In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
```

In [2]:

```
data=pd.read_csv('C:/Users/KORRA SRINU/Downloads/Salary.csv')
```

In [3]:

```
print(data)
```

```
    YearsExperience  Salary
0               1.1   39343
1               1.3   46205
2               1.5   37731
3               2.0   43525
4               2.2   39891
5               2.9   56642
6               3.0   60150
7               3.2   54445
8               3.2   64445
9               3.7   57189
10              3.9   63218
11              4.0   55794
12              4.0   56957
13              4.1   57081
14              4.5   61111
15              4.9   67938
16              5.1   66029
17              5.3   83088
18              5.9   81363
19              6.0   93940
20              6.8   91738
21              7.1   98273
22              7.9  101302
23              8.2  113812
24              8.7  109431
25              9.0  105582
26              9.5  116969
27              9.6  112635
28             10.3  122391
29             10.5  121872
30             11.2  127345
31             11.5  126756
32             12.3  128765
33             12.9  135675
34             13.5  139465
```

In [4]:

```
data.head()
```

Out[4]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891

In [5]:

```
data.head(10)
```

Out[5]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891
5	2.9	56642
6	3.0	60150
7	3.2	54445

8	3.2	64445
9	3.7	57189

In [6]:

```
data.tail(6)
```

Out[6]:

	YearsExperience	Salary
29	10.5	121872
30	11.2	127345
31	11.5	126756
32	12.3	128765
33	12.9	135675
34	13.5	139465

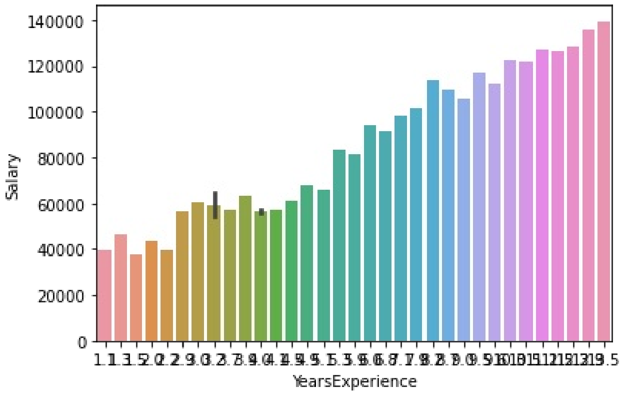
In [8]:

```
sns.barplot(data['YearsExperience'],data['Salary'])
```

```
C:\Users\KORRA SRINU\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following var
iables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing o
ther arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[8]:

```
<AxesSubplot:xlabel='YearsExperience', ylabel='Salary'>
```



In [9]:

```
#satatistical calculation
data.describe()
```

Out[9]:

	YearsExperience	Salary
count	35.000000	35.000000
mean	6.308571	83945.600000
std	3.618610	32162.673003
min	1.100000	37731.000000
25%	3.450000	57019.000000
50%	5.300000	81363.000000
75%	9.250000	113223.500000
max	13.500000	139465.000000

In [10]:

```
#for missing value
data.isnull()
```

Out[10]:

	YearsExperience	Salary
0	False	False
1	False	False
2	False	False
3	False	False

4	False	False
5	False	False
6	False	False
7	False	False
8	False	False
9	False	False
10	False	False
11	False	False
12	False	False
13	False	False
14	False	False
15	False	False
16	False	False
17	False	False
18	False	False
19	False	False
20	False	False
21	False	False
22	False	False
23	False	False
24	False	False
25	False	False
26	False	False
27	False	False
28	False	False
29	False	False
30	False	False
31	False	False
32	False	False
33	False	False
34	False	False

In [12]:

```
data.isnull().any()
#here it shows there is no missing values as boolean expression
```

Out[12]:

```
YearsExperience    False
Salary             False
dtype: bool
```

In [13]:

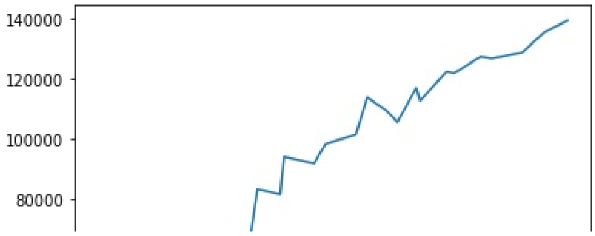
```
data.isnull().sum()
```

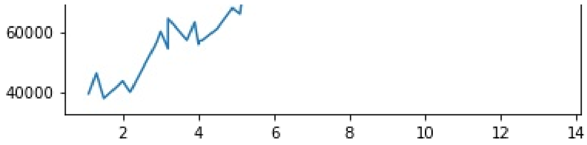
Out[13]:

```
YearsExperience    0
Salary             0
dtype: int64
```

In [20]:

```
#if we want to observe the line graph
plt.plot(data['YearsExperience'],data['Salary'])
plt.show()
```





In [29]:

```
from sklearn.linear_model import LinearRegression
```

In [30]:

```
L=LinearRegression()
```

In [31]:

```
x=data.drop('Salary',axis=1)
```

In [32]:

```
y=data['Salary']
```

In [33]:

```
print(x)
```

```
    YearsExperience
0               1.1
1               1.3
2               1.5
3               2.0
4               2.2
5               2.9
6               3.0
7               3.2
8               3.2
9               3.7
10              3.9
11              4.0
12              4.0
13              4.1
14              4.5
15              4.9
16              5.1
17              5.3
18              5.9
19              6.0
20              6.8
21              7.1
22              7.9
23              8.2
24              8.7
25              9.0
26              9.5
27              9.6
28             10.3
29             10.5
30             11.2
31             11.5
32             12.3
33             12.9
34             13.5
```

In [34]:

```
print(y)
```

```
0      39343
1      46205
2      37731
3      43525
4      39891
5      56642
6      60150
7      54445
8      64445
9      57189
10     63218
11     55794
12     56957
13     57081
14     61111
15     67938
16     66029
17     83088
```

```
18     81363
19     93940
20     91738
21     98273
22    101302
23    113812
24    109431
25    105582
26    116969
27    112635
28    122391
29    121872
30    127345
31    126756
32    128765
33    135675
34    139465
Name: Salary, dtype: int64
```

In [108…

```python
from sklearn.model_selection import train_test_split
```

In [36]:

```python
x_train,x_test,y_train,y_test=train_test_split(x,y)
```

In [37]:

```python
L.fit(x_train,y_train)
```

Out[37]:

```
LinearRegression()
```

In [109…

```python
y_predict=L.predict(x_test)
```

In [110…

```python
y_predict
```

Out[110…

```
array([ 64660.48448749,  68153.26090631, 146740.73032988, 100461.44278044,
        90856.30762868,  40211.04955571, 141501.56570164,  38464.6613463 ,
        41957.43776513])
```

In [113…

```python
y_predict=L.predict([[20]])
y_predict
```

Out[113…

```
array([203498.34713578])
```

In [115…

```python
#accuracy
print(L.score(x_test,y_test))
```

```
0.9644958697360632
```

In [55]:

```python
#without using the train and test_test_split


X=np.array(x)
Y=np.array(y)
```

In [56]:

```python
print(X)
```

```
[[ 1.1]
 [ 1.3]
 [ 1.5]
 [ 2. ]
 [ 2.2]
 [ 2.9]
 [ 3. ]
 [ 3.2]
 [ 3.2]
 [ 3.7]
 [ 3.9]
 [ 4. ]
 [ 4. ]
 [ 4.1]
 [ 4.5]
 [ 4.9]
 [ 5.1]
 [ 5.3]
 [ 5.9]
 [ 6. ]
 [ 6.8]
 [ 7.1]
 [ 7.9]
 [ 8.2]
 [ 8.7]
 [ 9. ]
 [ 9.5]
 [ 9.6]
 [10.3]
 [10.5]
 [11.2]
 [11.5]
 [12.3]
 [12.9]
 [13.5]]
```

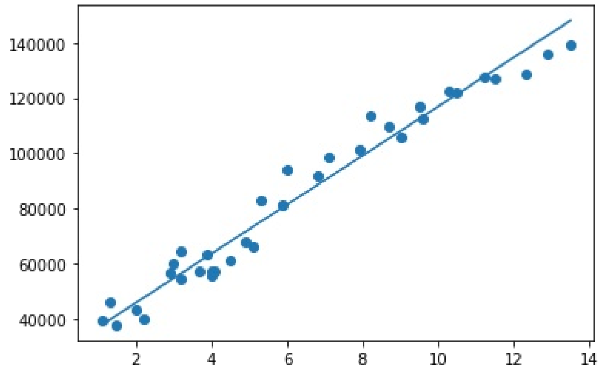
In [57]:

```
print(Y)
```

```
[ 39343  46205  37731  43525  39891  56642  60150  54445  64445  57189
  63218  55794  56957  57081  61111  67938  66029  83088  81363  93940
  91738  98273 101302 113812 109431 105582 116969 112635 122391 121872
 127345 126756 128765 135675 139465]
```

In [90]:

```
plt.scatter(X,Y)
plt.plot(X,L.predict(X))
plt.show()
```



In [91]:

```
L.fit(X,Y)
```

Out[91]:

```
LinearRegression()
```

In [92]:

```
Y_PREDICT=L.predict(X)
```

In [93]:

```
Y_PREDICT
```

Out[93]:

```
array([ 38464.6613463 ,  40211.04955571,  41957.43776513,  46323.40828866,
        48069.79649807,  54182.15523101,  55055.34933572,  56801.73754513,
        56801.73754513,  61167.70806866,  62914.09627808,  63787.29038278,
        63787.29038278,  64660.48448749,  68153.26090631,  71646.03732514,
        73392.42553455,  75138.81374396,  80377.9783722 ,  81251.17247691,
        88236.72531456,  90856.30762868,  97841.86046633, 100461.44278044,
       104827.41330398, 107446.99561809, 111812.96614163, 112686.16024633,
       118798.51897928, 120544.90718869, 126657.26592163, 129276.84823575,
       136262.4010734 , 141501.56570164, 146740.73032988])
```

In [101…

```
#after 20 years experience the salary will be
Y_PREDICT=L.predict([[20]])
```

In [102…

```
Y_PREDICT
```

Out[102…

```
array([203498.34713578])
```

In [104…

```
#accuracy of both salary and experiance
print(L.score(X,Y))
```

```
0.9651633106751443
```

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js