DataPreprocessing_Assignment11

Step 1: Importing the libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
sns.set(color_codes=True)
from sklearn.preprocessing import LabelEncoder
```

Step 2: Importing dataset

In [2]:
```
dataset = pd.read_csv('Data.csv')
```

In [3]:
```
dataset.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]:
```
dataset.shape
```

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

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

Out[5]:
```
Index(['Country', 'Age', 'Salary', 'Purchased'], dtype='object')
```

Step 3: Handling the missing data

In [6]:
```
dataset.isna().any()
```

Out[6]:
```
Country      False
Age           True
Salary        True
Purchased    False
dtype: bool
```

In [7]:
```
dataset.isna().sum()
```

Out[7]:
```
Country      0
Age          1
Salary       1
Purchased    0
dtype: int64
```

In [8]:
```
X = dataset[["Country","Age","Salary"]].values
```

In [9]:
```
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, 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)
```

In [10]:
```
Y = dataset[["Purchased"]].values
```

In [11]:
```
Y
```

Out[11]:
```
array([['No'],
       ['Yes'],
       ['No'],
       ['No'],
       ['Yes'],
       ['Yes'],
       ['No'],
       ['Yes'],
       ['No'],
       ['Yes']], dtype=object)
```

In [12]:
```
from sklearn.impute import SimpleImputer
```

In [13]:
```
imputer = SimpleImputer(missing_values = np.nan , strategy = "mean")
```

In [14]:
```
imputer = imputer.fit(X[:,1:3])
```

In [15]:
```
X[:,1:3] = imputer.transform(X[:,1:3])
```

In [16]:
```
X
```

Out[16]:
```
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)
```

Step 4: Encoding categorical data

In [17]:
```
from sklearn.preprocessing import LabelEncoder
```

In [18]:
```
label_encoder_X = LabelEncoder()
```

In [19]:
```
X[:,0] = label_encoder_X.fit_transform(X[:,0])
```

In [20]:
```
X
```

Out[20]:
```
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 [21]:
```
labelencoder_Y = LabelEncoder()
```

In [22]:
```
Y = labelencoder_Y.fit_transform(Y)
```

```
C:\Users\swara\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\preprocessing\_label.py:115: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

In [23]:
```
Y
```

Out[23]:
```
array([0, 1, 0, 0, 1, 1, 0, 1, 0, 1])
```

Step 5: Creating a dummy variable

In [24]:
```
from sklearn.preprocessing import OneHotEncoder
```

In [25]:
```
onehotencoder = OneHotEncoder()
```

In [26]:
```
onehotencoder.fit_transform(dataset.Country.values.reshape(-1,1)).toarray()
```

Out[26]:
```
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.]])
```

Step 6: Splitting the datasets into training sets and Test sets

In [27]:
```
from sklearn.model_selection import train_test_split
```

In [28]:
```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size = 0.2, random_state = 0)
```

In [29]:
```
X_train
```

Out[29]:
```
array([[1, 40.0, 63777.77777777778],
       [0, 37.0, 67000.0],
       [2, 27.0, 48000.0],
       [2, 38.77777777777778, 52000.0],
       [0, 48.0, 79000.0],
       [2, 38.0, 61000.0],
       [0, 44.0, 72000.0],
       [0, 35.0, 58000.0]], dtype=object)
```

In [30]:
```
X_test
```

Out[30]:
```
array([[1, 30.0, 54000.0],
       [1, 50.0, 83000.0]], dtype=object)
```

In [31]:
```
Y_train
```

Out[31]:
```
array([1, 1, 1, 0, 1, 0, 0, 1])
```

In [32]:
```
Y_test
```

Out[32]:
```
array([0, 0])
```

Step 7: Feature Scaling

In [33]:
```
from sklearn.preprocessing import StandardScaler
```

In [34]:
```
sc_X = StandardScaler()
```

In [35]:
```
X_train = sc_X.fit_transform(X_train)
```

In [36]:
```
X_test = sc_X.transform(X_test)
```

In [37]:
```
X_train
```

Out[37]:
```
array([[ 0.13483997,  0.26306757,  0.12381479],
       [-0.94387981, -0.25350148,  0.46175632],
       [ 1.21355975, -1.97539832, -1.53093341],
       [ 1.21355975,  0.05261351, -1.11141978],
       [-0.94387981,  1.64058505,  1.7202972 ],
       [ 1.21355975, -0.0813118 , -0.16751412],
       [-0.94387981,  0.95182631,  0.98614835],
       [-0.94387981, -0.59788085, -0.48214934]])
```

In [38]:
```
X_test
```

Out[38]:
```
array([[ 0.13483997, -1.45882927, -0.90166297],
       [ 0.13483997,  1.98496442,  2.13981082]])
```