In [1]:

```
#Step-1 :Importing all the required libraries
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]:

```
#Step-2: Reading the Dataset
df=pd.read_csv(r"C:\Users\thara\Downloads\data.csv")
df
```

Out[3]:

	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 [4]:

```
df = df[['sqft_living','sqft_lot']]
#Taking only selected two attributes from dataset
df.columns = ['sqft_living','sqft_lot']
```

In [5]:

```
df.head()
```

Out[5]:

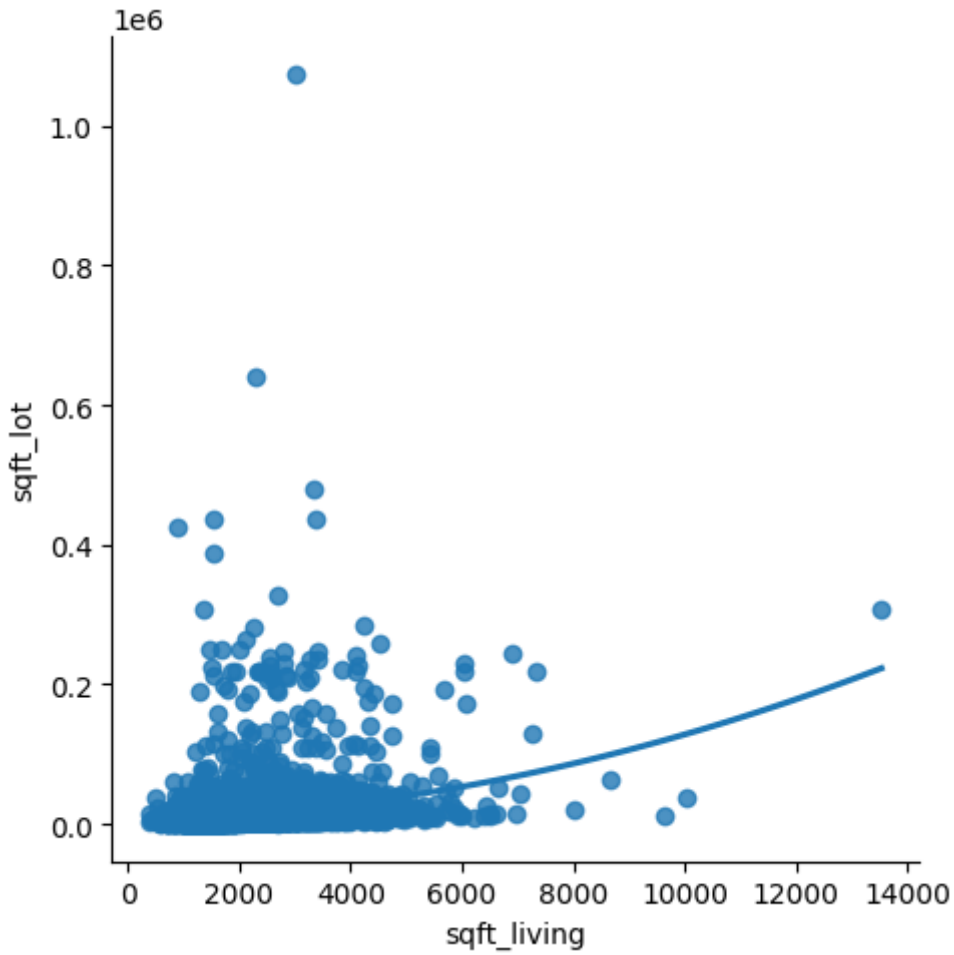
	sqft_living	sqft_lot
0	1340	7912
1	3650	9050
2	1930	11947
3	2000	8030
4	1940	10500

In [7]:

```
#Step-3: Exploring the Data Scatter - plotting the data scatter
sns.lmplot(x="sqft_living",y="sqft_lot",data=df,order=2,ci=None)
```

Out[7]:

```
<seaborn.axisgrid.FacetGrid at 0x20266573dd0>
```



In [8]:

```
#step 4: Dta cleaning - eliminating NON and missing input numbers
df.fillna(method='ffill',inplace=True)
```

```
C:\Users\smb06\AppData\Local\Temp\ipykernel_8724\477090421.py:2: SettingWi
thCopyWarning:
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)
  df.fillna(method='ffill',inplace=True)
```

In [11]:

```
#step 5:Training our model
X=np.array(df['sqft_living']).reshape(-1, 1)
y=np.array(df['sqft_lot']).reshape(-1, 1)
#separating the data into independent and dependent variables and convert
#how each dataframe contains only one coloumn
```

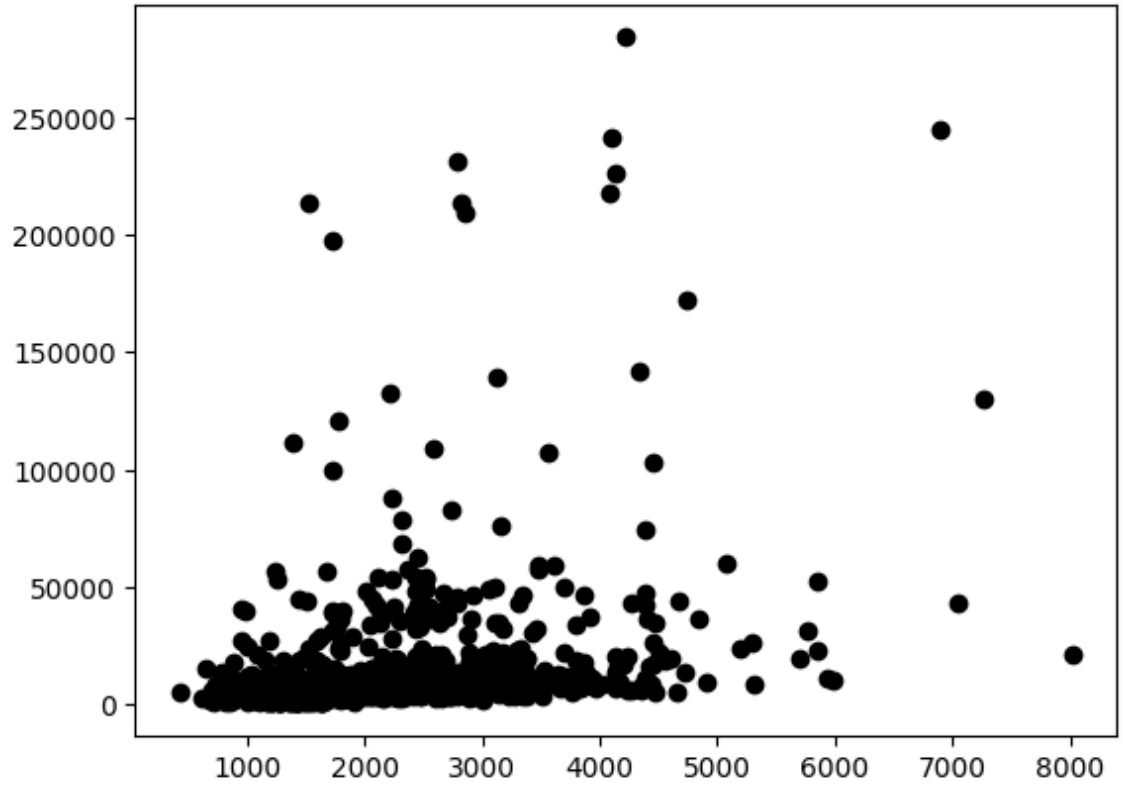
In [12]:

```
X_train,X_test,y_train,y_test = train_test_split(X, y, test_size = 0.25)
# Splitting the data into training data and test data
regr = LinearRegression()
regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

```
0.08765408422912568
```

In [13]:

```
#step-6: Exploring Our Results
y_pred = regr.predict(X_test)
plt.scatter(X_test, y_test, color = 'k')
plt.show()
# Data scatter of predicted values
```

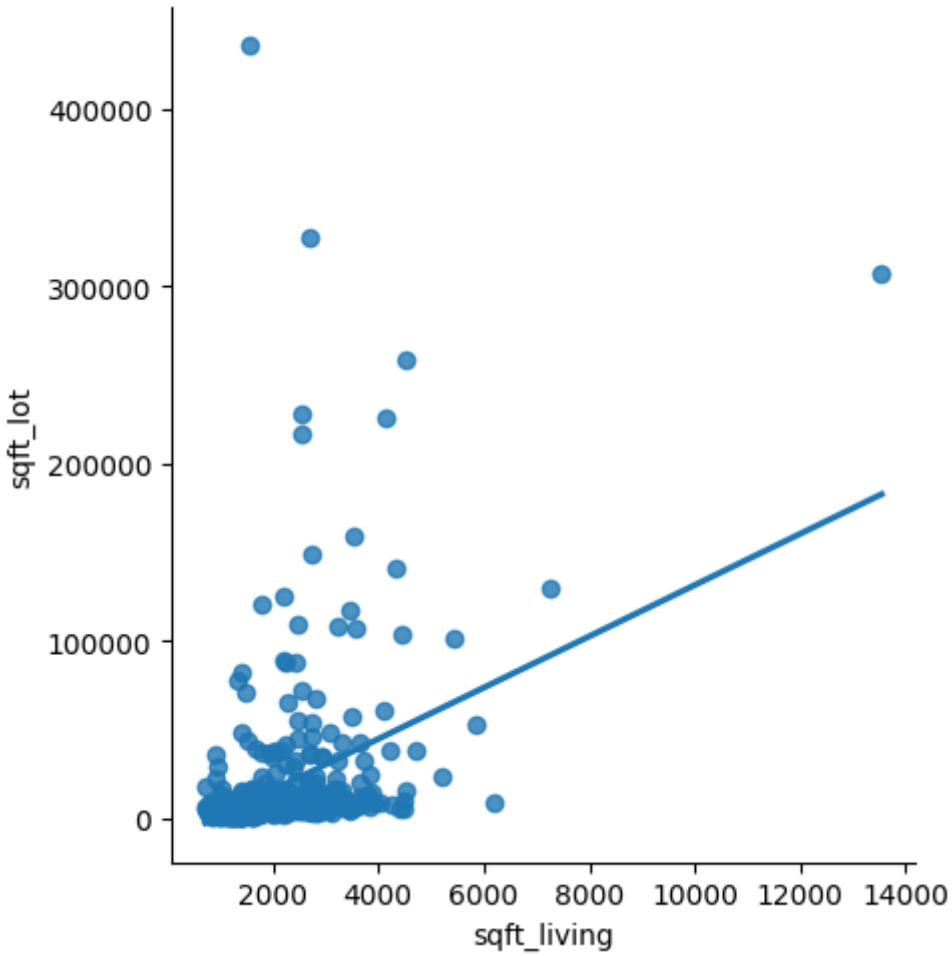


In [14]:

```
# Step-7: Working with a smaller Dataset
df500 = df[:][:500]
# Selecting the 1st 500 rows of the data
sns.lmplot(x = "sqft_living", y = "sqft_lot", data = df500, order = 1, ci = None)
```

Out[14]:

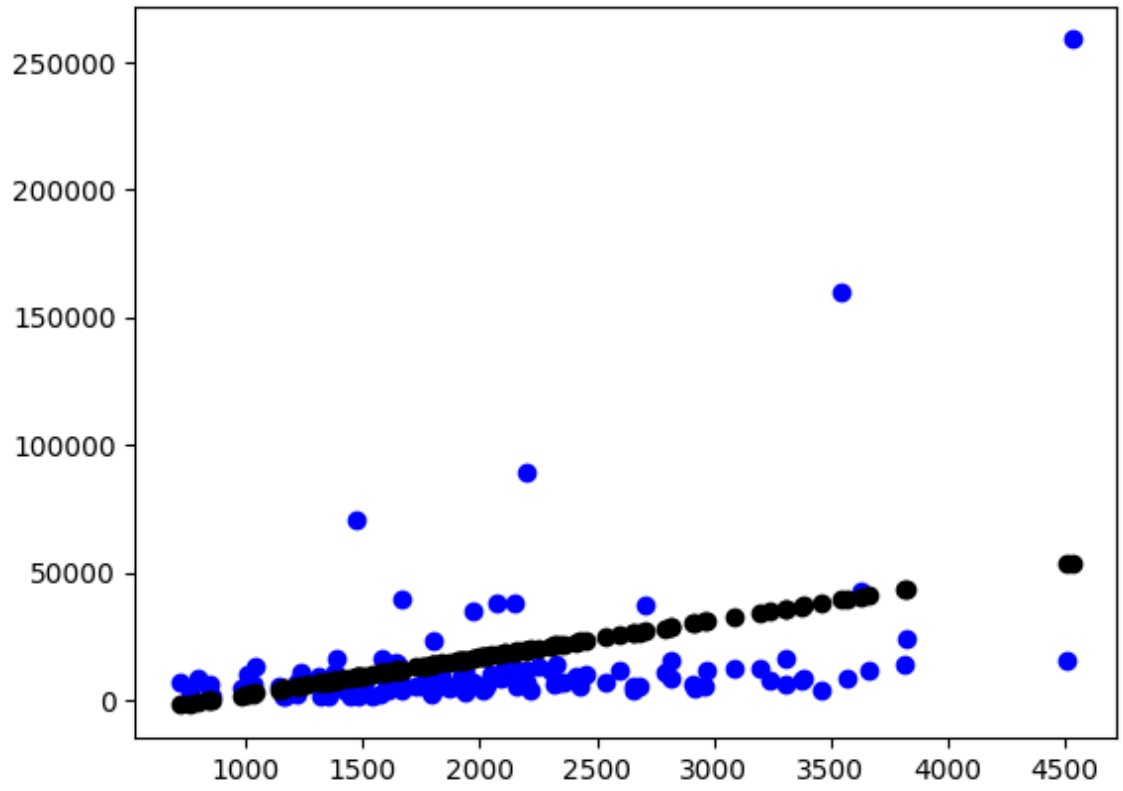
```
<seaborn.axisgrid.FacetGrid at 0x20268cb7390>
```



In [15]:

```
df500.fillna(method = 'ffill', inplace = True)
X = np.array(df500['sqft_living']).reshape(-1, 1)
y = np.array(df500['sqft_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: 0.12180914153519584
```



In [16]:

```
#Step-8: Evaluation of model

from sklearn.linear_model import LinearRegression

from sklearn.metrics import r2_score

#Train the model

model = LinearRegression()

model.fit(X_train, y_train)

#Evaluating the model on the test set

y_pred = model.predict(X_test)

r2 = r2_score(y_test, y_pred)

print("R2 score:",r2)
```

```
R2 score: 0.12180914153519584
```

In [ ]:

```
#step9:conclusion
Dataset we have taken is poor for Linear Model,but w
```