In [1]:
```
# Import the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

```
/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146: UserWarning: A NumPy v
ersion >=1.16.5 and <1.23.0 is required for this version of SciPy (detected version
1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [2]:
```
# Load the data set using the read function
salary=pd.read_csv('/kaggle/input/salary-dataset-simple-linear-regression/Salary_datas
salary.head()
```

Out[2]:

	Unnamed: 0	YearsExperience	Salary
0	0	1.2	39344.0
1	1	1.4	46206.0
2	2	1.6	37732.0
3	3	2.1	43526.0
4	4	2.3	39892.0

In [3]:
```
# Checking the null values
salary.isna().sum()
```

Out[3]:
```
Unnamed: 0         0
YearsExperience    0
Salary             0
dtype: int64
```

In [4]:
```
# Let's  drop the unwanted columns
salary=salary.drop('Unnamed: 0',axis=1)
salary.head()
```

Out[4]:

	YearsExperience	Salary
0	1.2	39344.0
1	1.4	46206.0
2	1.6	37732.0
3	2.1	43526.0
4	2.3	39892.0

In [5]:
```
salary.columns
```

Out[5]:
```
Index(['YearsExperience', 'Salary'], dtype='object')
```

Linear Regression Model

Absolutely, let's discuss the linear regression model in the context of predicting salary based on years of experience.

In this scenario:

- (y) represents the salary (dependent variable).
- (m) is the slope of the line, which signifies how much the salary changes with a unit change in years of experience.
- (c) is the y-axis intercept of the regression line, indicating the starting salary when years of experience ((x)) is 0.
- (x) corresponds to years of experience (independent variable).

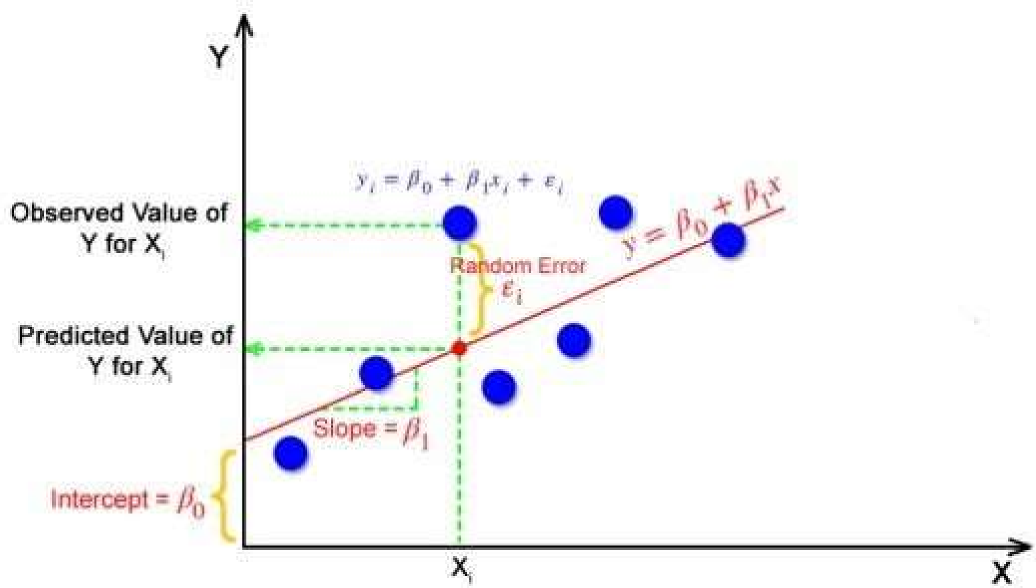
The formula for the linear regression model in this context is: [y = mx + c]

So, if we have a specific value for years of experience (let's say (x = 5.5)), we can plug it into the formula to predict the corresponding salary ((y)).

For example, if (x = 5.5): [y = m \times 5.5 + c]

Please note that (m) and (c) would be determined by fitting the linear regression model to the actual data. Once the model is trained, it can be used to make predictions like the one above.

If you have a dataset with actual years of experience and corresponding salaries, you can use libraries like `scikit-learn` to train a linear regression model and then use the model to make predictions for new values of years of experience.



In [6]:
```
# Split the data into dependent and independent variable
X=salary['YearsExperience']
y=salary['Salary']
```

In [7]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score,mean_squared_error
```

In [8]:
```
# Let's install the Linearregression model
linear_model=LinearRegression()
# fit the model with linear model
linear_model.fit(X.array.reshape(-1,1),y)
```

Out[8]:
```
▾ LinearRegression
LinearRegression()
```

In [9]:
```
# Predict the values with test data
y_pred=linear_model.predict(X.array.reshape(-1,1))
y_pred
```

Out[9]:
```
array([ 36188.15875227,  38078.15121656,  39968.14368085,  44693.12484158,
        46583.11730587,  53198.09093089,  54143.08716303,  56033.07962732,
        56033.07962732,  60758.06078805,  62648.05325234,  63593.04948449,
        63593.04948449,  64538.04571663,  68318.03064522,  72098.0155738 ,
        73988.00803809,  75878.00050238,  81547.97789525,  82492.9741274 ,
        90052.94398456,  92887.932681  , 100447.90253816, 103282.8912346 ,
       108007.87239533, 110842.86109176, 115567.84225249, 116512.83848464,
       123127.81210966, 125017.80457395])
```

In [10]:
```
# Checking the coefiencnt
coeff=linear_model.coef_[0]
coeff
```

Out[10]:
```
9449.962321455077
```

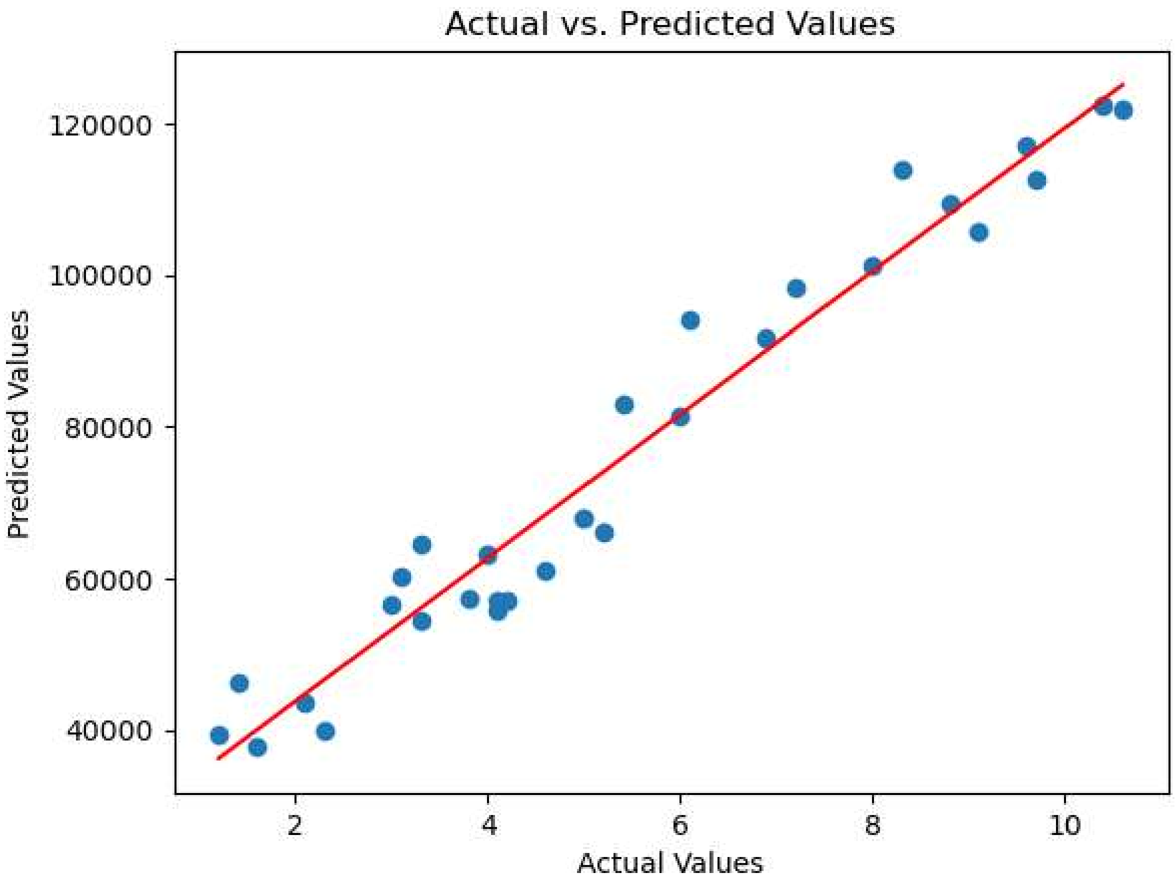
In [11]:
```
# intercept of the model
intercept=linear_model.intercept_
intercept
```

Out[11]:
```
24848.203966523193
```

In [12]:
```
# Checking the model score
print('Mean_squared_error',format(mean_squared_error(y,y_pred)))
print('R2_score ',format(r2_score(y,y_pred)))
```

```
Mean_squared_error 31270951.722280953
R2_score  0.9569566641435086
```

In [13]:
```
# Let's Create a scatter plot for the actual and preidict values
plt.scatter(X, y, label='Original Data')
plt.plot(X, y_pred, color='red', label='Regression Line')
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.title('Actual vs. Predicted Values')
plt.show()
```



In [14]:
```
# How to calculate the linearregression model
# formula=y=m*x+c
#intercept values
m=9449.962321455077
#year of expreice
x=5.5
# slop of the values
c=24848.203
predicted_values=m*x+c
predicted_values
```

Out[14]:
```
76822.99576800293
```