In [1]:

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

In [2]:

```
df=pd.read_csv(r'C:\Users\user\Downloads\fiat500_VehicleSelection_Dataset (2).csv')
df
```

Out[2]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price	Unna
0	1.0	lounge	51.0	882.0	25000.0	1.0	44.907242	8.611559868	8900	
1	2.0	pop	51.0	1186.0	32500.0	1.0	45.666359	12.24188995	8800	
2	3.0	sport	74.0	4658.0	142228.0	1.0	45.503300	11.41784	4200	
3	4.0	lounge	51.0	2739.0	160000.0	1.0	40.633171	17.63460922	6000	
4	5.0	pop	73.0	3074.0	106880.0	1.0	41.903221	12.49565029	5700	
...	...	...	...	...	...	...	...	...	...	
1544	NaN	NaN	NaN	NaN	NaN	NaN	NaN	length	5	
1545	NaN	NaN	NaN	NaN	NaN	NaN	NaN	concat	lonprice	
1546	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Null values	NO	
1547	NaN	NaN	NaN	NaN	NaN	NaN	NaN	find	1	
1548	NaN	NaN	NaN	NaN	NaN	NaN	NaN	search	1	

1549 rows × 11 columns

In [3]:

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

Out[3]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price	Unnamed
0	1.0	lounge	51.0	882.0	25000.0	1.0	44.907242	8.611559868	8900	Na
1	2.0	pop	51.0	1186.0	32500.0	1.0	45.666359	12.24188995	8800	Na
2	3.0	sport	74.0	4658.0	142228.0	1.0	45.503300	11.41784	4200	Na
3	4.0	lounge	51.0	2739.0	160000.0	1.0	40.633171	17.63460922	6000	Na
4	5.0	pop	73.0	3074.0	106880.0	1.0	41.903221	12.49565029	5700	Na
5	6.0	pop	74.0	3623.0	70225.0	1.0	45.000702	7.68227005	7900	Na
6	7.0	lounge	51.0	731.0	11600.0	1.0	44.907242	8.611559868	10750	Na
7	8.0	lounge	51.0	1521.0	49076.0	1.0	41.903221	12.49565029	9190	Na
8	9.0	sport	73.0	4049.0	76000.0	1.0	45.548000	11.54946995	5600	Na
9	10.0	sport	51.0	3653.0	89000.0	1.0	45.438301	10.99170017	6000	Na

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	Unnamed: 9
count	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	0.0
mean	769.500000	51.904421	1650.980494	53396.011704	1.123537	43.541361	NaN
std	444.126671	3.988023	1289.522278	40046.830723	0.416423	2.133518	NaN
min	1.000000	51.000000	366.000000	1232.000000	1.000000	36.855839	NaN
25%	385.250000	51.000000	670.000000	20006.250000	1.000000	41.802990	NaN
50%	769.500000	51.000000	1035.000000	39031.000000	1.000000	44.394096	NaN
75%	1153.750000	51.000000	2616.000000	79667.750000	1.000000	45.467960	NaN
max	1538.000000	77.000000	4658.000000	235000.000000	4.000000	46.795612	NaN

In [6]:

```
df.columns
```

Out[6]:

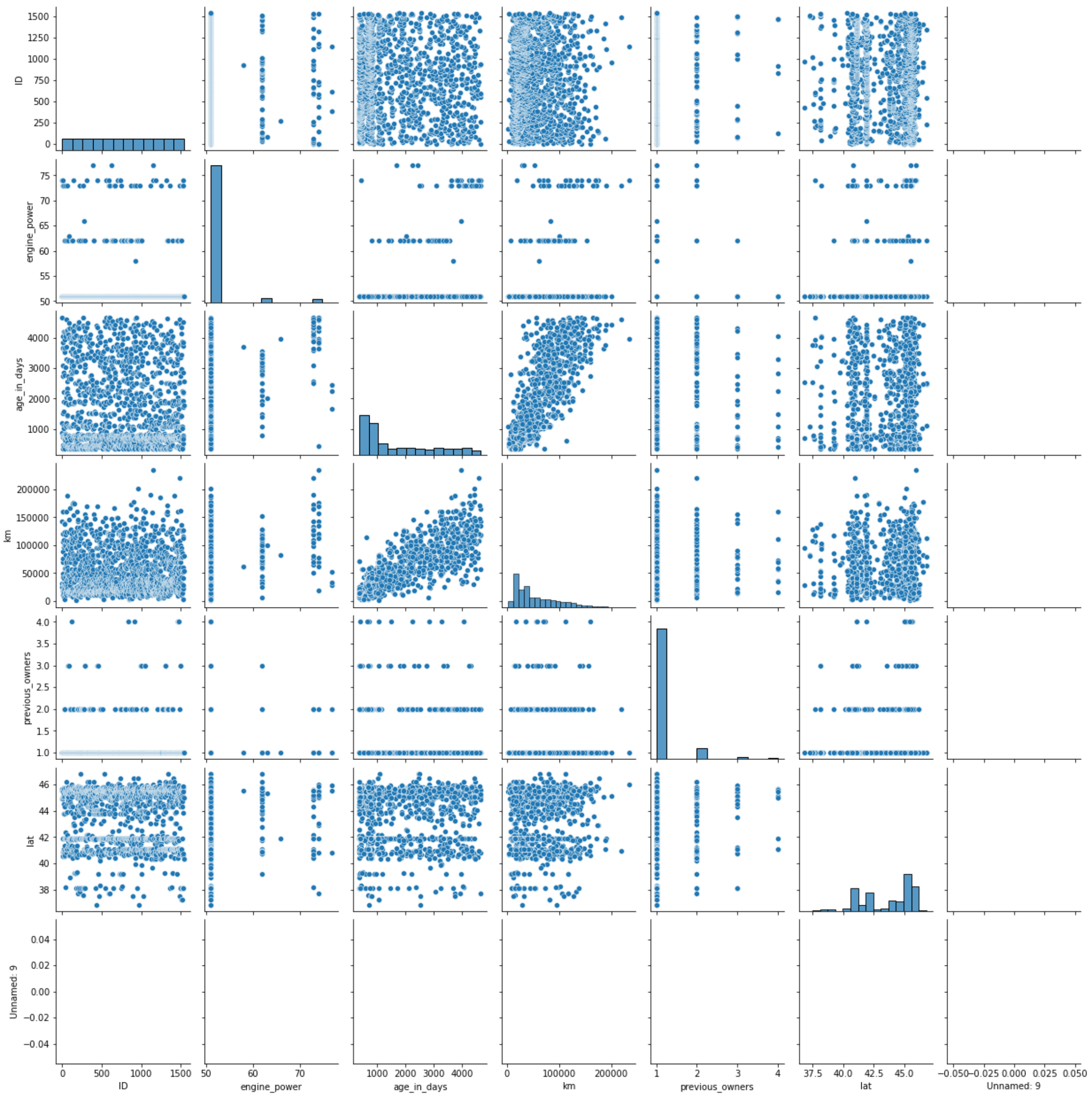
```
Index(['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owners',
       'lat', 'lon', 'price', 'Unnamed: 9', 'Unnamed: 10'],
      dtype='object')
```

In [7]:

```
sns.pairplot(df)
```

Out[7]:

```
<seaborn.axisgrid.PairGrid at 0x2b1837ceee0>
```



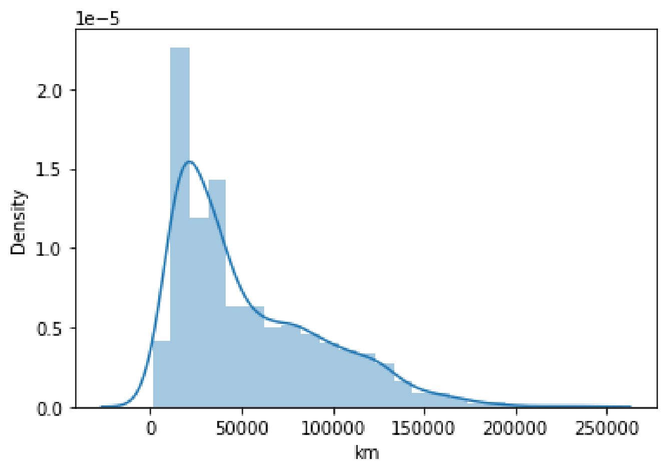
In [8]:

```
sns.distplot(df['km'])
```

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

Out[8]:

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

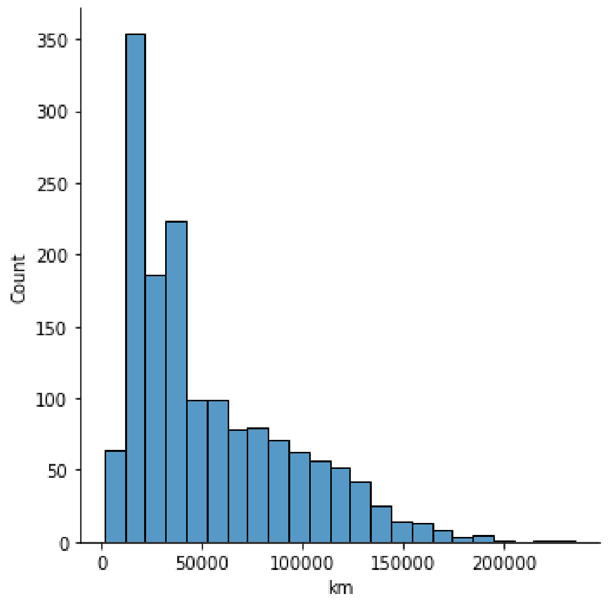


In [9]:

```
sns.displot(df["km"])
```

Out[9]:

```
<seaborn.axisgrid.FacetGrid at 0x2b18556bb20>
```



In [10]:

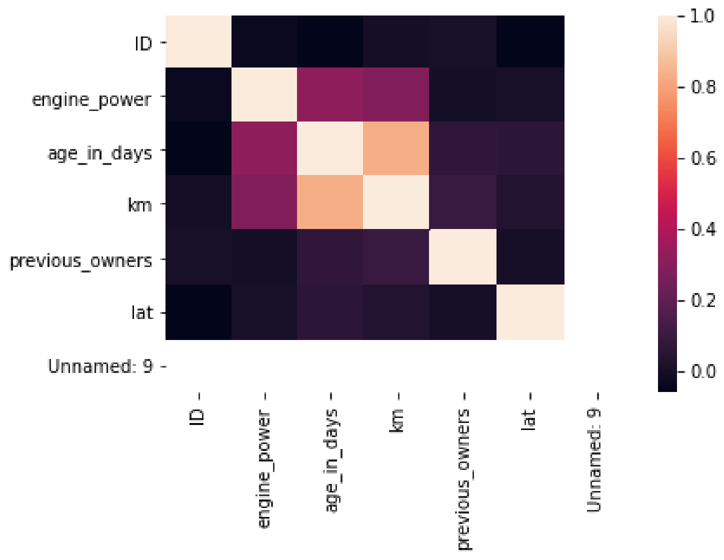
```
df1=df[['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owners',
       'lat', 'lon', 'price', 'Unnamed: 9', 'Unnamed: 10']]
```

In [11]:

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

Out[11]:

```
<AxesSubplot:>
```



In [12]:

```
x=df1[['ID', 'engine_power', 'age_in_days','previous_owners','lat', 'Unnamed: 9']]
y=df1[['km']]
```

In [13]:

```
from sklearn.model_selection import train_test_split
```

In [14]:

```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [20]:

```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)#ValueError: Input contains NaN, infinity or a value too large for dtype('float64',
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-20-58bc3c346312> in <module>
      2 
      3 lr=LinearRegression()
----> 4 lr.fit(x_train,y_train)#ValueError: Input contains NaN, infinity or a value too lar
ge for dtype('float64')

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_base.py in fit(self, X, y,
sample_weight)
    516         accept_sparse = False if self.positive else ['csr', 'csc', 'coo']
    517 
--> 518         X, y = self._validate_data(X, y, accept_sparse=accept_sparse,
    519                                    y_numeric=True, multi_output=True)
    520 

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py in _validate_data(self, X, y, re
set, validate_separately, **check_params)
    431                 y = check_array(y, **check_y_params)
    432             else:
--> 433                 X, y = check_X_y(X, y, **check_params)
    434             out = X, y
    435 

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args, **
kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64 
     65             # extra_args > 0

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in check_X_y(X, y, a
ccept_sparse, accept_large_sparse, dtype, order, copy, force_all_finite, ensure_2d, allow_n
d, multi_output, ensure_min_samples, ensure_min_features, y_numeric, estimator)
    812         raise ValueError("y cannot be None")
    813 
--> 814     X = check_array(X, accept_sparse=accept_sparse,
    815                     accept_large_sparse=accept_large_sparse,
    816                     dtype=dtype, order=order, copy=copy,

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args, **
kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64 
     65             # extra_args > 0

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in check_array(arra
y, accept_sparse, accept_large_sparse, dtype, order, copy, force_all_finite, ensure_2d, all
ow_nd, ensure_min_samples, ensure_min_features, estimator)
    661 
    662         if force_all_finite:
--> 663             _assert_all_finite(array,
    664                                allow_nan=force_all_finite == 'allow-nan')
    665 

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in _assert_all_finit
e(X, allow_nan, msg_dtype)
    101                 not allow_nan and not np.isfinite(X).all()):
    102             type_err = 'infinity' if allow_nan else 'NaN, infinity'
--> 103             raise ValueError(
    104                     msg_err.format
    105                     (type_err,

ValueError: Input contains NaN, infinity or a value too large for dtype('float64').
```

In [16]:

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

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-16-182bb45ab960> in <module>
----> 1 print(lr.intercept_)

AttributeError: 'LinearRegression' object has no attribute 'intercept_'
```

In [17]:

```
coef= pd.DataFrame(lr.coef_)
coef
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-17-0ff321f0a2a5> in <module>
----> 1 coef= pd.DataFrame(lr.coef_)
      2 coef

AttributeError: 'LinearRegression' object has no attribute 'coef_'
```

```
In [18]:
print(lr.score(x_test,y_test))
```

```
---------------------------------------------------------------------------
NotFittedError                            Traceback (most recent call last)
<ipython-input-18-6bc23016a4ce> in <module>
----> 1 print(lr.score(x_test,y_test))

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py in score(self, X, y, sample_weig
ht)
    551 
    552         from .metrics import r2_score
--> 553         y_pred = self.predict(X)
    554         return r2_score(y, y_pred, sample_weight=sample_weight)
    555 

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_base.py in predict(self,
X)
    236             Returns predicted values.
    237         """
--> 238         return self._decision_function(X)
    239 
    240     _preprocess_data = staticmethod(_preprocess_data)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_base.py in _decision_funct
ion(self, X)
    216 
    217     def _decision_function(self, X):
--> 218         check_is_fitted(self)
    219 
    220         X = check_array(X, accept_sparse=['csr', 'csc', 'coo'])

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args, **
kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64 
     65             # extra_args > 0

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in check_is_fitted(e
stimator, attributes, msg, all_or_any)
   1039 
   1040     if not attrs:
-> 1041         raise NotFittedError(msg % {'name': type(estimator).__name__})
   1042 
   1043 

NotFittedError: This LinearRegression instance is not fitted yet. Call 'fit' with appropria
te arguments before using this estimator.
```

In [19]:

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

```
---------------------------------------------------------------------------
NotFittedError                            Traceback (most recent call last)
<ipython-input-19-10d398fd7dc3> in <module>
----> 1 prediction = lr.predict(x_test)
      2 plt.scatter(y_test,prediction)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_base.py in predict(self,
X)
    236             Returns predicted values.
    237         """
--> 238         return self._decision_function(X)
    239 
    240     _preprocess_data = staticmethod(_preprocess_data)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_base.py in _decision_funct
ion(self, X)
    216 
    217     def _decision_function(self, X):
--> 218         check_is_fitted(self)
    219 
    220         X = check_array(X, accept_sparse=['csr', 'csc', 'coo'])

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args, **
kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64 
     65             # extra_args > 0

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in check_is_fitted(e
stimator, attributes, msg, all_or_any)
   1039 
   1040     if not attrs:
-> 1041         raise NotFittedError(msg % {'name': type(estimator).__name__})
   1042 
   1043 

NotFittedError: This LinearRegression instance is not fitted yet. Call 'fit' with appropria
te arguments before using this estimator.
```