In [ ]:
```
C:\Users\ubinl\Downloads\kc_house_data.csv.zip
```

In [2]:
```
#step 1:importing
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
#reading the data set
```

In [3]:
```
df=pd.read_csv(r"C:\Users\ubinl\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

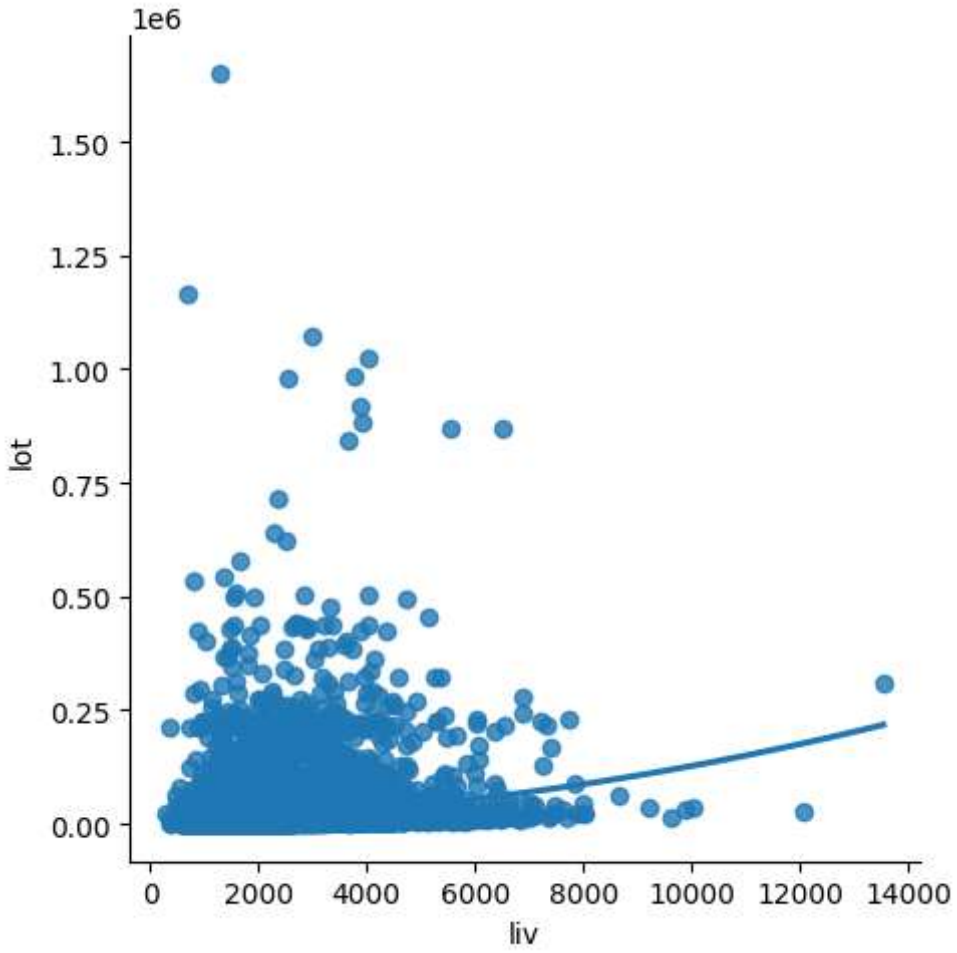
In [5]:
```
df=df[['sqft_living','sqft_lot']]
df.columns=['liv','lot']
df.head(10)
```

Out[5]:

	liv	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 [7]:
```
#step 3:exploring
sns.lmplot(x="liv",y="lot",data=df,order=2,ci=None)
```

Out[7]: <seaborn.axisgrid.FacetGrid at 0x20ddccb1710>



In [9]:
```
df.describe()
```

Out[9]:

	liv	lot
count	21613.000000	2.161300e+04
mean	2079.899736	1.510697e+04
std	918.440897	4.142051e+04
min	290.000000	5.200000e+02
25%	1427.000000	5.040000e+03
50%	1910.000000	7.618000e+03
75%	2550.000000	1.068800e+04
max	13540.000000	1.651359e+06

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

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

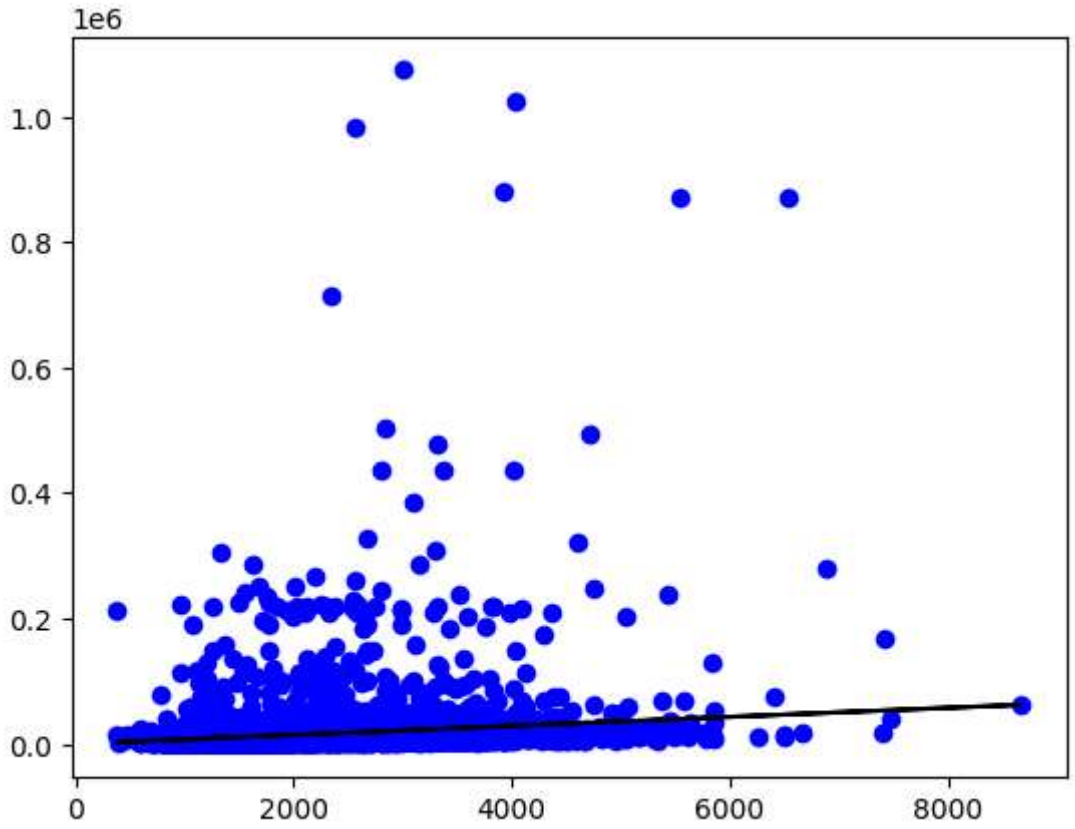
In [11]:
```
#step 4:
df.fillna(method='ffill',inplace=True)
```

```
C:\Users\ubinl\AppData\Local\Temp\ipykernel_29080\3632936489.py:2: SettingWit
hCopyWarning:
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]:
```
#step 5:training model
x=np.array(df['liv']).reshape(-1,1)
y=np.array(df['lot']).reshape(-1,1)
#seperating
#column
df.dropna(inplace=True)
#droping values
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
#spliting data
regr=LinearRegression()
regr.fit(x_train,y_train)
print(regr.score(x_test,y_test))
```

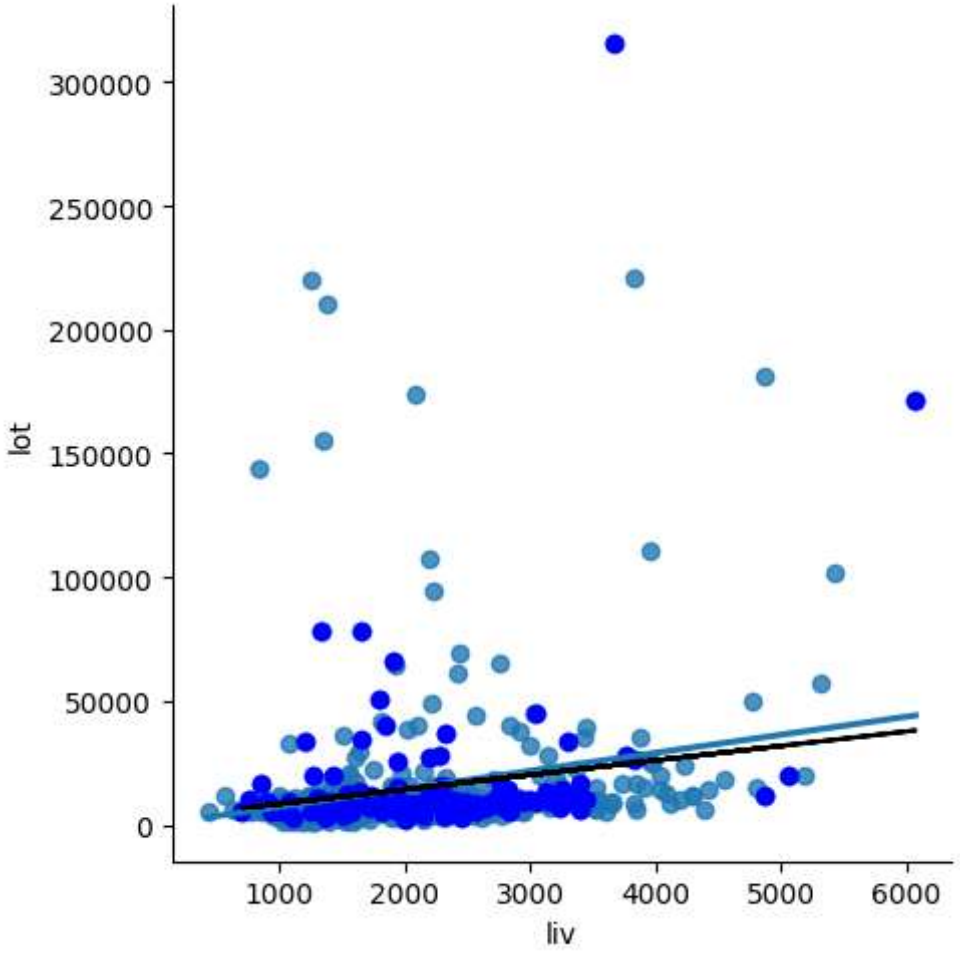
```
0.033943086060982175

C:\Users\ubinl\AppData\Local\Temp\ipykernel_29080\378609693.py:6: SettingWith
CopyWarning:
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.dropna(inplace=True)
```

In [13]:
```
#step 6:exploring results
y_pred=regr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
#scatter
```



In [15]:
```
#step7:working with a smaller data set
df500=df[:][:500]
#selecting
sns.lmplot(x="liv",y="lot",data=df500,order=1,ci=None)
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['liv']).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.plot(x_test,y_pred,color='k')
plt.show()
```

```
Regression: 0.07299996099800554
```



In [16]:
```
#step 8:
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
#train
model=LinearRegression()
model.fit(x_train,y_train)
#evaluate
y_pred=model.predict(x_test)
r2=r2_score(y_test,y_pred)
print("r2_score:",r2)
```

```
r2_score: 0.07299996099800554
```

In [ ]: