In [ ]:

```
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]:

```
#Step-2:Reading the dataset
ds=pd.read_csv(r"C:\Users\91955\Downloads\data.csv")
ds
```

Out[2]:

	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront
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]:

```
ds=ds[['sqft_living','sqft_lot']]
#Taking only the selected two attributes from the dataset
ds.columns=['Living','Lot']
#Renaming the columns for easier writing of the code
```

In [4]:

```
ds.head(10)
#Displaying only the 1st 10 rows
```

Out[4]:

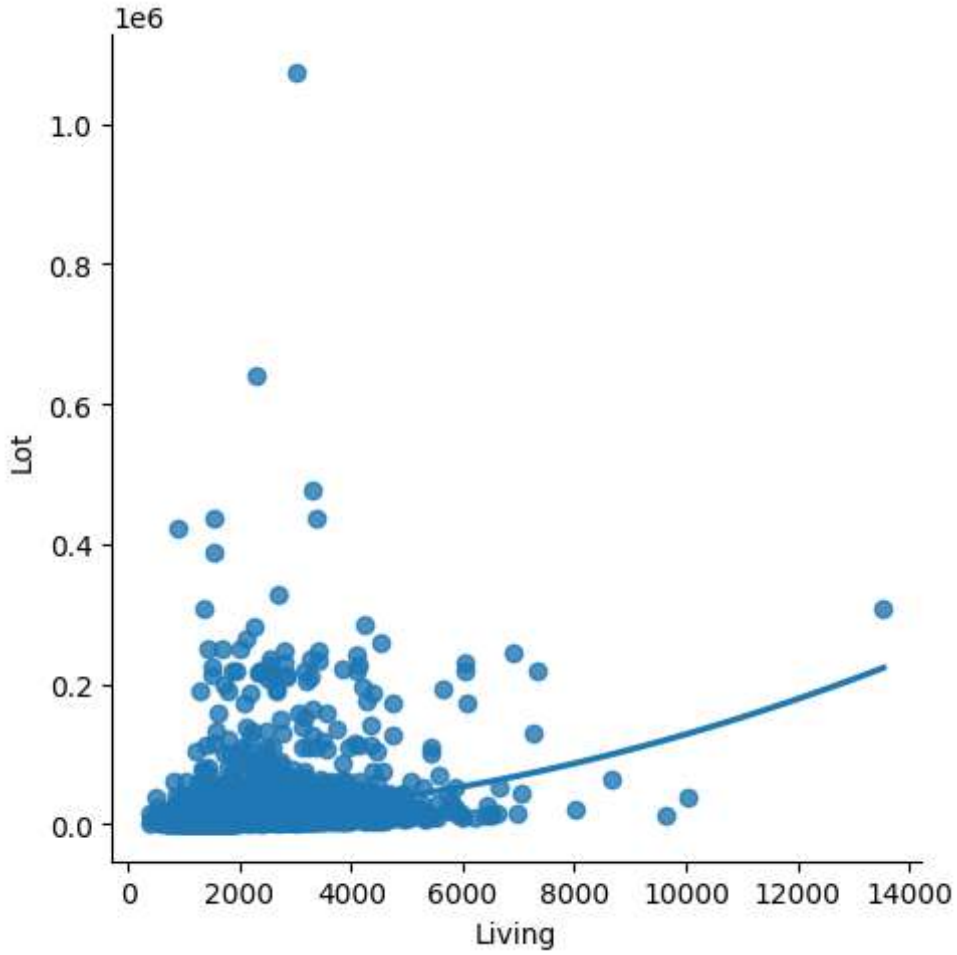
	Living	Lot
0	1340	7912
1	3650	9050
2	1930	11947
3	2000	8030
4	1940	10500
5	880	6380
6	1350	2560
7	2710	35868
8	2430	88426
9	1520	6200

In [5]:

```
1 sns.lmplot(x='Living',y='Lot',data=ds,order=2,ci=None)
```

Out[5]:

```
<seaborn.axisgrid.FacetGrid at 0x196c8821090>
```



In [6]:

```
1 ds.describe()
```

Out[6]:

	Living	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]:

```
1 ds.info()
```

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

In [8]:

```
1 ds.fillna(method='ffill')
```

Out[8]:

	Living	Lot
0	1340	7912
1	3650	9050
2	1930	11947
3	2000	8030
4	1940	10500
...	...	...
4595	1510	6360
4596	1460	7573
4597	3010	7014
4598	2090	6630
4599	1490	8102

4600 rows × 2 columns

In [10]:

```
1 x=np.array(ds['Living']).reshape(-1,1)
2 y=np.array(ds['Lot']).reshape(-1,1)
```

In [11]:

```
1 ds.dropna(inplace=True)
```

```
C:\Users\91955\AppData\Local\Temp\ipykernel_12408\2725967003.py:1: Setting
WithCopyWarning:
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://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  ds.dropna(inplace=True)
```

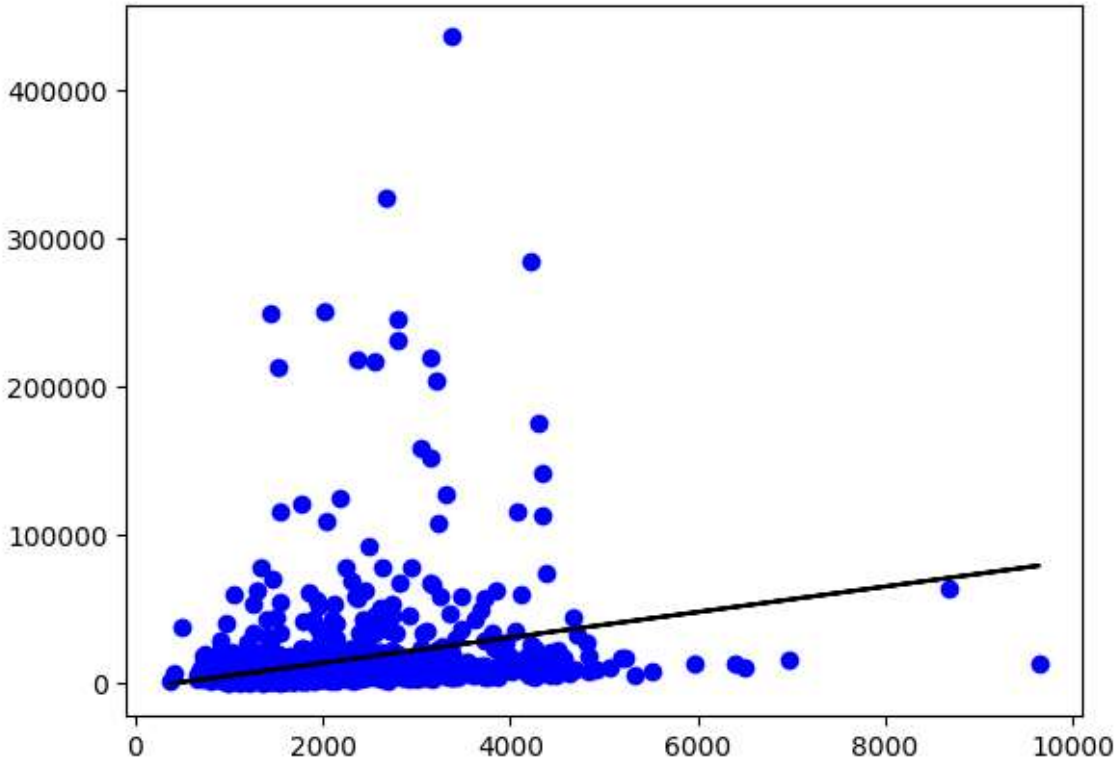
In [12]:

```
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
reg=LinearRegression()
reg.fit(X_train,y_train)
print(reg.score(X_test,y_test))
```

```
0.021139310760844854
```

In [13]:

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

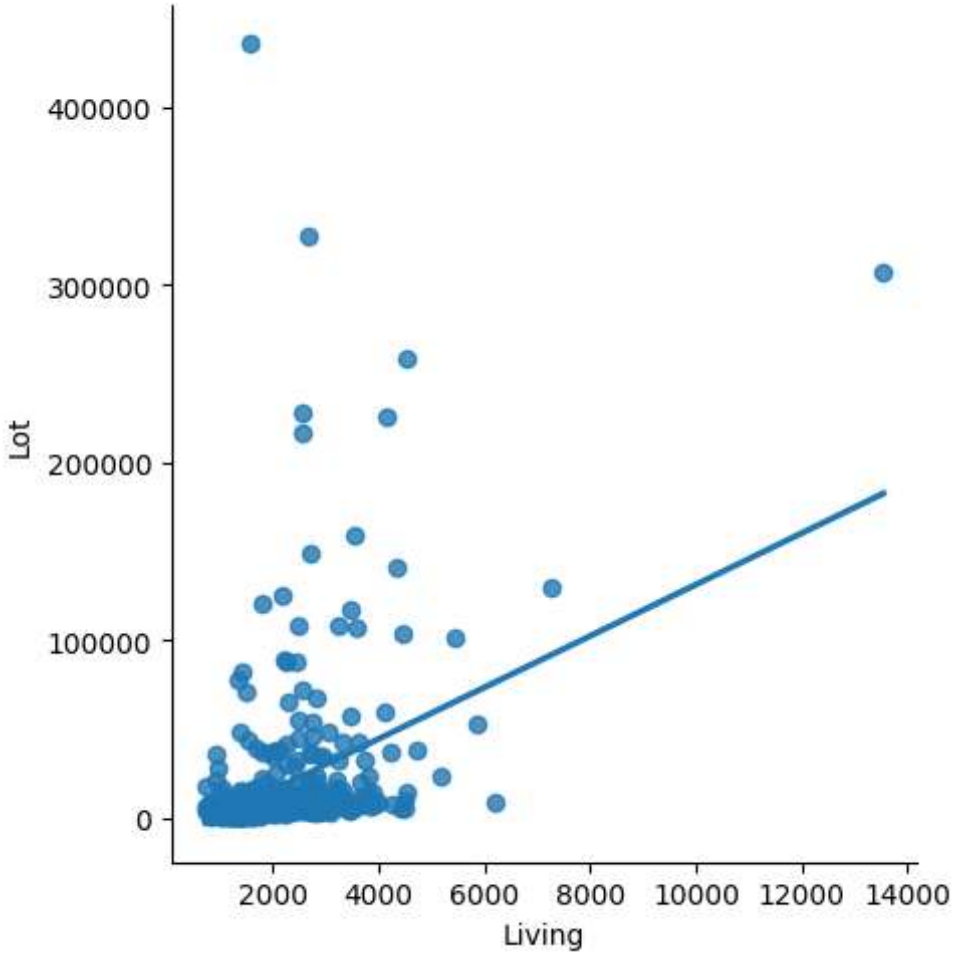


In [14]:

```
ds500=ds[:][:500]
sns.lmplot(x='Living',y='Lot',data=ds500,order=1,ci=None)
```

Out[14]:

```
<seaborn.axisgrid.FacetGrid at 0x196b65706d0>
```



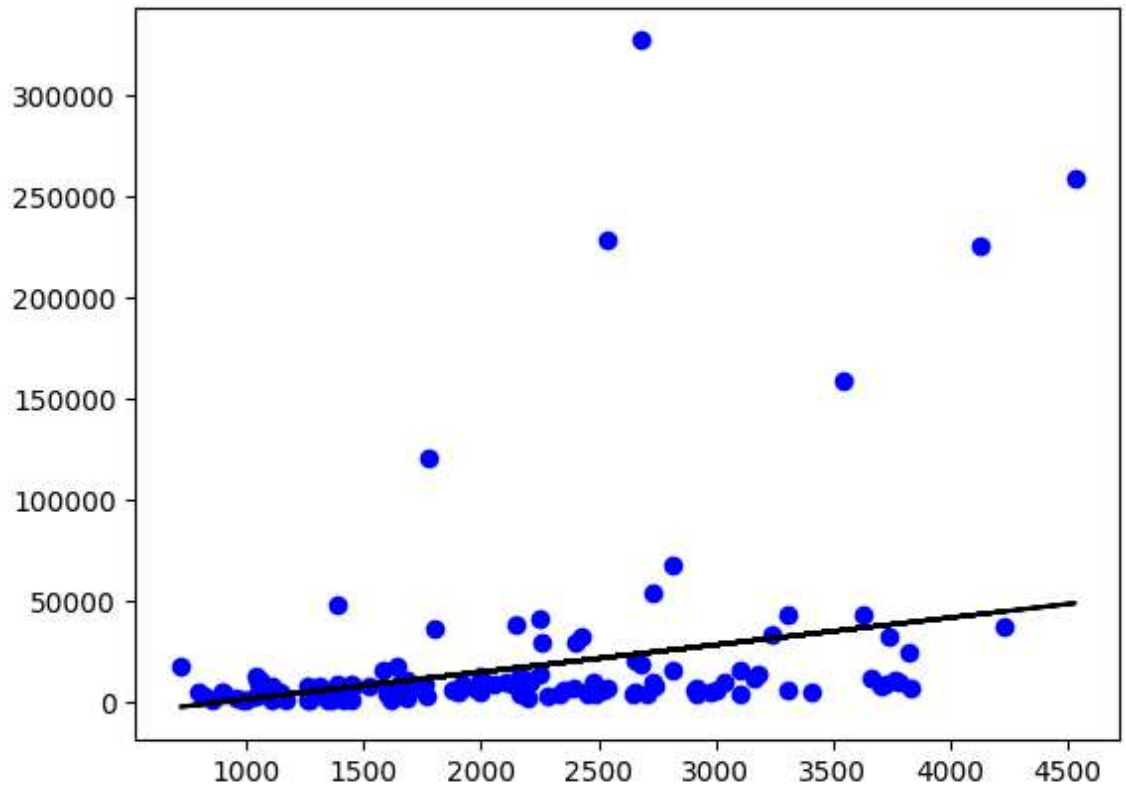
In [15]:

```
ds500.fillna(method='ffill',inplace=True)
X=np.array(ds500['Living']).reshape(-1,1)
y=np.array(ds500['Lot']).reshape(-1,1)
ds500.dropna(inplace=True)
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25)
reg=LinearRegression()
reg.fit(X_train,y_train)
print("Regression:",reg.score(X_test,y_test))
y_pred=reg.predict(X_test)
plt.scatter(X_test,y_test,color='b')
plt.plot(X_test,y_pred,color='k')
plt.show
```

```
Regression: 0.10313515292212616
```

Out[15]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [16]:

```
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:  0.10313515292212616
```

#conclusion:Linear regression is the best fit for the model