300.000000	13.880000	34.196900	5.500000	
max	34404.000000	864863.000000	5351.000000	31.140000	37.034000	11.130000	

8 rows × 70 columns



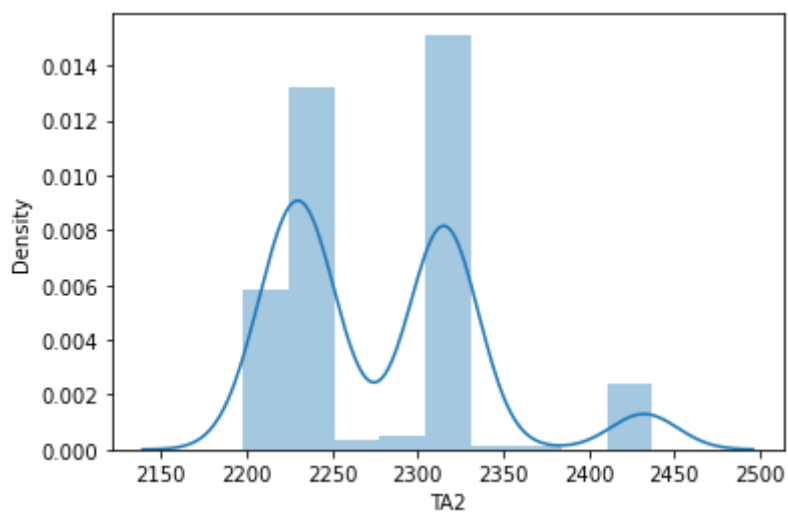
In [14]: `df.columns`

Out[14]: Index(['Cst_Cnt', 'Btl_Cnt', 'Sta_ID', 'Depth_ID', 'Depthm', 'T_degC',
 'Salnty', 'O2ml_L', 'STheta', 'O2Sat', 'Oxy_μmol/Kg', 'BtlNum',
 'RecInd', 'T_prec', 'T_qual', 'S_prec', 'S_qual', 'P_qual', 'O_qual',
 'SThta', 'O2Satq', 'ChlorA', 'Chlqua', 'Phaeop', 'Phaqua', 'PO4uM',
 'PO4q', 'SiO3uM', 'SiO3qu', 'NO2uM', 'NO2q', 'NO3uM', 'NO3q', 'NH3uM',
 'NH3q', 'C14As1', 'C14A1p', 'C14A1q', 'C14As2', 'C14A2p', 'C14A2q',
 'DarkAs', 'DarkAp', 'DarkAq', 'MeanAs', 'MeanAp', 'MeanAq', 'IncTim',
 'LightP', 'R_Depth', 'R_TEMP', 'R_POTEMP', 'R_SALINITY', 'R_SIGMA',
 'R_SVA', 'R_DYNHT', 'R_O2', 'R_O2Sat', 'R_SIO3', 'R_PO4', 'R_NO3',
 'R_NO2', 'R_NH4', 'R_CHLA', 'R_PHAEO', 'R_PRES', 'R_SAMP', 'DIC1',
 'DIC2', 'TA1', 'TA2', 'pH2', 'pH1', 'DIC Quality Comment'],
 dtype='object')

In [15]: `sb.distplot(df["TA2"])`

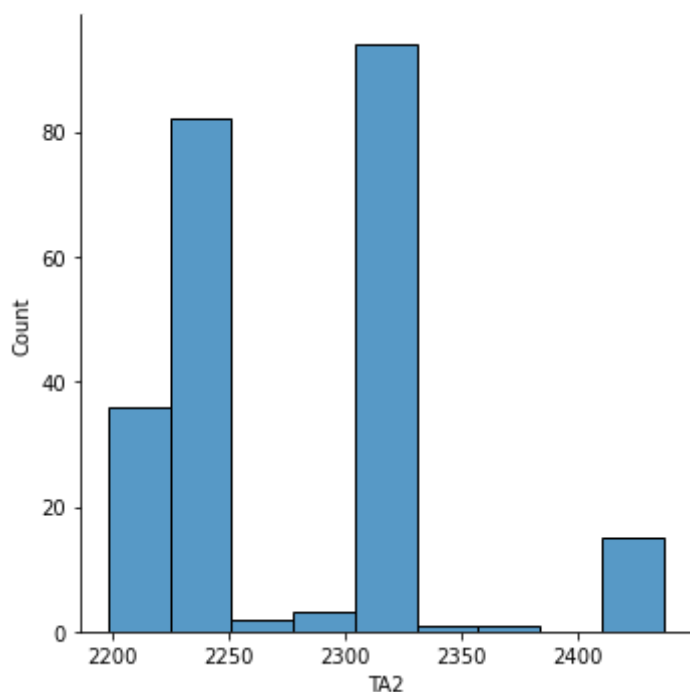
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function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

Out[15]: <AxesSubplot:xlabel='TA2', ylabel='Density'>



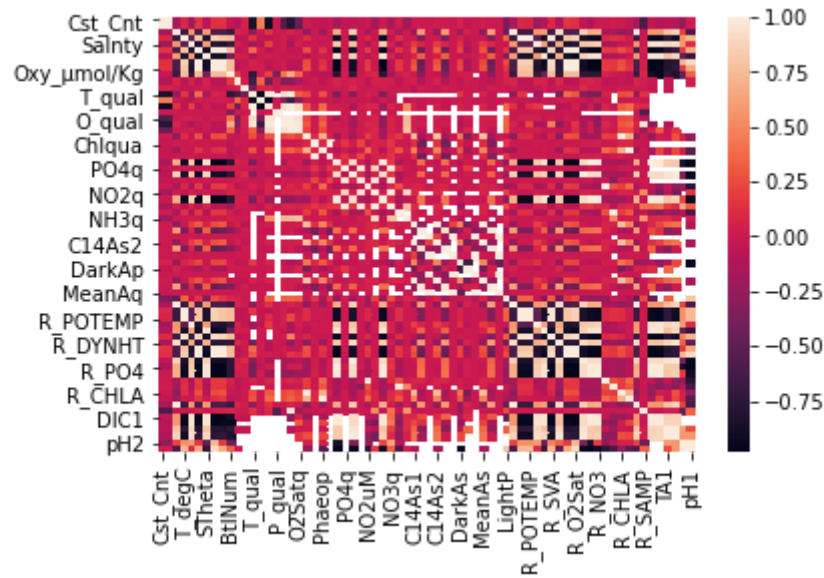
```
In [17]: sb.displot(df["TA2"])
```

```
Out[17]: <seaborn.axisgrid.FacetGrid at 0x21218412160>
```



```
In [19]: sb.heatmap(df.corr())
```

```
Out[19]: <AxesSubplot:>
```



```
In [ ]:
```

In [26]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as pp
```

In [27]:

```
import seaborn as sb
```

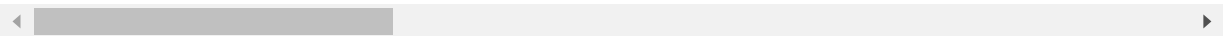
In [29]:

```
df = pd.read_csv(r"C:\Users\user\Desktop\8_BreastCancerPrediction.csv")
df
```

Out[29]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mea
0	842302	M	17.99	10.38	122.80	1001.0	0.1184
1	842517	M	20.57	17.77	132.90	1326.0	0.0847
2	84300903	M	19.69	21.25	130.00	1203.0	0.1096
3	84348301	M	11.42	20.38	77.58	386.1	0.1425
4	84358402	M	20.29	14.34	135.10	1297.0	0.1003
...
564	926424	M	21.56	22.39	142.00	1479.0	0.1110
565	926682	M	20.13	28.25	131.20	1261.0	0.0978
566	926954	M	16.60	28.08	108.30	858.1	0.0845
567	927241	M	20.60	29.33	140.10	1265.0	0.1178
568	92751	B	7.76	24.54	47.92	181.0	0.0526

569 rows × 33 columns



In [30]:

```
df.head(10)
```

Out[30]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean
0	842302	M	17.99	10.38	122.80	1001.0	0.11840
1	842517	M	20.57	17.77	132.90	1326.0	0.08474
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960
3	84348301	M	11.42	20.38	77.58	386.1	0.14250
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030
5	843786	M	12.45	15.70	82.57	477.1	0.12780
6	844359	M	18.25	19.98	119.60	1040.0	0.09463
7	84458202	M	13.71	20.83	90.20	577.9	0.11890
8	844981	M	13.00	21.82	87.50	519.8	0.12730
9	84501001	M	12.46	24.04	83.97	475.9	0.11860

10 rows × 33 columns

In [31]:

df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 33 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   id                                     569 non-null    int64
1   diagnosis                             569 non-null    object
2   radius_mean                           569 non-null    float64
3   texture_mean                           569 non-null    float64
4   perimeter_mean                         569 non-null    float64
5   area_mean                             569 non-null    float64
6   smoothness_mean                       569 non-null    float64
7   compactness_mean                      569 non-null    float64
8   concavity_mean                        569 non-null    float64
9   concave points_mean                   569 non-null    float64
10  symmetry_mean                         569 non-null    float64
11  fractal_dimension_mean                 569 non-null    float64
12  radius_se                             569 non-null    float64
13  texture_se                             569 non-null    float64
14  perimeter_se                           569 non-null    float64
15  area_se                               569 non-null    float64
16  smoothness_se                         569 non-null    float64
17  compactness_se                        569 non-null    float64
18  concavity_se                          569 non-null    float64
19  concave points_se                     569 non-null    float64
20  symmetry_se                           569 non-null    float64
21  fractal_dimension_se                   569 non-null    float64
22  radius_worst                          569 non-null    float64
23  texture_worst                         569 non-null    float64
24  perimeter_worst                       569 non-null    float64
25  area_worst                            569 non-null    float64
26  smoothness_worst                      569 non-null    float64
27  compactness_worst                     569 non-null    float64
28  concavity_worst                       569 non-null    float64
29  concave points_worst                   569 non-null    float64
30  symmetry_worst                         569 non-null    float64
31  fractal_dimension_worst                569 non-null    float64
32  Unnamed: 32                           0 non-null      float64
dtypes: float64(31), int64(1), object(1)
memory usage: 146.8+ KB
```

In [32]:

df.describe()

Out[32]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	c
count	5.690000e+02	569.000000	569.000000	569.000000	569.000000	569.000000	
mean	3.037183e+07	14.127292	19.289649	91.969033	654.889104	0.096360	
std	1.250206e+08	3.524049	4.301036	24.298981	351.914129	0.014064	
min	8.670000e+03	6.981000	9.710000	43.790000	143.500000	0.052630	
25%	8.692180e+05	11.700000	16.170000	75.170000	420.300000	0.086370	
50%	9.060240e+05	13.370000	18.840000	86.240000	551.100000	0.095870	
75%	8.813129e+06	15.780000	21.800000	104.100000	782.700000	0.105300	
max	9.113205e+08	28.110000	39.280000	188.500000	2501.000000	0.163400	

8 rows × 32 columns

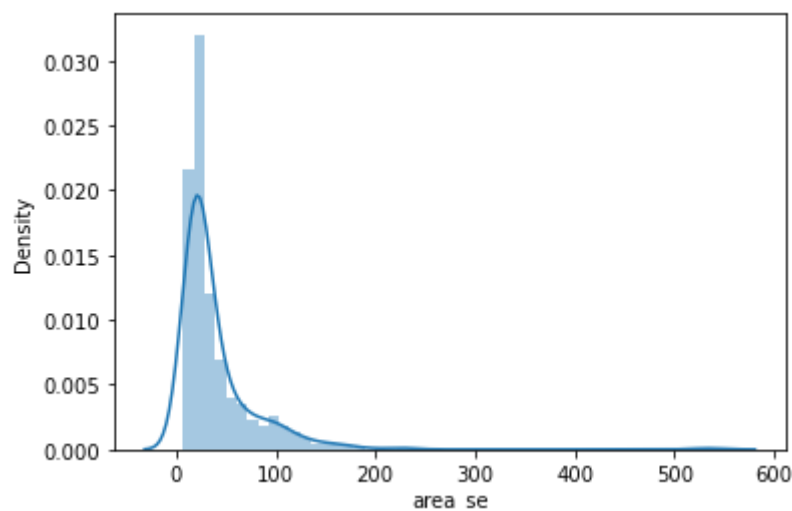
```
In [33]: df.columns
```

```
Out[33]: Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',  
              'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',  
              'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',  
              'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',  
              'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',  
              'fractal_dimension_se', 'radius_worst', 'texture_worst',  
              'perimeter_worst', 'area_worst', 'smoothness_worst',  
              'compactness_worst', 'concavity_worst', 'concave points_worst',  
              'symmetry_worst', 'fractal_dimension_worst', 'Unnamed: 32'],  
              dtype='object')
```

```
In [34]: sb.distplot(df["area_se"])
```

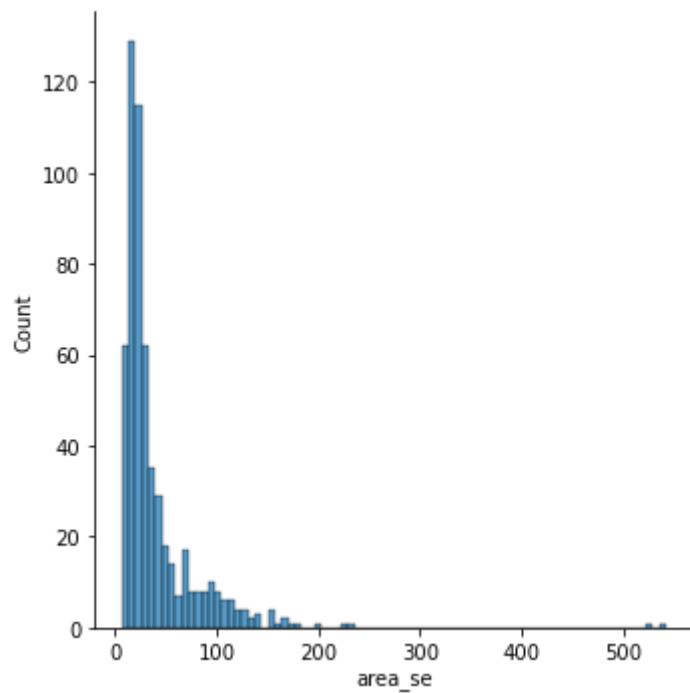
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function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

```
Out[34]: <AxesSubplot:xlabel='area_se', ylabel='Density'>
```



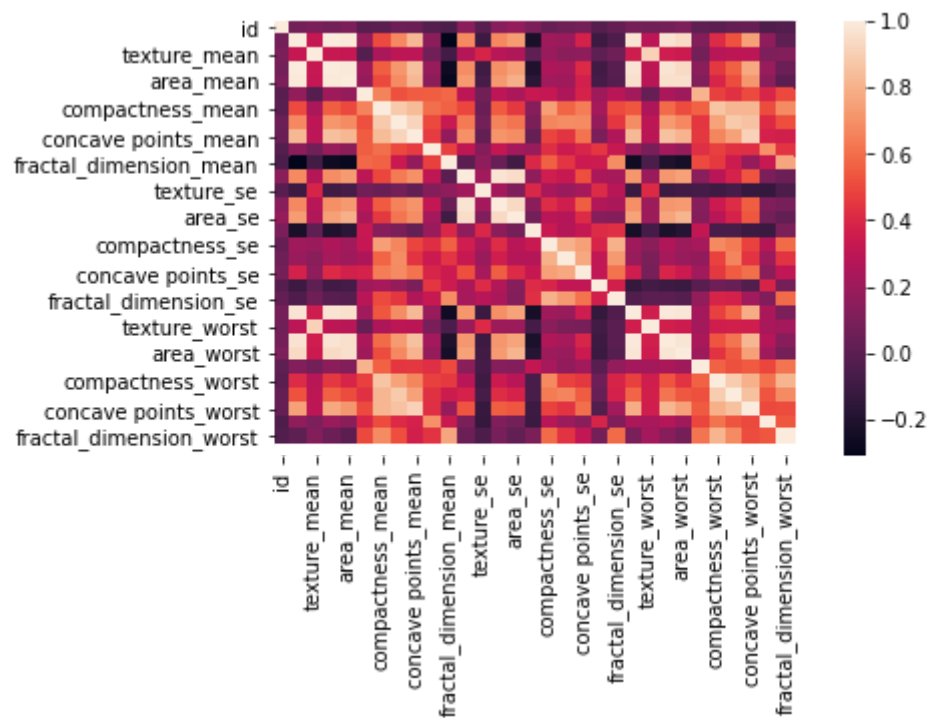
```
In [35]: sb.displot(df["area_se"])
```

```
Out[35]: <seaborn.axisgrid.FacetGrid at 0x21226b70700>
```



In [36]: `sb.heatmap(df.corr())`

Out[36]: <AxesSubplot:>



In []:

In [26]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as pp
```

In [27]:

```
import seaborn as sb
```

In [37]:

```
df = pd.read_csv(r"C:\Users\user\Desktop\6_Salesworkload1.csv")
df
```

Out[37]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLeas
0	10.2016	1.0	United Kingdom	88253.0	London (I)	1.0	Dry	3184.764	0.
1	10.2016	1.0	United Kingdom	88253.0	London (I)	2.0	Frozen	1582.941	0.
2	10.2016	1.0	United Kingdom	88253.0	London (I)	3.0	other	47.205	0.
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	0.
4	10.2016	1.0	United Kingdom	88253.0	London (I)	5.0	Fruits & Vegetables	1759.173	0.
...
7653	06.2017	9.0	Sweden	29650.0	Gothenburg	12.0	Checkout	6322.323	0.
7654	06.2017	9.0	Sweden	29650.0	Gothenburg	16.0	Customer Services	4270.479	0.
7655	06.2017	9.0	Sweden	29650.0	Gothenburg	11.0	Delivery	0	0.
7656	06.2017	9.0	Sweden	29650.0	Gothenburg	17.0	others	2224.929	0.
7657	06.2017	9.0	Sweden	29650.0	Gothenburg	18.0	all	39652.2	0.

7658 rows × 14 columns

In [38]:

```
df.head(10)
```

Out[38]:

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLease	
0	10.2016	1.0	United Kingdom	88253.0	London (I)	1.0	Dry	3184.764	0.0	398
1	10.2016	1.0	United Kingdom	88253.0	London (I)	2.0	Frozen	1582.941	0.0	82
2	10.2016	1.0	United Kingdom	88253.0	London (I)	3.0	other	47.205	0.0	438
3	10.2016	1.0	United Kingdom	88253.0	London (I)	4.0	Fish	1623.852	0.0	309

	MonthYear	Time index	Country	StoreID	City	Dept_ID	Dept. Name	HoursOwn	HoursLease	
4	10.2016	1.0	United Kingdom	88253.0	London (l)	5.0	Fruits & Vegetables	1759.173	0.0	165
5	10.2016	1.0	United Kingdom	88253.0	London (l)	6.0	Meat	8270.316	0.0	1713
6	10.2016	1.0	United Kingdom	88253.0	London (l)	13.0	Food	16468.251	0.0	3107
7	10.2016	1.0	United Kingdom	88253.0	London (l)	7.0	Clothing	4698.471	0.0	213
8	10.2016	1.0	United Kingdom	88253.0	London (l)	8.0	Household	1183.272	0.0	54
9	10.2016	1.0	United Kingdom	88253.0	London (l)	9.0	Hardware	2029.815	0.0	59

In [39]:

df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7658 entries, 0 to 7657
Data columns (total 14 columns):
#   Column                Non-Null Count  Dtype
---  -
0   MonthYear              7658 non-null  object
1   Time index             7650 non-null  float64
2   Country                7650 non-null  object
3   StoreID                7650 non-null  float64
4   City                   7650 non-null  object
5   Dept_ID                7650 non-null  float64
6   Dept. Name             7650 non-null  object
7   HoursOwn               7650 non-null  object
8   HoursLease             7650 non-null  float64
9   Sales units            7650 non-null  float64
10  Turnover               7650 non-null  float64
11  Customer               0 non-null     float64
12  Area (m2)              7650 non-null  object
13  Opening hours          7650 non-null  object
dtypes: float64(7), object(7)
memory usage: 837.7+ KB
```

In [40]:

df.describe()

Out[40]:

	Time index	StoreID	Dept_ID	HoursLease	Sales units	Turnover	Customer
count	7650.000000	7650.000000	7650.000000	7650.000000	7.650000e+03	7.650000e+03	0.0
mean	5.000000	61995.220000	9.470588	22.036078	1.076471e+06	3.721393e+06	NaN
std	2.582158	29924.581631	5.337429	133.299513	1.728113e+06	6.003380e+06	NaN
min	1.000000	12227.000000	1.000000	0.000000	0.000000e+00	0.000000e+00	NaN
25%	3.000000	29650.000000	5.000000	0.000000	5.457125e+04	2.726798e+05	NaN
50%	5.000000	75400.500000	9.000000	0.000000	2.932300e+05	9.319575e+05	NaN
75%	7.000000	87703.000000	14.000000	0.000000	9.175075e+05	3.264432e+06	NaN
max	9.000000	98422.000000	18.000000	3984.000000	1.124296e+07	4.271739e+07	NaN

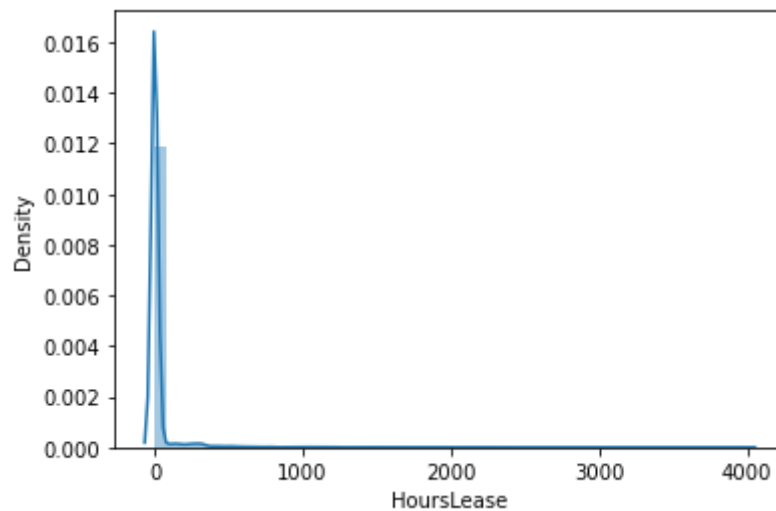
```
In [41]: df.columns
```

```
Out[41]: Index(['MonthYear', 'Time index', 'Country', 'StoreID', 'City', 'Dept_ID',  
              'Dept. Name', 'HoursOwn', 'HoursLease', 'Sales units', 'Turnover',  
              'Customer', 'Area (m2)', 'Opening hours'],  
             dtype='object')
```

```
In [42]: sb.distplot(df["HoursLease"])
```

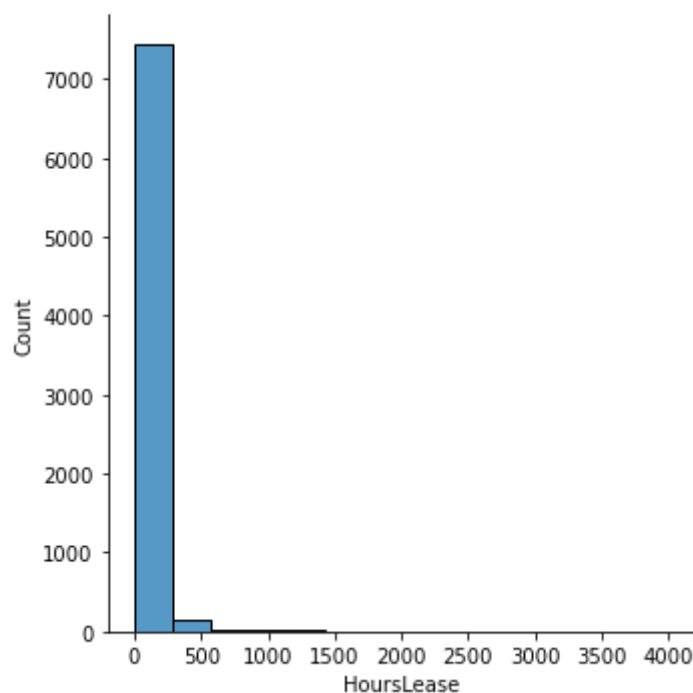
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function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

```
Out[42]: <AxesSubplot:xlabel='HoursLease', ylabel='Density'>
```



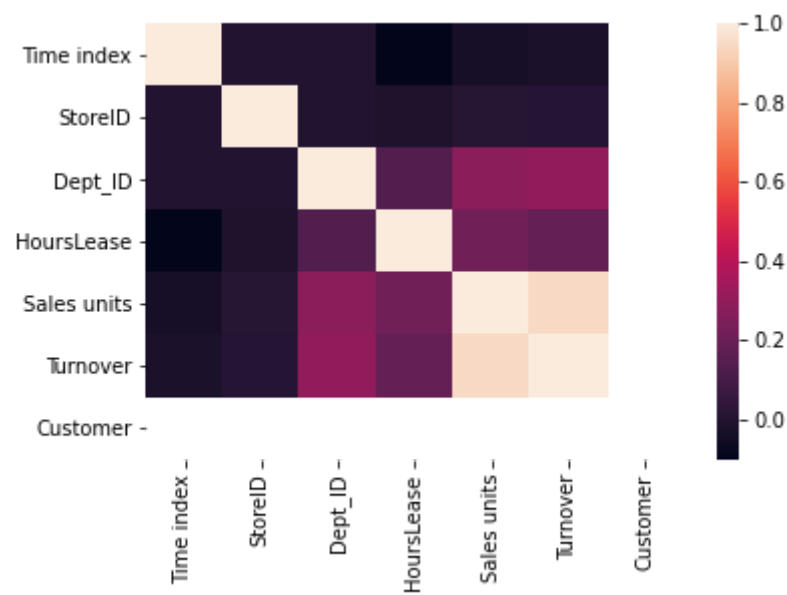
```
In [43]: sb.displot(df["HoursLease"])
```

```
Out[43]: <seaborn.axisgrid.FacetGrid at 0x21226ed6a90>
```



```
In [44]: sb.heatmap(df.corr())
```

Out[44]: <AxesSubplot:>



In []:

```
In [26]: import numpy as np
import pandas as pd
import matplotlib.pyplot as pp
```

```
In [27]: import seaborn as sb
```

```
In [45]: df = pd.read_csv(r"C:\Users\user\Desktop\3_Fitness-1.csv")
df
```

Out[45]:

	Row Labels	Sum of Jan	Sum of Feb	Sum of Mar	Sum of Total Sales
0	A	5.62%	7.73%	6.16%	75
1	B	4.21%	17.27%	19.21%	160
2	C	9.83%	11.60%	5.17%	101
3	D	2.81%	21.91%	7.88%	127
4	E	25.28%	10.57%	11.82%	179
5	F	8.15%	16.24%	18.47%	167
6	G	18.54%	8.76%	17.49%	171
7	H	25.56%	5.93%	13.79%	170
8	Grand Total	100.00%	100.00%	100.00%	1150

```
In [46]: df.head(10)
```

Out[46]:

	Row Labels	Sum of Jan	Sum of Feb	Sum of Mar	Sum of Total Sales
0	A	5.62%	7.73%	6.16%	75
1	B	4.21%	17.27%	19.21%	160
2	C	9.83%	11.60%	5.17%	101
3	D	2.81%	21.91%	7.88%	127
4	E	25.28%	10.57%	11.82%	179
5	F	8.15%	16.24%	18.47%	167
6	G	18.54%	8.76%	17.49%	171
7	H	25.56%	5.93%	13.79%	170
8	Grand Total	100.00%	100.00%	100.00%	1150

```
In [47]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9 entries, 0 to 8
Data columns (total 5 columns):
#   Column              Non-Null Count  Dtype
---  -
0   Row Labels          9 non-null      object
1   Sum of Jan          9 non-null      object
2   Sum of Feb          9 non-null      object
```

```

3   Sum of Mar          9 non-null   object
4   Sum of Total Sales  9 non-null   int64
dtypes: int64(1), object(4)
memory usage: 488.0+ bytes

```

In [48]: `df.describe()`

Out[48]:

	Sum of Total Sales
count	9.000000
mean	255.555556
std	337.332963
min	75.000000
25%	127.000000
50%	167.000000
75%	171.000000
max	1150.000000

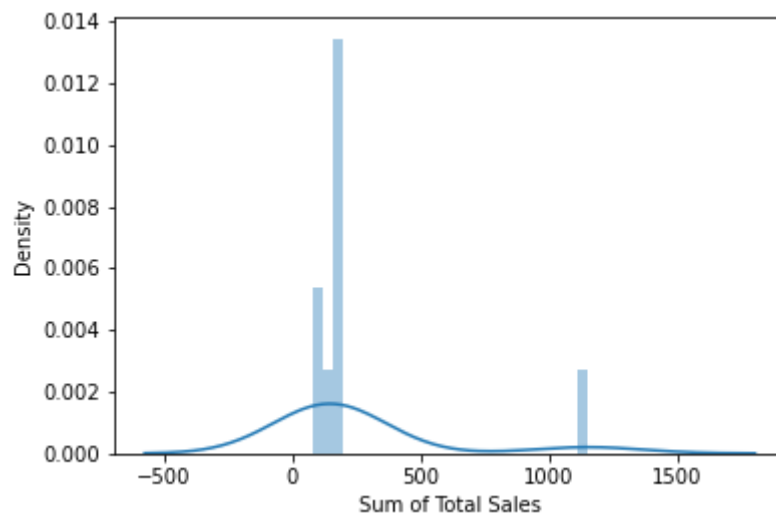
In [49]: `df.columns`

Out[49]: Index(['Row Labels', 'Sum of Jan', 'Sum of Feb', 'Sum of Mar',
'Sum of Total Sales'],
dtype='object')

In [50]: `sb.distplot(df["Sum of Total Sales"])`

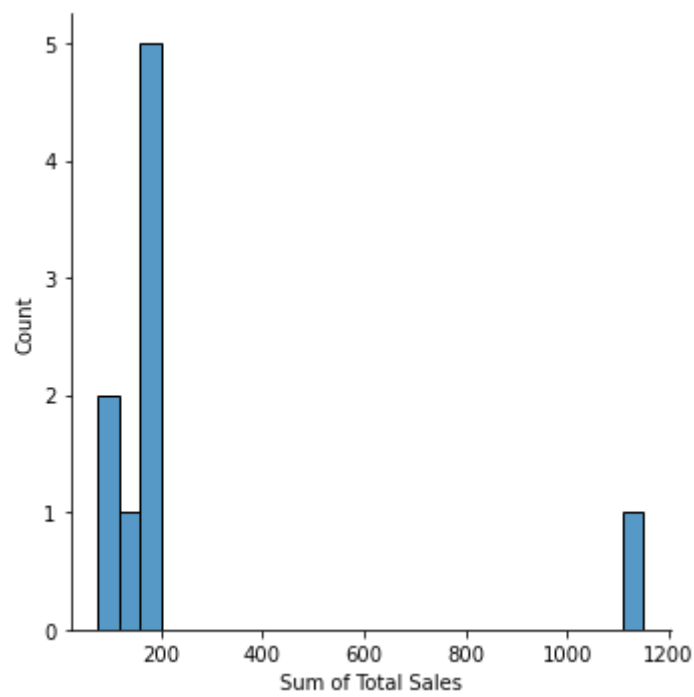
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function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

Out[50]: <AxesSubplot:xlabel='Sum of Total Sales', ylabel='Density'>



In [51]: `sb.displot(df["Sum of Total Sales"])`

Out[51]: <seaborn.axisgrid.FacetGrid at 0x212270c2310>



```
In [54]: sb.heatmap(df.corr())
```

Out[54]: <AxesSubplot:>



```
In [ ]:
```

```
In [26]: import numpy as np
import pandas as pd
import matplotlib.pyplot as pp
```

```
In [27]: import seaborn as sb
```

```
In [55]: df = pd.read_csv(r"C:\Users\user\Desktop\4_drug200.csv")
df
```

Out[55]:

	Age	Sex	BP	Cholesterol	Na_to_K	Drug
0	23	F	HIGH	HIGH	25.355	drugY
1	47	M	LOW	HIGH	13.093	drugC
2	47	M	LOW	HIGH	10.114	drugC
3	28	F	NORMAL	HIGH	7.798	drugX
4	61	F	LOW	HIGH	18.043	drugY
...
195	56	F	LOW	HIGH	11.567	drugC
196	16	M	LOW	HIGH	12.006	drugC
197	52	M	NORMAL	HIGH	9.894	drugX
198	23	M	NORMAL	NORMAL	14.020	drugX
199	40	F	LOW	NORMAL	11.349	drugX

200 rows × 6 columns

```
In [56]: df.head(10)
```

Out[56]:

	Age	Sex	BP	Cholesterol	Na_to_K	Drug
0	23	F	HIGH	HIGH	25.355	drugY
1	47	M	LOW	HIGH	13.093	drugC
2	47	M	LOW	HIGH	10.114	drugC
3	28	F	NORMAL	HIGH	7.798	drugX
4	61	F	LOW	HIGH	18.043	drugY
5	22	F	NORMAL	HIGH	8.607	drugX
6	49	F	NORMAL	HIGH	16.275	drugY
7	41	M	LOW	HIGH	11.037	drugC
8	60	M	NORMAL	HIGH	15.171	drugY
9	43	M	LOW	NORMAL	19.368	drugY

```
In [57]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype
---  -
 0   Age             200 non-null   int64
 1   Sex             200 non-null   object
 2   BP              200 non-null   object
 3   Cholesterol     200 non-null   object
 4   Na_to_K         200 non-null   float64
 5   Drug            200 non-null   object
dtypes: float64(1), int64(1), object(4)
memory usage: 9.5+ KB
```

```
In [58]: df.describe()
```

```
Out[58]:
```

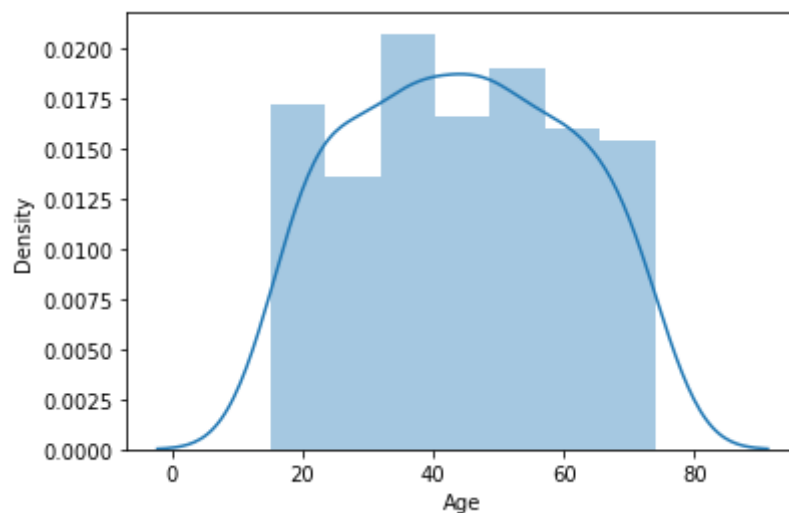
	Age	Na_to_K
count	200.000000	200.000000
mean	44.315000	16.084485
std	16.544315	7.223956
min	15.000000	6.269000
25%	31.000000	10.445500
50%	45.000000	13.936500
75%	58.000000	19.380000
max	74.000000	38.247000

```
In [59]: df.columns
```

```
Out[59]: Index(['Age', 'Sex', 'BP', 'Cholesterol', 'Na_to_K', 'Drug'], dtype='object')
```

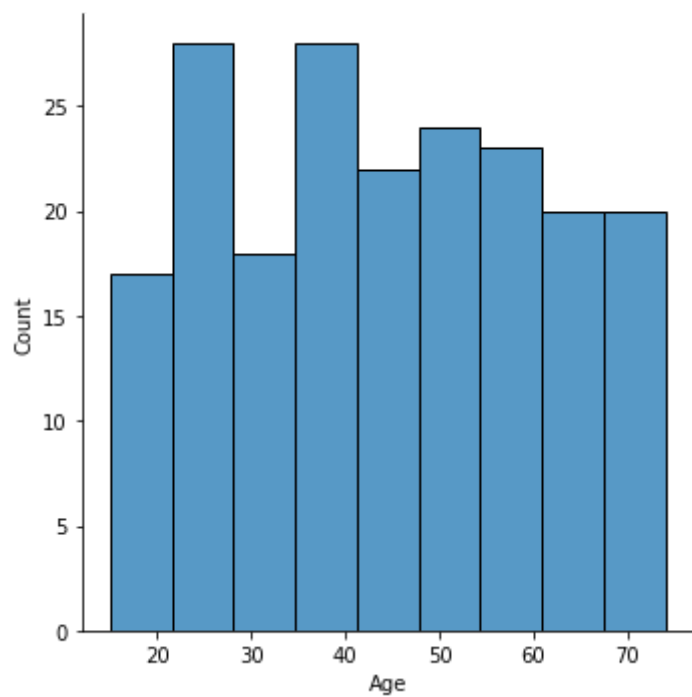
```
In [61]: sb.distplot(df["Age"])
```

```
Out[61]: <AxesSubplot:xlabel='Age', ylabel='Density'>
```



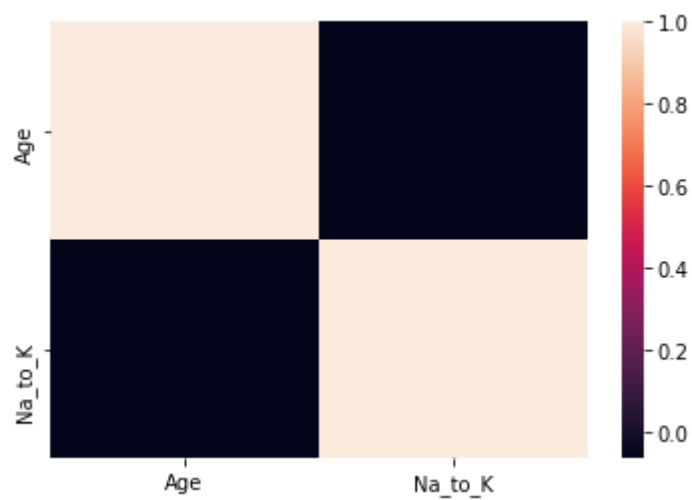
```
In [62]: sb.displot(df["Age"])
```

Out[62]: <seaborn.axisgrid.FacetGrid at 0x21218cd3070>



In [63]: `sb.heatmap(df.corr())`

Out[63]: <AxesSubplot:>



In []:

In [26]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as pp
```

In [27]:

```
import seaborn as sb
```

In [64]:

```
df = pd.read_csv(r"C:\Users\user\Desktop\10_USA_Housing.csv")
df
```

Out[64]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferry Ap 674\nLaurabury, N 3701
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson View Suite 079\nLak Kathleen, CA
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Elizabet Stravenue\nDanieltow WI 06482
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFPO A 4482
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nFP AE 0938
...	
4995	60567.944140	7.830362	6.137356	3.46	22837.361035	1.060194e+06	USNS Williams\nFP AP 30153-765
4996	78491.275435	6.999135	6.576763	4.02	25616.115489	1.482618e+06	PSC 9258, Bc 8489\nAPO AA 4299 335
4997	63390.686886	7.250591	4.805081	2.13	33266.145490	1.030730e+06	4215 Tracy Garde Suite 076\nJoshualan VA 01
4998	68001.331235	5.534388	7.130144	5.44	42625.620156	1.198657e+06	USS Wallace\nFPO A 7331
4999	65510.581804	5.992305	6.792336	4.07	46501.283803	1.298950e+06	37778 George Ridge Apt. 509\nEast Holl NV 2

5000 rows × 7 columns

In [65]:

```
df.head(10)
```

Out[65]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Address
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	208 Michael Ferry Apt. 674\nLaurabury, NE 3701...
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	188 Johnson Views Suite 079\nLake Kathleen, CA...
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	9127 Elizabeth Stravenue\nDanieltown, WI 06482...
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	USS Barnett\nFPO AP 44820
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	USNS Raymond\nFPO AE 09386
5	80175.754159	4.988408	6.104512	4.04	26748.428425	1.068138e+06	06039 Jennifer Islands Apt. 443\nTracyport, KS...
6	64698.463428	6.025336	8.147760	3.41	60828.249085	1.502056e+06	4759 Daniel Shoals Suite 442\nNguyenburgh, CO ...
7	78394.339278	6.989780	6.620478	2.42	36516.358972	1.573937e+06	972 Joyce Viaduct\nLake William, TN 17778-6483
8	59927.660813	5.362126	6.393121	2.30	29387.396003	7.988695e+05	USS Gilbert\nFPO AA 20957
9	81885.927184	4.423672	8.167688	6.10	40149.965749	1.545155e+06	Unit 9446 Box 0958\nDPO AE 97025

In [66]:

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 7 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Avg. Area Income                      5000 non-null   float64
1   Avg. Area House Age                   5000 non-null   float64
2   Avg. Area Number of Rooms             5000 non-null   float64
3   Avg. Area Number of Bedrooms          5000 non-null   float64
4   Area Population                       5000 non-null   float64
5   Price                                 5000 non-null   float64
6   Address                               5000 non-null   object
dtypes: float64(6), object(1)
memory usage: 273.6+ KB
```

In [67]:

```
df.describe()
```

Out[67]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5.000000e+03
mean	68583.108984	5.977222	6.987792	3.981330	36163.516039	1.232073e+06
std	10657.991214	0.991456	1.005833	1.234137	9925.650114	3.531176e+05
min	17796.631190	2.644304	3.236194	2.000000	172.610686	1.593866e+04
25%	61480.562388	5.322283	6.299250	3.140000	29403.928702	9.975771e+05
50%	68804.286404	5.970429	7.002902	4.050000	36199.406689	1.232669e+06
75%	75783.338666	6.650808	7.665871	4.490000	42861.290769	1.471210e+06
max	107701.748378	9.519088	10.759588	6.500000	69621.713378	2.469066e+06

In [68]:

```
df.columns
```

```
Out[68]: Index(['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
               'Avg. Area Number of Bedrooms', 'Area Population', 'Price', 'Address'],
              dtype='object')
```

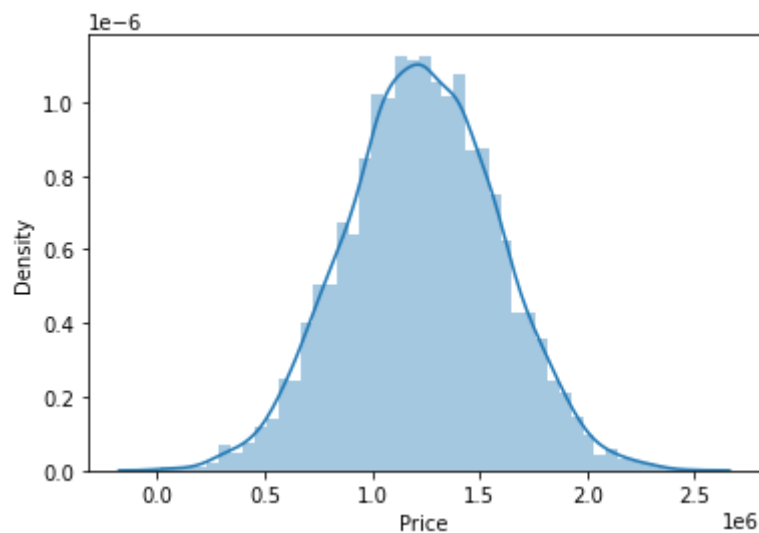
In [70]:

```
sb.distplot(df["Price"])
```

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)

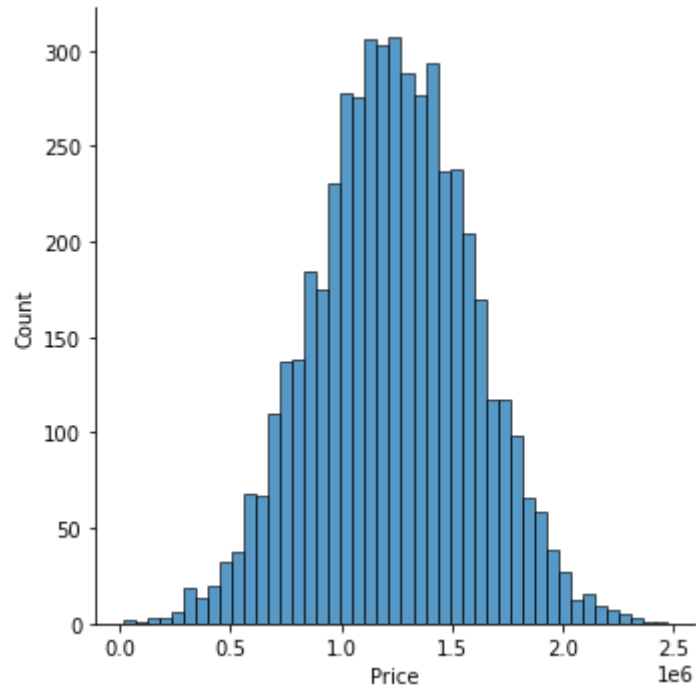
```
Out[70]: <AxesSubplot:xlabel='Price', ylabel='Density'>
```



In [71]:

```
sb.displot(df["Price"])
```

```
Out[71]: <seaborn.axisgrid.FacetGrid at 0x21218ed1f40>
```



```
In [72]: sb.heatmap(df.corr())
```

Out[72]: <AxesSubplot:>



```
In [ ]:
```


In [26]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as pp
```

In [27]:

```
import seaborn as sb
```

In [73]:

```
df = pd.read_csv(r"C:\Users\user\Desktop\7_uber.csv")
df
```

Out[73]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744
...
199995	42598914	2012-10-28 10:49:00.00000053	3.0	2012-10-28 10:49:00 UTC	-73.987042	40.739
199996	16382965	2014-03-14 01:09:00.0000008	7.5	2014-03-14 01:09:00 UTC	-73.984722	40.736
199997	27804658	2009-06-29 00:42:00.00000078	30.9	2009-06-29 00:42:00 UTC	-73.986017	40.756
199998	20259894	2015-05-20 14:56:25.0000004	14.5	2015-05-20 14:56:25 UTC	-73.997124	40.725
199999	11951496	2010-05-15 04:08:00.00000076	14.1	2010-05-15 04:08:00 UTC	-73.984395	40.720

200000 rows × 9 columns

In [74]:

```
df.head(10)
```

Out[74]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085
5	44470845	2011-02-12 02:27:09.00000006	4.9	2011-02-12 02:27:09 UTC	-73.969019	40.755910
6	48725865	2014-10-12 07:04:00.00000002	24.5	2014-10-12 07:04:00 UTC	-73.961447	40.693965
7	44195482	2012-12-11 13:52:00.000000029	2.5	2012-12-11 13:52:00 UTC	0.000000	0.000000
8	15822268	2012-02-17 09:32:00.000000043	9.7	2012-02-17 09:32:00 UTC	-73.975187	40.745767
9	50611056	2012-03-29 19:06:00.0000000273	12.5	2012-03-29 19:06:00 UTC	-74.001065	40.741787

In [75]:

df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 9 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Unnamed: 0            200000 non-null int64
1   key                   200000 non-null object
2   fare_amount           200000 non-null float64
3   pickup_datetime       200000 non-null object
4   pickup_longitude      200000 non-null float64
5   pickup_latitude       200000 non-null float64
6   dropoff_longitude     199999 non-null float64
7   dropoff_latitude     199999 non-null float64
8   passenger_count       200000 non-null int64
dtypes: float64(5), int64(2), object(2)
memory usage: 13.7+ MB
```

In [76]:

df.describe()

	Unnamed: 0	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_la
count	2.000000e+05	200000.000000	200000.000000	200000.000000	199999.000000	199999.0
mean	2.771250e+07	11.359955	-72.527638	39.935885	-72.525292	39.9
std	1.601382e+07	9.901776	11.437787	7.720539	13.117408	6.7
min	1.000000e+00	-52.000000	-1340.648410	-74.015515	-3356.666300	-881.9
25%	1.382535e+07	6.000000	-73.992065	40.734796	-73.991407	40.7
50%	2.774550e+07	8.500000	-73.981823	40.752592	-73.980093	40.7
75%	4.155530e+07	12.500000	-73.967154	40.767158	-73.963658	40.7
max	5.542357e+07	499.000000	57.418457	1644.421482	1153.572603	872.6

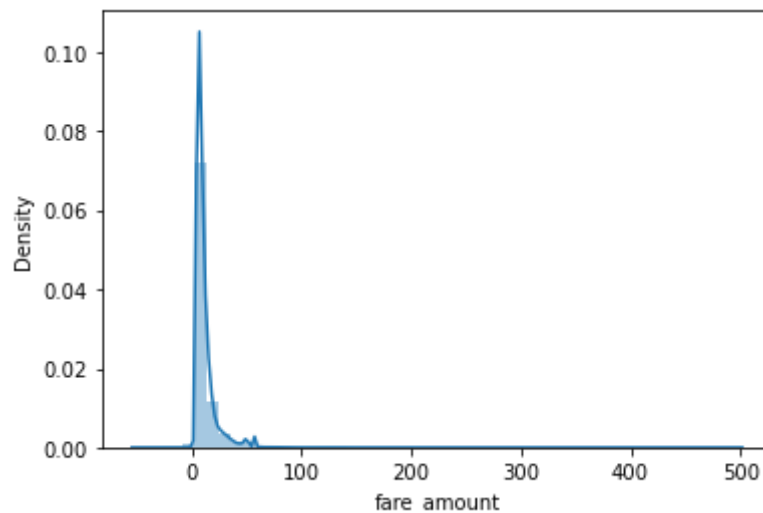
```
In [77]: df.columns
```

```
Out[77]: Index(['Unnamed: 0', 'key', 'fare_amount', 'pickup_datetime',  
              'pickup_longitude', 'pickup_latitude', 'dropoff_longitude',  
              'dropoff_latitude', 'passenger_count'],  
             dtype='object')
```

```
In [78]: sb.distplot(df["fare_amount"])
```

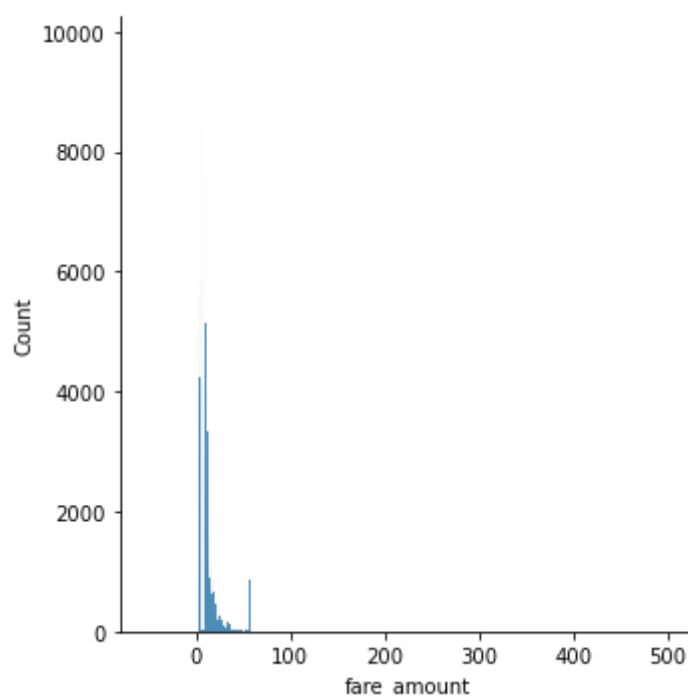
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function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

```
Out[78]: <AxesSubplot:xlabel='fare_amount', ylabel='Density'>
```



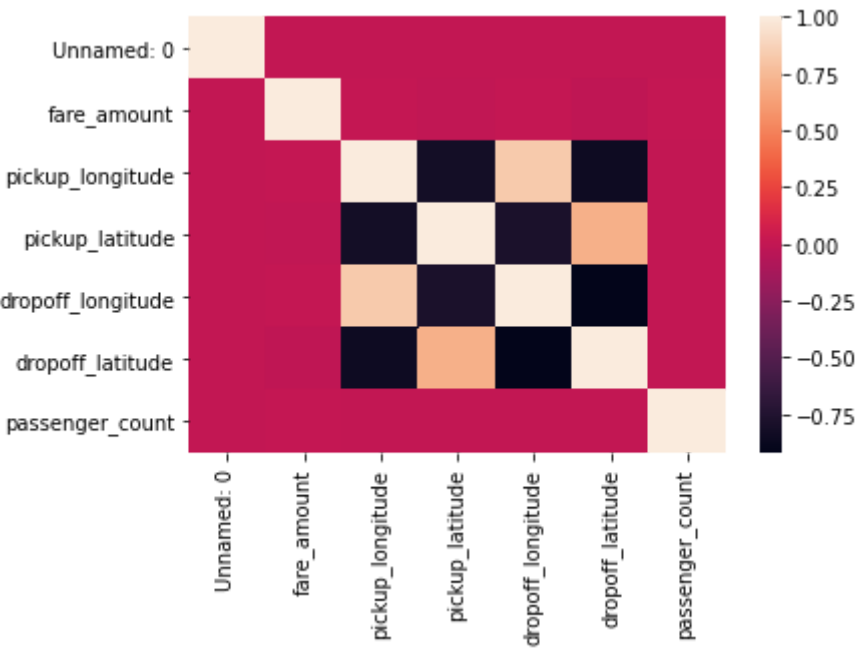
```
In [79]: sb.displot(df["fare_amount"])
```

```
Out[79]: <seaborn.axisgrid.FacetGrid at 0x21219032e20>
```



```
In [80]: sb.heatmap(df.corr())
```

Out[80]: <AxesSubplot:>



In []:

In [26]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as pp
```

In [27]:

```
import seaborn as sb
```

In [81]:

```
df = pd.read_csv(r"C:\Users\user\Desktop\2015.csv")
df
```

Out[81]:

	Country	Region	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedo
0	Switzerland	Western Europe	1	7.587	0.03411	1.39651	1.34951	0.94143	0.6655
1	Iceland	Western Europe	2	7.561	0.04884	1.30232	1.40223	0.94784	0.6287
2	Denmark	Western Europe	3	7.527	0.03328	1.32548	1.36058	0.87464	0.6493
3	Norway	Western Europe	4	7.522	0.03880	1.45900	1.33095	0.88521	0.6697
4	Canada	North America	5	7.427	0.03553	1.32629	1.32261	0.90563	0.6329
...
153	Rwanda	Sub-Saharan Africa	154	3.465	0.03464	0.22208	0.77370	0.42864	0.5920
154	Benin	Sub-Saharan Africa	155	3.340	0.03656	0.28665	0.35386	0.31910	0.4845
155	Syria	Middle East and Northern Africa	156	3.006	0.05015	0.66320	0.47489	0.72193	0.1568
156	Burundi	Sub-Saharan Africa	157	2.905	0.08658	0.01530	0.41587	0.22396	0.1185
157	Togo	Sub-Saharan Africa	158	2.839	0.06727	0.20868	0.13995	0.28443	0.3645

158 rows × 12 columns

In [82]:

```
df.head(10)
```

Out[82]:

	Country	Region	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom
0	Switzerland	Western Europe	1	7.587	0.03411	1.39651	1.34951	0.94143	0.66557
1	Iceland	Western Europe	2	7.561	0.04884	1.30232	1.40223	0.94784	0.62877
2	Denmark	Western Europe	3	7.527	0.03328	1.32548	1.36058	0.87464	0.64938
3	Norway	Western Europe	4	7.522	0.03880	1.45900	1.33095	0.88521	0.66973
4	Canada	North America	5	7.427	0.03553	1.32629	1.32261	0.90563	0.63297
5	Finland	Western Europe	6	7.406	0.03140	1.29025	1.31826	0.88911	0.64169
6	Netherlands	Western Europe	7	7.378	0.02799	1.32944	1.28017	0.89284	0.61576
7	Sweden	Western Europe	8	7.364	0.03157	1.33171	1.28907	0.91087	0.65980
8	New Zealand	Australia and New Zealand	9	7.286	0.03371	1.25018	1.31967	0.90837	0.63938
9	Australia	Australia and New Zealand	10	7.284	0.04083	1.33358	1.30923	0.93156	0.65124



In [83]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 158 entries, 0 to 157
Data columns (total 12 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Country                               158 non-null    object
1   Region                                158 non-null    object
2   Happiness Rank                        158 non-null    int64
3   Happiness Score                       158 non-null    float64
4   Standard Error                       158 non-null    float64
5   Economy (GDP per Capita)              158 non-null    float64
6   Family                                158 non-null    float64
7   Health (Life Expectancy)              158 non-null    float64
8   Freedom                               158 non-null    float64
9   Trust (Government Corruption)         158 non-null    float64
10  Generosity                            158 non-null    float64
11  Dystopia Residual                      158 non-null    float64
dtypes: float64(9), int64(1), object(2)
memory usage: 14.9+ KB
```

In [84]:

```
df.describe()
```

Out[84]:

	Happiness Rank	Happiness Score	Standard Error	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	(Govern Corrup
count	158.000000	158.000000	158.000000	158.000000	158.000000	158.000000	158.000000	158.00
mean	79.493671	5.375734	0.047885	0.846137	0.991046	0.630259	0.428615	0.14
std	45.754363	1.145010	0.017146	0.403121	0.272369	0.247078	0.150693	0.12
min	1.000000	2.839000	0.018480	0.000000	0.000000	0.000000	0.000000	0.00
25%	40.250000	4.526000	0.037268	0.545808	0.856823	0.439185	0.328330	0.00
50%	79.500000	5.232500	0.043940	0.910245	1.029510	0.696705	0.435515	0.10
75%	118.750000	6.243750	0.052300	1.158448	1.214405	0.811013	0.549092	0.18
max	158.000000	7.587000	0.136930	1.690420	1.402230	1.025250	0.669730	0.51

In [85]:

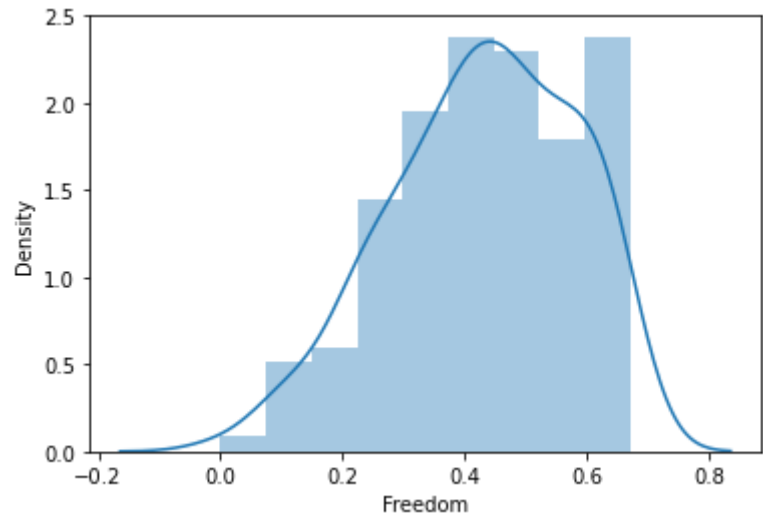
```
df.columns
```

Out[85]: Index(['Country', 'Region', 'Happiness Rank', 'Happiness Score', 'Standard Error', 'Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)', 'Freedom', 'Trust (Government Corruption)', 'Generosity', 'Dystopia Residual'], dtype='object')

In [87]:

```
sb.distplot(df["Freedom"])
```

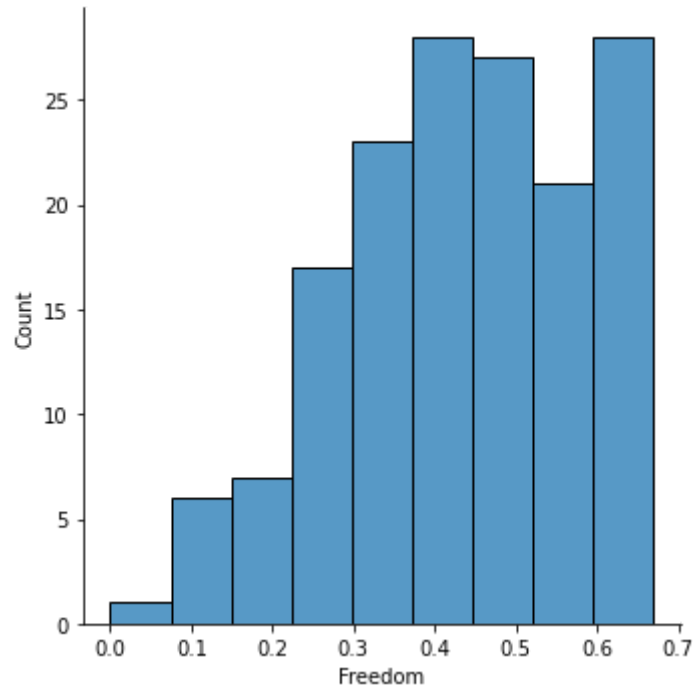
Out[87]: <AxesSubplot:xlabel='Freedom', ylabel='Density'>



In [88]:

```
sb.displot(df["Freedom"])
```

Out[88]: <seaborn.axisgrid.FacetGrid at 0x21223af47c0>



```
In [89]: sb.heatmap(df.corr())
```

Out[89]: <AxesSubplot:>



```
In [ ]:
```



```
In [26]: import numpy as np
import pandas as pd
import matplotlib.pyplot as pp
```

```
In [27]: import seaborn as sb
```

```
In [90]: df = pd.read_csv(r"C:\Users\user\Desktop\fiat500_VehicleSelection_Dataset.csv")
df
```

Out[90]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon
0	1.0	lounge	51.0	882.0	25000.0	1.0	44.907242	8.611559868
1	2.0	pop	51.0	1186.0	32500.0	1.0	45.666359	12.24188995
2	3.0	sport	74.0	4658.0	142228.0	1.0	45.503300	11.41784
3	4.0	lounge	51.0	2739.0	160000.0	1.0	40.633171	17.63460922
4	5.0	pop	73.0	3074.0	106880.0	1.0	41.903221	12.49565029
...
1544	NaN	NaN	NaN	NaN	NaN	NaN	NaN	length
1545	NaN	NaN	NaN	NaN	NaN	NaN	NaN	concat
1546	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Null values
1547	NaN	NaN	NaN	NaN	NaN	NaN	NaN	find
1548	NaN	NaN	NaN	NaN	NaN	NaN	NaN	search

1549 rows × 11 columns



```
In [91]: df.head(10)
```

Out[91]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	1.0	lounge	51.0	882.0	25000.0	1.0	44.907242	8.611559868	89000
1	2.0	pop	51.0	1186.0	32500.0	1.0	45.666359	12.24188995	88000
2	3.0	sport	74.0	4658.0	142228.0	1.0	45.503300	11.41784	42000
3	4.0	lounge	51.0	2739.0	160000.0	1.0	40.633171	17.63460922	60000
4	5.0	pop	73.0	3074.0	106880.0	1.0	41.903221	12.49565029	57000
5	6.0	pop	74.0	3623.0	70225.0	1.0	45.000702	7.68227005	79000
6	7.0	lounge	51.0	731.0	11600.0	1.0	44.907242	8.611559868	107000
7	8.0	lounge	51.0	1521.0	49076.0	1.0	41.903221	12.49565029	91000
8	9.0	sport	73.0	4049.0	76000.0	1.0	45.548000	11.54946995	56000

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
9	10.0	sport	51.0	3653.0	89000.0	1.0	45.438301	10.99170017	60000.0

In [92]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1549 entries, 0 to 1548
Data columns (total 11 columns):
#   Column                Non-Null Count  Dtype
---  -
0   ID                    1538 non-null   float64
1   model                 1538 non-null   object
2   engine_power          1538 non-null   float64
3   age_in_days           1538 non-null   float64
4   km                    1538 non-null   float64
5   previous_owners       1538 non-null   float64
6   lat                   1538 non-null   float64
7   lon                   1549 non-null   object
8   price                 1549 non-null   object
9   Unnamed: 9            0 non-null      float64
10  Unnamed: 10           1 non-null      object
dtypes: float64(7), object(4)
memory usage: 133.2+ KB
```

In [93]:

```
df.describe()
```

Out[93]:

	ID	engine_power	age_in_days	km	previous_owners	lat	Unnamed: 9	Unnamed: 10
count	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000		
mean	769.500000	51.904421	1650.980494	53396.011704	1.123537	43.541361		
std	444.126671	3.988023	1289.522278	40046.830723	0.416423	2.133518		
min	1.000000	51.000000	366.000000	1232.000000	1.000000	36.855839		
25%	385.250000	51.000000	670.000000	20006.250000	1.000000	41.802990		
50%	769.500000	51.000000	1035.000000	39031.000000	1.000000	44.394096		
75%	1153.750000	51.000000	2616.000000	79667.750000	1.000000	45.467960		
max	1538.000000	77.000000	4658.000000	235000.000000	4.000000	46.795612		

In [94]:

```
df.columns
```

Out[94]:

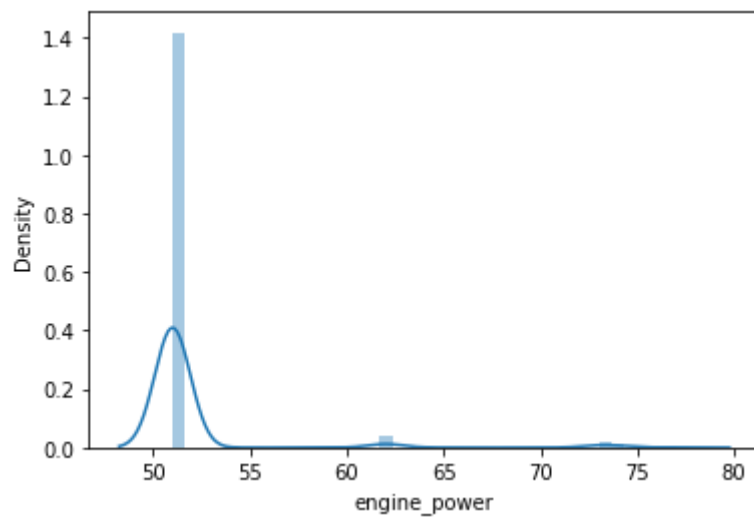
```
Index(['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owners',
       'lat', 'lon', 'price', 'Unnamed: 9', 'Unnamed: 10'],
      dtype='object')
```

In [95]:

```
sb.distplot(df["engine_power"])
```

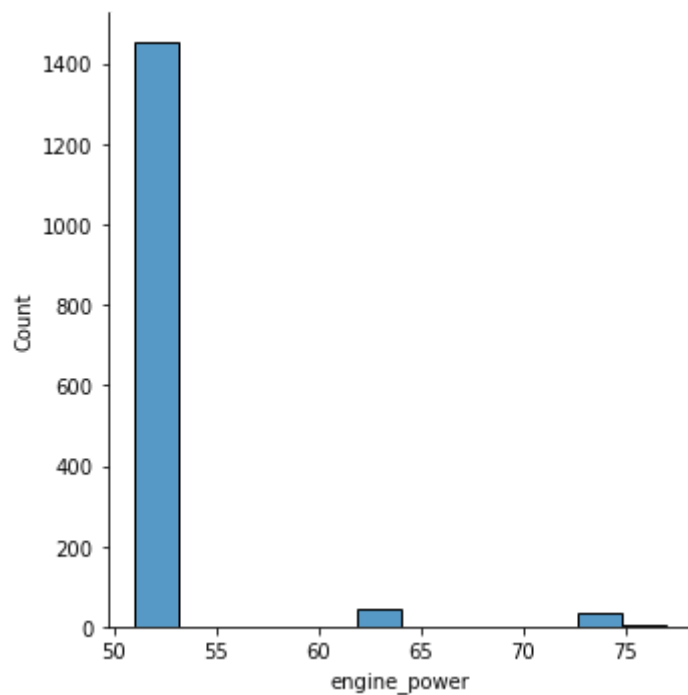
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function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

Out[95]: <AxesSubplot:xlabel='engine_power', ylabel='Density'>



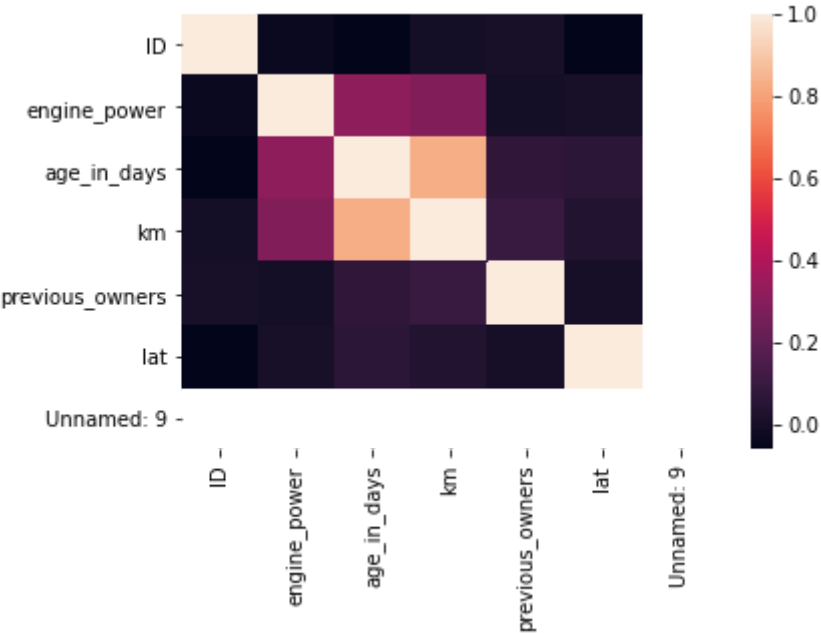
In [96]: `sb.displot(df["engine_power"])`

Out[96]: <seaborn.axisgrid.FacetGrid at 0x212241645b0>



In [97]: `sb.heatmap(df.corr())`

Out[97]: <AxesSubplot:>



In []:

```
In [26]: import numpy as np
import pandas as pd
import matplotlib.pyplot as pp
```

```
In [27]: import seaborn as sb
```

```
In [98]: df = pd.read_csv(r"C:\Users\user\Desktop\5_Instagram data.csv")
df
```

Out[98]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Follower
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	1
115	5731	1923	1368	2266	65	135	4	1	148	20	
116	4139	1133	1538	1367	33	36	0	1	92	34	

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Follows
117	32695	11815	3147	17414	170	1095	2	75	549	148	2
118	36919	13473	4176	16444	2547	653	5	26	443	611	2

119 rows × 13 columns

In [99]:

df.head(10)

Out[99]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Follows
0	3920	2586	1028	619	56	98	9	5	162	35	2
1	5394	2727	1838	1174	78	194	7	14	224	48	10
2	4021	2085	1188	0	533	41	11	1	131	62	12
3	4528	2700	621	932	73	172	10	7	213	23	8
4	2518	1704	255	279	37	96	5	4	123	8	0
5	3884	2046	1214	329	43	74	7	10	144	9	2

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Follows
6	2621	1543	599	333	25	22	5	1	76	26	0
7	3541	2071	628	500	60	135	4	9	124	12	6
8	3749	2384	857	248	49	155	6	8	159	36	4
9	4115	2609	1104	178	46	122	6	3	191	31	6

In [100...

```
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 [101...

```
df.describe()
```

Out[101...

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments
count	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
std	4843.780105	1489.386348	1884.361443	2613.026132	289.431031	156.317731	3.544576
min	1941.000000	1133.000000	116.000000	0.000000	9.000000	22.000000	0.000000

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments
25%	3467.000000	1945.000000	726.000000	157.500000	38.000000	65.000000	4.000000
50%	4289.000000	2207.000000	1278.000000	326.000000	74.000000	109.000000	6.000000
75%	6138.000000	2602.500000	2363.500000	689.500000	196.000000	169.000000	8.000000
max	36919.000000	13473.000000	11817.000000	17414.000000	2547.000000	1095.000000	19.000000

In [102...

```
df.columns
```

Out[102...

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

In [103...

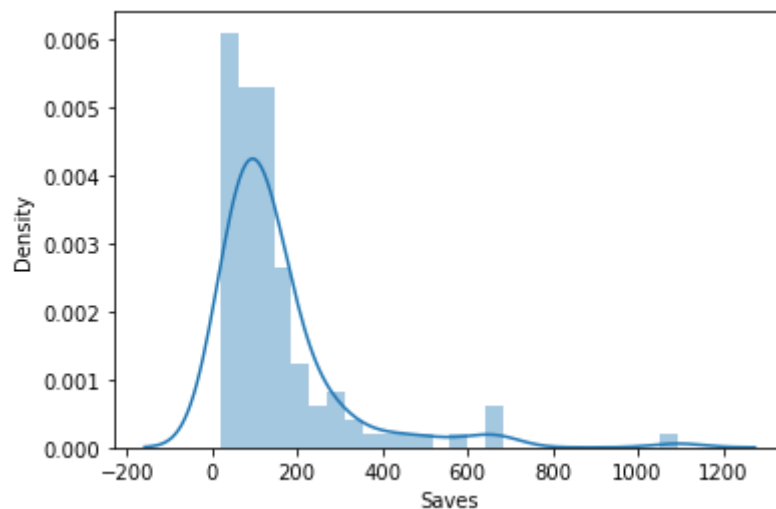
```
sb.distplot(df["Saves"])
```

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[103...

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

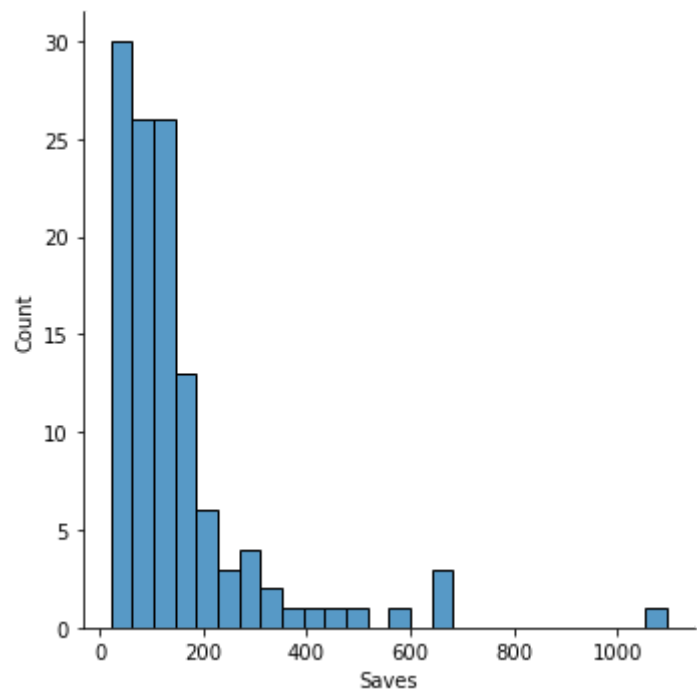


In [104...

```
sb.displot(df["Saves"])
```

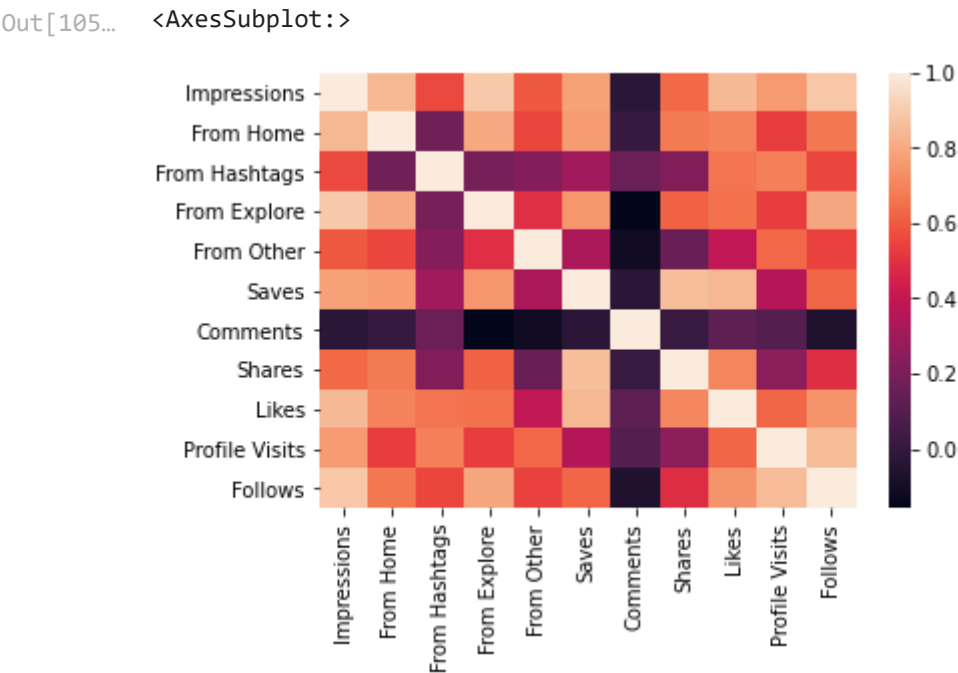
Out[104...

```
<seaborn.axisgrid.FacetGrid at 0x21223d2bb80>
```

In [105...

sb.heatmap(df.corr())



In []: