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\sravya\Downloads\fiat500_VehicleSelection_Dataset (2).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[['lat','lon']]
df.columns=['la','lo']
```

In [4]:

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

Out[4]:

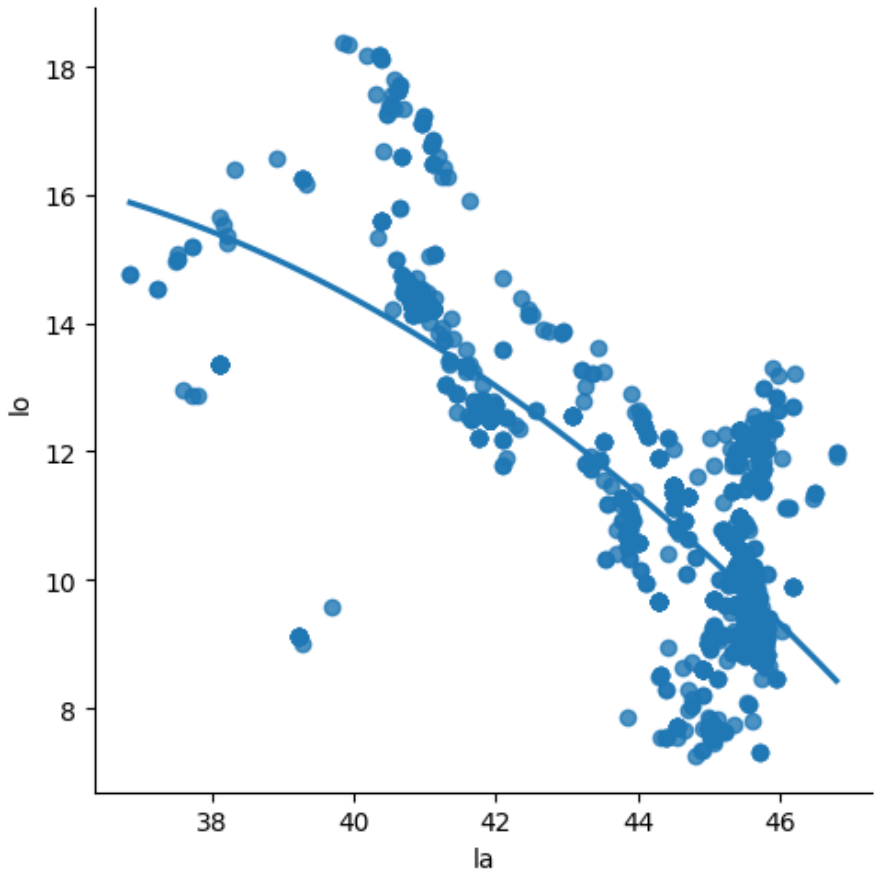
	la	lo
0	44.907242	8.611560
1	45.666359	12.241890
2	45.503300	11.417840
3	40.633171	17.634609
4	41.903221	12.495650
5	45.000702	7.682270
6	44.907242	8.611560
7	41.903221	12.495650
8	45.548000	11.549470
9	45.438301	10.991700

In [6]:

```
sns.lmplot(x="la",y="lo",data=df,order=2,ci=None)
```

Out[6]:

```
<seaborn.axisgrid.FacetGrid at 0x2947eeb6c90>
```



In [7]:

```
df.describe()
```

Out[7]:

	la	lo
count	1538.000000	1538.000000
mean	43.541361	11.563428
std	2.133518	2.328190
min	36.855839	7.245400
25%	41.802990	9.505090
50%	44.394096	11.869260
75%	45.467960	12.769040
max	46.795612	18.365520

In [8]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1538 entries, 0 to 1537
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   la      1538 non-null   float64
 1   lo      1538 non-null   float64
dtypes: float64(2)
memory usage: 24.2 KB
```

In [11]:

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

```
C:\Users\sravya\AppData\Local\Temp\ipykernel_12872\4116506308.py:1: SettingWithCopyWa
rning:
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/us
er_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pand
as-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df.fillna(method='ffill',inplace=True)
```

In [12]:

```
x=np.array(df['la']).reshape(-1,1)
y=np.array(df['lo']).reshape(-1,1)
```

In [13]:

```
df.dropna(inplace=True)
```

```
C:\Users\sravya\AppData\Local\Temp\ipykernel_12872\1379821321.py:1: SettingWithCopyWa
rning:
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/us
er_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pand
as-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df.dropna(inplace=True)
```

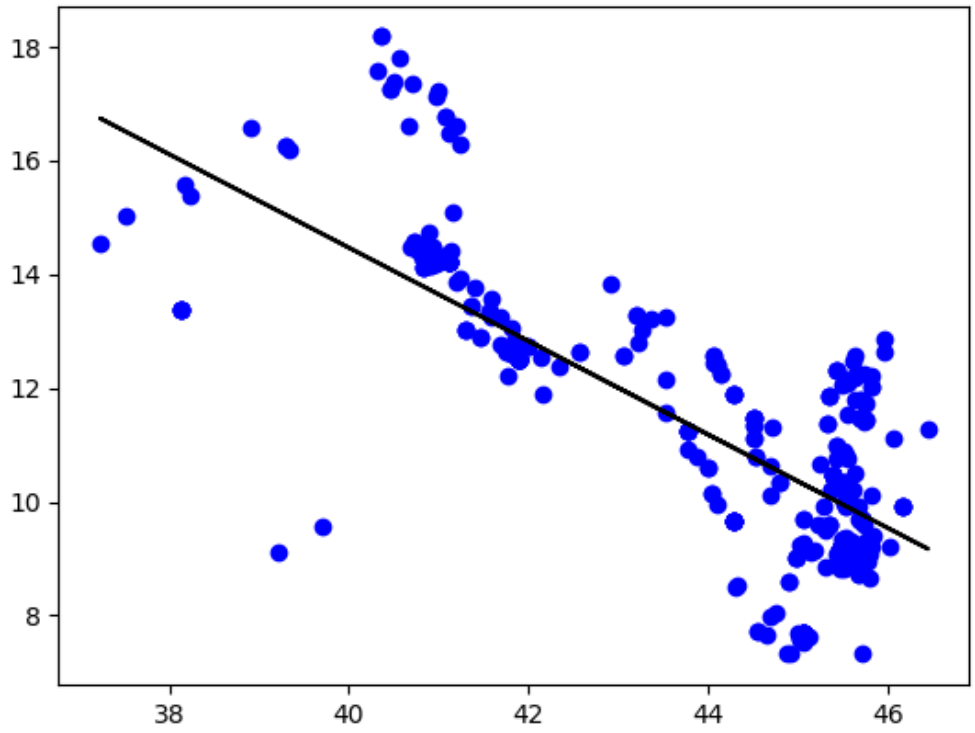
In [14]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.27)
regr=LinearRegression()
regr.fit(x_train,y_train)
print(regr.score(x_test,y_test))
```

```
0.636234433872235
```

In [21]:

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

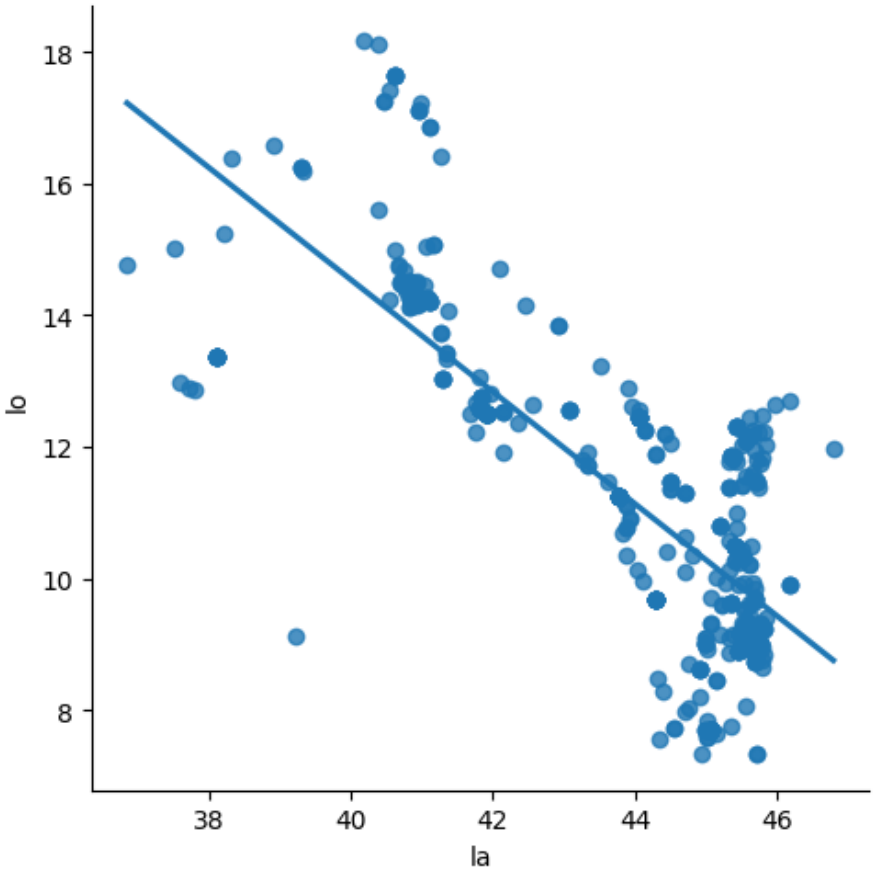


In [22]:

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

Out[22]:

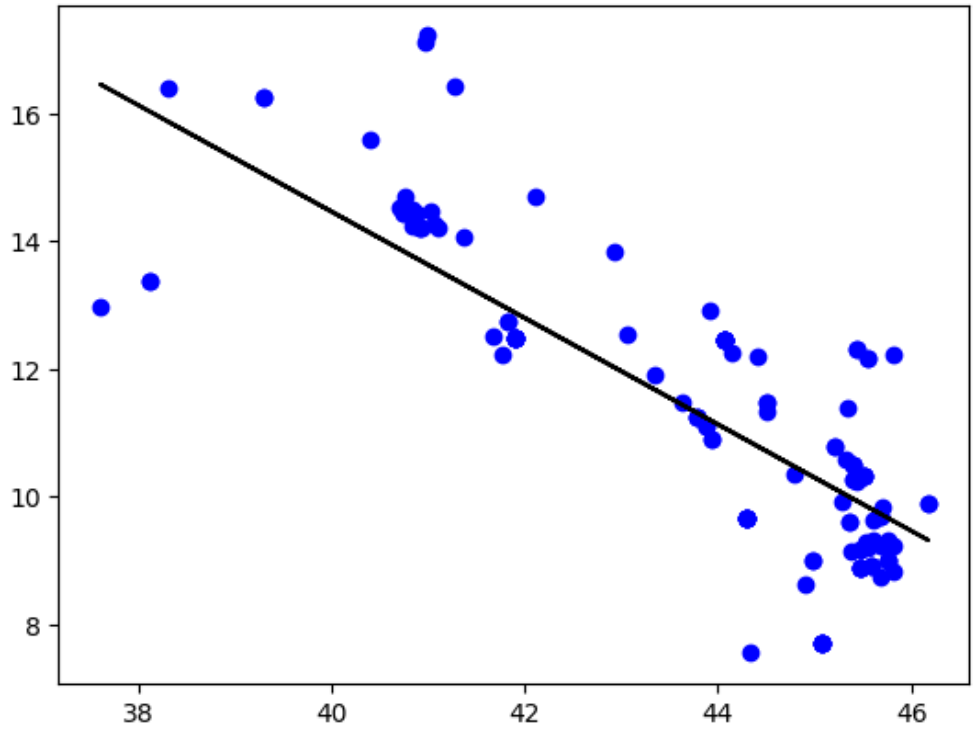
```
<seaborn.axisgrid.FacetGrid at 0x2947f6445d0>
```



In [24]:

```
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['la']).reshape(-1,1)
y=np.array(df500['lo']).reshape(-1,1)
df500.dropna(inplace=True)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.27)
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: 0.6846696276497934
```



In [25]:

```
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: 0.6846696276497934
```

conclusion

Data set we have taken is poor for linear model but with the smaller data works well linear model