D1

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

In [2]:
```
df=pd.read_csv(r"C:\Users\user\Downloads\1_fiat500_VehicleSelection_Dataset.csv
df
```

Out[2]:
```
        ID   model  engine_power  age_in_days        km  previous_owners        lat            l

   0   1.0  lounge          51.0        882.0   25000.0              1.0  44.907242  8.6115598
   1   2.0     pop          51.0       1186.0   32500.0              1.0  45.666359  12.241889
   2   3.0   sport          74.0       4658.0  142228.0              1.0  45.503300     11.417
   3   4.0  lounge          51.0       2739.0  160000.0              1.0  40.633171  17.634609
   4   5.0     pop          73.0       3074.0  106880.0              1.0  41.903221  12.495650
 ...   ...     ...           ...          ...       ...              ...        ...
1544   NaN     NaN           NaN          NaN       NaN              NaN        NaN       leng
1545   NaN     NaN           NaN          NaN       NaN              NaN        NaN       con
1546   NaN     NaN           NaN          NaN       NaN              NaN        NaN  Null valu
1547   NaN     NaN           NaN          NaN       NaN              NaN        NaN         fi
1548   NaN     NaN           NaN          NaN       NaN              NaN        NaN       sear

1549 rows × 11 columns
```

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

Out[3]:

	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
5	6.0	pop	74.0	3623.0	70225.0	1.0	45.000702	7.68227005
6	7.0	lounge	51.0	731.0	11600.0	1.0	44.907242	8.611559868
7	8.0	lounge	51.0	1521.0	49076.0	1.0	41.903221	12.49565029
8	9.0	sport	73.0	4049.0	76000.0	1.0	45.548000	11.54946995
9	10.0	sport	51.0	3653.0	89000.0	1.0	45.438301	10.99170017

In [4]:
```
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 [5]:
```
df.describe()
```

Out[5]:

	ID	engine_power	age_in_days	km	previous_owners	lat
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 [6]:
```
dft=df.drop(["Unnamed: 9","Unnamed: 10"],axis=1)
```

In [7]:
```
dff=dft.dropna()
```

In [8]:
```
dff.isnull().sum()
```

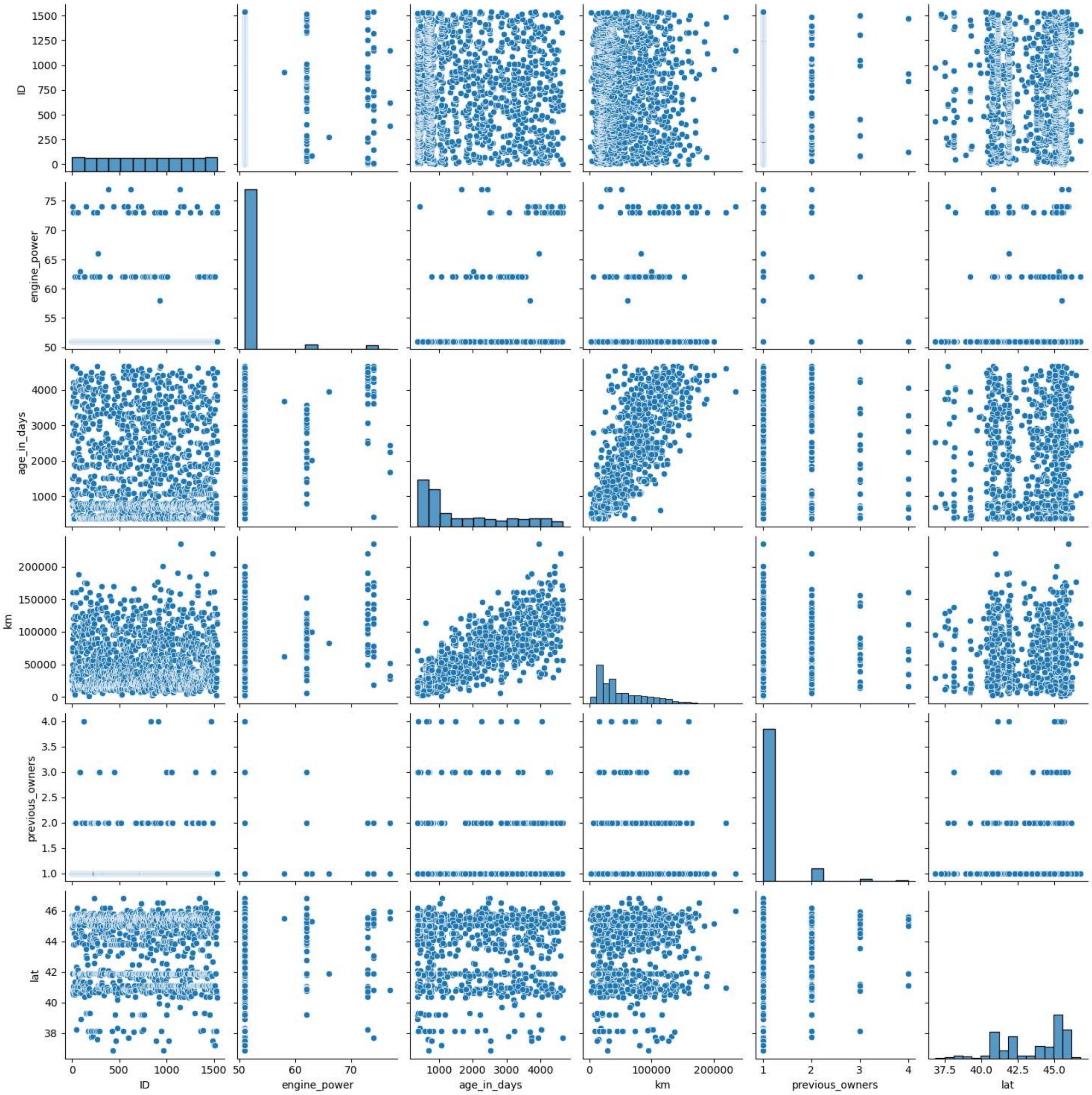
Out[8]:
```
ID                 0
model              0
engine_power       0
age_in_days        0
km                 0
previous_owners    0
lat                0
lon                0
price              0
dtype: int64
```

In [9]:
```
dff.columns
```

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

```
In [10]: sns.pairplot(dff)
```

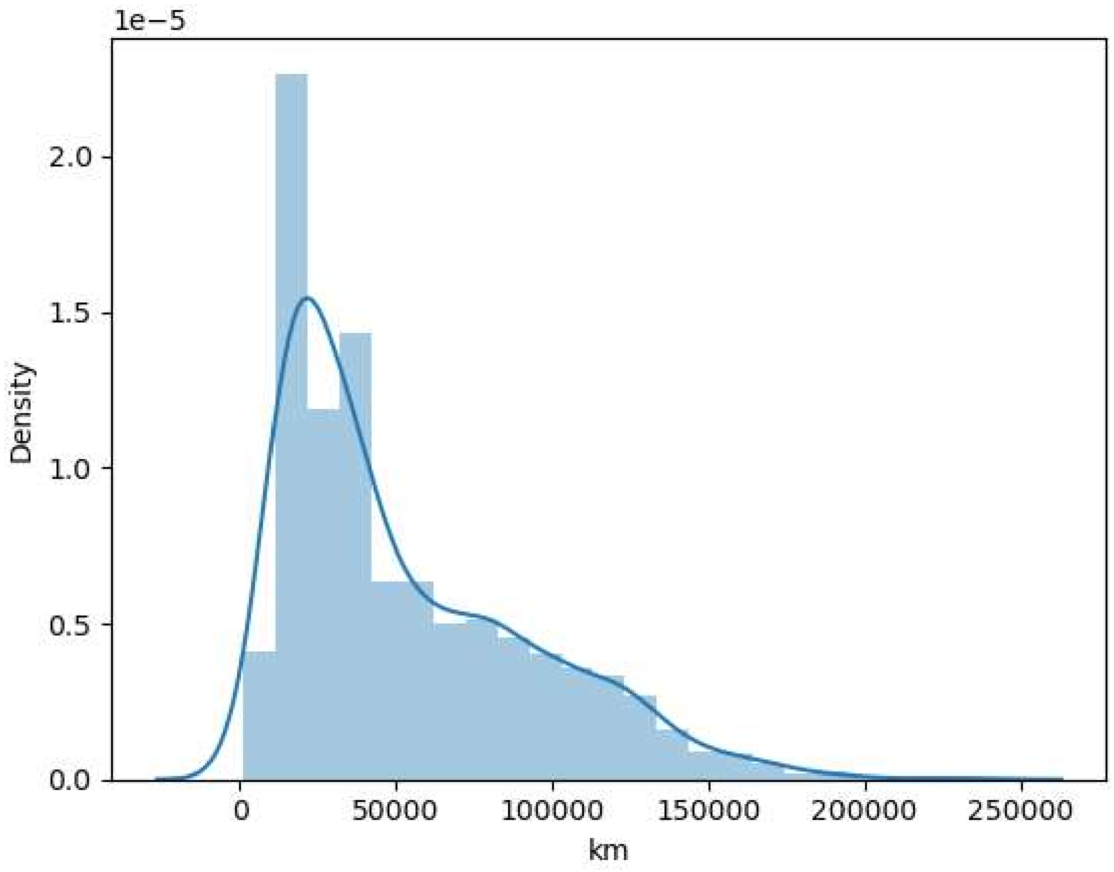
```
Out[10]: <seaborn.axisgrid.PairGrid at 0x1ad26e0eed0>
```



In [11]:
```
sns.distplot(dff["km"])
```

