In [28]:
```
import numpy as np
import pandas as pd
```

import dataset

In [29]:
```
dataset=pd.read_csv("C:\\Users\\YC\\Documents\\Data_processing.csv")
dataset
```

Out[29]:

	Country	Age	Salary	purchased
0	France	44.0	72000.0	No
1	Spain	27.0	48000.0	Yes
2	Germany	30.0	54000.0	No
3	Spain	38.0	61000.0	No
4	Germany	40.0	NaN	Yes
5	France	35.0	58000.0	Yes
6	Spain	NaN	52000.0	No
7	France	48.0	79000.0	Yes
8	Germany	50.0	83000.0	No
9	France	37.0	67000.0	Yes

In [46]:
```
print(dataset.columns)
```

```
Index(['Country ', 'Age', 'Salary', 'purchased'], dtype='object')
```

independent variable

In [48]:
```
x=dataset[['Country ', 'Age', 'Salary']].values
x
```

Out[48]:
```
array([['France', 44.0, 72000.0],
       ['Spain', 27.0, 48000.0],
       ['Germany', 30.0, 54000.0],
       ['Spain', 38.0, 61000.0],
       ['Germany', 40.0, nan],
       ['France', 35.0, 58000.0],
       ['Spain', nan, 52000.0],
       ['France', 48.0, 79000.0],
       ['Germany', 50.0, 83000.0],
       ['France', 37.0, 67000.0]], dtype=object)
```

Dependent variable

In [50]:
```
y=dataset[['purchased']].values
y
```

Out[50]:
```
array([['No'],
       ['Yes'],
       ['No'],
       ['No'],
       ['Yes'],
       ['Yes'],
       ['No'],
       ['Yes'],
       ['No'],
       ['Yes']], dtype=object)
```

Handling missing values

In [53]:
```
from sklearn.impute import SimpleImputer
imputer=SimpleImputer(missing_values=np.NaN, strategy='mean')
imputer=imputer.fit(x[:,1:3])
x[:,1:3]=imputer.transform(x[:,1:3])
```

In [55]:
```
x
```

Out[55]:
```
array([['France', 44.0, 72000.0],
       ['Spain', 27.0, 48000.0],
       ['Germany', 30.0, 54000.0],
       ['Spain', 38.0, 61000.0],
       ['Germany', 40.0, 63777.77777777778],
       ['France', 35.0, 58000.0],
       ['Spain', 38.77777777777778, 52000.0],
       ['France', 48.0, 79000.0],
       ['Germany', 50.0, 83000.0],
       ['France', 37.0, 67000.0]], dtype=object)
```

Encoding categorical data

In [58]:
```
from sklearn.preprocessing import LabelEncoder
label_encoder_x=LabelEncoder()
x[:,0]=label_encoder_x.fit_transform(x[:,0])
```

In [69]:
```
x
```

Out[69]:
```
array([[0, 44.0, 72000.0],
       [2, 27.0, 48000.0],
       [1, 30.0, 54000.0],
       [2, 38.0, 61000.0],
       [1, 40.0, 63777.77777777778],
       [0, 35.0, 58000.0],
       [2, 38.77777777777778, 52000.0],
       [0, 48.0, 79000.0],
       [1, 50.0, 83000.0],
       [0, 37.0, 67000.0]], dtype=object)
```

In [70]:
```
print(dataset.columns)
```

```
Index(['Country ', 'Age', 'Salary', 'purchased'], dtype='object')
```

Dumy Encoding(0,1)

In [73]:
```
from sklearn.preprocessing import OneHotEncoder
```

```
onehotencoder = OneHotEncoder()
x = onehotencoder.fit_transform(dataset['Country '].values.reshape(-1, 1)).toarray(
```

In [74]:
```
x
```

Out[74]:
```
array([[1., 0., 0.],
       [0., 0., 1.],
       [0., 1., 0.],
       [0., 0., 1.],
       [0., 1., 0.],
       [1., 0., 0.],
       [0., 0., 1.],
       [1., 0., 0.],
       [0., 1., 0.],
       [1., 0., 0.]])
```

For y dumy encoding

In [76]:
```
label_y=LabelEncoder()
y=label_y.fit_transform(y)
y
```

```
C:\Users\YC\anaconda3\Lib\site-packages\sklearn\preprocessing\_label.py:114: DataC
onversionWarning: A column-vector y was passed when a 1d array was expected. Pleas
e change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[76]:
```
array([0, 1, 0, 0, 1, 1, 0, 1, 0, 1])
```

Data set split(train,test)

In [78]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

In [79]:
```
x_train
```

Out[79]:
```
array([[0., 1., 0.],
       [1., 0., 0.],
       [0., 0., 1.],
       [0., 0., 1.],
       [1., 0., 0.],
       [0., 0., 1.],
       [1., 0., 0.],
       [1., 0., 0.]])
```

In [80]:
```
x_test
```

Out[80]:
```
array([[0., 1., 0.],
       [0., 1., 0.]])
```

In [81]:
```
y_train
```

Out[81]:
```
array([1, 1, 1, 0, 1, 0, 0, 1])
```

In [83]:
```
y_test
```

Out[83]:
```
array([0, 0])
```

Feature scaling

In [85]:
```
from sklearn.preprocessing import StandardScaler
sc_x=StandardScaler()
x_train=sc_x.fit_transform(x_train)
x_test=sc_x.transform(x_test)
```

In [87]:
```
x_train
```

Out[87]:
```
array([[-1.        ,  2.64575131, -0.77459667],
       [ 1.        , -0.37796447, -0.77459667],
       [-1.        , -0.37796447,  1.29099445],
       [-1.        , -0.37796447,  1.29099445],
       [ 1.        , -0.37796447, -0.77459667],
       [-1.        , -0.37796447,  1.29099445],
       [ 1.        , -0.37796447, -0.77459667],
       [ 1.        , -0.37796447, -0.77459667]])
```