

Simple Linear Regression

Regression models (both linear and non-linear) are used for predicting a real value, like salary for example. If your independent variable is time, then you are forecasting future values, otherwise your model is predicting present but unknown values. Regression technique vary from Linear Regression to SVR and Random Forests Regression.

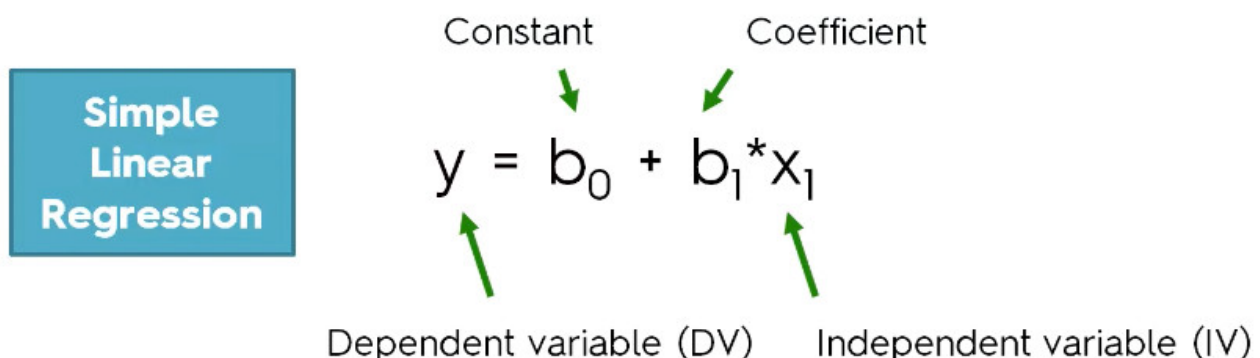
Linear regression is a *basic* and *commonly* used type of **predictive analysis**. The overall idea of regression is to examine two things:

- Does a set of **predictor variables** do a good job in predicting an **outcome** (dependent) variable?
- Which variables in particular are **significant predictors** of the outcome variable, and in what way they do **impact** the outcome variable?

These regression estimates are used to explain the **relationship between one dependent variable and one or more independent variables**. The simplest form of the regression equation with one dependent and one independent variable is defined by the formula :

$$y = \beta_0 + \beta_1 x$$

In this part, you will understand and learn how to implement the **Simple Linear Regression** models:



The diagram illustrates the Simple Linear Regression equation $y = b_0 + b_1 * x_1$. A blue box on the left contains the text "Simple Linear Regression". Green arrows point from labels to parts of the equation: "Constant" points to b_0 , "Coefficient" points to b_1 , "Dependent variable (DV)" points to y , and "Independent variable (IV)" points to x_1 .

Importing the libraries

In [1]:

```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [2]:

```
# Importing the sklearn libraries to train and test and to perform linear regression.
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

Importing the dataset

In [3]:

```
# Importing the dataset
df = pd.read_csv('../Data/Salary_Data.csv')
```

In [4]:

```
# See the data frame information.
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 2 columns):
YearsExperience    30 non-null float64
Salary            30 non-null float64
dtypes: float64(2)
memory usage: 560.0 bytes
```

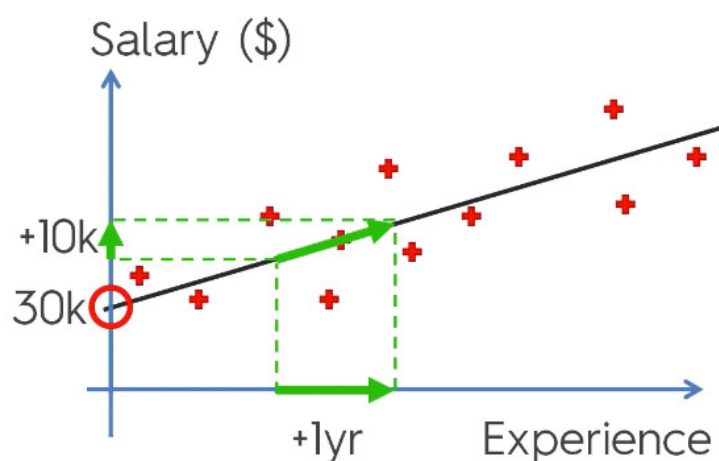
In [5]:

```
# See the dataframe header information.
df.head()
```

Out[5]:

	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

Simple Linear Regression:



$$y = b_0 + b_1 * x$$

↓

$$\text{Salary} = \textcircled{b_0} + \textcircled{b_1} * \text{Experience}$$

Preparing data for X and Y

In [6]:

```
X = df.iloc[:, :-1].values  
y = df.iloc[:, 1].values
```

Splitting X and y into training and test datasets

In [7]:

```
# Splitting the dataset into the Training set and Test set  
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 1/3, random_state  
= 0)
```

Fitting Simple Linear Regression to the Training set

In [8]:

```
# Fitting Simple Linear Regression to the Training set  
linreg = LinearRegression()  
linreg.fit(X_train, y_train)
```

Out[8]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,  
normalize=False)
```

Predicting the Test set results

In [9]:

```
# Predicting the Test set results
y_pred = linreg.predict(X_test)

df_sample = pd.DataFrame({'experience': X_test.tolist(), 'test_salary': y_test, 'predict_salary': y_pred})
# print(X_test, y_test, y_pred)

df_sample
```

Out[9]:

	experience	test_salary	predict_salary
0	[1.5]	37731.0	40835.105909
1	[10.3]	122391.0	123079.399408
2	[4.1]	57081.0	65134.556261
3	[3.9]	63218.0	63265.367772
4	[9.5]	116969.0	115602.645454
5	[8.7]	109431.0	108125.891499
6	[9.6]	112635.0	116537.239698
7	[4.0]	55794.0	64199.962017
8	[5.3]	83088.0	76349.687193
9	[7.9]	101302.0	100649.137545

Visualising the Training set results

In [10]:

```
# Visualising the Training set results  
plt.scatter(X_train, y_train, color = 'red')  
plt.plot(X_train, linreg.predict(X_train), color = 'blue')  
plt.title('Salary vs Experience (Training set)')  
plt.xlabel('Years of Experience')  
plt.ylabel('Salary')  
plt.show()
```



Visualising the Test set results

In [11]:

```
# Visualising the Test set results  
plt.scatter(X_test, y_test, color = 'red')  
plt.plot(X_train, linreg.predict(X_train), color = 'blue')  
plt.title('Salary vs Experience (Test set)')  
plt.xlabel('Years of Experience')  
plt.ylabel('Salary')  
plt.show()
```

