In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing, svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [2]:
```
df=pd.read_csv(r"C:\Users\pavan\Downloads\fiat500_VehicleSelection_Dataset.csv")
df
```

Out[2]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price
0	1	lounge	51	882	25000	1	44.907242	8.611560	8900
1	2	pop	51	1186	32500	1	45.666359	12.241890	8800
2	3	sport	74	4658	142228	1	45.503300	11.417840	4200
3	4	lounge	51	2739	160000	1	40.633171	17.634609	6000
4	5	pop	73	3074	106880	1	41.903221	12.495650	5700
...	...	...	...	...	...	...	...	...	...
1533	1534	sport	51	3712	115280	1	45.069679	7.704920	5200
1534	1535	lounge	74	3835	112000	1	45.845692	8.666870	4600
1535	1536	pop	51	2223	60457	1	45.481541	9.413480	7500
1536	1537	lounge	51	2557	80750	1	45.000702	7.682270	5990
1537	1538	pop	51	1766	54276	1	40.323410	17.568270	7900

1538 rows × 9 columns

In [3]:
```
df=df[['engine_power','age_in_days']]
df.columns=['power','age']
```

In [4]:
```
df.head(10)
```

Out[4]:

	power	age
0	51	882
1	51	1186
2	74	4658
3	51	2739
4	73	3074
5	74	3623
6	51	731
7	51	1521
8	73	4049
9	51	3653

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

Out[5]:

	power	age
count	1538.000000	1538.000000
mean	51.904421	1650.980494
std	3.988023	1289.522278
min	51.000000	366.000000
25%	51.000000	670.000000
50%	51.000000	1035.000000
75%	51.000000	2616.000000
max	77.000000	4658.000000

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1538 entries, 0 to 1537
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   power   1538 non-null   int64
 1   age     1538 non-null   int64
dtypes: int64(2)
memory usage: 24.2 KB
```

In [7]:
```
df.fillna(method='ffill',inplace=True)
```

```
C:\Users\pavan\AppData\Local\Temp\ipykernel_6604\4116506308.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df.fillna(method='ffill',inplace=True)
```

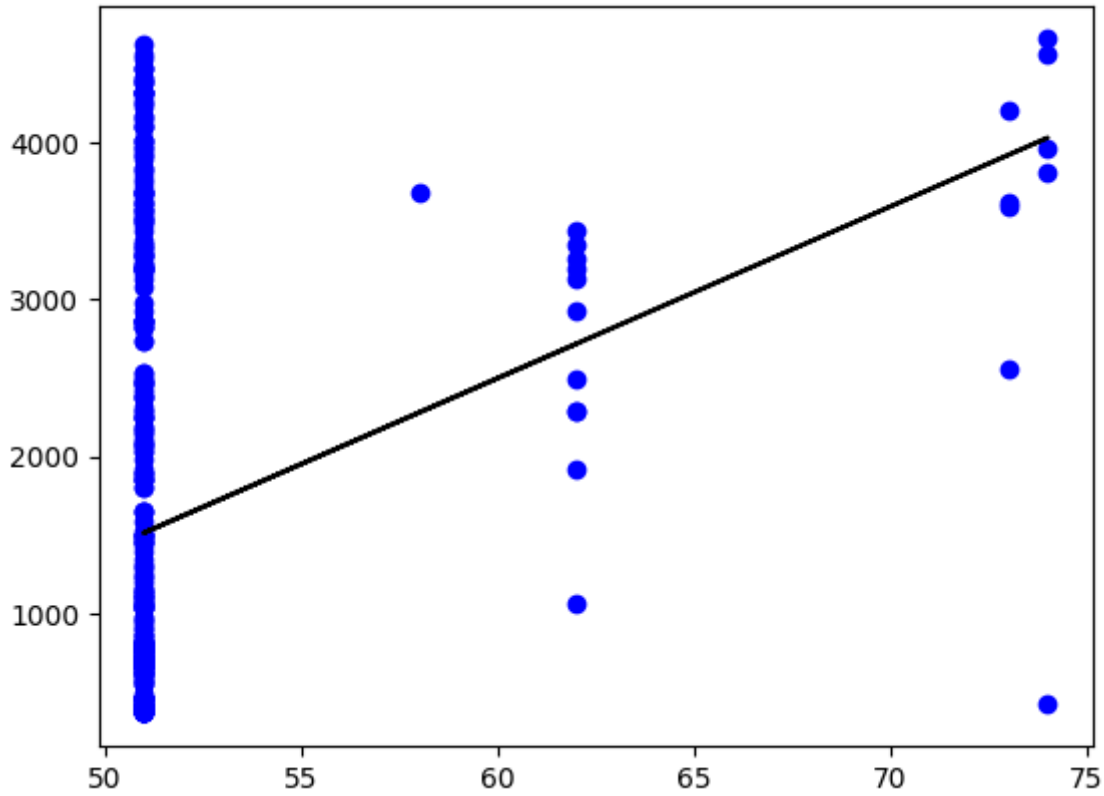
In [8]:
```
x=np.array(df['power']).reshape(-1,1)
y=np.array(df['age']).reshape(-1,1)
```

In [9]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
```

In [10]:
```
regr=LinearRegression()
regr.fit(x_train,y_train)
print(regr.score(x_test,y_test))
```

```
0.036829935706227945
```

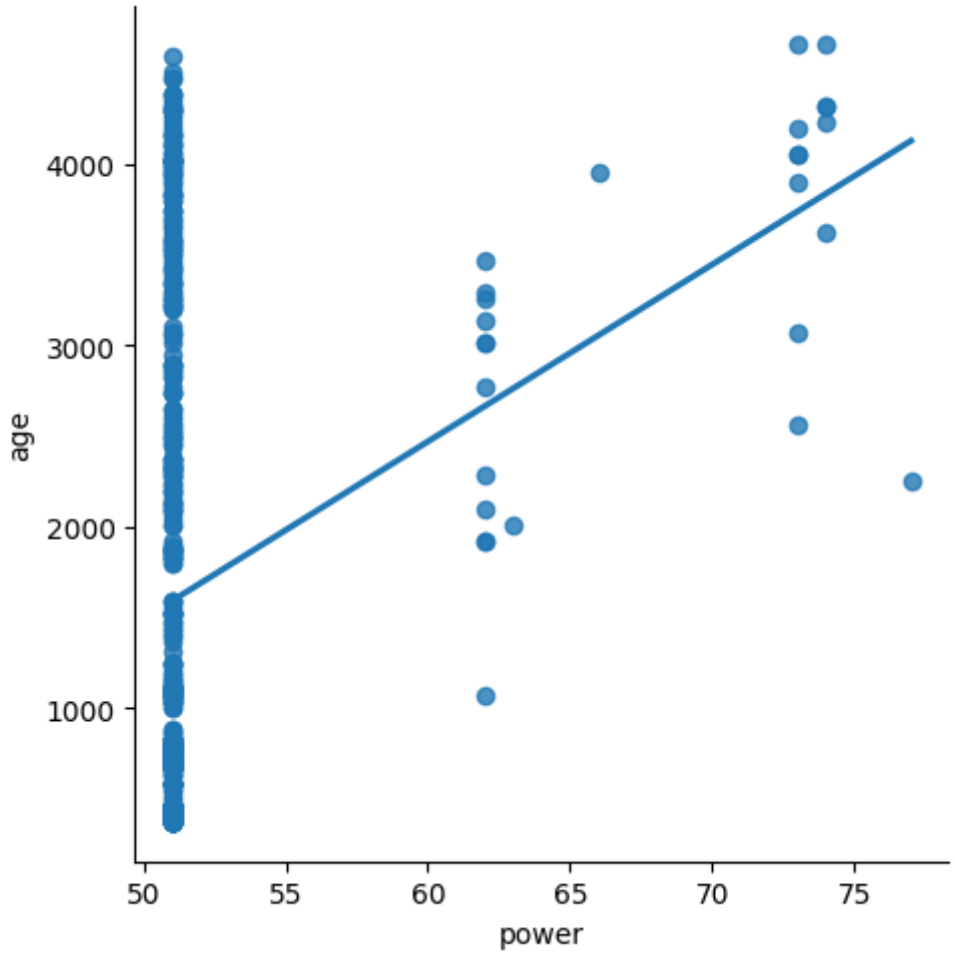
In [11]:
```
y_pred=regr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```



In [12]:
```
df500=df[:][:500]
sns.lmplot(x="power",y="age",data=df500,order=1,ci=None)
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['living']).reshape(-1,1)
y=np.array(df500['above']).reshape(-1,1)
df500.dropna(inplace=True)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
regr=LinearRegression
regr.fit(x_train,y_train)
print("Regression:",regr.score(x_test,y_test))
y_pred=regr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\pandas\core\indexes\base.py:3652, in Index.get_loc
(self, key)
   3651 try:
-> 3652     return self._engine.get_loc(casted_key)
   3653 except KeyError as err:

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\pandas\_libs\index.pyx:147, in pandas._libs.index.I
ndexEngine.get_loc()

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\pandas\_libs\index.pyx:176, in pandas._libs.index.I
ndexEngine.get_loc()

File pandas\_libs\hashtable_class_helper.pxi:7080, in pandas._libs.hashtable.PyObjectHashTable.get_item()

File pandas\_libs\hashtable_class_helper.pxi:7088, in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'living'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[12], line 4
      2 sns.lmplot(x="power",y="age",data=df500,order=1,ci=None)
      3 df500.fillna(method='ffill',inplace=True)
----> 4 x=np.array(df500['living']).reshape(-1,1)
      5 y=np.array(df500['above']).reshape(-1,1)
      6 df500.dropna(inplace=True)

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\pandas\core\frame.py:3761, in DataFrame.__getitem__
(self, key)
   3759 if self.columns.nlevels > 1:
   3760     return self._getitem_multilevel(key)
-> 3761 indexer = self.columns.get_loc(key)
   3762 if is_integer(indexer):
   3763     indexer = [indexer]

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\pandas\core\indexes\base.py:3654, in Index.get_loc
(self, key)
   3652     return self._engine.get_loc(casted_key)
```

```
   3653 except KeyError as err:
-> 3654     raise KeyError(key) from err
   3655 except TypeError:
   3656     # If we have a listlike key, _check_indexing_error will raise
   3657     #  InvalidIndexError. Otherwise we fall through and re-raise
   3658     #  the TypeError.
   3659     self._check_indexing_error(key)

KeyError: 'living'
```



In [13]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
model=LinearRegression()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
r2=r2_score(y_test,y_pred)
print("R2 score:",r2)
```

```
R2 score: 0.036829935706227945
```

In [ ]:
```

```