Simple Linear Regression - HR Analytics

In [1]:
```
import pandas as pd
```

Approach for building a regression problem in Python

- 1) Read and access the data
- 2) Identify the independent and dependent variables
- 3) Splliting the data into train and test
- 4) Building the model
- 5) Identify the equation using the slope and intercept and find R-Sqaured
- 6) Predict the test data, using your model

In [2]:
```
ctc = pd.read_csv(r'C:\Users\kaushikswaroop\OneDrive - KPMG\Documents\DSP
ctc
```

Out[2]:

	CTCoffered	LastCTC	Interview rating	Skill Set Index	Highest qualification	Total years of work exp
0	19	18	4	3	3	8.5
1	17	16	4	3	3	7.7
2	17	16	4	3	3	7.9
3	9	8	3	1	2	2.7
4	10	9	5	4	4	9.7
...	...	...	...	...	...	...
186	7	5	4	2	2	5.5
187	21	19	3	2	2	5.3
188	14	14	5	4	4	10.3
189	10	8	5	4	4	9.5
190	15	15	4	3	3	7.7

191 rows × 6 columns

In [3]:
```
ctc_new = ctc[['CTCoffered','LastCTC']]
ctc_new.head()
```

Out[3]:

	CTCoffered	LastCTC
0	19	18
1	17	16
2	17	16
3	9	8
4	10	9

In [4]:
```
x = ctc_new[['LastCTC']]
y = ctc_new[['CTCoffered']]
```

In [5]:
```
x.head()
```

Out[5]:

	LastCTC
0	18
1	16
2	16
3	8
4	9

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

Out[6]:

	CTCoffered
0	19
1	17
2	17
3	9
4	10

Splitting the data into train and test

In [7]:
```
from sklearn.model_selection import train_test_split
```

In [8]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,train_size=0.8,random
```

In [9]:
```
len(x_train),len(x_test),len(y_train),len(y_test)
```

Out[9]:
```
(152, 39, 152, 39)
```

In [10]:
```
x_train.head()
```

Out[10]:

	LastCTC
35	18
100	9
55	11
135	15
78	6

Building a linear regression

In [11]:
```
from sklearn.linear_model import LinearRegression
```

In [12]:
```
lr = LinearRegression()
model = lr.fit(x_train,y_train)
```

CTCOffered = (m*lastctc)+c

In [13]:
```
#To find the slope, we would use the coef_ function
model.coef_
```

Out[13]:
```
array([[0.9401281]])
```

In [14]:
```
#To find the constant, we would use intercept_ function
model.intercept_
```

Out[14]:
```
array([1.80548038])
```

In [15]:
```
#To find the R-square value, we will use the score function
model.score(x_train,y_train)
```

Out[15]:
```
0.9709568208423041
```

Ctc = (0.94*LastCTC)+1.8

In [16]:
```
CTC = (0.94*10)+1.8
CTC
```

Out[16]:
```
11.2
```

Predicting on test data

In [17]:
```
y_test.head()
```

Out[17]:

	CTCoffered
152	18
75	21
158	8
66	17
60	17

In [18]:
```
y_test['Predicted CTC'] = model.predict(x_test)
y_test.head()
```

```
<ipython-input-18-f448d3c6d2ff>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  y_test['Predicted CTC'] = model.predict(x_test)
```

Out[18]:

	CTCoffered	Predicted CTC
152	18	17.787658
75	21	20.608042
158	8	7.446249
66	17	16.847530
60	17	16.847530

Evaluating the model by finding out the RMSE value

RMSE - Root Mean Squared Error

In [20]:
```
y_test['Error'] = y_test['CTCoffered'] - y_test['Predicted CTC']
y_test.head()
```

```
<ipython-input-20-52b8d927da44>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  y_test['Error'] = y_test['CTCoffered'] - y_test['Predicted CTC']
```

Out[20]:

	CTCoffered	Predicted CTC	Error
152	18	17.787658	0.212342
75	21	20.608042	0.391958
158	8	7.446249	0.553751
66	17	16.847530	0.152470
60	17	16.847530	0.152470

In [22]:
```
9**2
```

Out[22]: 81

In [24]:
```
y_test['Sqaured_Error'] = y_test['Error']**2
y_test.head()
```

```
<ipython-input-24-d1e7a20492cc>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  y_test['Sqaured_Error'] = y_test['Error']**2
```

Out[24]:

	CTCoffered	Predicted CTC	Error	Sqaured_Error
152	18	17.787658	0.212342	0.045089
75	21	20.608042	0.391958	0.153631
158	8	7.446249	0.553751	0.306640
66	17	16.847530	0.152470	0.023247
60	17	16.847530	0.152470	0.023247

In [25]:
```
mse = y_test['Sqaured_Error'].mean()
mse
```

Out[25]: 0.5104422481086017

In [26]:
```
from math import sqrt
```

In [27]:
```
rmse = sqrt(mse)
rmse
```

Out[27]:
```
0.7144524113673364
```

Let assume a candidate with 12 lakhs salary get selected. What would be the predicted ctc

In [41]:
```
pred_salary = model.predict([[14]])
```

In [42]:
```
pred_salary
```

Out[42]:
```
array([[14.96727382]])
```

In [43]:
```
upper_band = pred_salary+rmse
lower_band = pred_salary-rmse
```

In [44]:
```
upper_band
```

Out[44]:
```
array([[15.68172623]])
```

In [45]:
```
lower_band
```

Out[45]:
```
array([[14.25282141]])
```