```
C:\Users\user\AppData\Local\Temp\ipykernel_8148\596799067.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (https://gis
t.github.com/mwaskom/de44147ed2974457ad6372750bbe5751)

  sns.distplot(dff["km"])
```

Out[11]:
```
<Axes: xlabel='km', ylabel='Density'>
```



In [12]:
```
df1=dff[['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owners'
       'lat', 'lon', 'price']]
```

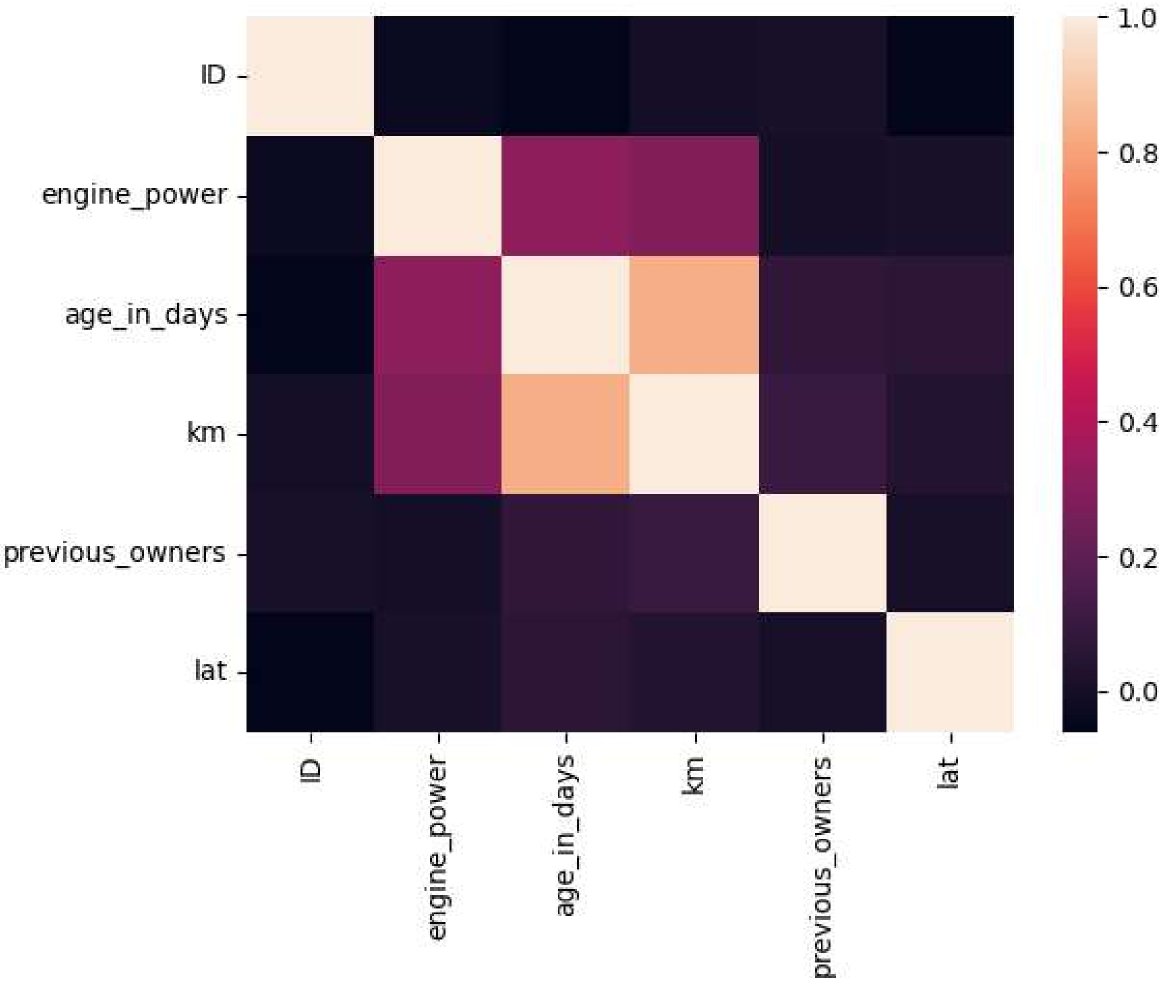
In [13]:

```
sns.heatmap(df1.corr())
```

```
C:\Users\user\AppData\Local\Temp\ipykernel_8148\781785195.py:1: FutureWarnin
g: The default value of numeric_only in DataFrame.corr is deprecated. In a fu
ture version, it will default to False. Select only valid columns or specify
the value of numeric_only to silence this warning.
  sns.heatmap(df1.corr())
```

Out[13]:

```
<Axes: >
```



In [14]:

```
x=df1[['ID', 'engine_power', 'age_in_days', 'previous_owners',
       'lat']]
y=df1['km']
```

In [15]:

```
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 [16]:

```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[16]:

```
▾ LinearRegression

LinearRegression()
```

In [17]:

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

```
-3742.018457785569
```

In [18]:

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

Out[18]:

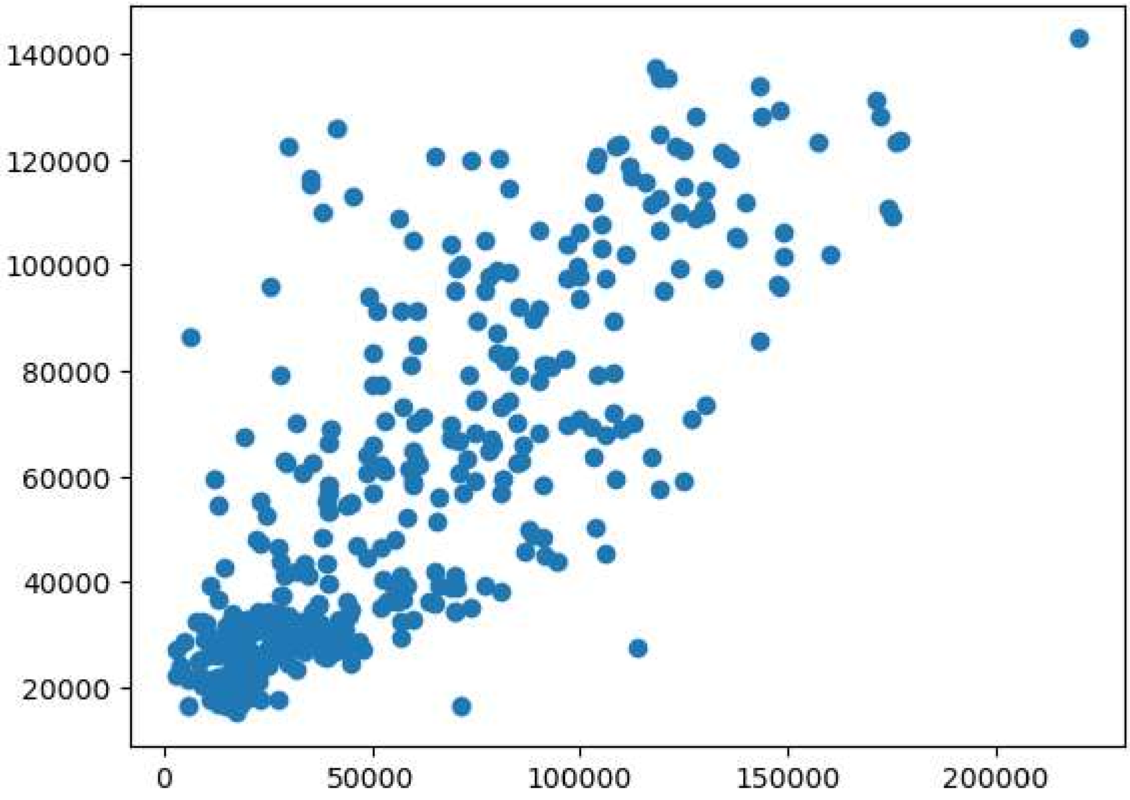
	Co-efficient
ID	4.733338
engine_power	310.433868
age_in_days	25.803759
previous_owners	3766.548261
lat	-224.134735

In [19]:

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

Out[19]:

```
<matplotlib.collections.PathCollection at 0x1ad2c5b4c10>
```



In [20]:

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

```
0.6518748273974636
```

```
In [21]: from sklearn.linear_model import Ridge,Lasso
```

```
In [22]: rr=Ridge(alpha=10)
         rr.fit(x_train,y_train)
```

```
Out[22]: ▾      Ridge
         Ridge(alpha=10)
```

```
In [23]: rr.score(x_test,y_test)
```

```
Out[23]: 0.6519347562577196
```

```
In [24]: la=Lasso(alpha=10)
         la.fit(x_train,y_train)
```

```
Out[24]: ▾      Lasso
         Lasso(alpha=10)
```

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

```
Out[25]: 0.6518956328194522
```

```
In [ ]:
```