Data Preprocessing Tools ادوات المعالجة المسبقة للبيانات (تجهيز البيانات)

Importing the libraries استيراد المكتبات البرمجية

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Importing the dataset استيراد البيانات وتحميلها من ملف

In [2]:

```
dataset = pd.read_csv('Data.csv')
```

In [3]:

```
dataset
```

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
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
10	France	37.0	67000.0	Yes

Drop Duplicated data علاج البيانات المكررة

In [4]:

```
dup=dataset.duplicated()
dup.value_counts()
```

Out[4]:

```
False    10
True      2
dtype: int64
```

In [5]:

```
dataset=dataset.drop_duplicates()
dup=dataset.duplicated()
dup.value_counts()
```

Out[5]:

```
False    10
dtype: int64
```

Taking care of missing data علاج البيانات المفقودة

In [6]:

```
m=dataset.isnull().sum()
m
```

Out[6]:

```
Country      0
Age          1
Salary       1
Purchased    0
dtype: int64
```

In [7]:

```
dataset=dataset.dropna()
m=dataset.isnull().sum()
m
```

Out[7]:

```
Country      0
Age          0
Salary       0
Purchased    0
dtype: int64
```

In [8]:

```
print(dataset)
```

```
   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
5   France  35.0  58000.0       Yes
7   France  48.0  79000.0       Yes
8  Germany  50.0  83000.0        No
9   France  37.0  67000.0       Yes
```

In [9]:

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

In [10]:

```
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(missing_values=np.nan, strategy='mean') # للقيم المفقودة للعمر والمرتبات
imputer.fit(X[:, 1:3])
X[:, 1:3] = imputer.transform(X[:, 1:3])
```

In [11]:

```
print(X)
```

```
[['France' 44.0 72000.0]
 ['Spain' 27.0 48000.0]
 ['Germany' 30.0 54000.0]
 ['Spain' 38.0 61000.0]
 ['France' 35.0 58000.0]
 ['France' 48.0 79000.0]
 ['Germany' 50.0 83000.0]
 ['France' 37.0 67000.0]]
```

Encoding categorical data ترميز البيانات الفئوية

Encoding the Independent Variable (عمود البلد) ترميز المتغير المستقل

In [12]:

```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [0])], remainder='passthr
X = np.array(ct.fit_transform(X))
```

In [13]:

```
print(X)
```

```
[[1.0 0.0 0.0 44.0 72000.0]
 [0.0 0.0 1.0 27.0 48000.0]
 [0.0 1.0 0.0 30.0 54000.0]
 [0.0 0.0 1.0 38.0 61000.0]
 [1.0 0.0 0.0 35.0 58000.0]
 [1.0 0.0 0.0 48.0 79000.0]
 [0.0 1.0 0.0 50.0 83000.0]
 [1.0 0.0 0.0 37.0 67000.0]]
```

Encoding the Dependent Variable (عمود الشراء) ترميز المتغير التابع

In [14]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)
```

In [15]:

```
print(y)
```

```
[0 1 0 0 1 1 0 1]
```

Splitting the dataset into the Training set and Test set تقسيم البيانات الي بيانات التدريب وبيانات الاختبار

In [33]:

```
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 = 1
```

In [34]:

```
print(X_train)
```

```
[[0.0 0.0 1.0 27.0 48000.0]
 [0.0 1.0 0.0 50.0 83000.0]
 [1.0 0.0 0.0 44.0 72000.0]
 [1.0 0.0 0.0 35.0 58000.0]
 [0.0 0.0 1.0 38.0 61000.0]
 [1.0 0.0 0.0 48.0 79000.0]]
```

In [35]:

```
print(X_test)
```

```
[[1.0 0.0 0.0 37.0 67000.0]
 [0.0 1.0 0.0 30.0 54000.0]]
```

In [36]:

```
print(y_train)
```

```
[1 0 0 1 0 1]
```

In [37]:

```
print(y_test)
```

```
[1 0]
```

Feature Scaling تحجيم الخصائص

In [38]:

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train[:, 3:] = sc.fit_transform(X_train[:, 3:])
X_test[:, 3:] = sc.transform(X_test[:, 3:])
```

In [39]:

```
print(X_train)
```

```
[[0.0 0.0 1.0 -1.6813254068367947 -1.5353204183985696]
 [0.0 1.0 0.0 1.2189609199566755 1.3179299166784189]
 [1.0 0.0 0.0 0.46236448688011816 0.42119409708279393]
 [1.0 0.0 0.0 -0.672530162734718 -0.7201060369480015]
 [0.0 0.0 1.0 -0.29423194619643933 -0.475541722512831]
 [1.0 0.0 0.0 0.9667621089311564 0.9918441640981916]]
```

In [40]:

```
print(X_test)
```

```
[[1.0 0.0 0.0 -0.4203313517091989 0.013586906357509865]
 [0.0 1.0 0.0 -1.303027190298516 -1.0461917895282287]]
```

In [ ]:

In [ ]: