In [70]:

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

In [71]:

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

Out[71]:

| | ID | model | engine_power | age_in_days | km | previous_owners | lat | l |
|---|---|---|---|---|---|---|---|---|
| **0** | 1 | lounge | 51 | 882 | 25000 | 1 | 44.907242 | 8.6115 |
| **1** | 2 | pop | 51 | 1186 | 32500 | 1 | 45.666359 | 12.2418 |
| **2** | 3 | sport | 74 | 4658 | 142228 | 1 | 45.503300 | 11.4178 |
| **3** | 4 | lounge | 51 | 2739 | 160000 | 1 | 40.633171 | 17.6346 |
| **4** | 5 | pop | 73 | 3074 | 106880 | 1 | 41.903221 | 12.4956 |
| **...** | ... | ... | ... | ... | ... | ... | ... | |
| **1533** | 1534 | sport | 51 | 3712 | 115280 | 1 | 45.069679 | 7.7049 |
| **1534** | 1535 | lounge | 74 | 3835 | 112000 | 1 | 45.845692 | 8.6668 |
| **1535** | 1536 | pop | 51 | 2223 | 60457 | 1 | 45.481541 | 9.4134 |
| **1536** | 1537 | lounge | 51 | 2557 | 80750 | 1 | 45.000702 | 7.6822 |
| **1537** | 1538 | pop | 51 | 1766 | 54276 | 1 | 40.323410 | 17.5682 |

1538 rows × 9 columns

In [72]:

```
a=a.head(10)
a
```

Out[72]:

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

In [73]:

```
# to find
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   ID               10 non-null     int64  
 1   model            10 non-null     object 
 2   engine_power     10 non-null     int64  
 3   age_in_days      10 non-null     int64  
 4   km               10 non-null     int64  
 5   previous_owners  10 non-null     int64  
 6   lat              10 non-null     float64
 7   lon              10 non-null     float64
 8   price            10 non-null     int64  
dtypes: float64(2), int64(6), object(1)
memory usage: 848.0+ bytes
```

In [74]:

```
# to display summary of statastic
a.describe()
```

Out[74]:

| | ID | engine_power | age_in_days | km | previous_owners | lat | |
|---|---|---|---|---|---|---|---|
| **count** | 10.00000 | 10.000000 | 10.000000 | 10.000000 | 10.0 | 10.000000 | 10.( |
| **mean** | 5.50000 | 60.000000 | 2611.600000 | 76250.900000 | 1.0 | 44.141076 | 11.: |
| **std** | 3.02765 | 11.623731 | 1427.557214 | 49399.679798 | 0.0 | 1.887936 | 2.8 |
| **min** | 1.00000 | 51.000000 | 731.000000 | 11600.000000 | 1.0 | 40.633171 | 7.( |
| **25%** | 3.25000 | 51.000000 | 1269.750000 | 36644.000000 | 1.0 | 42.654226 | 9.: |
| **50%** | 5.50000 | 51.000000 | 2906.500000 | 73112.500000 | 1.0 | 44.953972 | 11.4 |
| **75%** | 7.75000 | 73.000000 | 3645.500000 | 102410.000000 | 1.0 | 45.487050 | 12.4 |
| **max** | 10.00000 | 74.000000 | 4658.000000 | 160000.000000 | 1.0 | 45.666359 | 17.( |

In [75]:

```
# to display colum heading
a.columns
```

Out[75]:

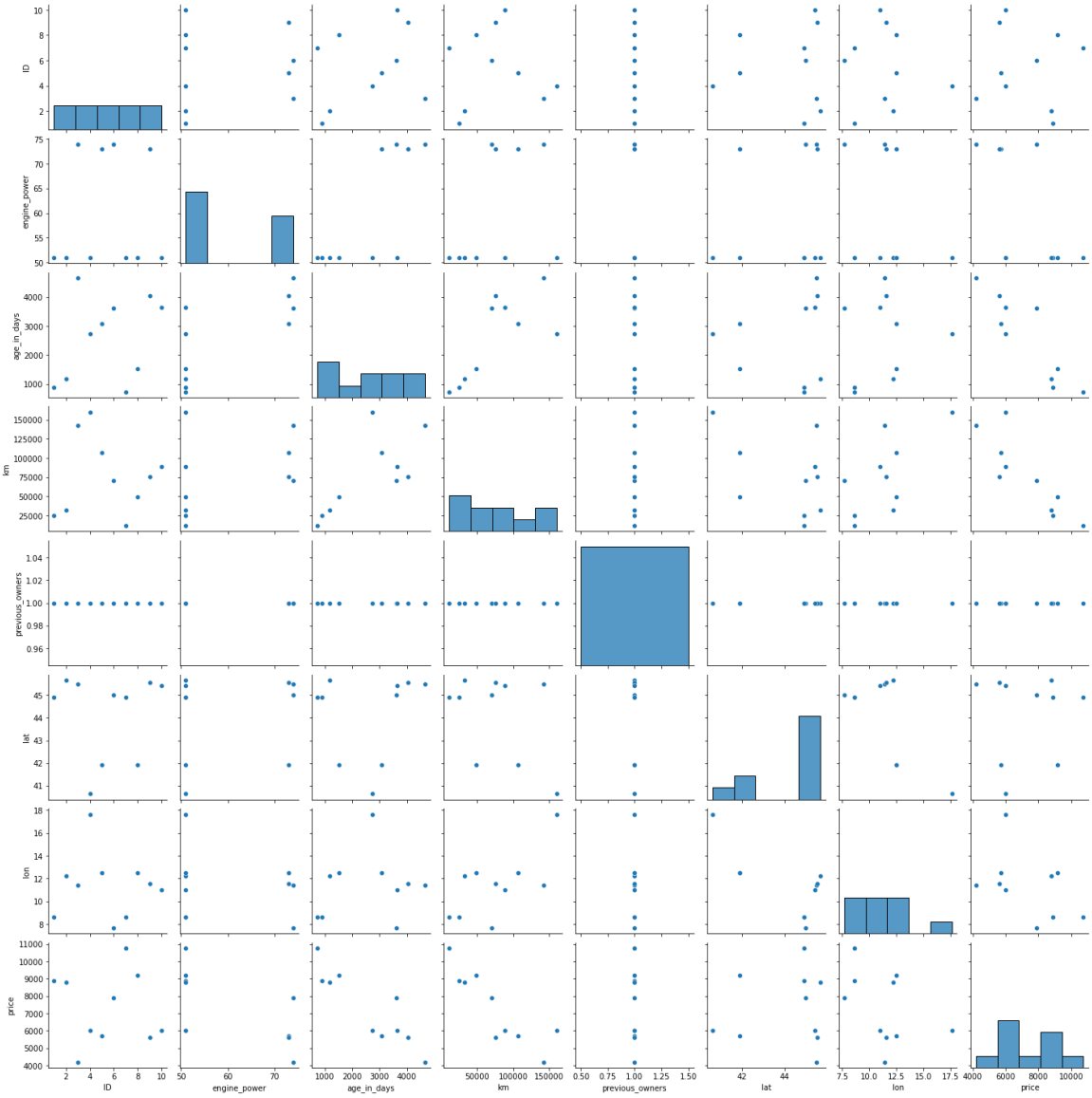
```
Index(['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owner
s',
       'lat', 'lon', 'price'],
      dtype='object')
```

In [76]:

```
sns.pairplot(a)
```

Out[76]:

```
<seaborn.axisgrid.PairGrid at 0x243ea33ad30>
```

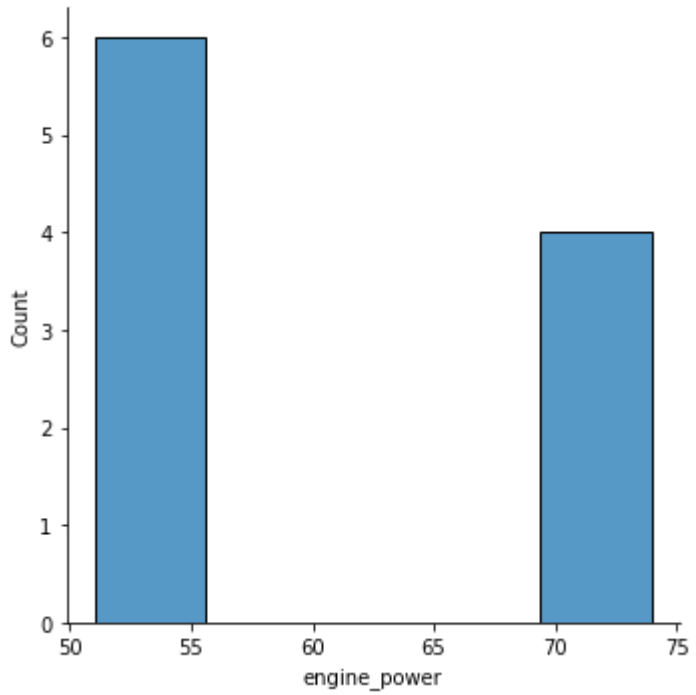


In [77]:

```
sns.displot(a["engine_power"])
```

Out[77]:

```
<seaborn.axisgrid.FacetGrid at 0x243ed33b2e0>
```




In [78]:

```
b=a[['engine_power', 'age_in_days', 'km', 'previous_owners']]
b
```

Out[78]:

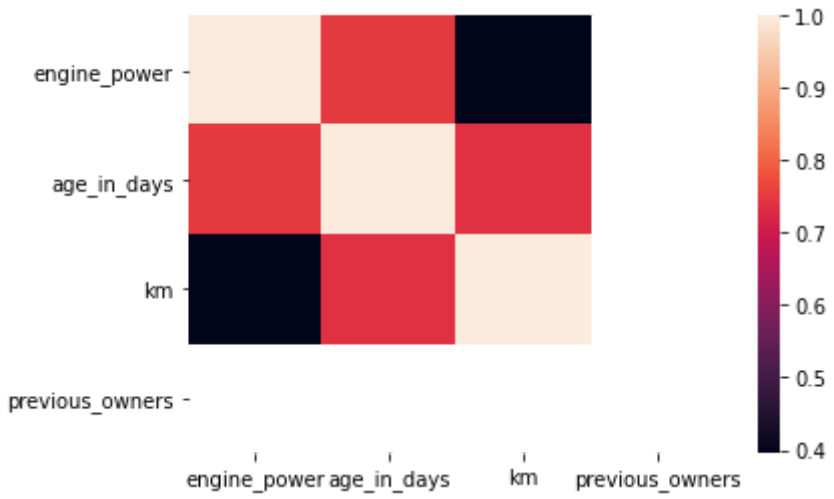
| | engine_power | age_in_days | km | previous_owners |
|---|---|---|---|---|
| **0** | 51 | 882 | 25000 | 1 |
| **1** | 51 | 1186 | 32500 | 1 |
| **2** | 74 | 4658 | 142228 | 1 |
| **3** | 51 | 2739 | 160000 | 1 |
| **4** | 73 | 3074 | 106880 | 1 |
| **5** | 74 | 3623 | 70225 | 1 |
| **6** | 51 | 731 | 11600 | 1 |
| **7** | 51 | 1521 | 49076 | 1 |
| **8** | 73 | 4049 | 76000 | 1 |
| **9** | 51 | 3653 | 89000 | 1 |

In [79]:

```
sns.heatmap(b.corr())
```

Out[79]:

<AxesSubplot:>




In [81]:

```
x=a[['engine_power', 'age_in_days', 'km', 'previous_owners']]
y=a['age_in_days']
```

In [82]:

```
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 [83]:

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

Out[83]:

```
LinearRegression()
```

In [84]:

```
lr.intercept_
```

Out[84]:

```
-5.9117155615240335e-12
```

In [85]:

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

Out[85]:

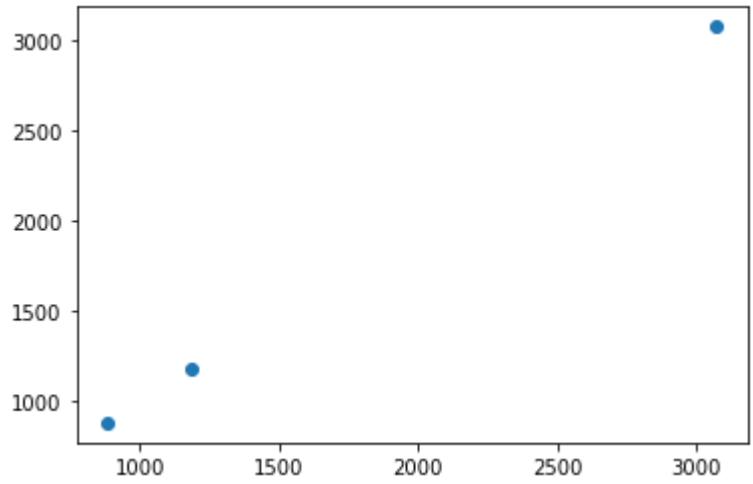
| | Co-efficient |
|---|---|
| **engine_power** | 9.657998e-14 |
| **age_in_days** | 1.000000e+00 |
| **km** | 8.712301e-17 |
| **previous_owners** | 0.000000e+00 |

In [86]:

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

Out[86]:

```
<matplotlib.collections.PathCollection at 0x243edd334f0>
```




In [87]:

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

Out[87]:

```
1.0
```

In [88]:

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

Out[88]:

```
1.0
```

In [89]:

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

In [90]:

```
rr=Ridge(alpha=10)
rr.fit(x_test,y_test)
```

Out[90]:

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

In [91]:

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

Out[91]:

```
0.9999999939239826
```

In [92]:

```
la=Lasso(alpha=10)
la.fit(x_test,y_test)
```

Out[92]:

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

In [93]:

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

Out[93]:

```
0.9999834224889977
```

In [94]:

```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[94]:

```
ElasticNet()
```

In [95]:

```
en.coef_
```

Out[95]:

```
array([0.00000000e+00, 9.99992754e-01, 1.26973653e-07, 0.00000000e+00])
```

In [96]:

```
en.intercept_
```

Out[96]:

```
0.01086161492185056
```

In [97]:

```
prediction=en.predict(x_test)
prediction
```

Out[97]:

```
array([ 882.00764497, 1186.00639448, 3074.00215831])
```

In [98]:

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

Out[98]:

```
0.9999999999631309
```

# EVALUATION METRICS

In [99]:

```
from sklearn import metrics
```

In [100]:

```
print("Mean Absolute Error:",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute Error: 0.005399254093238899
```

In [101]:

```
print("Mean Squared Error",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Squared Error 3.466443186858874e-05
```

In [102]:

```
print("Root Mean Squared Error",np.sqrt(metrics.mean_squared_error(y_test,prediction)))
```

```
Root Mean Squared Error 0.00588765079370276
```

In [ ]:

```

```