In [1]:
```
#importing libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [2]:
```
#importing Dataset
df = pd.read_csv("Data.csv")
```

In [3]:
```
#View The Data
df.head()
```

Out[3]:

	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

In [4]:
```
#View The Data Info
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Country    10 non-null     object 
 1   Age        9 non-null      float64
 2   Salary     9 non-null      float64
 3   Purchased  10 non-null     object 
dtypes: float64(2), object(2)
memory usage: 448.0+ bytes
```

In [5]:
```
#View The Shape of Data
df.shape
```

Out[5]:
```
(10, 4)
```

In [6]:
```
#Check if There is Any NULl Values in Data
df.isnull().sum()
```

Out[6]:
```
Country      0
Age          1
Salary       1
Purchased    0
dtype: int64
```

In [7]:
```
#Defining Features & Label of Data
X = df.iloc[:, :-1].values
y = df.iloc[:, 3].values
```

In [8]:
```
#Import SimpleImputer Scikit-Learn Library to Handle the Missing Numeric Data

from sklearn.impute import SimpleImputer

SI = SimpleImputer(missing_values=np.nan, strategy='mean')
X[:, 1:3] = SI.fit_transform(X[:, 1:3])
```

In [9]:
```
#View Data
X
```

Out[9]:
```
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)
```

In [10]:
```
#View Categories in Country Column
df["Country"].unique()
```

Out[10]:
```
array(['France', 'Spain', 'Germany'], dtype=object)
```

In [11]:
```
#Import LabelEncoder Scikit-Learn Library to Handle the Categorical Data

from sklearn.preprocessing import LabelEncoder

LE = LabelEncoder()
X[:, 0] = LE.fit_transform(X[:, 0])
```

In [12]:
```
#View Data
X
```

Out[12]:
```
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 [13]:
```
#Import OneHotEncoder Scikit-Learn Library to Handle the Categorical Data

from sklearn.preprocessing import OneHotEncoder

OHE = OneHotEncoder(categories = 'auto', sparse_output = False)
X = OHE.fit_transform(df[['Country']])
```

In [14]:
```
#View Data
X
```

Out[14]:
```
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.]])
```

In [15]:
```
#Assign LabelEncoder on y Label to Handle Categorical Data
LE_y = LabelEncoder()
y = LE_y.fit_transform(y)
```

In [16]:
```
#View Label
y
```

Out[16]:
```
array([0, 1, 0, 0, 1, 1, 0, 1, 0, 1])
```

In [17]:
```
#Splitting Data into Train & Test
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 [18]:
```
#Import StandardScaler Scikit-Learn Library to Scale Data
from sklearn.preprocessing import StandardScaler

SC = StandardScaler()
X_train = SC.fit_transform(X_train)
X_test = SC.transform(X_test)
```

In [19]:
```
#View X_train Data
X_train
```

Out[19]:
```
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]])
```

In [20]:
```
#View X_test Data
X_test
```

Out[20]:
```
array([[-1.        ,  2.64575131, -0.77459667],
       [-1.        ,  2.64575131, -0.77459667]])
```

In [ ]:
```

```