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\HP\Downloads\Bengaluru_House_Data.csv")
df
```

Out[2]:

	area_type	availability	location	size	society	total_sqft	bath	balcony	
0	Super built-up Area	19-Dec	Electronic City Phase II	2 BHK	Coomee	1056	2.0	1.0	
1	Plot Area	Ready To Move	Chikka Tirupathi	4 Bedroom	Theanmp	2600	5.0	3.0	1
2	Built-up Area	Ready To Move	Uttarahalli	3 BHK	NaN	1440	2.0	3.0	
3	Super built-up Area	Ready To Move	Lingadheeranahalli	3 BHK	Soiewre	1521	3.0	1.0	
4	Super built-up Area	Ready To Move	Kothanur	2 BHK	NaN	1200	2.0	1.0	
...	...	...	...	...	...	...	...	...	
13315	Built-up Area	Ready To Move	Whitefield	5 Bedroom	ArsiaEx	3453	4.0	0.0	2
13316	Super built-up Area	Ready To Move	Richards Town	4 BHK	NaN	3600	5.0	NaN	4
13317	Built-up Area	Ready To Move	Raja Rajeshwari Nagar	2 BHK	Mahla T	1141	2.0	1.0	
13318	Super built-up Area	18-Jun	Padmanabhanagar	4 BHK	SollyCl	4689	4.0	1.0	4
13319	Super built-up Area	Ready To Move	Doddathoguru	1 BHK	NaN	550	1.0	1.0	

13320 rows × 9 columns

In [3]:
```
df=df[['bath','balcony']]
df.columns=['bat','bal']
```

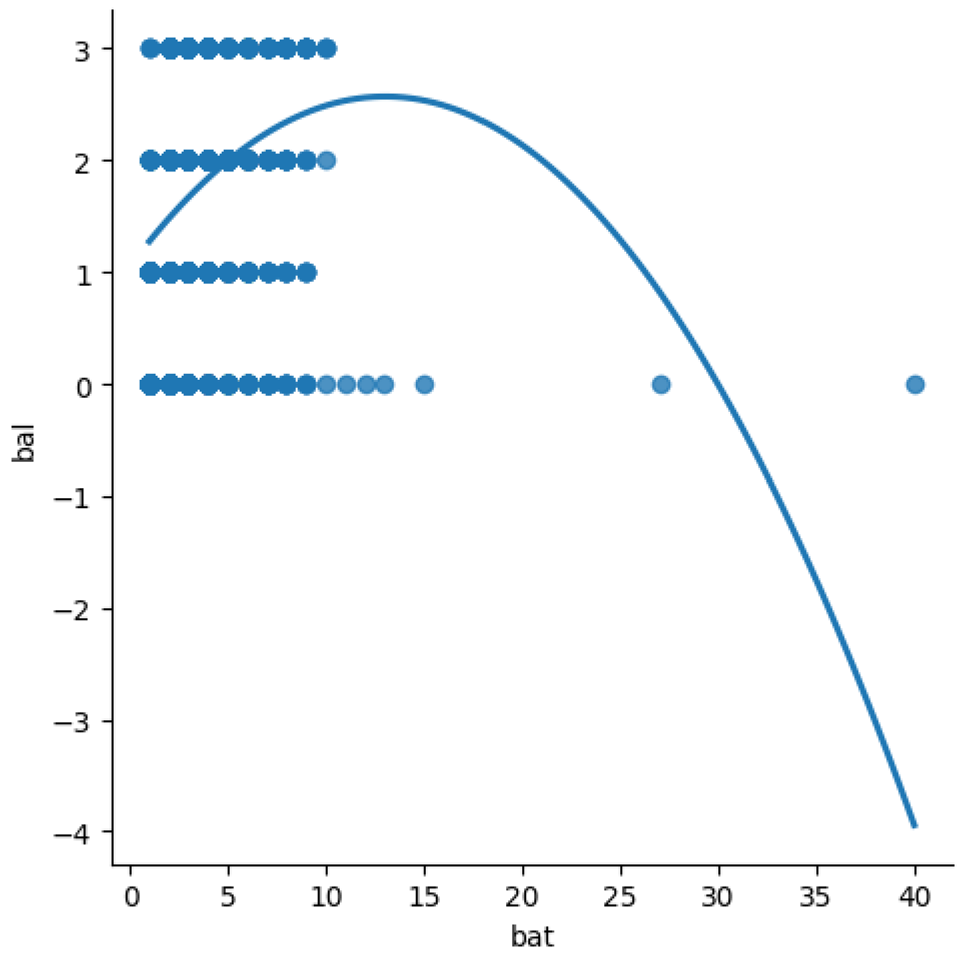
In [4]:
```
df.head(10)
```

Out[4]:

	bat	bal
0	2.0	1.0
1	5.0	3.0
2	2.0	3.0
3	3.0	1.0
4	2.0	1.0
5	2.0	1.0
6	4.0	NaN
7	4.0	NaN
8	3.0	1.0
9	6.0	NaN

In [6]:
```
sns.lmplot(x="bat",y="bal",data=df,order=2,ci=None)
```

Out[6]:
```
<seaborn.axisgrid.FacetGrid at 0x25404251780>
```



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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13320 entries, 0 to 13319
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   bat     13247 non-null  float64
 1   bal     12711 non-null  float64
dtypes: float64(2)
memory usage: 208.2 KB
```

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

```
C:\Users\HP\AppData\Local\Temp\ipykernel_9672\4116506308.py:1: SettingWithCo
pyWarning:
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://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  df.fillna(method='ffill',inplace=True)
```

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

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

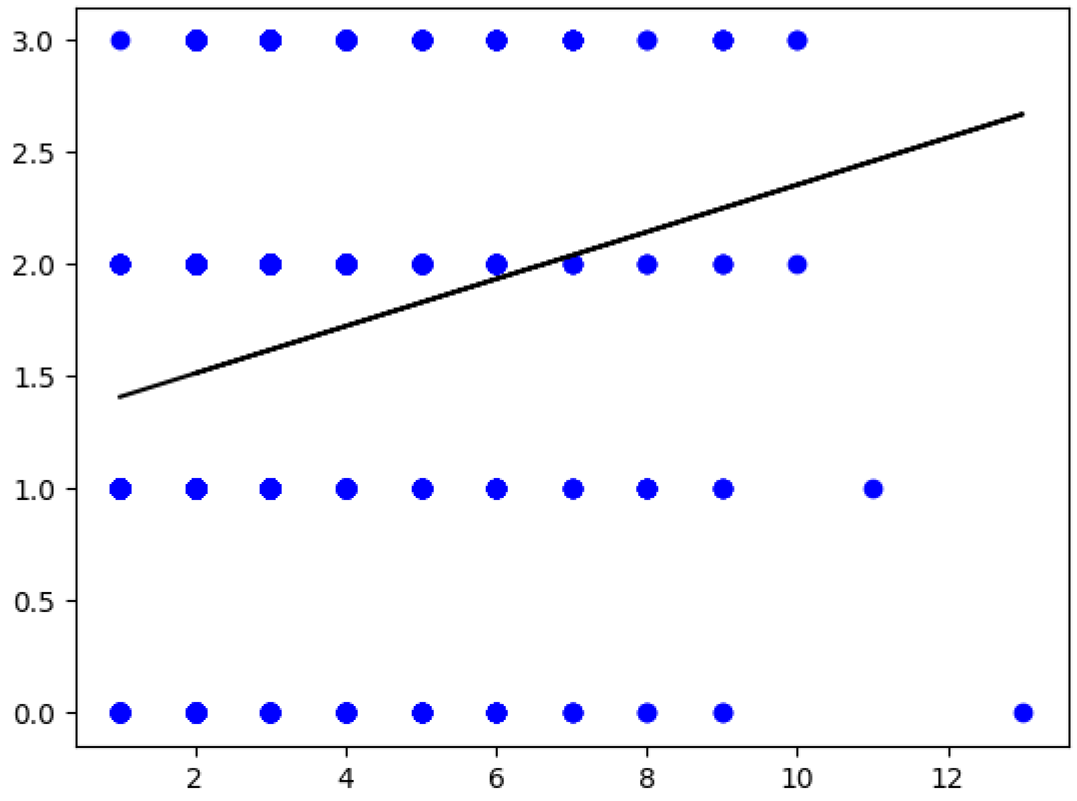
```
C:\Users\HP\AppData\Local\Temp\ipykernel_9672\1379821321.py:1: SettingWithCo
pyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  df.dropna(inplace=True)
```

In [11]:
```
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))
```

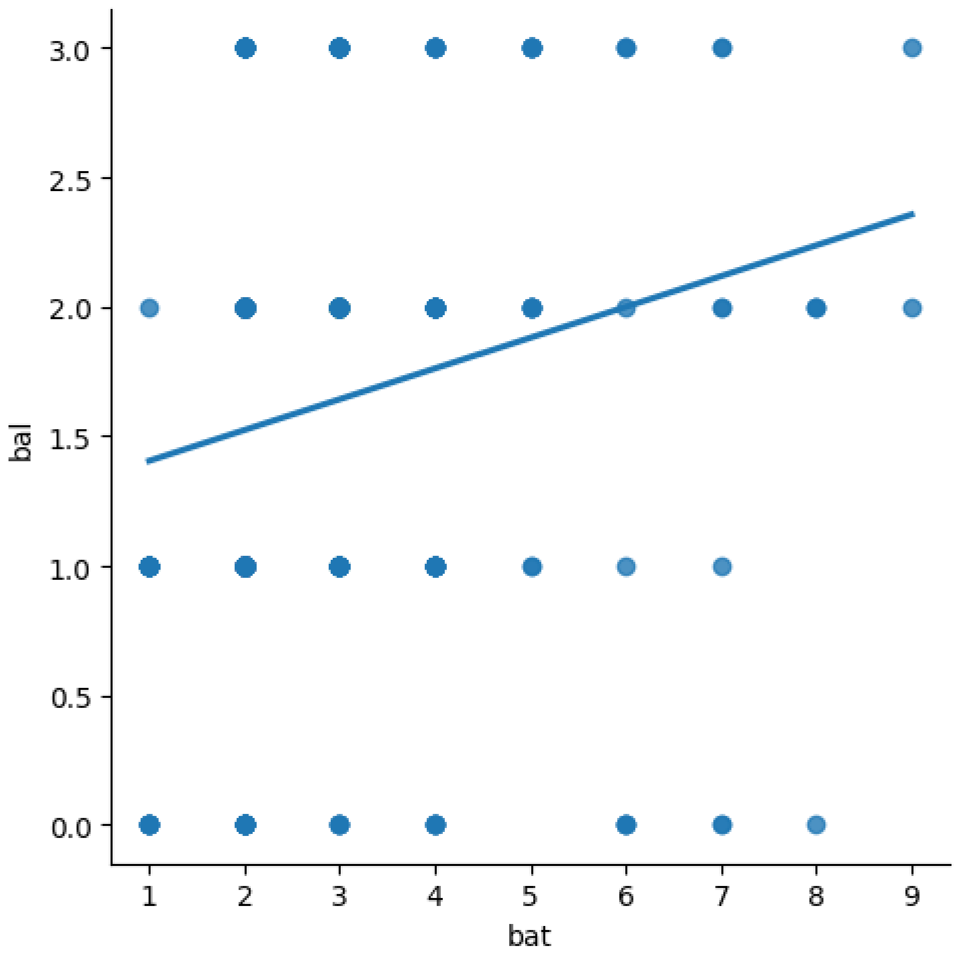
```
0.0384216404560247
```

In [12]:
```
y_pred=regr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```



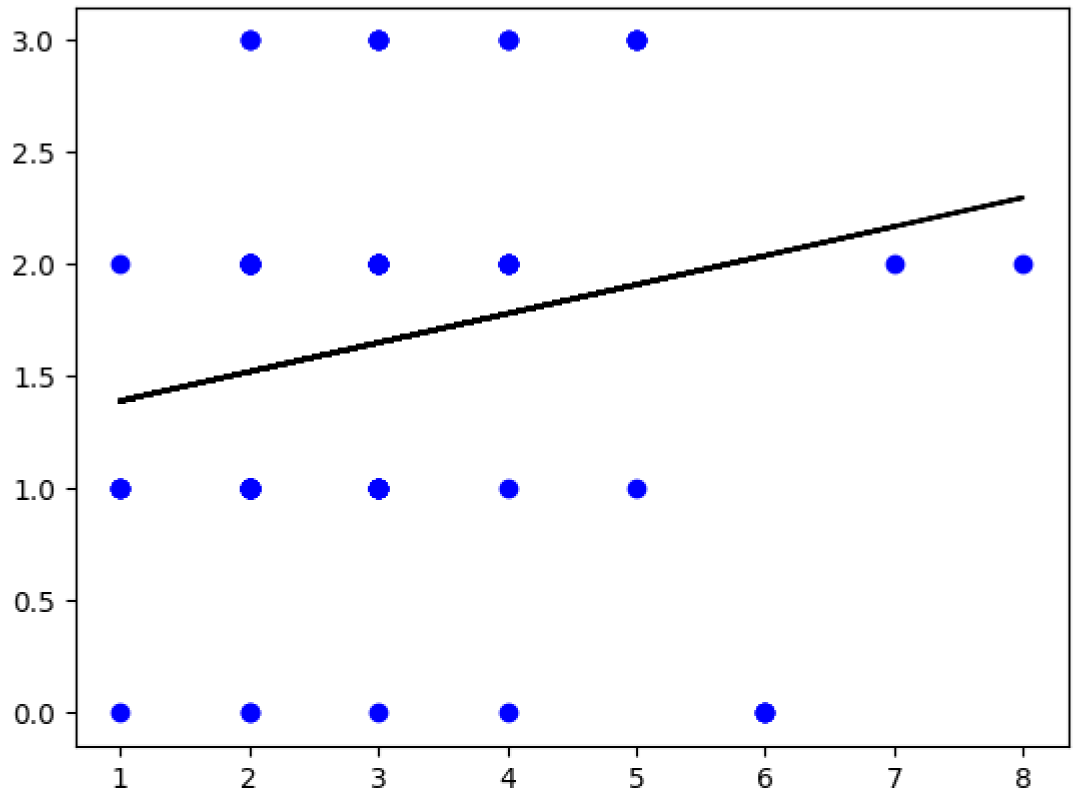
In [13]:
```
df500=df[:][:500]
sns.lmplot(x="bat",y="bal",data=df500,order=1,ci=None)
```

Out[13]:
```
<seaborn.axisgrid.FacetGrid at 0x25407096530>
```



In [14]:
```
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['bat']).reshape(-1,1)
y=np.array(df500['bal']).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: 0.013871747984256055
```



In [15]:
```
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: 0.013871747984256055
```

# conclusion

```
       Data set we have taken is poor for linear model but with the smaller
data works well with linear model.
```