In [1]:
```python
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]:
```python
df=pd.read_csv(r"C:\Users\Niranjan\Downloads\data.csv")
df
```

Out[2]:

	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	vi
0	2014-05-02 00:00:00	3.130000e+05	3.0	1.50	1340	7912	1.5	0	
1	2014-05-02 00:00:00	2.384000e+06	5.0	2.50	3650	9050	2.0	0	
2	2014-05-02 00:00:00	3.420000e+05	3.0	2.00	1930	11947	1.0	0	
3	2014-05-02 00:00:00	4.200000e+05	3.0	2.25	2000	8030	1.0	0	
4	2014-05-02 00:00:00	5.500000e+05	4.0	2.50	1940	10500	1.0	0	
...	...	...	...	...	...	...	...	...	
4595	2014-07-09 00:00:00	3.081667e+05	3.0	1.75	1510	6360	1.0	0	
4596	2014-07-09 00:00:00	5.343333e+05	3.0	2.50	1460	7573	2.0	0	
4597	2014-07-09 00:00:00	4.169042e+05	3.0	2.50	3010	7014	2.0	0	
4598	2014-07-10 00:00:00	2.034000e+05	4.0	2.00	2090	6630	1.0	0	
4599	2014-07-10 00:00:00	2.206000e+05	3.0	2.50	1490	8102	2.0	0	

4600 rows × 18 columns

In [3]:
```python
df=df[['sqft_living','sqft_lot']]
df.columns=['liv','lot']
```

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

Out[4]:

	liv	lot
0	1340	7912
1	3650	9050
2	1930	11947
3	2000	8030
4	1940	10500
5	880	6380
6	1350	2560
7	2710	35868
8	2430	88426
9	1520	6200

In [5]: `sns.lmplot(x='liv',y='lot',data=df,order=2,ci=None)`

Out[5]: `<seaborn.axisgrid.FacetGrid at 0x28d817bfa50>`
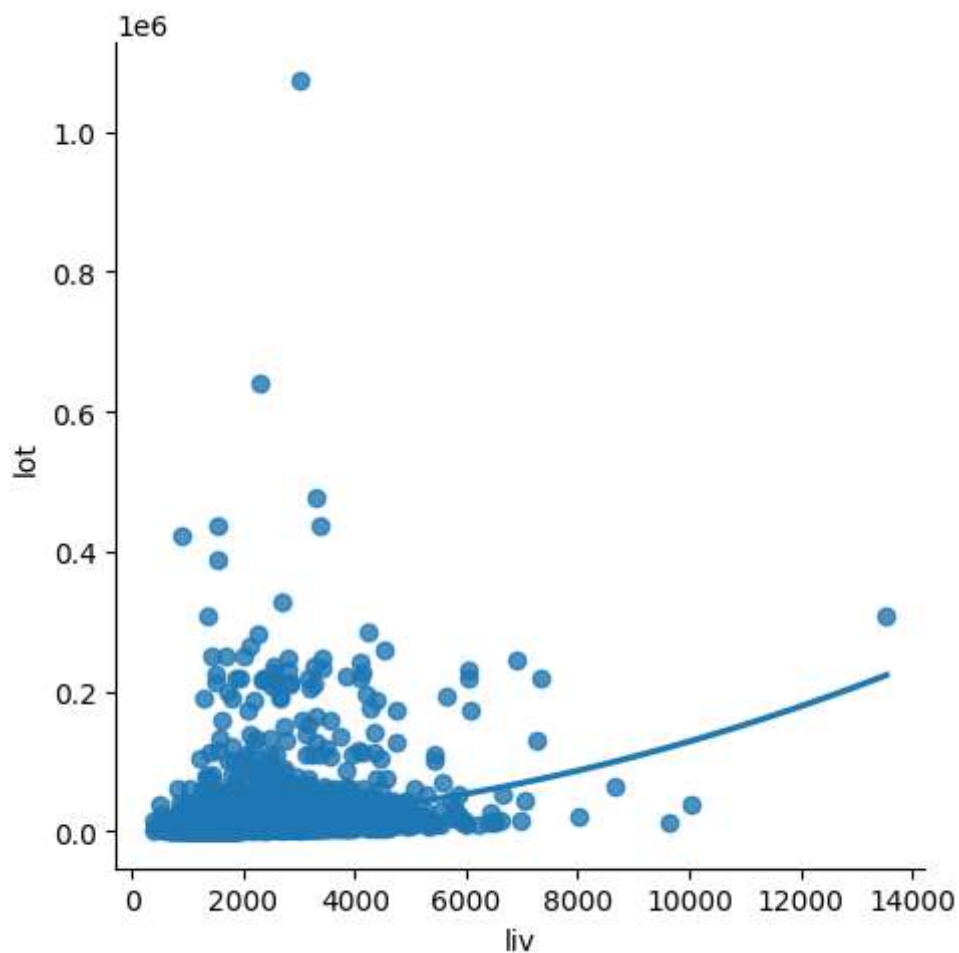
In [6]: `df.describe()`

Out[6]:

	liv	lot
count	4600.000000	4.600000e+03
mean	2139.346957	1.485252e+04
std	963.206916	3.588444e+04
min	370.000000	6.380000e+02
25%	1460.000000	5.000750e+03
50%	1980.000000	7.683000e+03
75%	2620.000000	1.100125e+04
max	13540.000000	1.074218e+06

In [7]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4600 entries, 0 to 4599
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   liv     4600 non-null   int64
 1   lot     4600 non-null   int64
dtypes: int64(2)
memory usage: 72.0 KB
```

In [8]:
```python
df.fillna(method='ffill',inplace=True)
```

```
C:\Users\Niranjan\AppData\Local\Temp\ipykernel_3836\4116506308.py:1: Settin
gWithCopyWarning:
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-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://p
andas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-vi
ew-versus-a-copy)
  df.fillna(method='ffill',inplace=True)
```

In [9]:
```python
x=np.array(df['liv']).reshape(-1,1)
y=np.array(df['lot']).reshape(-1,1)
```

In [10]:
```python
df.dropna(inplace=True)
```

```
C:\Users\Niranjan\AppData\Local\Temp\ipykernel_3836\1379821321.py:1: Settin
gWithCopyWarning:
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-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://p
andas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-vi
ew-versus-a-copy)
  df.dropna(inplace=True)
```

In [11]:
```python
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))
```
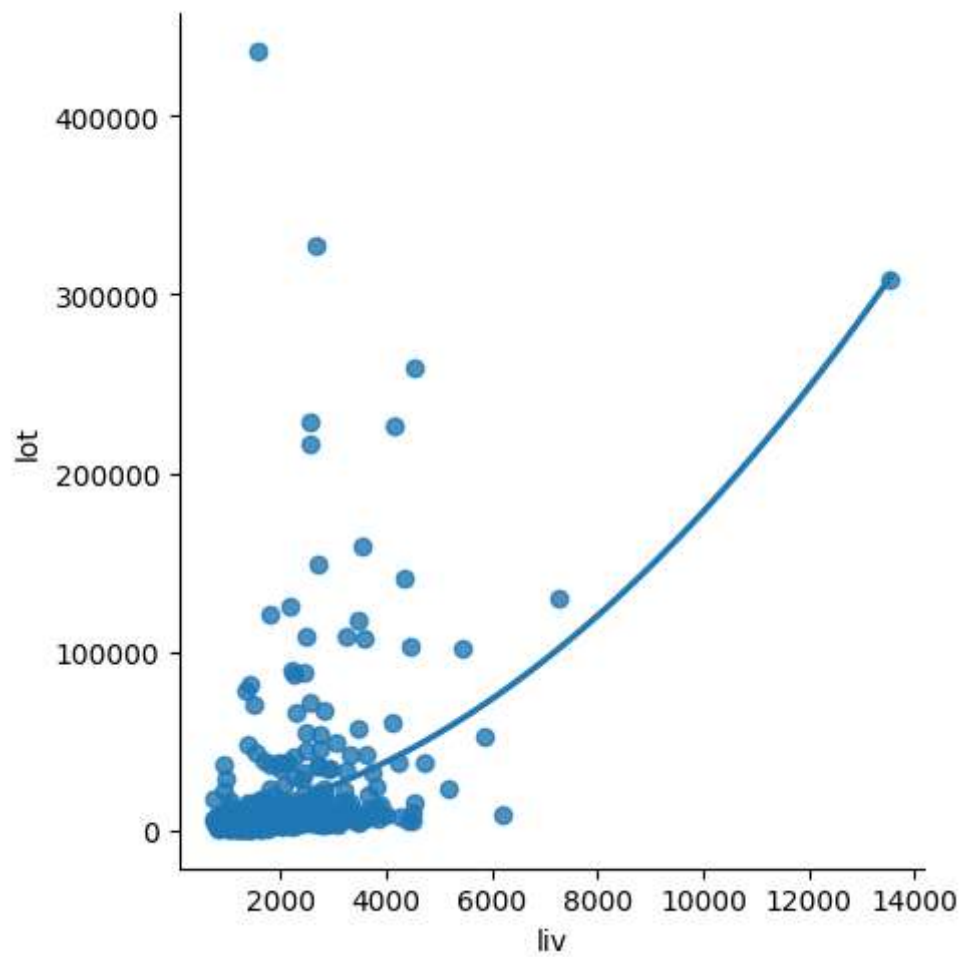
```
0.03550971588733709
```

In [12]:
```python
y_pred=regr.predict(X_test)
plt.scatter(X_test,y_test,color='b')
plt.scatter(X_test,y_pred,color='k')
plt.show()
```
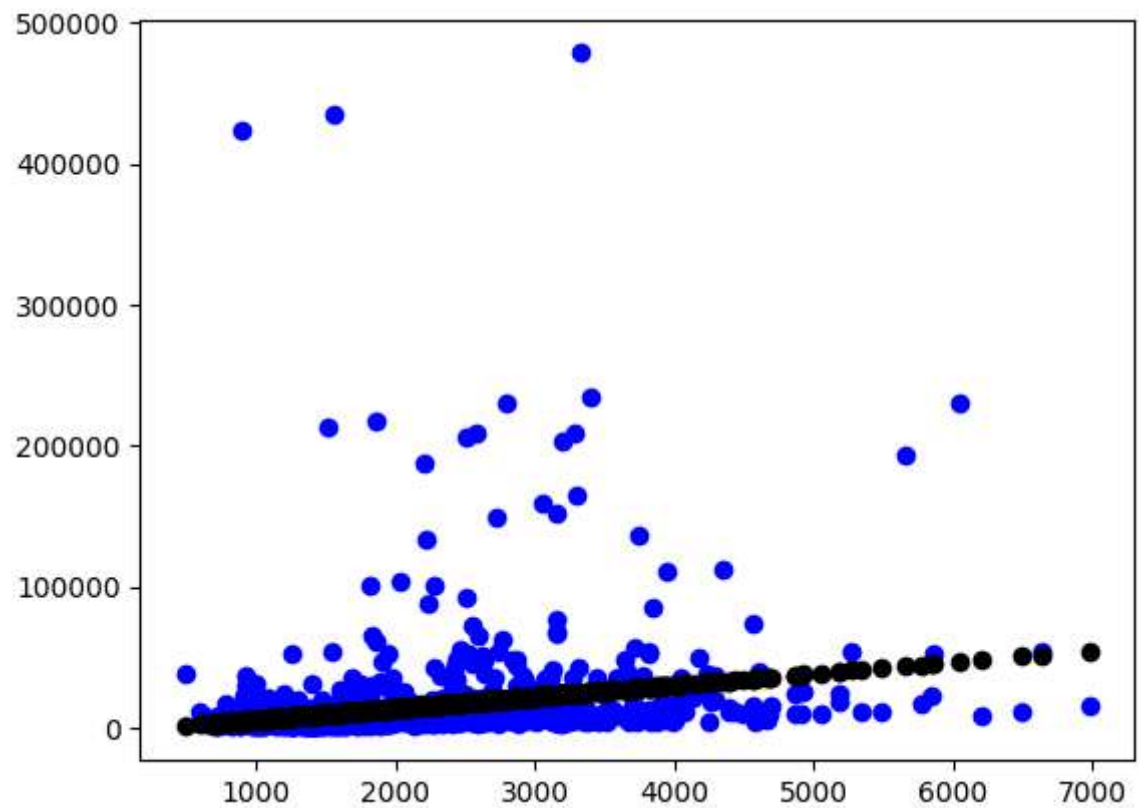
In [13]:
```
df500=df[:][:500]
sns.lmplot(x="liv",y="lot",data=df500,order=2,ci=None)
```

Out[13]: <seaborn.axisgrid.FacetGrid at 0x28d83869d90>

```python
In [14]: 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: 0.04950279161907423
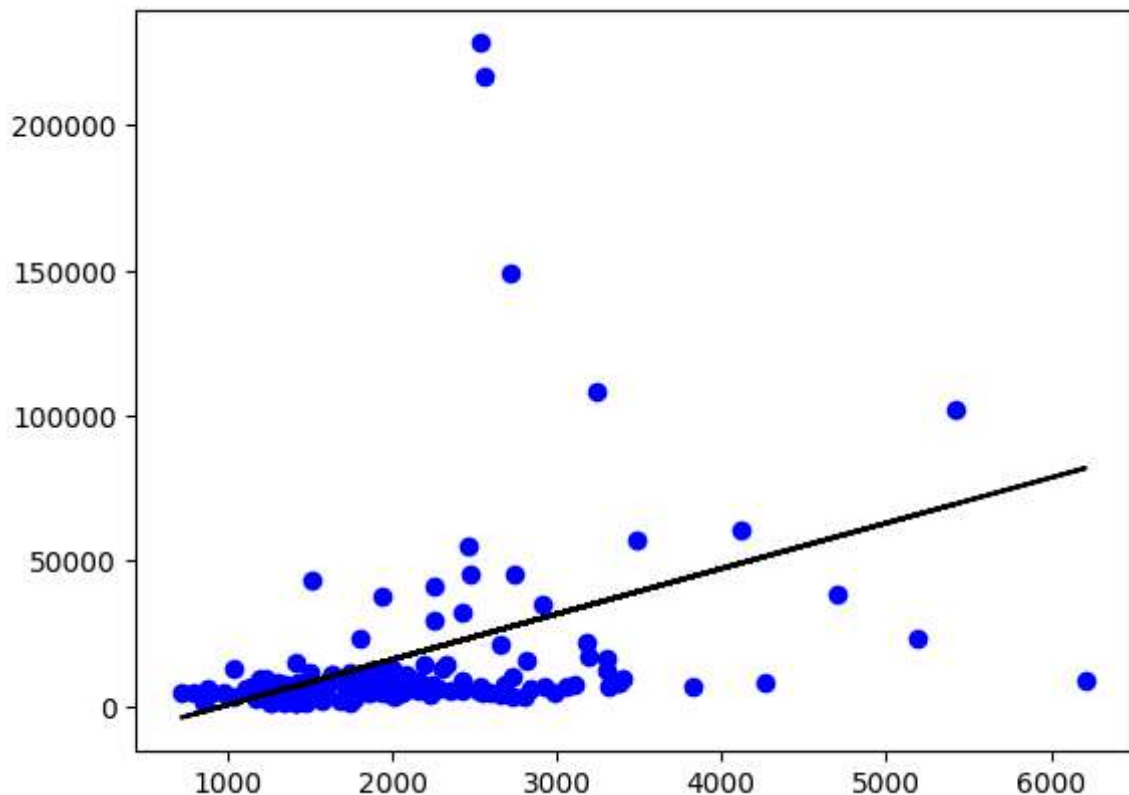

```python
In [15]: from sklearn.linear_model import LinearRegression
         from sklearn.metrics import r2_score
         mode1=LinearRegression()
         mode1.fit(X_train,y_train)
         y_pred=mode1.predict(X_test)
         r2=r2_score(y_test,y_pred)
         print("R2 score:",r2)
```

R2 score: 0.04950279161907423

In [ ]: