In [1]:

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

In [2]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\fiat500_VehicleSelection_Dataset (1).csv")
a
```

Out[2]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	
0	1.0	lounge	51.0	882.0	25000.0	1.0	44.907242	8.6115
1	2.0	pop	51.0	1186.0	32500.0	1.0	45.666359	12.241
2	3.0	sport	74.0	4658.0	142228.0	1.0	45.503300	11
3	4.0	lounge	51.0	2739.0	160000.0	1.0	40.633171	17.634
4	5.0	pop	73.0	3074.0	106880.0	1.0	41.903221	12.495
...	...	...	...	...	...	...	...	
1544	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1545	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1546	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Null
1547	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1548	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

1549 rows × 11 columns

In [3]:

```
b=a.head(100)
b
```

Out[3]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	
0	1.0	lounge	51.0	882.0	25000.0	1.0	44.907242	8.61155
1	2.0	pop	51.0	1186.0	32500.0	1.0	45.666359	12.2418
2	3.0	sport	74.0	4658.0	142228.0	1.0	45.503300	11.4
3	4.0	lounge	51.0	2739.0	160000.0	1.0	40.633171	17.6346
4	5.0	pop	73.0	3074.0	106880.0	1.0	41.903221	12.4956
...	...	...	...	...	...	...	...	
95	96.0	sport	51.0	4292.0	165600.0	1.0	44.715408	11.3083
96	97.0	pop	51.0	1066.0	28000.0	1.0	41.769051	12.6628
97	98.0	sport	51.0	2009.0	86000.0	2.0	40.633171	17.6346
98	99.0	lounge	51.0	456.0	18592.0	2.0	45.393600	10.4822
99	100.0	pop	51.0	731.0	41558.0	2.0	45.571220	9.15913

100 rows × 11 columns

In [4]:

```
b.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   ID               100 non-null    float64
 1   model            100 non-null    object 
 2   engine_power     100 non-null    float64
 3   age_in_days      100 non-null    float64
 4   km               100 non-null    float64
 5   previous_owners  100 non-null    float64
 6   lat              100 non-null    float64
 7   lon              100 non-null    object 
 8   price            100 non-null    object 
 9   Unnamed: 9       0 non-null      float64
 10  Unnamed: 10      0 non-null      object 
dtypes: float64(7), object(4)
memory usage: 8.7+ KB
```

In [5]:

```
b.describe()
```

Out[5]:

	ID	engine_power	age_in_days	km	previous_owners	lat
count	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000
mean	50.500000	53.010000	1935.300000	58812.180000	1.180000	43.612648
std	29.011492	6.014284	1414.251278	44728.034639	0.500101	2.083451
min	1.000000	51.000000	366.000000	4000.000000	1.000000	38.218128
25%	25.750000	51.000000	723.500000	19781.750000	1.000000	41.744165
50%	50.500000	51.000000	1446.000000	44032.000000	1.000000	44.831066
75%	75.250000	51.000000	3265.500000	95075.750000	1.000000	45.396568
max	100.000000	74.000000	4658.000000	188000.000000	3.000000	46.176498

In [6]:

```
c=b.dropna(axis=1)
c
```

Out[6]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	
0	1.0	lounge	51.0	882.0	25000.0	1.0	44.907242	8.61155
1	2.0	pop	51.0	1186.0	32500.0	1.0	45.666359	12.2418
2	3.0	sport	74.0	4658.0	142228.0	1.0	45.503300	11.4
3	4.0	lounge	51.0	2739.0	160000.0	1.0	40.633171	17.6346
4	5.0	pop	73.0	3074.0	106880.0	1.0	41.903221	12.4956
...	...	...	...	...	...	...	...	
95	96.0	sport	51.0	4292.0	165600.0	1.0	44.715408	11.3083
96	97.0	pop	51.0	1066.0	28000.0	1.0	41.769051	12.6628
97	98.0	sport	51.0	2009.0	86000.0	2.0	40.633171	17.6346
98	99.0	lounge	51.0	456.0	18592.0	2.0	45.393600	10.4822
99	100.0	pop	51.0	731.0	41558.0	2.0	45.571220	9.15913

100 rows × 9 columns

In [7]:

```
c.columns
```

Out[7]:

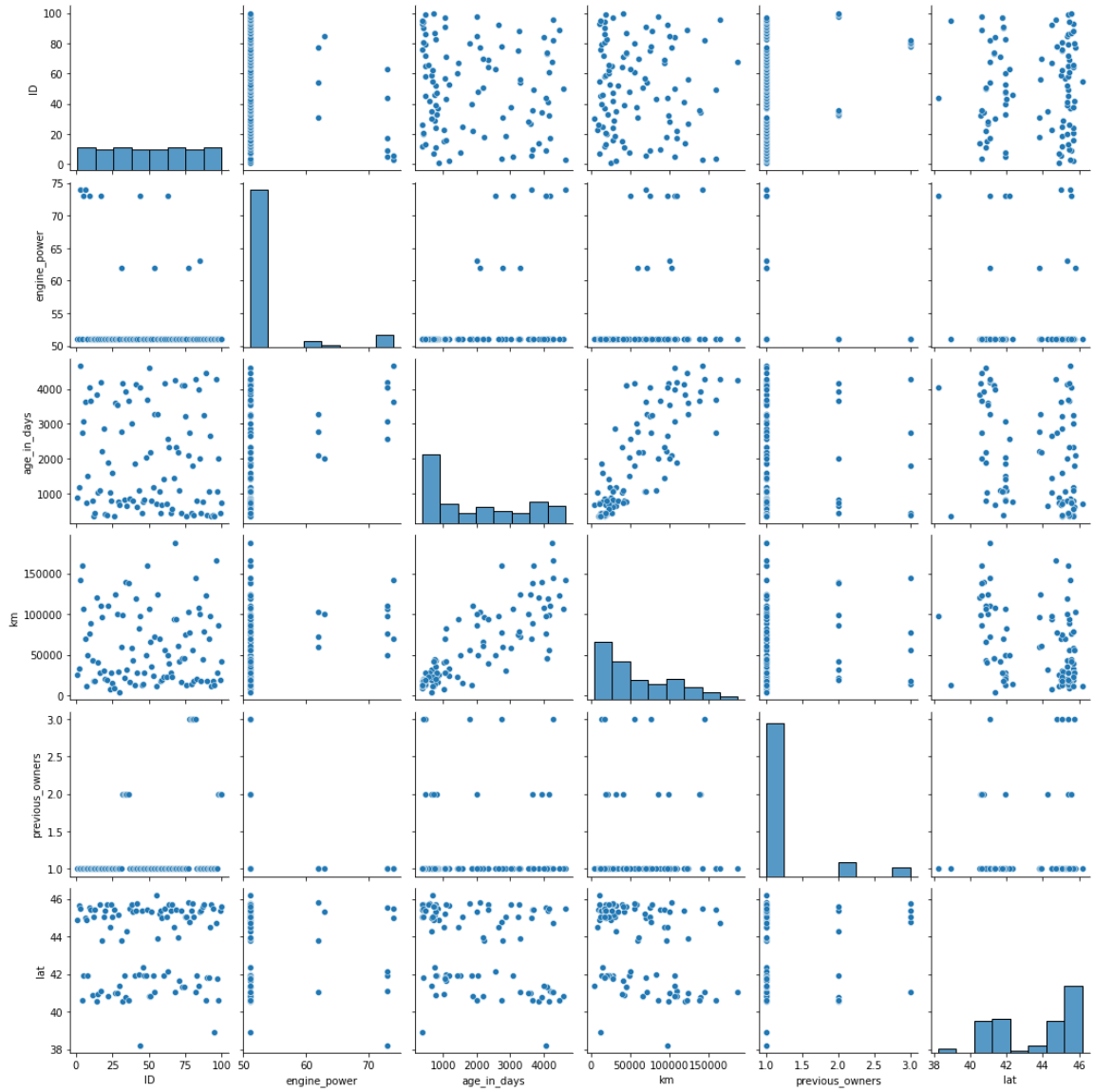
```
Index(['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owner
s',
       'lat', 'lon', 'price'],
      dtype='object')
```

In [8]:

```
sns.pairplot(c)
```

Out[8]:

```
<seaborn.axisgrid.PairGrid at 0x11c24fc1cd0>
```



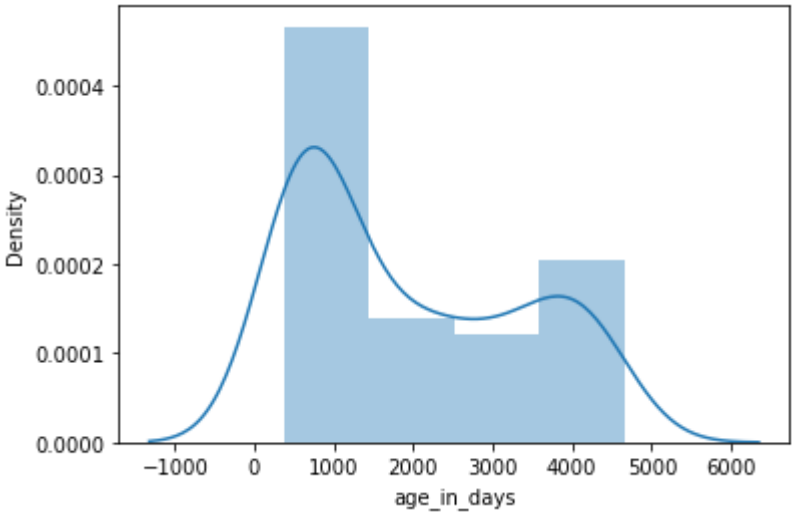
In [9]:

```
sns.distplot(c['age_in_days'])
```

```
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 fun
ction for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[9]:

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



In [10]:

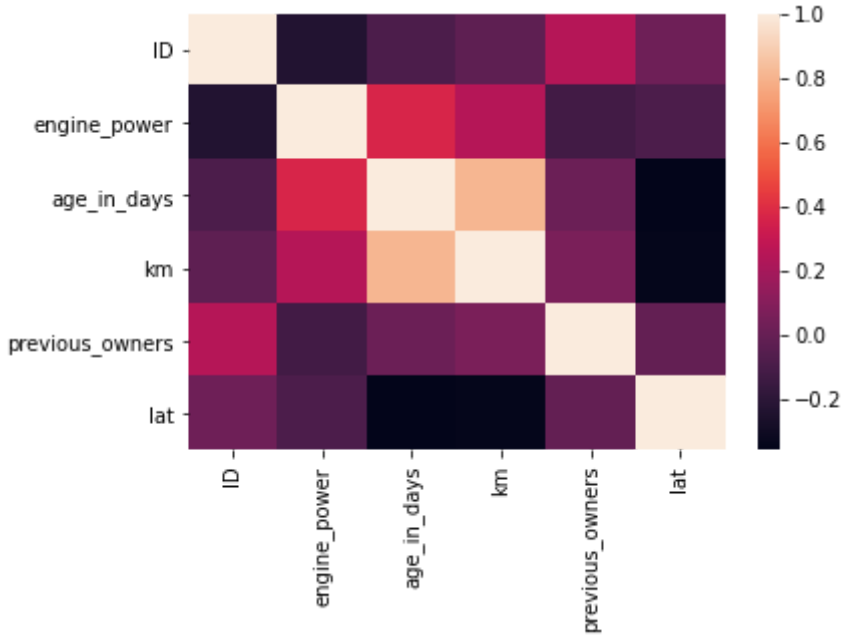
```
f=c[['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owners',
       'lat', 'lon', 'price']]
```

In [11]:

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

Out[11]:

```
<AxesSubplot:>
```



In [12]:

```
x=f[['ID','engine_power','km', 'previous_owners',
       'lat', 'lon', 'price']]
y=f['age_in_days']
```

In [13]:

```
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 [14]:

```
from sklearn.linear_model import LinearRegression

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

Out[14]:

```
LinearRegression()
```

In [15]:

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

```
7165.510059169973
```

In [16]:

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

Out[16]:

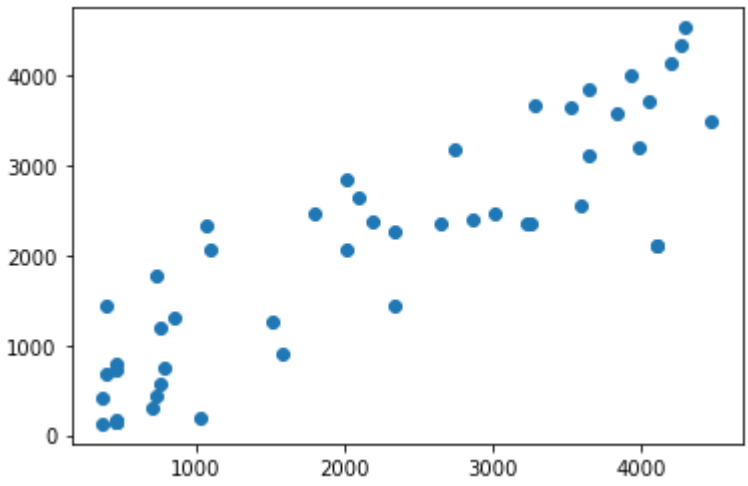
	Co-efficient
ID	-0.616914
engine_power	27.432007
km	0.005680
previous_owners	299.350722
lat	-48.495682
lon	-86.333966
price	-0.518771

In [17]:

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

Out[17]:

```
<matplotlib.collections.PathCollection at 0x11c275e8df0>
```



In [18]:

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

```
0.7682041262486937
```

In [19]:

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

Out[19]:

```
0.8920352363226728
```

RIDGE REGRESSION

In [20]:

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

In [21]:

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

Out[21]:

```
Ridge(alpha=10)
```

In [22]:

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

Out[22]:

```
0.7855916998020599
```

LASSO REGRESSION

In [23]:

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

Out[23]:

```
Lasso(alpha=10)
```

In [24]:

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

Out[24]:

```
0.7763901359381123
```

In [ ]:

```

```