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\HP\Downloads\bottle.csv.zip")
df
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_28448\508500159.py:1: DtypeWarning:
Columns (47,73) have mixed types. Specify dtype option on import or set low_m
emory=False.
  df=pd.read_csv(r"C:\Users\HP\Downloads\bottle.csv.zip")
```

Out[2]:

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2S
0	1	1	054.0 056.0	19-4903CR-HY-060-0930-05400560-0000A-3	0	10.500	33.4400	NaN	25.64900	Na
1	1	2	054.0 056.0	19-4903CR-HY-060-0930-05400560-0008A-3	8	10.460	33.4400	NaN	25.65600	Na
2	1	3	054.0 056.0	19-4903CR-HY-060-0930-05400560-0010A-7	10	10.460	33.4370	NaN	25.65400	Na
3	1	4	054.0 056.0	19-4903CR-HY-060-0930-05400560-0019A-3	19	10.450	33.4200	NaN	25.64300	Na
4	1	5	054.0 056.0	19-4903CR-HY-060-0930-05400560-0020A-7	20	10.450	33.4210	NaN	25.64300	Na
...	...	...	...	...	...	...	...	...	...	
864858	34404	864859	093.4 026.4	20-1611SR-MX-310-2239-09340264-0000A-7	0	18.744	33.4083	5.805	23.87055	108.
864859	34404	864860	093.4 026.4	20-1611SR-MX-310-2239-09340264-0002A-3	2	18.744	33.4083	5.805	23.87072	108.
864860	34404	864861	093.4 026.4	20-1611SR-MX-310-2239-09340264-0005A-3	5	18.692	33.4150	5.796	23.88911	108.
864861	34404	864862	093.4 026.4	20-1611SR-MX-310-2239-09340264-0010A-3	10	18.161	33.4062	5.816	24.01426	107.

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2S
864862	34404	864863	093.4 026.4	20-1611SR-MX-310-2239-09340264-0015A-3	15	17.533	33.3880	5.774	24.15297	105.

864863 rows × 74 columns

In [3]:
```
df=df[['Salnty','T_degC']]
df.columns=['Sal','Temp']
```

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

Out[4]:

	Sal	Temp
0	33.440	10.50
1	33.440	10.46
2	33.437	10.46
3	33.420	10.45
4	33.421	10.45
5	33.431	10.45
6	33.440	10.45
7	33.424	10.24
8	33.420	10.06
9	33.494	9.86

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 864863 entries, 0 to 864862
Data columns (total 2 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Sal     817509 non-null  float64
 1   Temp    853900 non-null  float64
dtypes: float64(2)
memory usage: 13.2 MB
```

In [6]:
```
df.fillna(method="ffill",inplace=True)
```

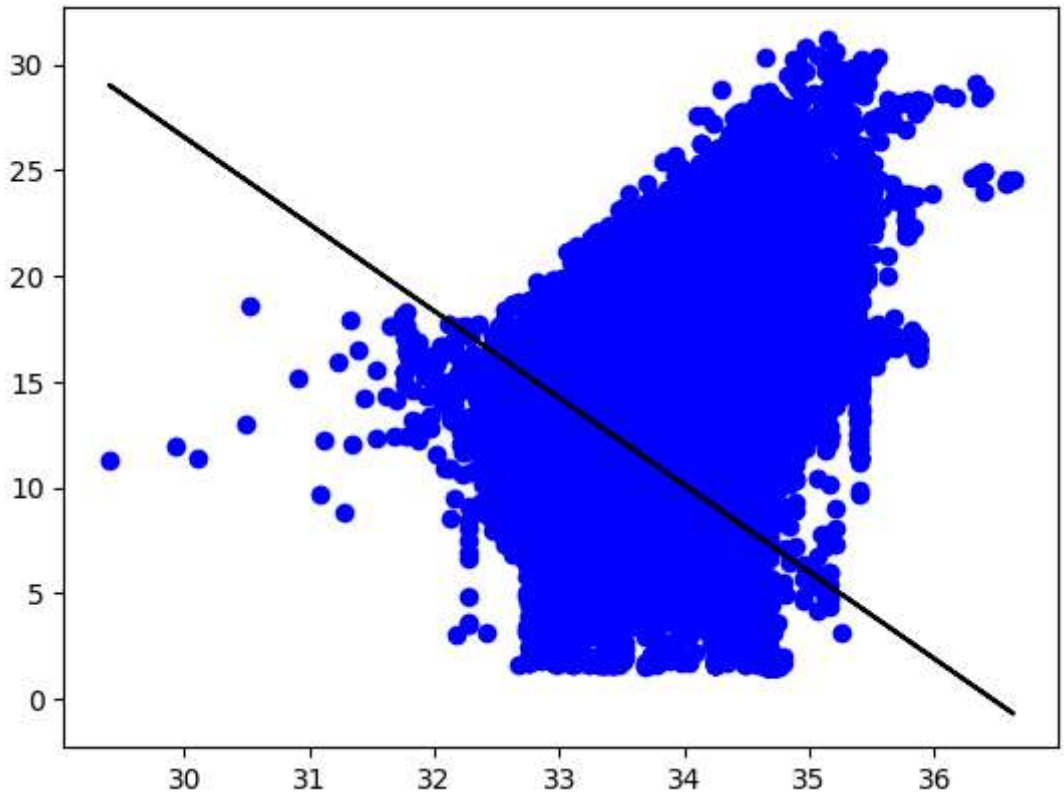
```
C:\Users\HP\AppData\Local\Temp\ipykernel_28448\1844562654.py:1: SettingWithCo
pyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

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

In [7]:
```
x=np.array(df['Sal']).reshape(-1,1)
y=np.array(df['Temp']).reshape(-1,1)
```

In [9]:
```
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(regr.score(x_test,y_test))
```

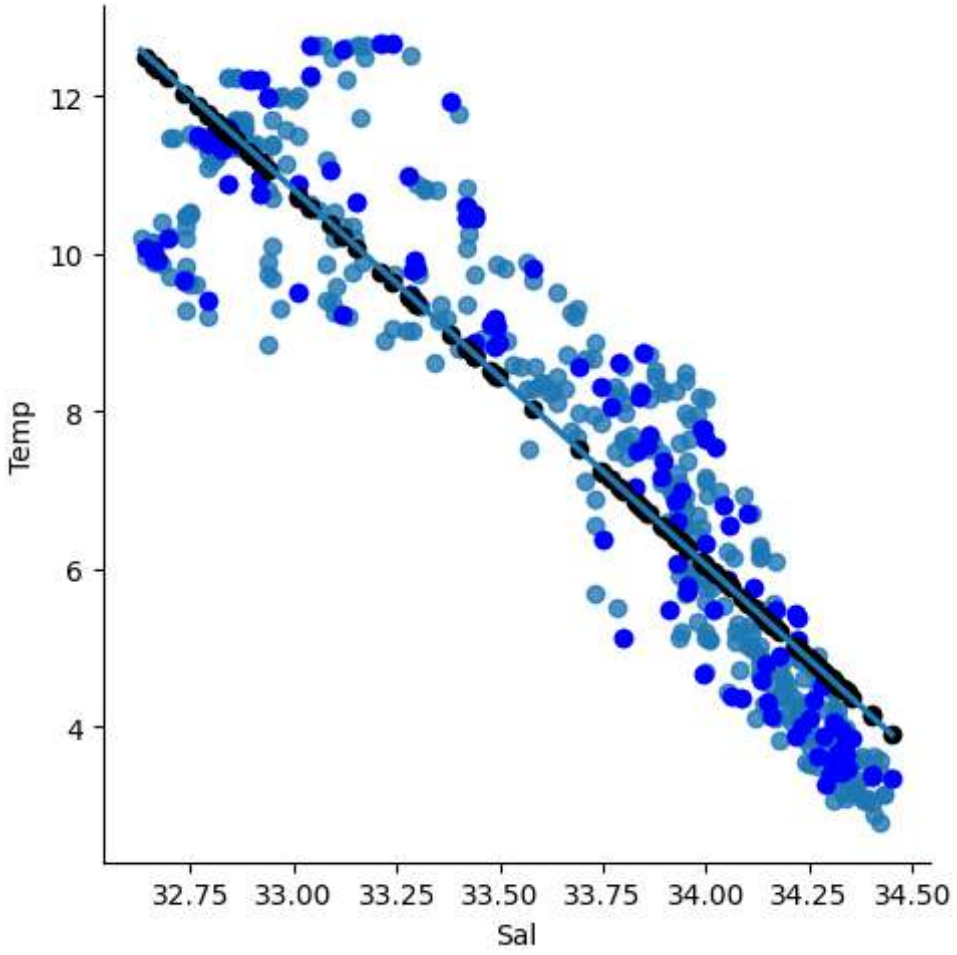
```
0.2022174608873305
```

In [10]:
```
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 [13]:
```
df500=df[:][:500]
sns.lmplot(x="Sal",y="Temp",data=df500,order=1,ci=None)
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['Sal']).reshape(-1,1)
y=np.array(df500['Temp']).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.scatter(x_test,y_pred,color='k')
plt.show()
```

```
Regression: 0.8462427959276528
```



In [17]:
```
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)
```

```
0.7130750653438346
```

vehical data collection

In [2]:
```
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 [7]:
```
df=pd.read_csv(r"C:\Users\HP\Downloads\fiat500_VehicleSelection_Dataset (2).cs
df
```

Out[7]:

	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
...	...	...	...	...	...	...	...	...
1533	1534	sport	51	3712	115280	1	45.069679	7.704920
1534	1535	lounge	74	3835	112000	1	45.845692	8.666870
1535	1536	pop	51	2223	60457	1	45.481541	9.413480
1536	1537	lounge	51	2557	80750	1	45.000702	7.682270
1537	1538	pop	51	1766	54276	1	40.323410	17.568270

1538 rows × 9 columns

In [8]:
```
df=df[['age_in_days','km']]
df.columns=['age','k']
```

In [9]:
```
df.head(10)
```

Out[9]:

	age	k
0	882	25000
1	1186	32500
2	4658	142228
3	2739	160000
4	3074	106880
5	3623	70225
6	731	11600
7	1521	49076
8	4049	76000
9	3653	89000

In [10]:
```
df.info()
```

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

In [11]:
```
df.fillna(method="ffill",inplace=True)
```

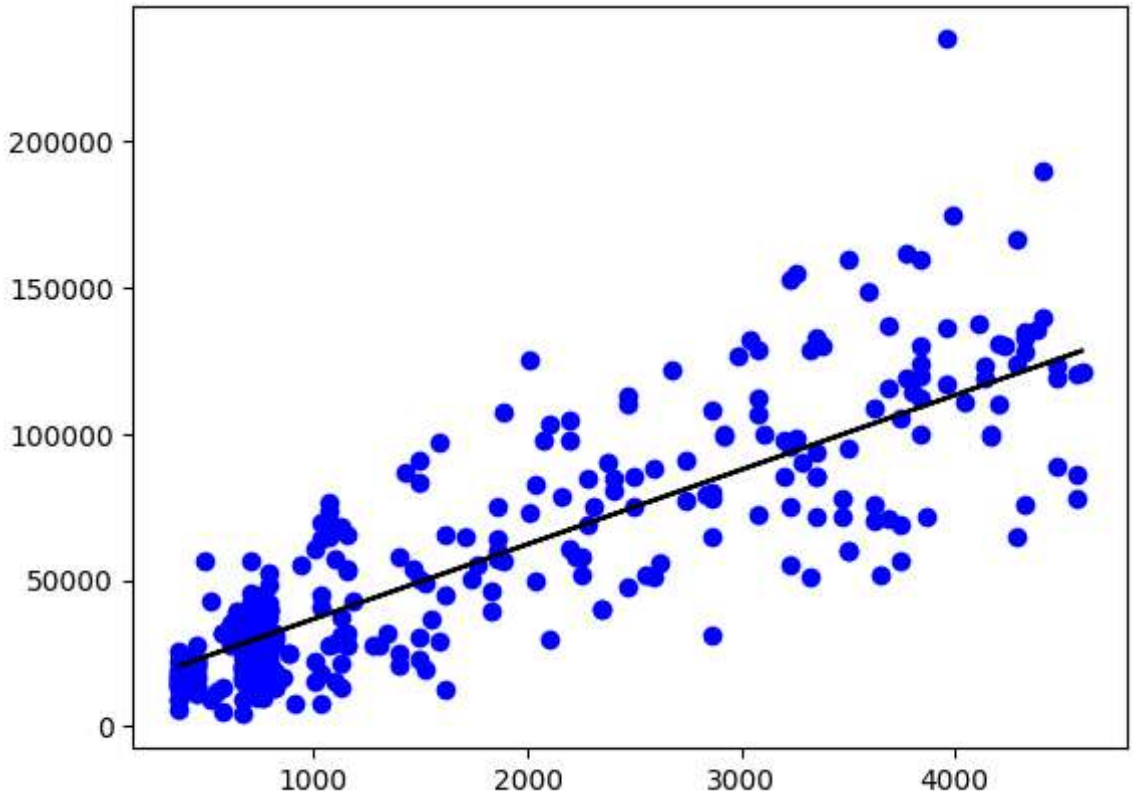
```
C:\Users\HP\AppData\Local\Temp\ipykernel_25884\1844562654.py:1: SettingWithCo
pyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

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

In [12]:
```
x=np.array(df['age']).reshape(-1,1)
y=np.array(df['k']).reshape(-1,1)
```

In [13]:
```
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(regr.score(x_test,y_test))
```

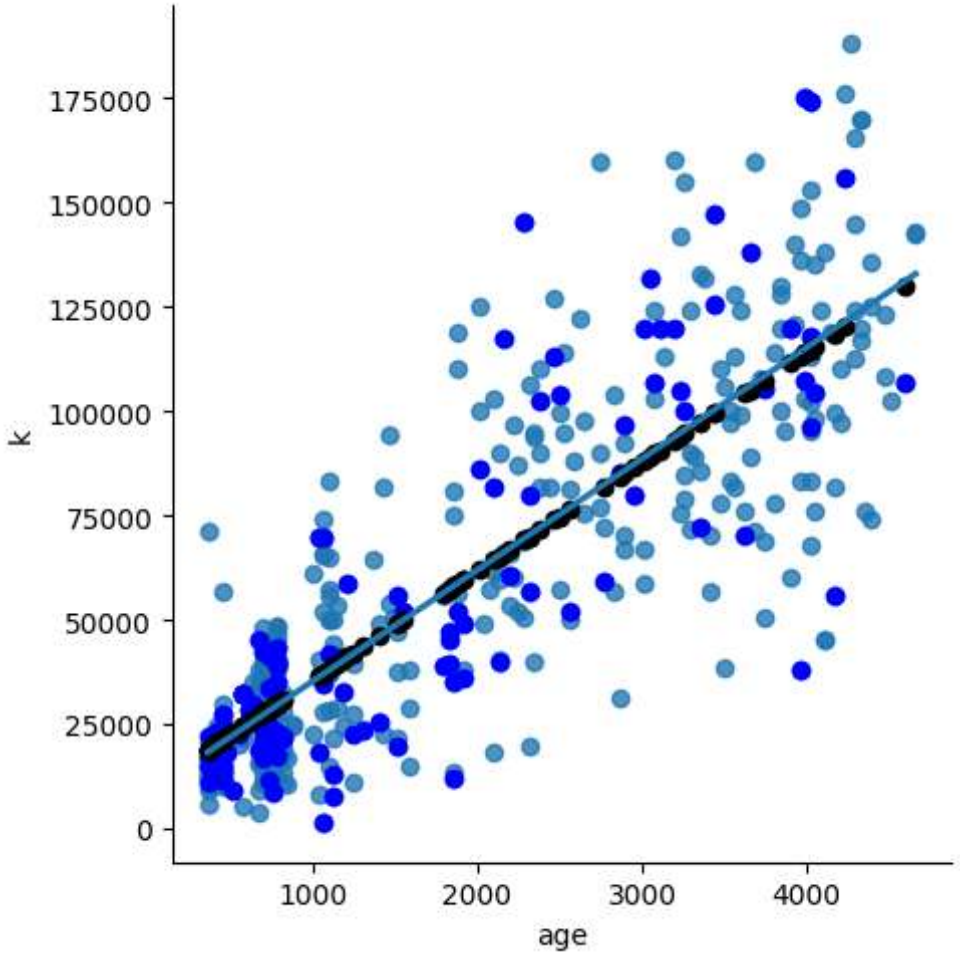
```
0.7293526366908099
```

In [14]:
```
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 [15]:
```
df500=df[:][:500]
sns.lmplot(x="age",y="k",data=df500,order=1,ci=None)
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['age']).reshape(-1,1)
y=np.array(df500['k']).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.scatter(x_test,y_pred,color='k')
plt.show()
```

```
Regression: 0.7130750653438346
```



In [14]:
```
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)
```

```
0.040148522263719566
```

House price prediction

In [2]:
```
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 [3]:
```
df=pd.read_csv(r"C:\Users\HP\Downloads\kc_house_data.csv.zip")
df
```

Out[3]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floo
0	7129300520	20141013T000000	221900.0	3	1.00	1180	5650	1
1	6414100192	20141209T000000	538000.0	3	2.25	2570	7242	2
2	5631500400	20150225T000000	180000.0	2	1.00	770	10000	1
3	2487200875	20141209T000000	604000.0	4	3.00	1960	5000	1
4	1954400510	20150218T000000	510000.0	3	2.00	1680	8080	1
...	...	...	...	...	...	...	...	
21608	263000018	20140521T000000	360000.0	3	2.50	1530	1131	3
21609	6600060120	20150223T000000	400000.0	4	2.50	2310	5813	2
21610	1523300141	20140623T000000	402101.0	2	0.75	1020	1350	2
21611	291310100	20150116T000000	400000.0	3	2.50	1600	2388	2
21612	1523300157	20141015T000000	325000.0	2	0.75	1020	1076	2

21613 rows × 21 columns

In [4]:
```
df=df[['sqft_living','sqft_lot']]
df.columns=['live','lot']
```

In [5]:
```
df.head(10)
```

Out[5]:

	live	lot
0	1180	5650
1	2570	7242
2	770	10000
3	1960	5000
4	1680	8080
5	5420	101930
6	1715	6819
7	1060	9711
8	1780	7470
9	1890	6560

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21613 entries, 0 to 21612
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   live    21613 non-null  int64
 1   lot     21613 non-null  int64
dtypes: int64(2)
memory usage: 337.8 KB
```

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

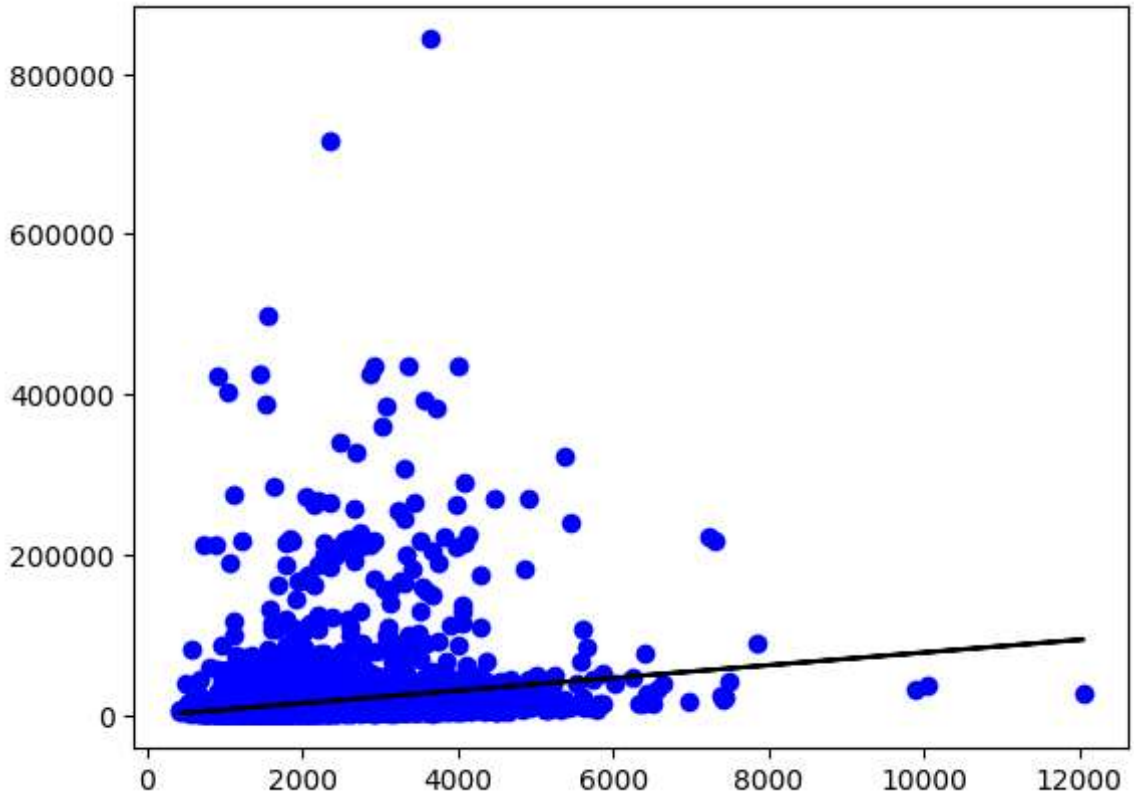
```
C:\Users\HP\AppData\Local\Temp\ipykernel_8876\1844562654.py:1: SettingWithCop
yWarning:
A value is trying to be set on a copy of a slice from a DataFrame

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

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

In [10]:
```
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(regr.score(x_test,y_test))
```

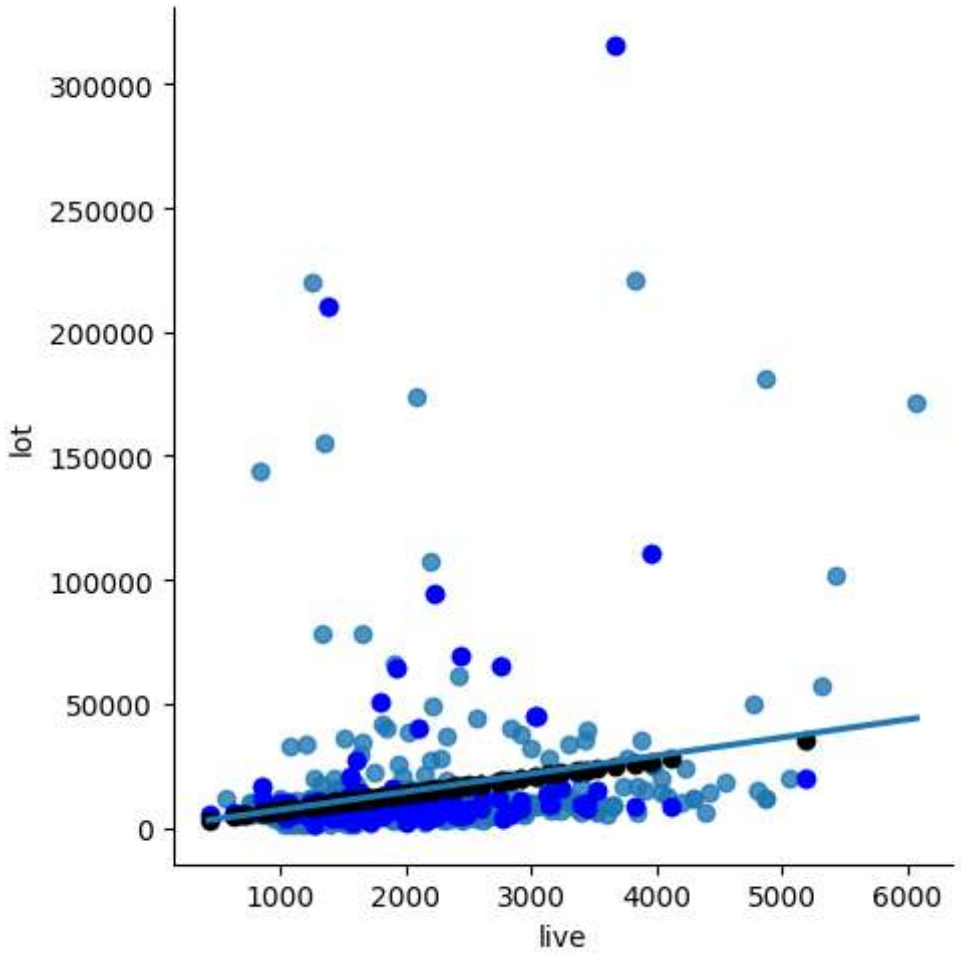
```
0.03525876524343974
```

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="live",y="lot",data=df500,order=1,ci=None)
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['live']).reshape(-1,1)
y=np.array(df500['lot']).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.scatter(x_test,y_pred,color='k')
plt.show()
```

```
Regression: 0.040148522263719566
```



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)
```

```
0.040148522263719566
```

In [ ]: