In [1]:
```
import pandas as pd
data=pd.read_csv("/home/placement/Downloads/fiat500.csv")
```

In [2]:
```
data.describe()
```

Out[2]:

	ID	engine_power	age_in_days	km	previous_owners	lat	lon	price
count	1538.000000	1538.000000	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	11.563428	8576.003901
std	444.126671	3.988023	1289.522278	40046.830723	0.416423	2.133518	2.328190	1939.958641
min	1.000000	51.000000	366.000000	1232.000000	1.000000	36.855839	7.245400	2500.000000
25%	385.250000	51.000000	670.000000	20006.250000	1.000000	41.802990	9.505090	7122.500000
50%	769.500000	51.000000	1035.000000	39031.000000	1.000000	44.394096	11.869260	9000.000000
75%	1153.750000	51.000000	2616.000000	79667.750000	1.000000	45.467960	12.769040	10000.000000
max	1538.000000	77.000000	4658.000000	235000.000000	4.000000	46.795612	18.365520	11100.000000

In [3]:
```
data.head(10)
```

Out[3]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	1	lounge	51	882	25000	1	44.907242	8.611560	8900
1	2	pop	51	1186	32500	1	45.666359	12.241890	8800
2	3	sport	74	4658	142228	1	45.503300	11.417840	4200
3	4	lounge	51	2739	160000	1	40.633171	17.634609	6000
4	5	pop	73	3074	106880	1	41.903221	12.495650	5700
5	6	pop	74	3623	70225	1	45.000702	7.682270	7900
6	7	lounge	51	731	11600	1	44.907242	8.611560	10750
7	8	lounge	51	1521	49076	1	41.903221	12.495650	9190
8	9	sport	73	4049	76000	1	45.548000	11.549470	5600
9	10	sport	51	3653	89000	1	45.438301	10.991700	6000

In [4]:
```
data['previous_owners'].unique()
```

Out[4]:
```
array([1, 2, 3, 4])
```

In [5]:
```
list(data.columns)
```

Out[5]:
```
['ID',
 'model',
 'engine_power',
 'age_in_days',
 'km',
 'previous_owners',
 'lat',
 'lon',
 'price']
```

In [6]:
```
data['model'].unique()
```

Out[6]:
```
array(['lounge', 'pop', 'sport'], dtype=object)
```

In [7]:
```
data.groupby(['previous_owners']).count()
```

Out[7]:

previous_owners	ID	model	engine_power	age_in_days	km	lat	lon	price
1	1389	1389	1389	1389	1389	1389	1389	1389
2	117	117	117	117	117	117	117	117
3	23	23	23	23	23	23	23	23
4	9	9	9	9	9	9	9	9

In [8]:
```
data.groupby(['model']).count()
```

Out[8]:

model	ID	engine_power	age_in_days	km	previous_owners	lat	lon	price
lounge	1094	1094	1094	1094	1094	1094	1094	1094
pop	358	358	358	358	358	358	358	358
sport	86	86	86	86	86	86	86	86

In [9]:
```
data1=data.drop(['lat','ID'],axis=1)
```

In [10]:
```
data1.head()
```

Out[10]:

	model	engine_power	age_in_days	km	previous_owners	lon	price
0	lounge	51	882	25000	1	8.611560	8900
1	pop	51	1186	32500	1	12.241890	8800
2	sport	74	4658	142228	1	11.417840	4200
3	lounge	51	2739	160000	1	17.634609	6000
4	pop	73	3074	106880	1	12.495650	5700

In [11]:
```
data['price'].sum()
```

Out[11]: 13189894

In [12]:
```
data2=data.loc[(data.model=='lounge')]
```

In [13]:
```
data2
```

Out[13]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	1	lounge	51	882	25000	1	44.907242	8.611560	8900
3	4	lounge	51	2739	160000	1	40.633171	17.634609	6000
6	7	lounge	51	731	11600	1	44.907242	8.611560	10750
7	8	lounge	51	1521	49076	1	41.903221	12.495650	9190
11	12	lounge	51	366	17500	1	45.069679	7.704920	10990
...	...	...	...	...	...	...	...	...	...
1528	1529	lounge	51	2861	126000	1	43.841980	10.515310	5500
1529	1530	lounge	51	731	22551	1	38.122070	13.361120	9900
1530	1531	lounge	51	670	29000	1	45.764648	8.994500	10800
1534	1535	lounge	74	3835	112000	1	45.845692	8.666870	4600
1536	1537	lounge	51	2557	80750	1	45.000702	7.682270	5990

1094 rows × 9 columns

In [14]:
```
data3=data.loc[(data.km<=50000)]
```

In [15]:
```
data3
```

Out[15]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	1	lounge	51	882	25000	1	44.907242	8.61156	8900
1	2	pop	51	1186	32500	1	45.666359	12.24189	8800
6	7	lounge	51	731	11600	1	44.907242	8.61156	10750
7	8	lounge	51	1521	49076	1	41.903221	12.49565	9190
10	11	pop	51	790	43286	1	40.871429	14.43896	8950
...	...	...	...	...	...	...	...	...	...
1525	1526	lounge	51	790	41870	1	45.707249	11.47760	9500
1526	1527	lounge	51	1705	23600	1	38.122070	13.36112	9300
1527	1528	pop	51	517	3000	1	40.748241	14.52835	9999
1529	1530	lounge	51	731	22551	1	38.122070	13.36112	9900
1530	1531	lounge	51	670	29000	1	45.764648	8.99450	10800

907 rows × 9 columns

In [16]:
```
data2=data.loc[(data.model=='lounge')&(data.previous_owners==1)]
```

In [17]:
```
data2
```

Out[17]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	1	lounge	51	882	25000	1	44.907242	8.611560	8900
3	4	lounge	51	2739	160000	1	40.633171	17.634609	6000
6	7	lounge	51	731	11600	1	44.907242	8.611560	10750
7	8	lounge	51	1521	49076	1	41.903221	12.495650	9190
11	12	lounge	51	366	17500	1	45.069679	7.704920	10990
...	...	...	...	...	...	...	...	...	...
1528	1529	lounge	51	2861	126000	1	43.841980	10.515310	5500
1529	1530	lounge	51	731	22551	1	38.122070	13.361120	9900
1530	1531	lounge	51	670	29000	1	45.764648	8.994500	10800
1534	1535	lounge	74	3835	112000	1	45.845692	8.666870	4600
1536	1537	lounge	51	2557	80750	1	45.000702	7.682270	5990

993 rows × 9 columns

In [18]:
```
data4=data.loc[(data.model=='lounge')|(data.model=='pop')]
```

In [19]:
```
data4
```

Out[19]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	1	lounge	51	882	25000	1	44.907242	8.611560	8900
1	2	pop	51	1186	32500	1	45.666359	12.241890	8800
3	4	lounge	51	2739	160000	1	40.633171	17.634609	6000
4	5	pop	73	3074	106880	1	41.903221	12.495650	5700
5	6	pop	74	3623	70225	1	45.000702	7.682270	7900
...	...	...	...	...	...	...	...	...	...
1532	1533	pop	51	1917	52008	1	45.548000	11.549470	9900
1534	1535	lounge	74	3835	112000	1	45.845692	8.666870	4600
1535	1536	pop	51	2223	60457	1	45.481541	9.413480	7500
1536	1537	lounge	51	2557	80750	1	45.000702	7.682270	5990
1537	1538	pop	51	1766	54276	1	40.323410	17.568270	7900

1452 rows × 9 columns

In [20]:
```
cor=data1.corr()
cor
```

```
/tmp/ipykernel_6888/870474124.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is
deprecated. In a future version, it will default to False. Select only valid columns or specify the value
of numeric_only to silence this warning.
  cor=data1.corr()
```

Out[20]:

	engine_power	age_in_days	km	previous_owners	lon	price
engine_power	1.000000	0.319190	0.285495	-0.005030	-0.005032	-0.277235
age_in_days	0.319190	1.000000	0.833890	0.075775	-0.042667	-0.893328
km	0.285495	0.833890	1.000000	0.097539	0.004839	-0.859373
previous_owners	-0.005030	0.075775	0.097539	1.000000	-0.026836	-0.076274
lon	-0.005032	-0.042667	0.004839	-0.026836	1.000000	-0.003541
price	-0.277235	-0.893328	-0.859373	-0.076274	-0.003541	1.000000

In [21]:
```
import warnings
warnings.filterwarnings('ignore')
```

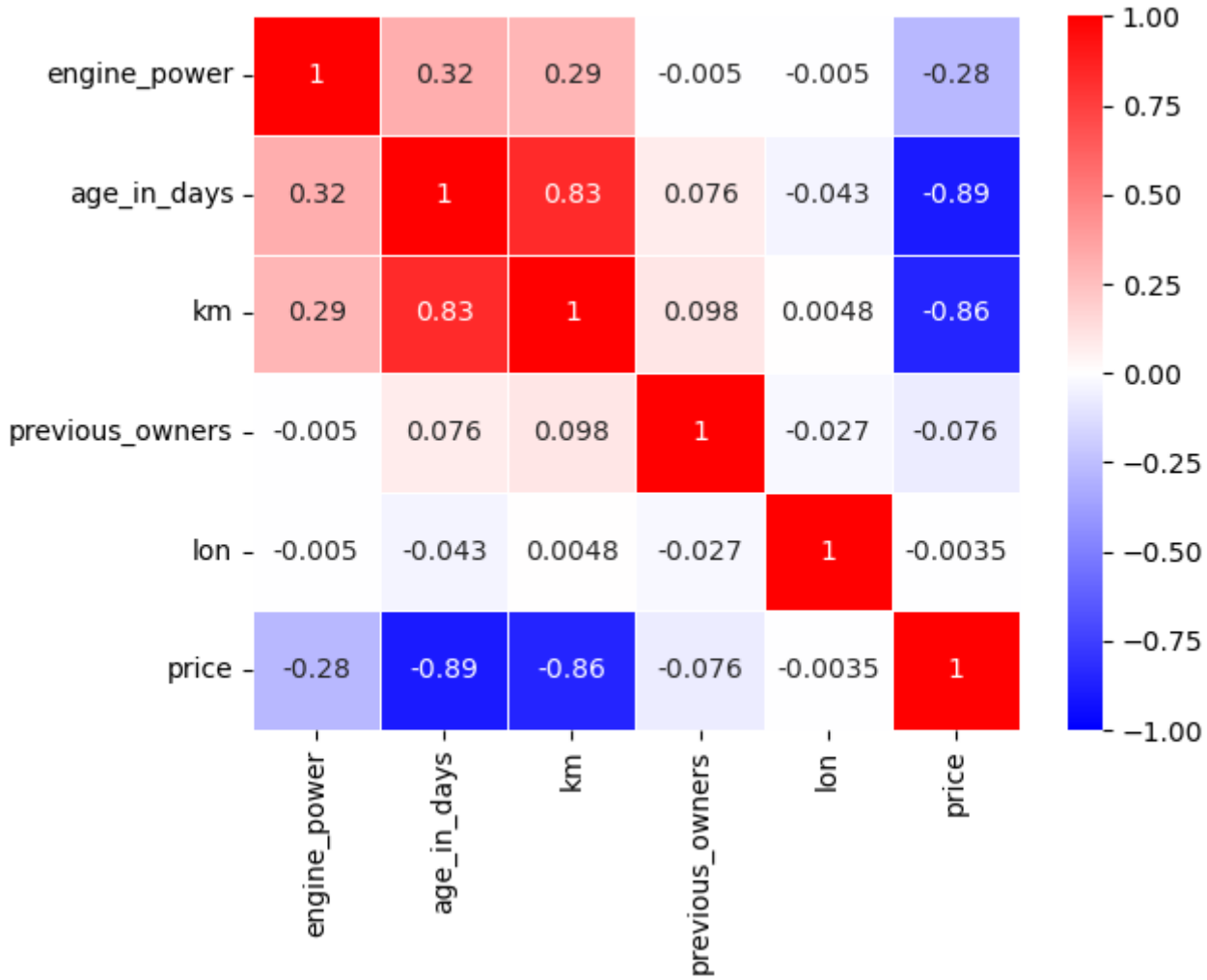
In [22]:
```
cor=data1.corr()
cor
```

Out[22]:

	engine_power	age_in_days	km	previous_owners	lon	price
engine_power	1.000000	0.319190	0.285495	-0.005030	-0.005032	-0.277235
age_in_days	0.319190	1.000000	0.833890	0.075775	-0.042667	-0.893328
km	0.285495	0.833890	1.000000	0.097539	0.004839	-0.859373
previous_owners	-0.005030	0.075775	0.097539	1.000000	-0.026836	-0.076274
lon	-0.005032	-0.042667	0.004839	-0.026836	1.000000	-0.003541
price	-0.277235	-0.893328	-0.859373	-0.076274	-0.003541	1.000000

In [23]:
```
import seaborn as sns
sns.heatmap(cor,vmax=1,vmin=-1,annot=True,linewidth=.5,cmap='bwr')
```

Out[23]: <Axes: >



In [ ]: