Solving simulatneous equation

In [71]:

```
# x+y+z=6 => x=1, y=1, z=1
# 2y+5z=-4 => x=0, y=2, z=5
# 2x+5y-z=27 => x=2, y=5, z=-1
```

In [72]:

```
#Import numpy and create a numpy matrix for the above equation
import numpy as np
```

In [73]:

```
X=np.matrix([[1,1,1],
             [0,2,5],
             [2,5,-1]])
print(X)
```

```
[[ 1  1  1]
 [ 0  2  5]
 [ 2  5 -1]]
```

In [74]:

```
#Create the output array and transpose to column vector for multiplication
Y=np.array([[6,-4,27]]).T
Y
```

Out[74]:

```
array([[ 6],
       [-4],
       [27]])
```

In [75]:

```
#Find the inverse of X
Xinv=np.linalg.pinv(X)
Xinv
```

Out[75]:

```
matrix([[ 1.28571429, -0.28571429, -0.14285714],
        [-0.47619048,  0.14285714,  0.23809524],
        [ 0.19047619,  0.14285714, -0.0952381 ]])
```

In [76]:

```
#Find the output vector [x,y,z]
O=Xinv*Y
O.T
```

Out[76]:

```
matrix([[ 5.,  3., -2.]])
```

Solution

x=5, y=3, z=-2

Employee Salary Regression Plot

In [77]:

```
#import required libs
import pandas as pd
import seaborn as sb
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
%matplotlib inline
```

In [78]:

```
#Read training data
EmpDF=pd.read_csv('Salary_Data.csv')
```

In [79]:

```
#Check the input DataFrame
EmpDF.head()
```

Out[79]:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

In [80]:

```
#Get the slope and intercept values
x=EmpDF['YearsExperience']
y=EmpDF['Salary']
slope, intercept, r_value, p_value, std_err = stats.linregress(x,y)
slope, intercept, r_value**2
```

Out[80]:

```
(9449.962321455077, 25792.20019866869, 0.9569566641435084)
```

Solution

Slope/Coefficient theta1 = 9449.96

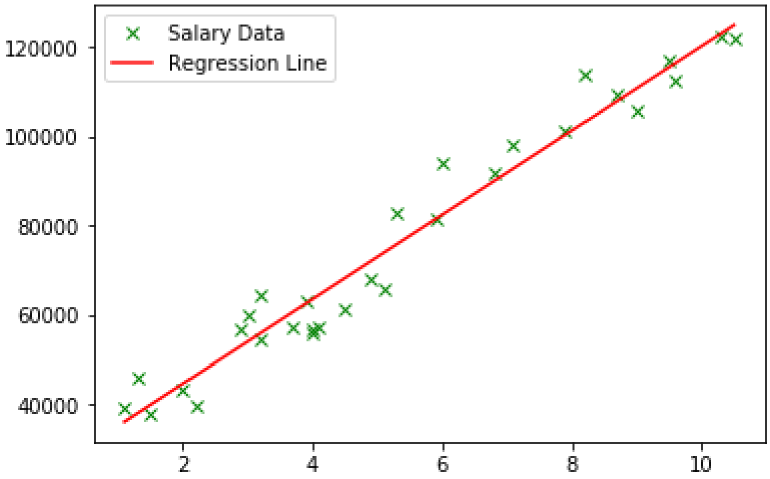
Intercept/Coefficient theta0 = 25792.2

Mean Error Squared term = 0.9569

Sample plot for the above model

In [81]:

```
plt.plot(x, y, 'x', color='g', label='Salary Data')
plt.plot(x, intercept + slope*x, 'r', label='Regression Line')
plt.legend()
plt.show()
```

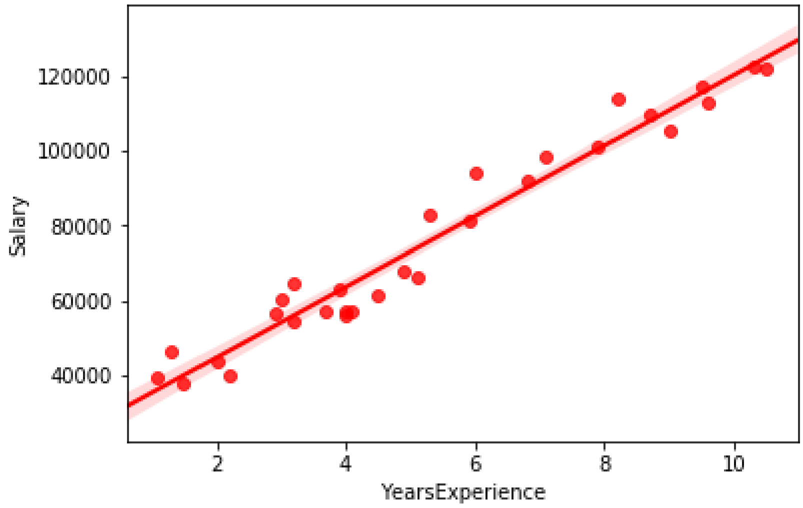


In [84]:

```
#Experiment the same with Seaborn
```

In [85]:

```
#Plot a regression line with 95% confidence using seaborn
p=sb.regplot(x=('YearsExperience'),y=('Salary'),data=EmpDF,color='red',fit_reg='true',ci=95,label=p.text)
slope, intercept, r_value, p_value, std_err = stats.linregress(x=p.get_lines()[0].get_xdata(),y=p.get_lines()[0].get_ydata())
```



In [86]:

```
slope, intercept, r_value, p_value, std_err
```

Out[86]:

```
(9449.962321455076, 25792.200198668696, 1.0, 0.0, 0.0)
```