

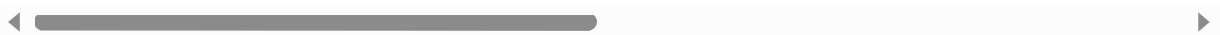
```
In [1]: import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing, svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

```
In [2]: df=pd.read_csv(r"C:\Users\Welcome\Downloads\data.csv")
df
```

```
Out[2]:
```

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

4600 rows × 18 columns



```
In [5]: df=df[['sqft_living','yr_built']]
df.columns=['sqft','yr']
```

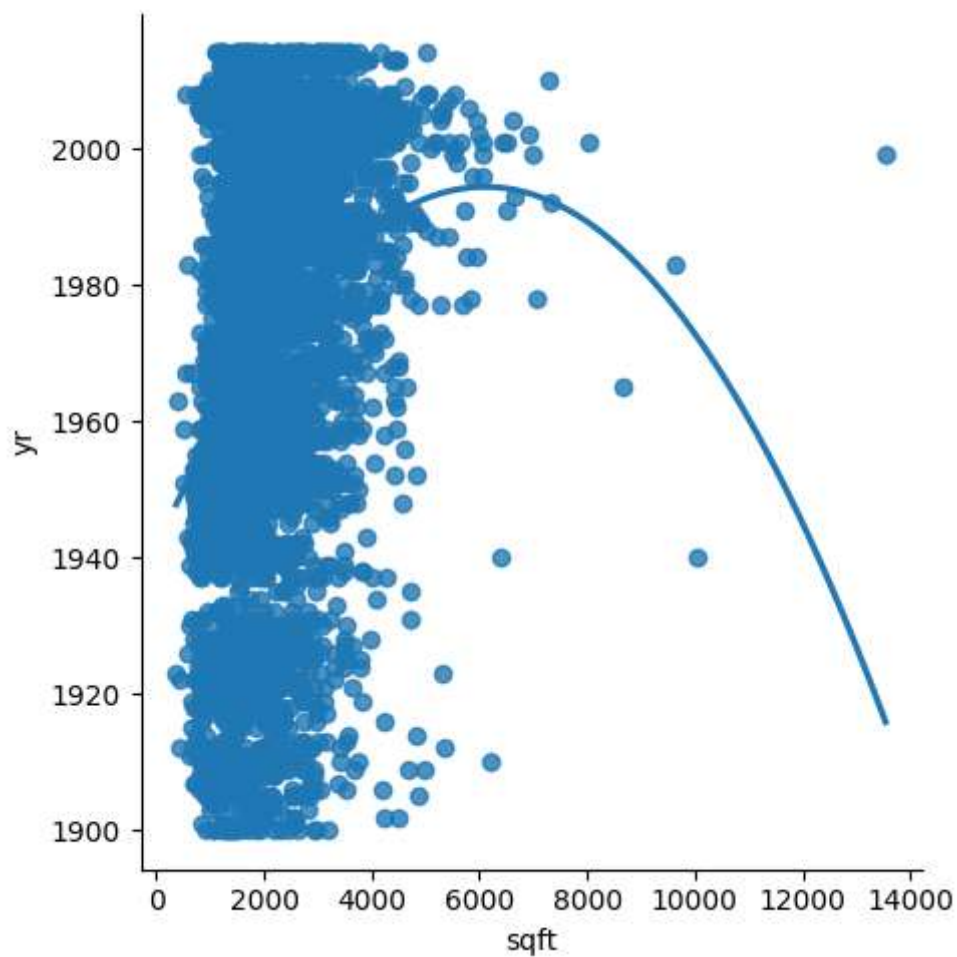
```
In [6]: df.head()
```

```
Out[6]:
```

	sqft	yr
0	1340	1955
1	3650	1921
2	1930	1966
3	2000	1963
4	1940	1976

```
In [7]: sns.lmplot(x="sqft",y="yr",data=df,order=2,ci=None)
```

```
Out[7]: <seaborn.axisgrid.FacetGrid at 0x2dacd337130>
```



In [8]: `df.describe()`

Out[8]:

	sqft	yr
count	4600.000000	4600.000000
mean	2139.346957	1970.786304
std	963.206916	29.731848
min	370.000000	1900.000000
25%	1460.000000	1951.000000
50%	1980.000000	1976.000000
75%	2620.000000	1997.000000
max	13540.000000	2014.000000

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

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

In [10]: `df.fillna(method='ffill',inplace=True)`

C:\Users\Welcome\AppData\Local\Temp\ipykernel_7716\4116506308.py:1: SettingWithCopyWarning:
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/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]: `x=np.array(df['sqft']).reshape(-1,1)`
`y=np.array(df['yr']).reshape(-1,1)`

```
In [12]: df.dropna(inplace=True)
```

C:\Users\Welcome\AppData\Local\Temp\ipykernel_7716\1379821321.py:1: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame

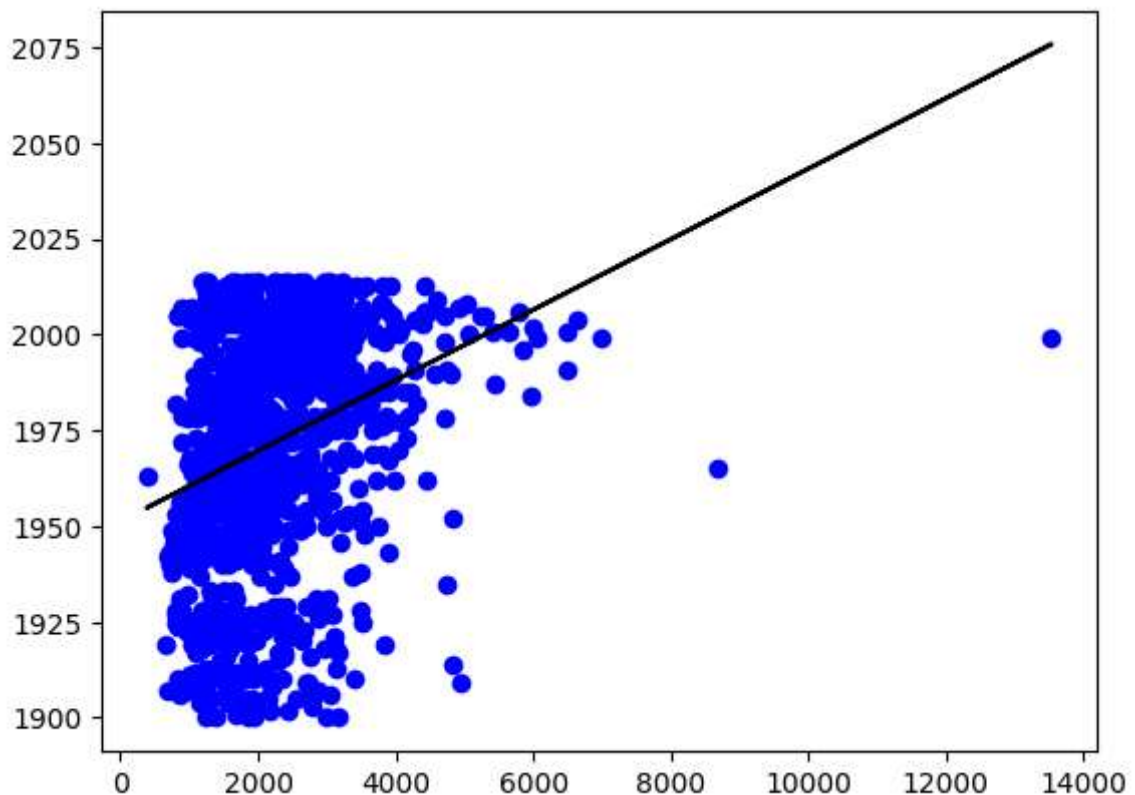
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

```
df.dropna(inplace=True)
```

```
In [13]: 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))
```

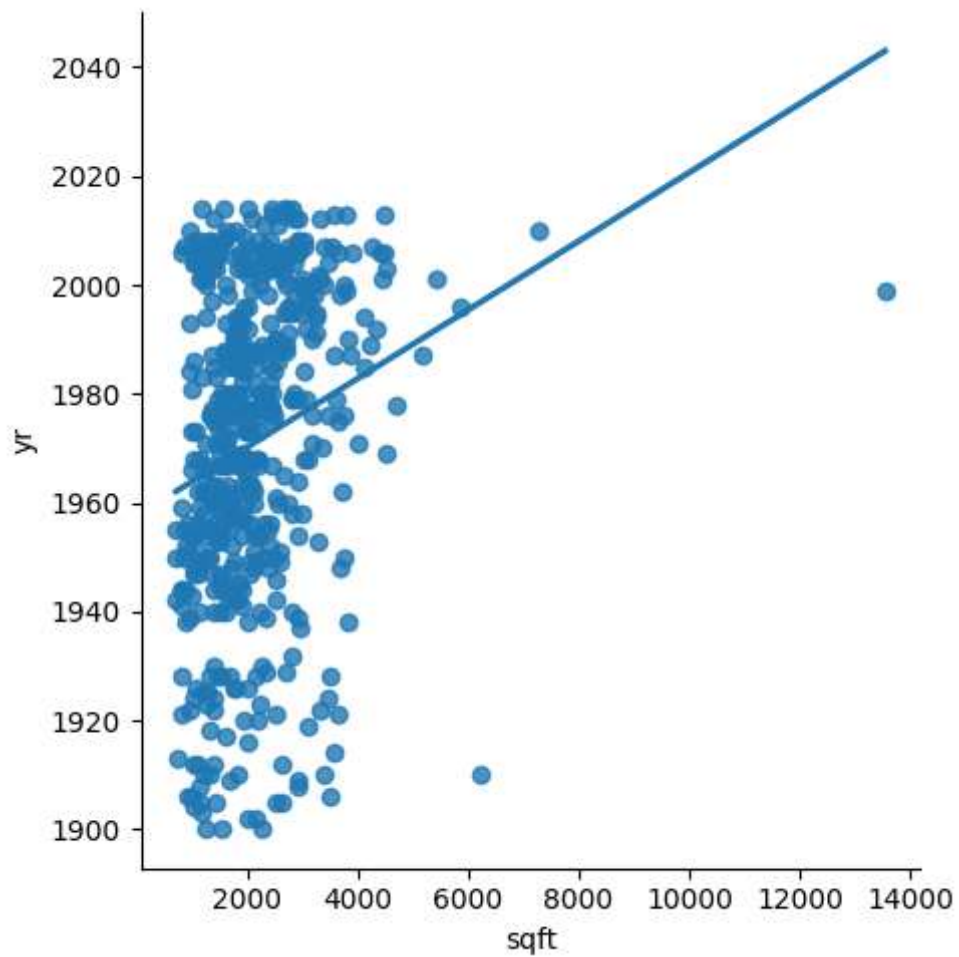
0.07157210989282503

```
In [14]: y_pred=reg.predict(X_test)
plt.scatter(X_test,y_test,color='b')
plt.plot(X_test,y_pred,color='k')
plt.show()
```



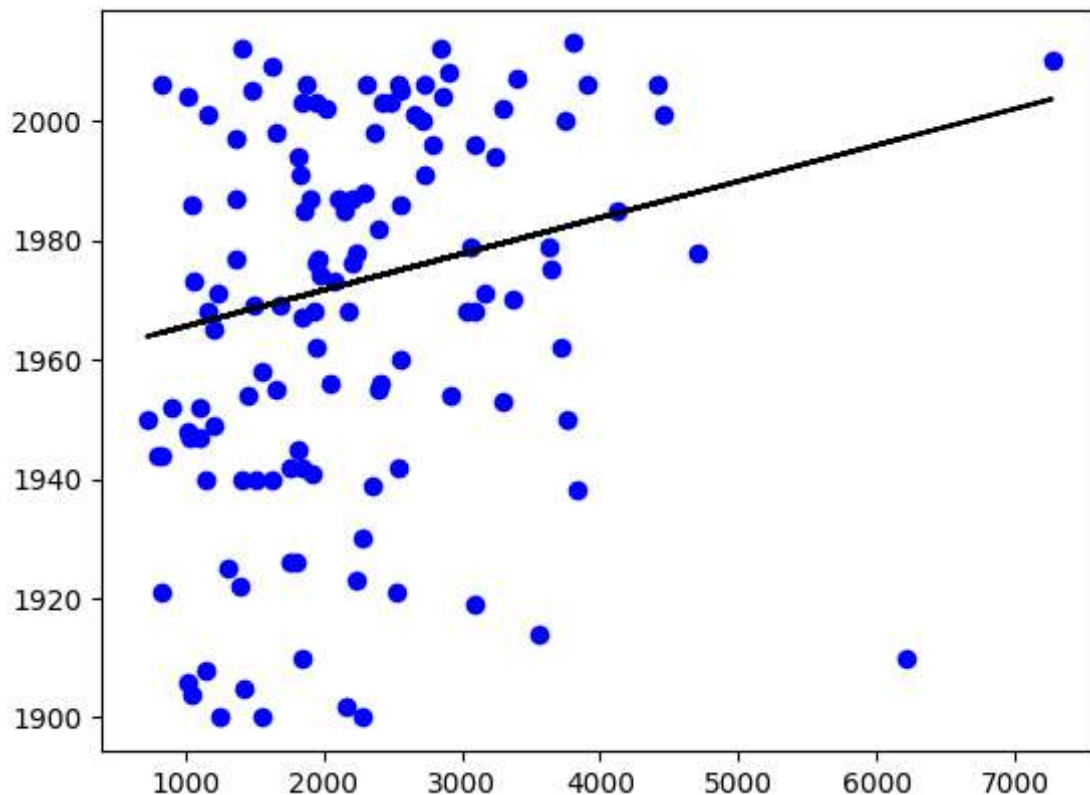
```
In [15]: df500=df[:][:500]  
sns.lmplot(x="sqft",y="yr",data=df500,order=1,ci=None)
```

```
Out[15]: <seaborn.axisgrid.FacetGrid at 0x2dacd4565f0>
```



```
In [16]: df500.fillna(method='ffill',inplace=True)
X=np.array(df500['sqft']).reshape(-1,1)
y=np.array(df500['yr']).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: 0.029020968600797725



```
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 score:",r2)
```

R2 score: 0.029020968600797725

In []:

