In [2]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
df=pd.read_csv(r"C:\Users\DHEEPAK\Desktop\fiat500_VehicleSelection_Dataset.csv")
print(df)
```

```
        ID   model  engine_power  age_in_days      km  previous_owners
0        1  lounge            51          882   25000                1  \
1        2     pop            51         1186   32500                1
2        3   sport            74         4658  142228                1
3        4  lounge            51         2739  160000                1
4        5     pop            73         3074  106880                1
...    ...     ...           ...          ...     ...              ...
1533  1534   sport            51         3712  115280                1
1534  1535  lounge            74         3835  112000                1
1535  1536     pop            51         2223   60457                1
1536  1537  lounge            51         2557   80750                1
1537  1538     pop            51         1766   54276                1

            lat        lon  price
0     44.907242   8.611560   8900
1     45.666359  12.241890   8800
2     45.503300  11.417840   4200
3     40.633171  17.634609   6000
4     41.903221  12.495650   5700
...         ...        ...    ...
1533  45.069679   7.704920   5200
1534  45.845692   8.666870   4600
1535  45.481541   9.413480   7500
1536  45.000702   7.682270   5990
1537  40.323410  17.568270   7900

[1538 rows x 9 columns]
```

In [3]:

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

Out[3]:

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

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1538 entries, 0 to 1537
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   ID               1538 non-null   int64  
 1   model            1538 non-null   object 
 2   engine_power     1538 non-null   int64  
 3   age_in_days      1538 non-null   int64  
 4   km               1538 non-null   int64  
 5   previous_owners  1538 non-null   int64  
 6   lat              1538 non-null   float64
 7   lon              1538 non-null   float64
 8   price            1538 non-null   int64  
dtypes: float64(2), int64(6), object(1)
memory usage: 108.3+ KB
```

In [5]:

```
df.describe()
```

Out[5]:

| | ID | engine_power | age_in_days | km | previous_owners | l |
|---|---|---|---|---|---|---|
| **count** | 1538.000000 | 1538.000000 | 1538.000000 | 1538.000000 | 1538.000000 | 1538.00000 |
| **mean** | 769.500000 | 51.904421 | 1650.980494 | 53396.011704 | 1.123537 | 43.54136 |
| **std** | 444.126671 | 3.988023 | 1289.522278 | 40046.830723 | 0.416423 | 2.13351 |
| **min** | 1.000000 | 51.000000 | 366.000000 | 1232.000000 | 1.000000 | 36.85583 |
| **25%** | 385.250000 | 51.000000 | 670.000000 | 20006.250000 | 1.000000 | 41.80299 |
| **50%** | 769.500000 | 51.000000 | 1035.000000 | 39031.000000 | 1.000000 | 44.39409 |
| **75%** | 1153.750000 | 51.000000 | 2616.000000 | 79667.750000 | 1.000000 | 45.46796 |
| **max** | 1538.000000 | 77.000000 | 4658.000000 | 235000.000000 | 4.000000 | 46.79561 |

In [6]:

```
df.columns
```

Out[6]:

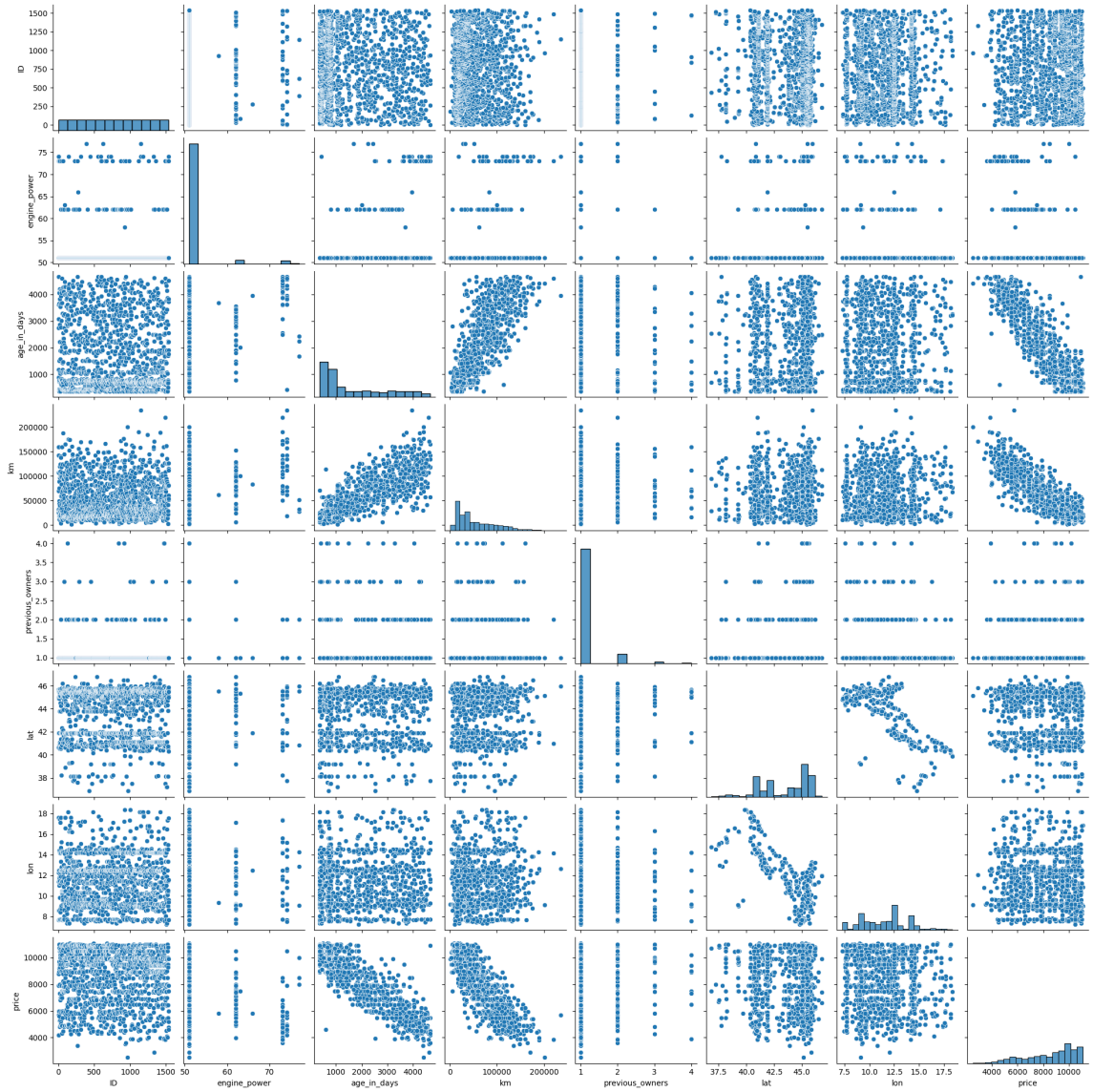
```
Index(['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owner
s',
       'lat', 'lon', 'price'],
      dtype='object')
```

In [7]:

```
#EDA
sns.pairplot(df)
```

Out[7]:

```
<seaborn.axisgrid.PairGrid at 0x121a4346650>
```

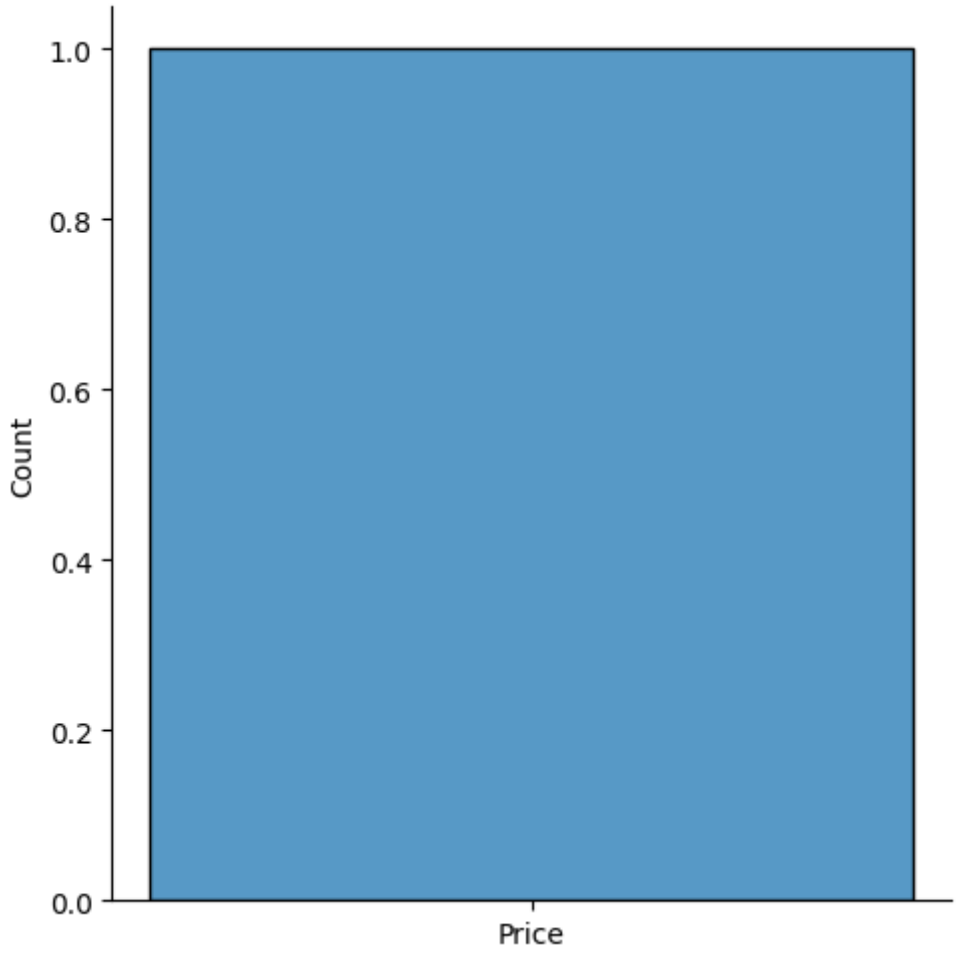


In [8]:

```
sns.displot(['Price'])
```

Out[8]:

```
<seaborn.axisgrid.FacetGrid at 0x121aa0e35d0>
```

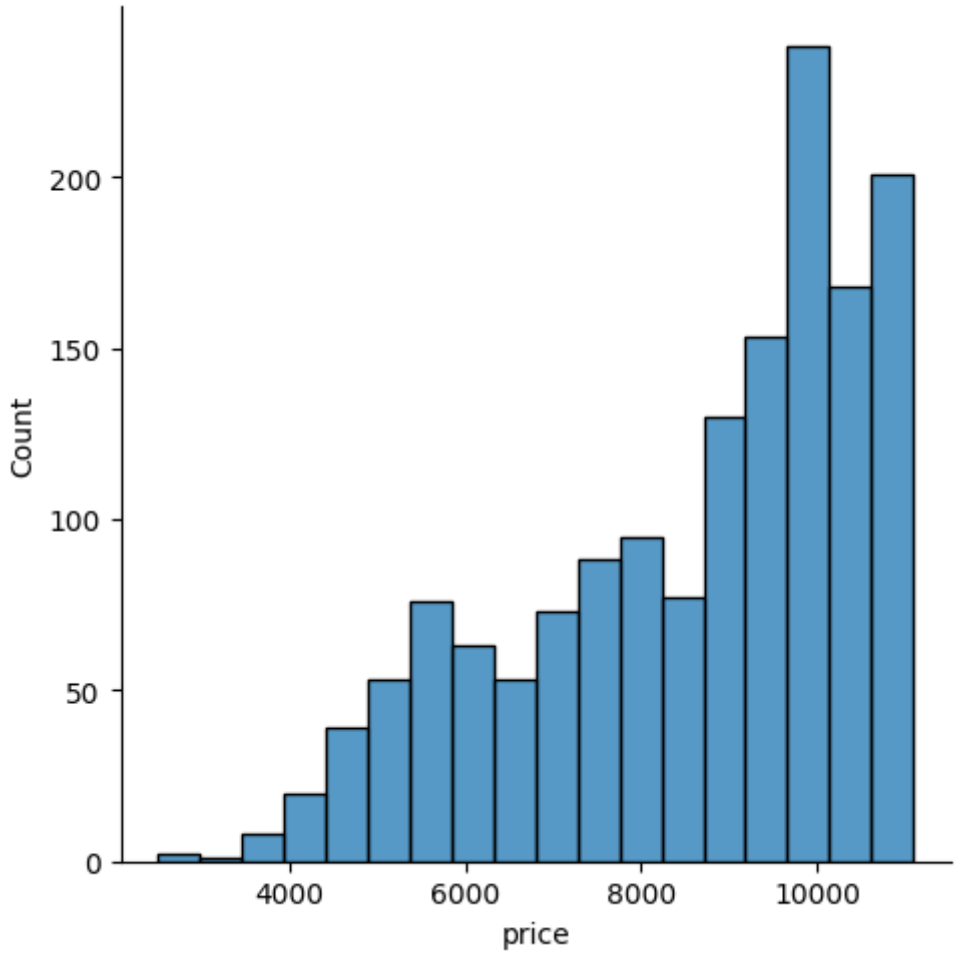


In [9]:

```
sns.displot(df['price'])
```

Out[9]:

```
<seaborn.axisgrid.FacetGrid at 0x121a9cf0890>
```

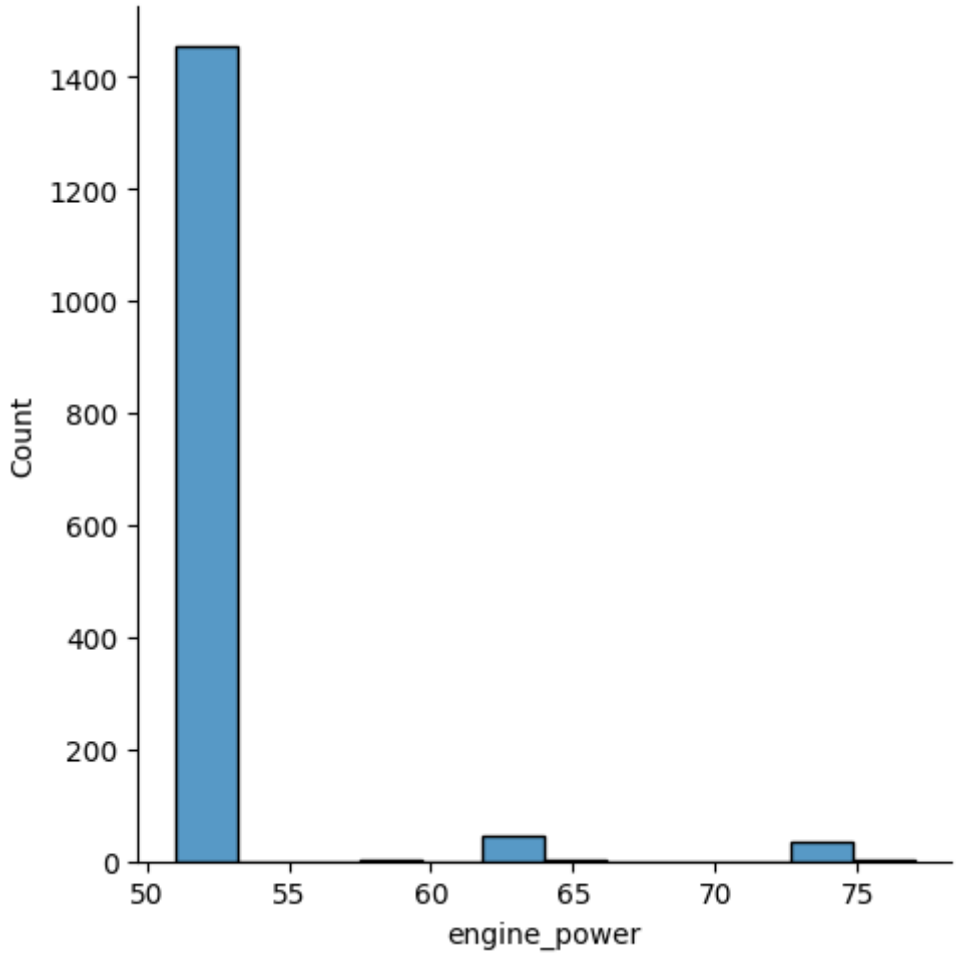


In [10]:

```
sns.displot(df['engine_power'])
```

Out[10]:

```
<seaborn.axisgrid.FacetGrid at 0x121a9df5ad0>
```

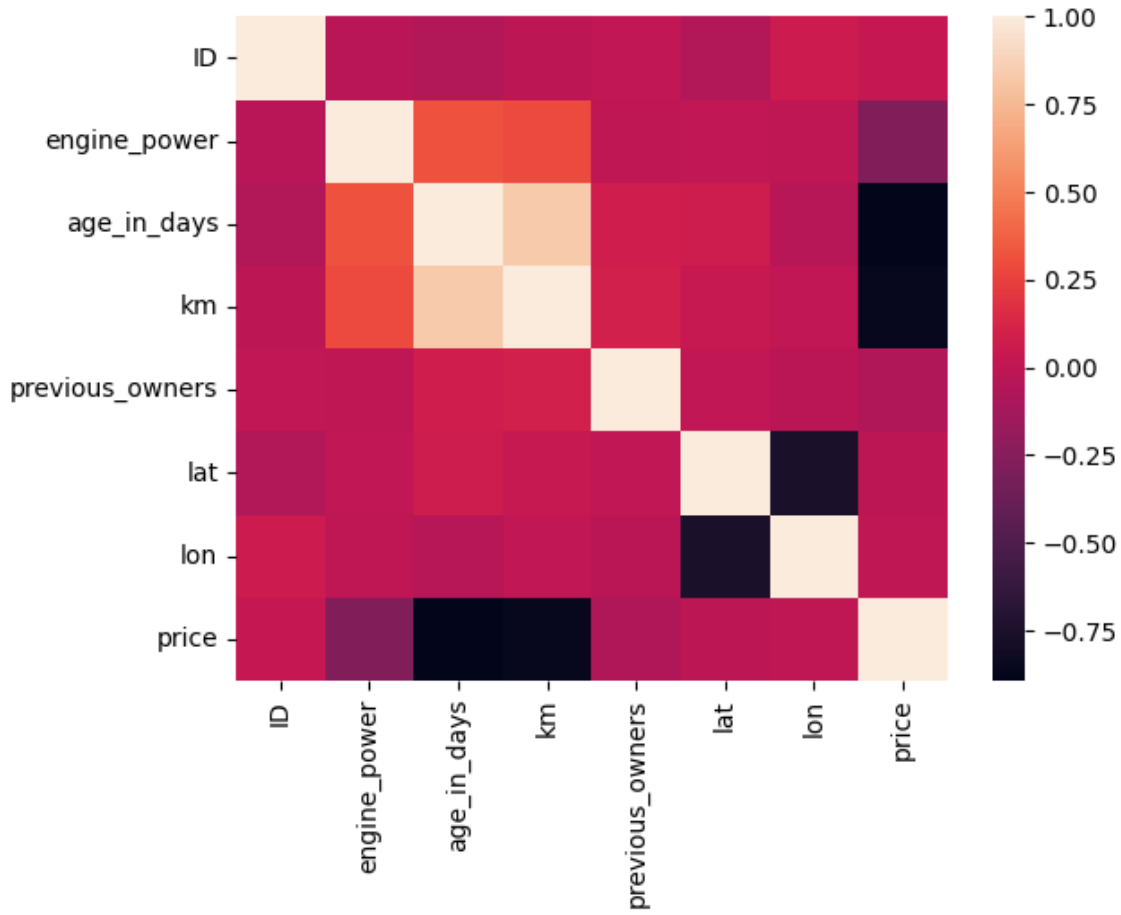


In [11]:

```
fiatdf=df[['ID', 'engine_power', 'age_in_days', 'km', 'previous_owners',
 'lat', 'lon','price']]
sns.heatmap(fiatdf.corr())
```

Out[11]:

<Axes: >

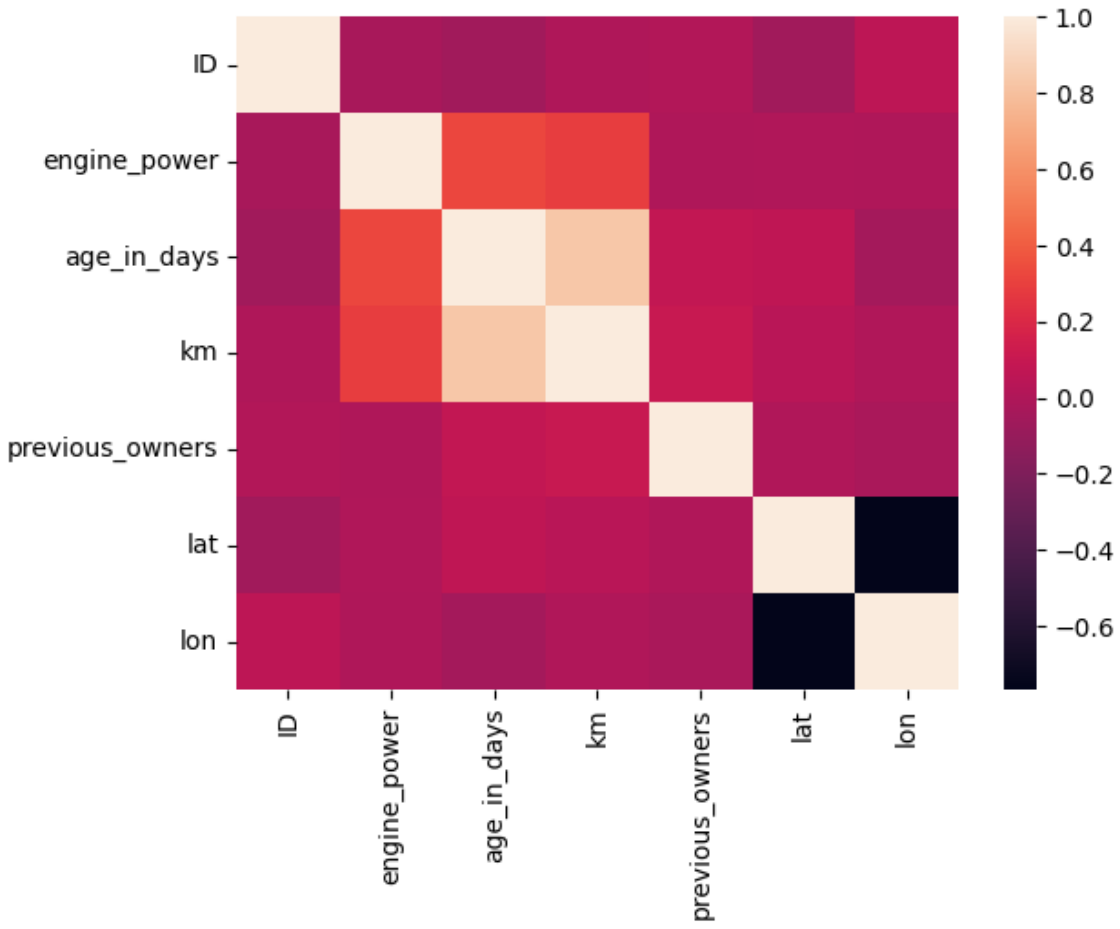


In [12]:

```
fiatdf=df[['ID', 'engine_power', 'age_in_days', 'km', 'previous_owners',
 'lat', 'lon']]
sns.heatmap(fiatdf.corr())#without price
```

Out[12]:

<Axes: >




In [13]:

```
X=fiatdf[['ID', 'engine_power', 'age_in_days', 'km', 'previous_owners',
 'lat', 'lon']]
y=df['price']
```

In [14]:

```
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,random_state=101)
from sklearn.linear_model import LinearRegression
regr=LinearRegression()
regr.fit(X_train,y_train)
print(regr.intercept_)
```

```
8971.19568349988
```

In [15]:

```
coeff_df=pd.DataFrame(regr.coef_,X.columns,columns=['coefficient'])
coeff_df
```

Out[15]:

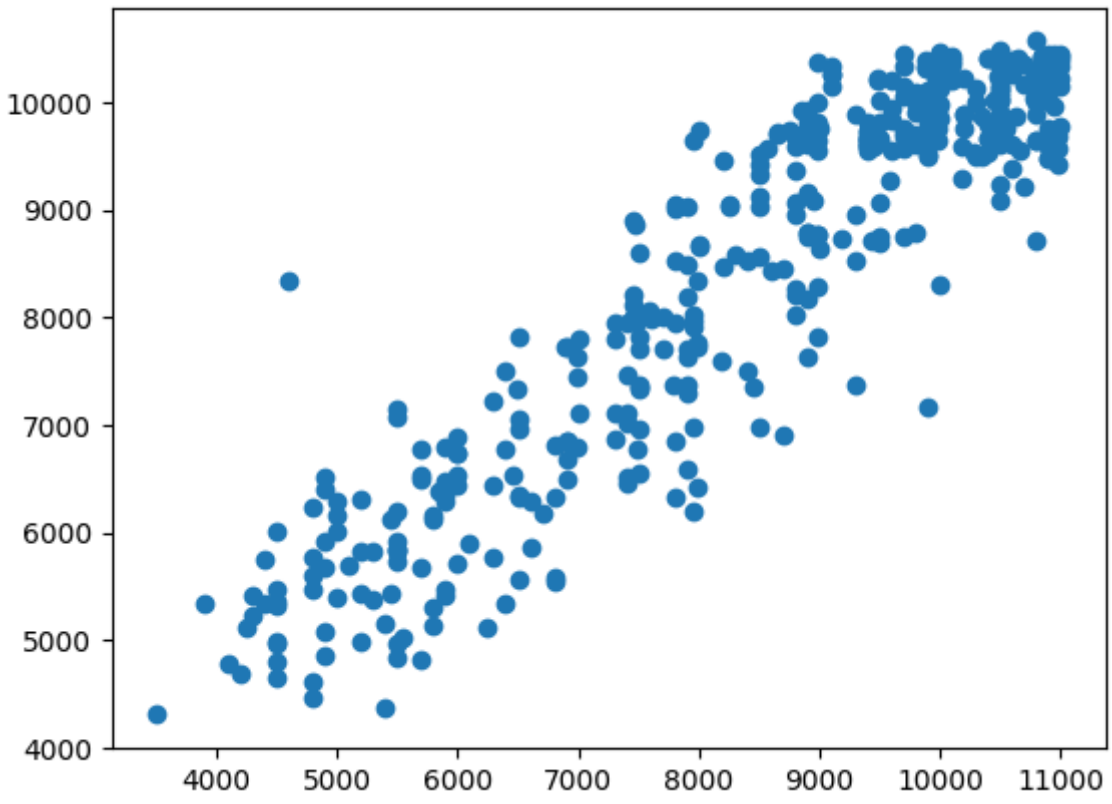
| | coefficient |
|---|---|
| **ID** | -0.046704 |
| **engine_power** | 11.646408 |
| **age_in_days** | -0.898018 |
| **km** | -0.017232 |
| **previous_owners** | 26.400886 |
| **lat** | 32.189709 |
| **lon** | 0.161073 |

In [16]:

```
predictions=regr.predict(X_test)
plt.scatter(y_test,predictions)
```

Out[16]:

```
<matplotlib.collections.PathCollection at 0x121ac3a9310>
```

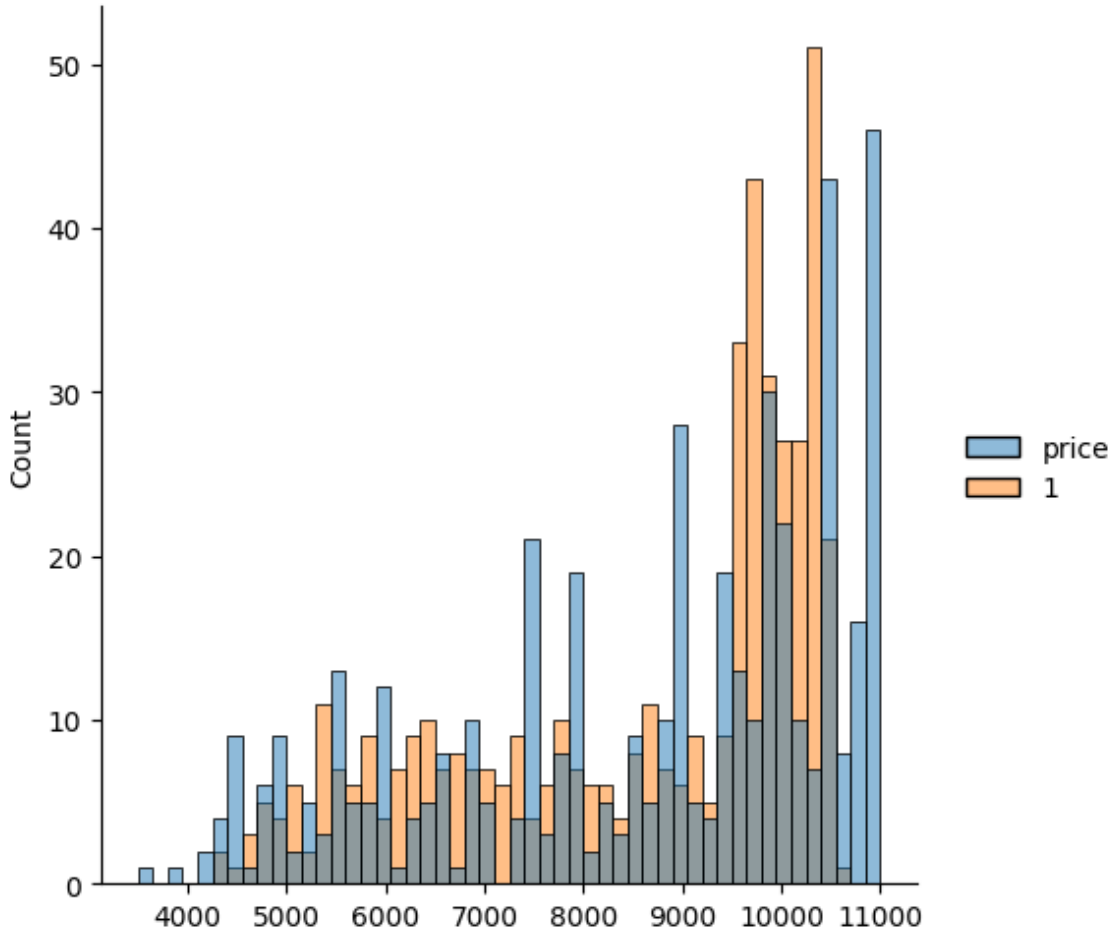


In [17]:

```
sns.displot((y_test,predictions),bins=50)#without semicolon
```

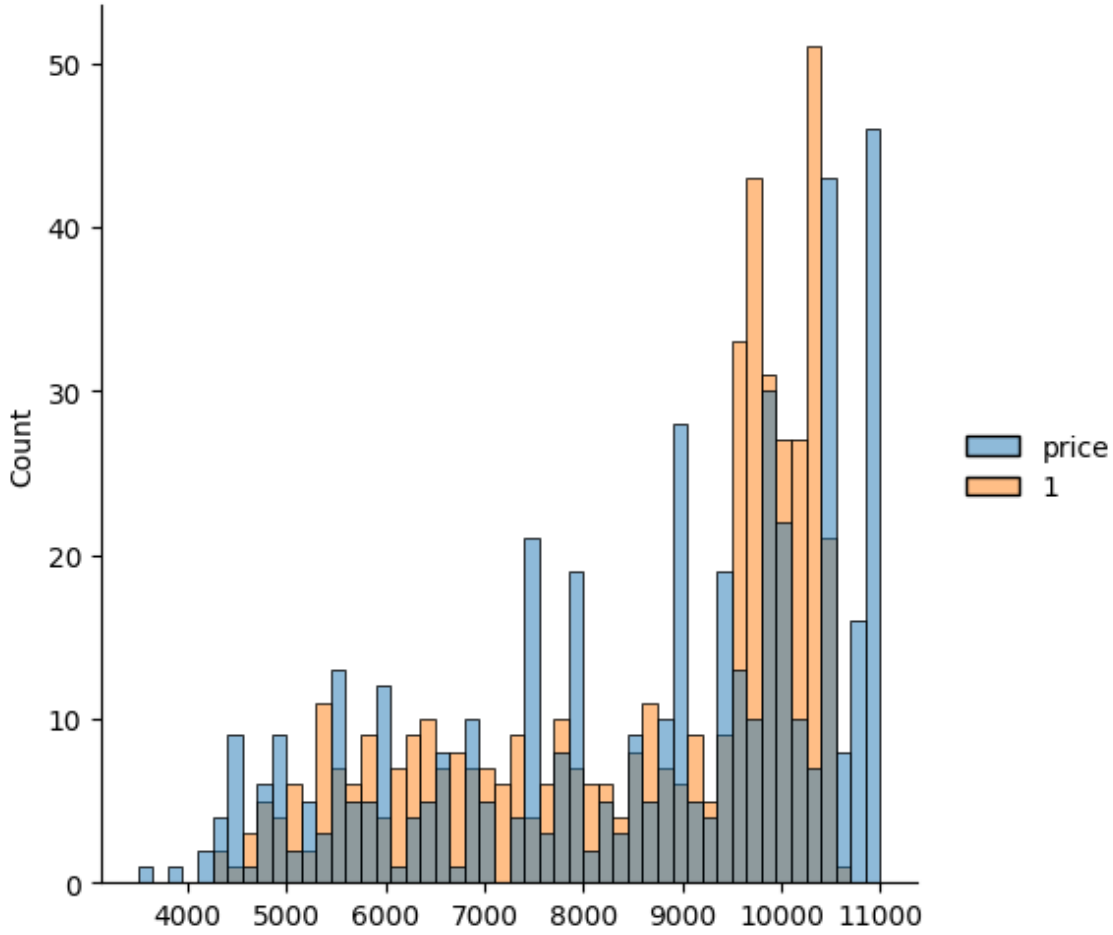
Out[17]:

```
<seaborn.axisgrid.FacetGrid at 0x121aa130790>
```



In [18]:

```
sns.displot((y_test,predictions),bins=50);#with semicolon
```




In [19]:

```
from sklearn import metrics
print('MAE:',metrics.mean_absolute_error(y_test,predictions))
print('MSE:',metrics.mean_squared_error(y_test,predictions))
print('MAE:',np.sqrt(metrics.mean_squared_error(y_test,predictions)))
```

```
MAE: 593.0876179519931
MSE: 551442.6799691801
MAE: 742.5918663500026
```

In [20]:

```
#accuracy
regr=LinearRegression()
regr.fit(X_train,y_train)
regr.fit(X_train,y_train)
print(regr.score(X_test,y_test))
```

```
0.8597136704308868
```

In [21]:

```
df.fillna(method='ffill',inplace=True)
```

In [22]:

```
x=np.array(df['age_in_days']).reshape(-1,1)
y=np.array(df['km']).reshape(-1,1)
df.dropna(inplace=True)
```

In [23]:

```
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
regr.fit(X_train,y_train)
regr.fit(X_train,y_train)
```

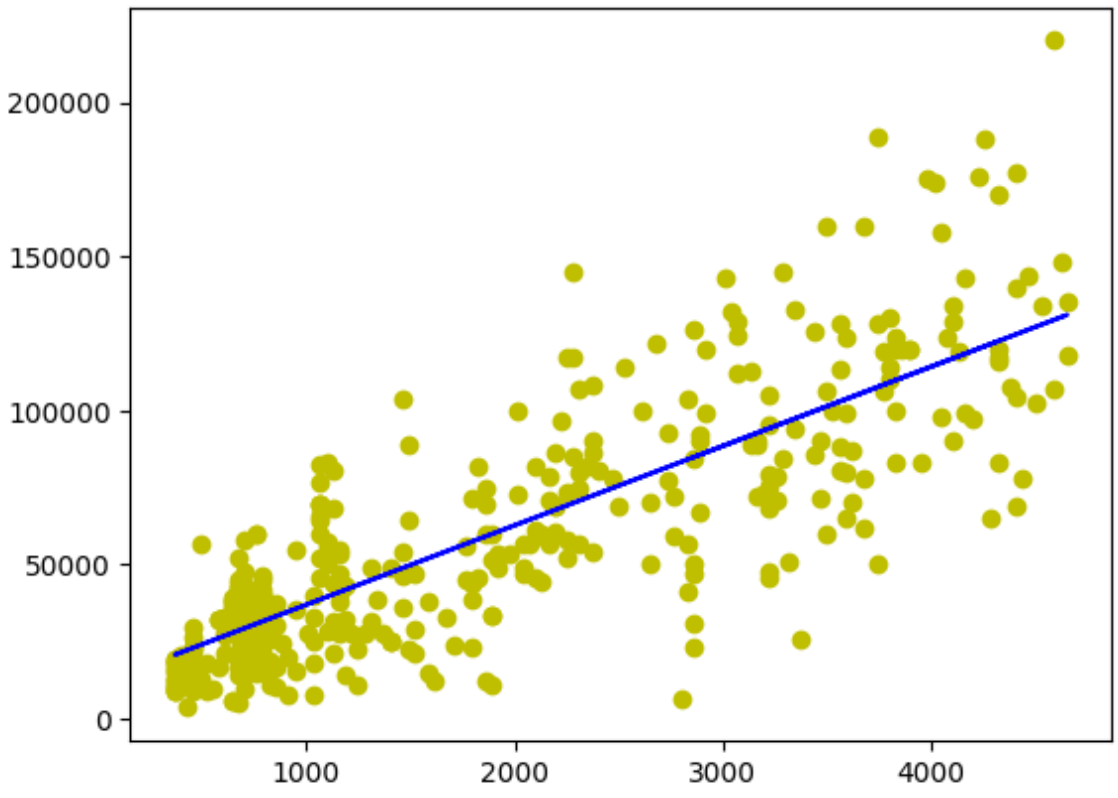
Out[23]:

LinearRegression

LinearRegression()

In [24]:

```
y_pred=regr.predict(X_test)
plt.scatter(X_test,y_test,color='y')
plt.plot(X_test,y_pred,color='b')
plt.show()
```



In [25]:

```
#elasticnet
from sklearn.linear_model import ElasticNet
regr=ElasticNet()
regr.fit(x,y)
print(regr.coef_)
print(regr.intercept_)
y_pred_elastic=regr.predict(X_train)
mean_squared_error=np.mean((y_pred_elastic-y_train)**2)
print("Mean Squared Error on test set",mean_squared_error)
```

```
[25.89689696]
[10640.73996329]
Mean Squared Error on test set 2725683166.086014
```

In [ ]:

```

```

In [ ]:

```

```