

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\Svijayalakshmi\Downloads\fiat500_VehicleSelection_Dataset.csv")
df
```

Out[2]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	
0	1	lounge	51	882	25000	1	44.907242	8.611
1	2	pop	51	1186	32500	1	45.666359	12.241
2	3	sport	74	4658	142228	1	45.503300	11.417
3	4	lounge	51	2739	160000	1	40.633171	17.634
4	5	pop	73	3074	106880	1	41.903221	12.495
...	...	...	...	...	...	...	...	...
1533	1534	sport	51	3712	115280	1	45.069679	7.704
1534	1535	lounge	74	3835	112000	1	45.845692	8.666
1535	1536	pop	51	2223	60457	1	45.481541	9.413
1536	1537	lounge	51	2557	80750	1	45.000702	7.682
1537	1538	pop	51	1766	54276	1	40.323410	17.568

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\Svijayalakshmi\AppData\Local\Temp\ipykernel\_16684\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) ([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)
```

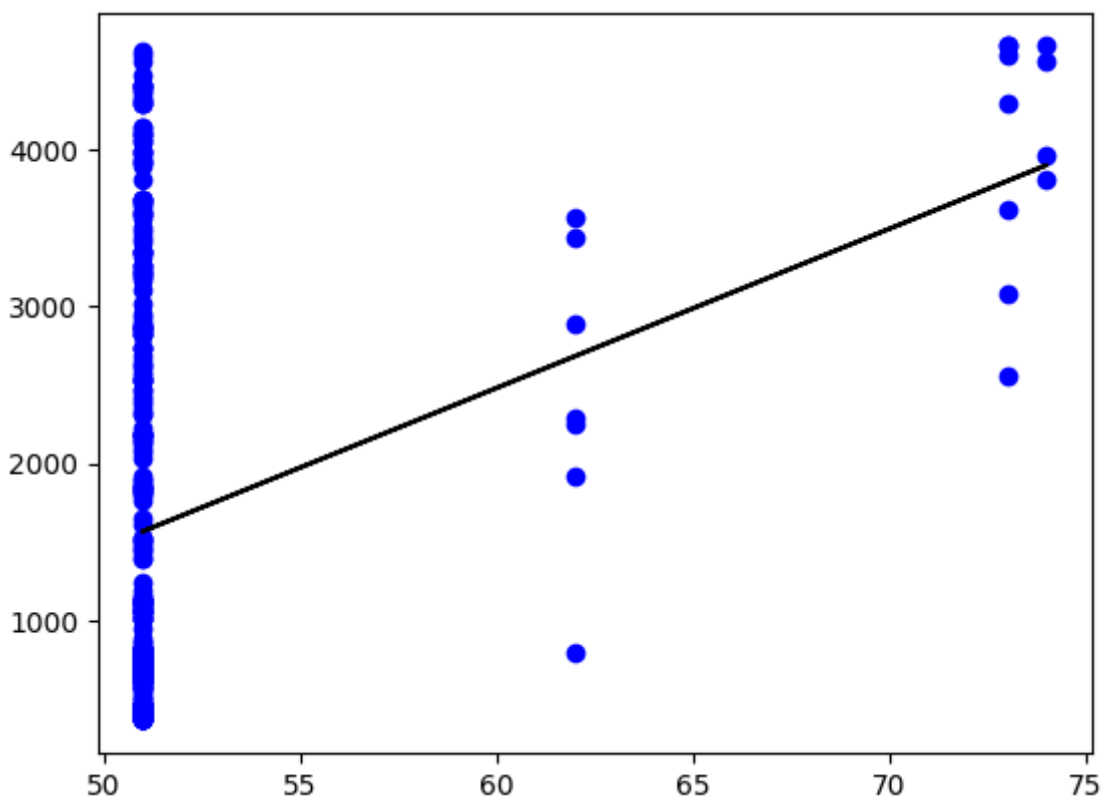
In [11]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
```

In [12]:

```
regr=LinearRegression()  
regr.fit(x_train,y_train)  
print(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()
```

0.10708384445065489

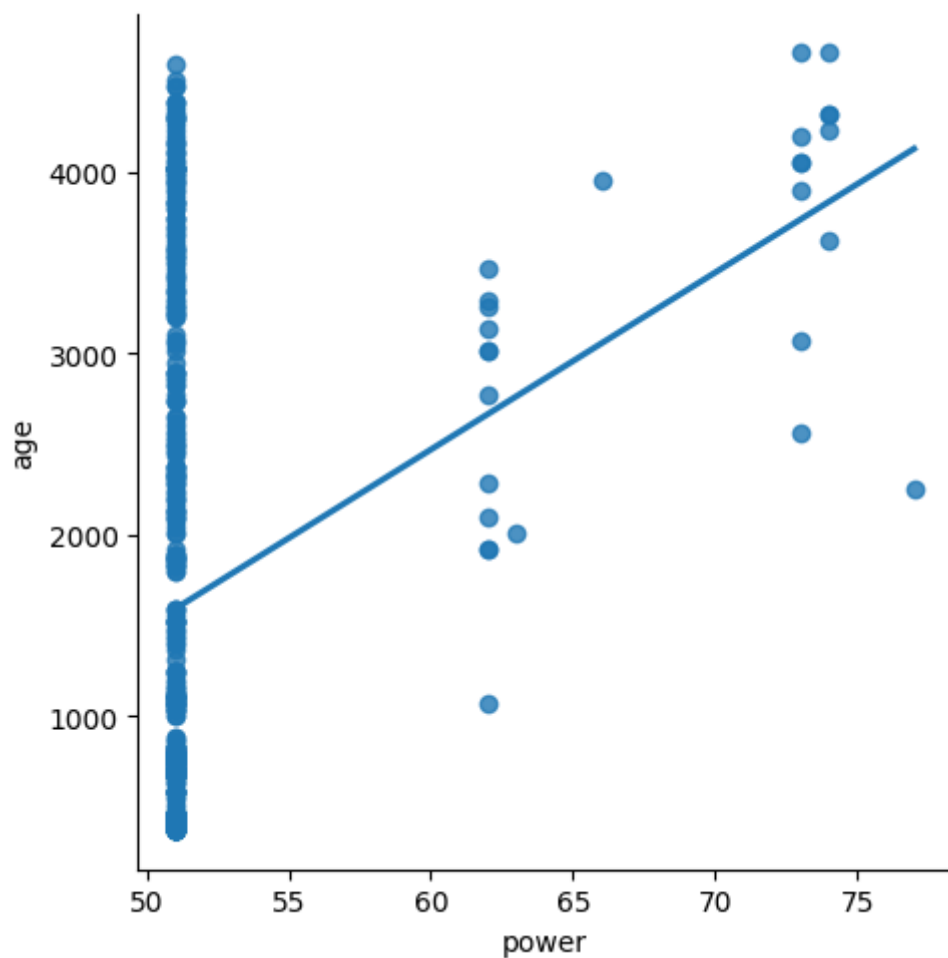


In [13]:

```
df500=df[:][:500]  
sns.lmplot(x="power",y="age",data=df500,order=1,ci=None)
```

Out[13]:

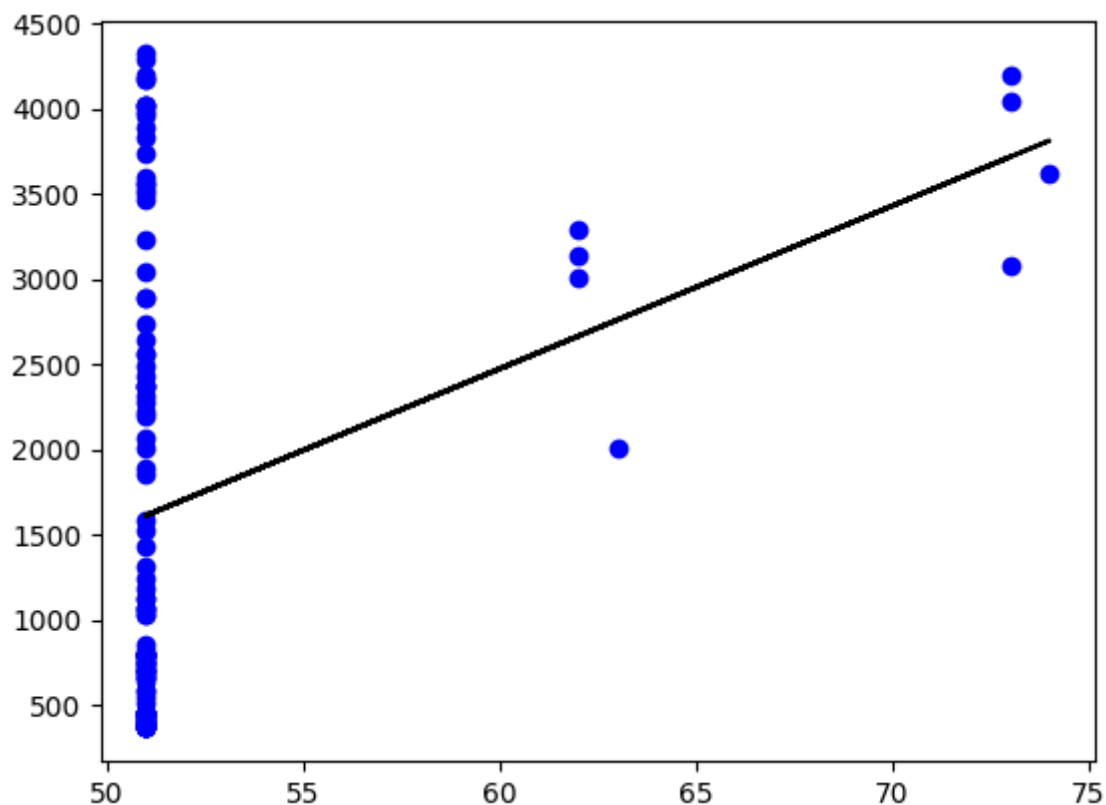
<seaborn.axisgrid.FacetGrid at 0x2379aed2c10>



In [14]:

```
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['power']).reshape(-1,1)
y=np.array(df500['age']).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()
```

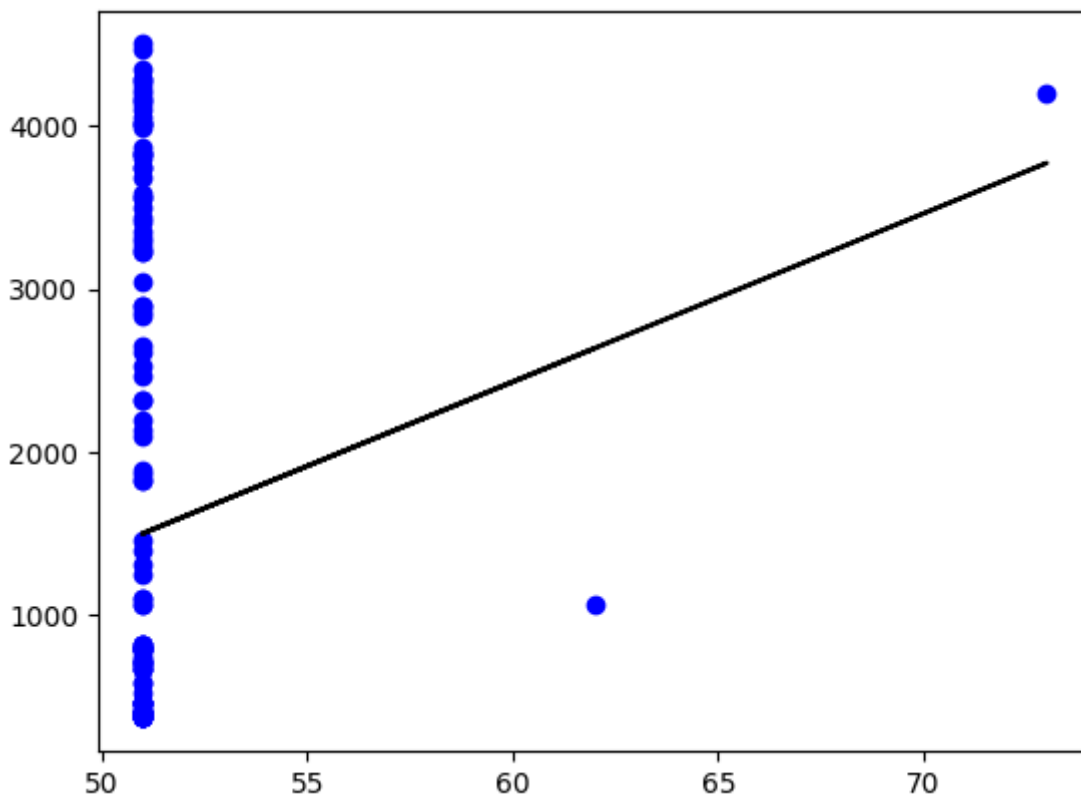
regression: 0.10975816804750838



In [15]:

```
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['power']).reshape(-1,1)
y=np.array(df500['age']).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()
```

regression: -0.04363300110093826



In [16]:

```
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: -0.04363300110093826

In [ ]:

