In [2]:

```
import pandas as pd
import numpy as np
dataset = pd.read_csv('incomplete-data.csv')
```

In [3]:

```
dataset.shape
```

Out[3]:

```
(20, 6)
```

In [4]:

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   User ID    20 non-null     int64  
 1   country    19 non-null     object 
 2   Gender     20 non-null     object 
 3   Age        17 non-null     float64
 4   salary     19 non-null     float64
 5   Purchased  20 non-null     int64  
dtypes: float64(2), int64(2), object(2)
memory usage: 1.1+ KB
```

In [5]:

```
dataset.describe()
```

Out[5]:

| | User ID | Age | salary | Purchased |
|---|---|---|---|---|
| **count** | 2.000000e+01 | 17.000000 | 19.000000 | 20.000000 |
| **mean** | 1.567881e+07 | 29.529412 | 57368.421053 | 0.400000 |
| **std** | 6.987218e+04 | 9.348246 | 33128.052105 | 0.502625 |
| **min** | 1.557077e+07 | 18.000000 | 18000.000000 | 0.000000 |
| **25%** | 1.561468e+07 | 25.000000 | 28000.000000 | 0.000000 |
| **50%** | 1.569626e+07 | 27.000000 | 57000.000000 | 0.000000 |
| **75%** | 1.572768e+07 | 35.000000 | 80000.000000 | 1.000000 |
| **max** | 1.581094e+07 | 47.000000 | 150000.000000 | 1.000000 |

In [6]:

```
dataset = dataset.drop(['User ID','Gender'],axis=1)
```

In [7]:

```
dataset
```

Out[7]:

| | country | Age | salary | Purchased |
|---|---|---|---|---|
| **0** | India | 19.0 | 19000.0 | 0 |
| **1** | USA | 35.0 | NaN | 1 |
| **2** | France | 26.0 | 43000.0 | 0 |
| **3** | USA | NaN | 57000.0 | 0 |
| **4** | France | 19.0 | 76000.0 | 0 |
| **5** | India | 27.0 | 58000.0 | 0 |
| **6** | India | 27.0 | 84000.0 | 1 |
| **7** | USA | NaN | 150000.0 | 1 |
| **8** | France | 25.0 | 33000.0 | 0 |
| **9** | USA | 35.0 | 65000.0 | 0 |
| **10** | India | 26.0 | 80000.0 | 0 |
| **11** | India | 26.0 | 52000.0 | 0 |
| **12** | France | 20.0 | 86000.0 | 0 |
| **13** | USA | 32.0 | 18000.0 | 1 |
| **14** | France | 18.0 | 82000.0 | 0 |
| **15** | India | 29.0 | 80000.0 | 0 |
| **16** | India | 47.0 | 25000.0 | 1 |
| **17** | NaN | 45.0 | 26000.0 | 1 |
| **18** | France | 46.0 | 28000.0 | 1 |
| **19** | India | NaN | 28000.0 | 1 |

In [8]:

```
dataset.isnull().sum()
```

Out[8]:

```
country      1
Age          3
salary       1
Purchased    0
dtype: int64
```

In [9]:

```
#filling missing value in country column
country_mode = dataset['country'].mode()[0]
dataset['country'].fillna(country_mode,inplace=True)
dataset.head()
```

Out[9]:

| | country | Age | salary | Purchased |
|---|---|---|---|---|
| **0** | India | 19.0 | 19000.0 | 0 |
| **1** | USA | 35.0 | NaN | 1 |
| **2** | France | 26.0 | 43000.0 | 0 |
| **3** | USA | NaN | 57000.0 | 0 |
| **4** | France | 19.0 | 76000.0 | 0 |

In [26]:

```
dataset.isna().sum()
```

Out[26]:

```
Age          0
salary       0
Purchased    0
France       0
India        0
USA          0
dtype: int64
```

In [27]:

```
#Encoding country column value into numerical form i.e India as 1, USA as 2 and france
#d1={'India':1,'USA':2,'France':3}
#dataset['country1']=dataset['country'].map(d1)
#dataset=dataset.drop('country',axis=1)
#df3=dataset[['country1']]
```

In [12]:

```
#dataset.drop('country1',axis=1,inplace=True)
#dataset.insert(0,'country',df3)
```

In [13]:

```
dataset.head()
```

Out[13]:

| | country | Age | salary | Purchased |
|---|---|---|---|---|
| **0** | India | 19.0 | 19000.0 | 0 |
| **1** | USA | 35.0 | NaN | 1 |
| **2** | France | 26.0 | 43000.0 | 0 |
| **3** | USA | NaN | 57000.0 | 0 |
| **4** | France | 19.0 | 76000.0 | 0 |

In [15]:

```
#creating dummy numerical column for each value in country column
df_cat = pd.get_dummies(dataset['country'],drop_first=False)
df_cat
```

Out[15]:

| | France | India | USA |
|---|---|---|---|
| **0** | 0 | 1 | 0 |
| **1** | 0 | 0 | 1 |
| **2** | 1 | 0 | 0 |
| **3** | 0 | 0 | 1 |
| **4** | 1 | 0 | 0 |
| **5** | 0 | 1 | 0 |
| **6** | 0 | 1 | 0 |
| **7** | 0 | 0 | 1 |
| **8** | 1 | 0 | 0 |
| **9** | 0 | 0 | 1 |
| **10** | 0 | 1 | 0 |
| **11** | 0 | 1 | 0 |
| **12** | 1 | 0 | 0 |
| **13** | 0 | 0 | 1 |
| **14** | 1 | 0 | 0 |
| **15** | 0 | 1 | 0 |
| **16** | 0 | 1 | 0 |
| **17** | 0 | 1 | 0 |
| **18** | 1 | 0 | 0 |
| **19** | 0 | 1 | 0 |

In [17]:

```
#joining dummy column with dataset
dataset = pd.concat([dataset,df_cat],axis=1)
```

In [18]:

```
dataset
```

Out[18]:

| | country | Age | salary | Purchased | France | India | USA |
|---|---|---|---|---|---|---|---|
| **0** | India | 19.0 | 19000.0 | 0 | 0 | 1 | 0 |
| **1** | USA | 35.0 | NaN | 1 | 0 | 0 | 1 |
| **2** | France | 26.0 | 43000.0 | 0 | 1 | 0 | 0 |
| **3** | USA | NaN | 57000.0 | 0 | 0 | 0 | 1 |
| **4** | France | 19.0 | 76000.0 | 0 | 1 | 0 | 0 |
| **5** | India | 27.0 | 58000.0 | 0 | 0 | 1 | 0 |
| **6** | India | 27.0 | 84000.0 | 1 | 0 | 1 | 0 |
| **7** | USA | NaN | 150000.0 | 1 | 0 | 0 | 1 |
| **8** | France | 25.0 | 33000.0 | 0 | 1 | 0 | 0 |
| **9** | USA | 35.0 | 65000.0 | 0 | 0 | 0 | 1 |
| **10** | India | 26.0 | 80000.0 | 0 | 0 | 1 | 0 |
| **11** | India | 26.0 | 52000.0 | 0 | 0 | 1 | 0 |
| **12** | France | 20.0 | 86000.0 | 0 | 1 | 0 | 0 |
| **13** | USA | 32.0 | 18000.0 | 1 | 0 | 0 | 1 |
| **14** | France | 18.0 | 82000.0 | 0 | 1 | 0 | 0 |
| **15** | India | 29.0 | 80000.0 | 0 | 0 | 1 | 0 |
| **16** | India | 47.0 | 25000.0 | 1 | 0 | 1 | 0 |
| **17** | India | 45.0 | 26000.0 | 1 | 0 | 1 | 0 |
| **18** | France | 46.0 | 28000.0 | 1 | 1 | 0 | 0 |
| **19** | India | NaN | 28000.0 | 1 | 0 | 1 | 0 |

In [19]:

```
#deleting country column
dataset.drop('country',axis=1,inplace=True)
```

In [20]:

```
dataset
```

Out[20]:

| | Age | salary | Purchased | France | India | USA |
|---|---|---|---|---|---|---|
| **0** | 19.0 | 19000.0 | 0 | 0 | 1 | 0 |
| **1** | 35.0 | NaN | 1 | 0 | 0 | 1 |
| **2** | 26.0 | 43000.0 | 0 | 1 | 0 | 0 |
| **3** | NaN | 57000.0 | 0 | 0 | 0 | 1 |
| **4** | 19.0 | 76000.0 | 0 | 1 | 0 | 0 |
| **5** | 27.0 | 58000.0 | 0 | 0 | 1 | 0 |
| **6** | 27.0 | 84000.0 | 1 | 0 | 1 | 0 |
| **7** | NaN | 150000.0 | 1 | 0 | 0 | 1 |
| **8** | 25.0 | 33000.0 | 0 | 1 | 0 | 0 |
| **9** | 35.0 | 65000.0 | 0 | 0 | 0 | 1 |
| **10** | 26.0 | 80000.0 | 0 | 0 | 1 | 0 |
| **11** | 26.0 | 52000.0 | 0 | 0 | 1 | 0 |
| **12** | 20.0 | 86000.0 | 0 | 1 | 0 | 0 |
| **13** | 32.0 | 18000.0 | 1 | 0 | 0 | 1 |
| **14** | 18.0 | 82000.0 | 0 | 1 | 0 | 0 |
| **15** | 29.0 | 80000.0 | 0 | 0 | 1 | 0 |
| **16** | 47.0 | 25000.0 | 1 | 0 | 1 | 0 |
| **17** | 45.0 | 26000.0 | 1 | 0 | 1 | 0 |
| **18** | 46.0 | 28000.0 | 1 | 1 | 0 | 0 |
| **19** | NaN | 28000.0 | 1 | 0 | 1 | 0 |

In [24]:

```
#filling missing values using imputer
from sklearn.impute import SimpleImputer
median_imputer = SimpleImputer(missing_values=np.nan,strategy='median')
result_median_imputer = median_imputer.fit_transform(dataset)
dataset = pd.DataFrame(result_median_imputer, columns=dataset.columns)
```

In [25]:

```
dataset
```

Out[25]:

| | Age | salary | Purchased | France | India | USA |
|---|---|---|---|---|---|---|
| **0** | 19.0 | 19000.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| **1** | 35.0 | 57000.0 | 1.0 | 0.0 | 0.0 | 1.0 |
| **2** | 26.0 | 43000.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| **3** | 27.0 | 57000.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| **4** | 19.0 | 76000.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| **5** | 27.0 | 58000.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| **6** | 27.0 | 84000.0 | 1.0 | 0.0 | 1.0 | 0.0 |
| **7** | 27.0 | 150000.0 | 1.0 | 0.0 | 0.0 | 1.0 |
| **8** | 25.0 | 33000.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| **9** | 35.0 | 65000.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| **10** | 26.0 | 80000.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| **11** | 26.0 | 52000.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| **12** | 20.0 | 86000.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| **13** | 32.0 | 18000.0 | 1.0 | 0.0 | 0.0 | 1.0 |
| **14** | 18.0 | 82000.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| **15** | 29.0 | 80000.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| **16** | 47.0 | 25000.0 | 1.0 | 0.0 | 1.0 | 0.0 |
| **17** | 45.0 | 26000.0 | 1.0 | 0.0 | 1.0 | 0.0 |
| **18** | 46.0 | 28000.0 | 1.0 | 1.0 | 0.0 | 0.0 |
| **19** | 27.0 | 28000.0 | 1.0 | 0.0 | 1.0 | 0.0 |

In [28]:

```
#Another way of filling missing Values
# find the mode of Age in data #calculate mode and substitute (Impute) this mode at th
#impute value for the Age column column
#age_mode=x['Age'].mode()[0]
#most repeated values assigned to age_mode varibale
#country_mode=x['country'].mode()[0]
#most repeated values assigned to age_mode varibale
#filling missing value with mode value of Age
#x['Age'].fillna(age_mode,inplace=True)
#filling missing value with mode value of country
#column x['country'].fillna(country_mode,inplace=True)
##filling missing value with median value of salary column
#median_val=x['salary'].median() x['salary'].fillna(median_val,inplace=True)
```

In [30]:

```
dataset.isna().sum()
```

Out[30]:

```
Age          0
salary       0
Purchased    0
France       0
India        0
USA          0
dtype: int64
```

In [31]:

```
dataset
```

Out[31]:

| | Age | salary | Purchased | France | India | USA |
|---|---|---|---|---|---|---|
| **0** | 19.0 | 19000.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| **1** | 35.0 | 57000.0 | 1.0 | 0.0 | 0.0 | 1.0 |
| **2** | 26.0 | 43000.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| **3** | 27.0 | 57000.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| **4** | 19.0 | 76000.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| **5** | 27.0 | 58000.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| **6** | 27.0 | 84000.0 | 1.0 | 0.0 | 1.0 | 0.0 |
| **7** | 27.0 | 150000.0 | 1.0 | 0.0 | 0.0 | 1.0 |
| **8** | 25.0 | 33000.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| **9** | 35.0 | 65000.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| **10** | 26.0 | 80000.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| **11** | 26.0 | 52000.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| **12** | 20.0 | 86000.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| **13** | 32.0 | 18000.0 | 1.0 | 0.0 | 0.0 | 1.0 |
| **14** | 18.0 | 82000.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| **15** | 29.0 | 80000.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| **16** | 47.0 | 25000.0 | 1.0 | 0.0 | 1.0 | 0.0 |
| **17** | 45.0 | 26000.0 | 1.0 | 0.0 | 1.0 | 0.0 |
| **18** | 46.0 | 28000.0 | 1.0 | 1.0 | 0.0 | 0.0 |
| **19** | 27.0 | 28000.0 | 1.0 | 0.0 | 1.0 | 0.0 |

In [32]:

```
dataset['France'] = dataset['France'].astype('int64')
dataset['India']=dataset['India'].astype('int64')
dataset['USA']=dataset['USA'].astype('int64')
dataset['Purchased']=dataset['Purchased'].astype('int64')
dataset.head()
```

Out[32]:

|   | Age | salary | Purchased | France | India | USA |
|---|---|---|---|---|---|---|
| **0** | 19.0 | 19000.0 | 0 | 0 | 1 | 0 |
| **1** | 35.0 | 57000.0 | 1 | 0 | 0 | 1 |
| **2** | 26.0 | 43000.0 | 0 | 1 | 0 | 0 |
| **3** | 27.0 | 57000.0 | 0 | 0 | 0 | 1 |
| **4** | 19.0 | 76000.0 | 0 | 1 | 0 | 0 |

In [33]:

```
# The code in cell of In [58] to In [60] is used to move Purchased column at the end
purchase_column=dataset['Purchased']
```

In [34]:

```
dataset = dataset.drop('Purchased',axis=1)
```

In [35]:

```
dataset.insert(5,"Purchased",purchase_column)
```

In [36]:

```
dataset
```

Out[36]:

|   | Age | salary | France | India | USA | Purchased |
|---|---|---|---|---|---|---|
| **0** | 19.0 | 19000.0 | 0 | 1 | 0 | 0 |
| **1** | 35.0 | 57000.0 | 0 | 0 | 1 | 1 |
| **2** | 26.0 | 43000.0 | 1 | 0 | 0 | 0 |
| **3** | 27.0 | 57000.0 | 0 | 0 | 1 | 0 |
| **4** | 19.0 | 76000.0 | 1 | 0 | 0 | 0 |
| **5** | 27.0 | 58000.0 | 0 | 1 | 0 | 0 |
| **6** | 27.0 | 84000.0 | 0 | 1 | 0 | 1 |
| **7** | 27.0 | 150000.0 | 0 | 0 | 1 | 1 |
| **8** | 25.0 | 33000.0 | 1 | 0 | 0 | 0 |
| **9** | 35.0 | 65000.0 | 0 | 0 | 1 | 0 |
| **10** | 26.0 | 80000.0 | 0 | 1 | 0 | 0 |
| **11** | 26.0 | 52000.0 | 0 | 1 | 0 | 0 |
| **12** | 20.0 | 86000.0 | 1 | 0 | 0 | 0 |
| **13** | 32.0 | 18000.0 | 0 | 0 | 1 | 1 |
| **14** | 18.0 | 82000.0 | 1 | 0 | 0 | 0 |
| **15** | 29.0 | 80000.0 | 0 | 1 | 0 | 0 |
| **16** | 47.0 | 25000.0 | 0 | 1 | 0 | 1 |
| **17** | 45.0 | 26000.0 | 0 | 1 | 0 | 1 |
| **18** | 46.0 | 28000.0 | 1 | 0 | 0 | 1 |
| **19** | 27.0 | 28000.0 | 0 | 1 | 0 | 1 |

In [37]:

```
#dividing datasets into input x and output y
x = dataset.loc[:,['Age','salary','France','India','USA']]
y = dataset.loc[:,['Purchased']]
```

In [38]:

```
y = np.array(y)
y = y.ravel()
y
```

Out[38]:

```
array([0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1],
      dtype=int64)
```

In [39]:

```
#divide input x and output y into training and testing sets
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=41)
```

In [40]:

```
print("Total records(rows) in X_train:",len(X_train))
print("Total records(rows) in y_train:",len(y_train))
print("Total records(rows) in X_test:",len(X_test))
print("Total records(rows) in y_test:",len(y_test))
```

```
Total records(rows) in X_train: 16
Total records(rows) in y_train: 16
Total records(rows) in X_test: 4
Total records(rows) in y_test: 4
```

In [41]:

```
#Since all input columns(features in X) values should be in common scale (0 to 1)
#so do the feature scaling
from sklearn.preprocessing import MinMaxScaler
sc=MinMaxScaler()
X_train=sc.fit_transform(X_train)
X_test=sc.transform(X_test)
```

In [42]:

```
X_train=np.array(X_train)
X_test=np.array(X_test)
```

In [43]:

```
X_train[:5]
```

Out[43]:

```
array([[0.        , 0.43939394, 1.        , 0.        , 0.        ],
       [0.35714286, 0.46969697, 0.        , 1.        , 0.        ],
       [1.        , 0.0530303 , 0.        , 1.        , 0.        ],
       [0.92857143, 0.06060606, 0.        , 1.        , 0.        ],
       [0.46428571, 0.        , 0.        , 0.        , 1.        ]])
```

In [44]:

```
X_train[:5]
```

Out[44]:

```
array([[0.        , 0.43939394, 1.        , 0.        , 0.        ],
       [0.35714286, 0.46969697, 0.        , 1.        , 0.        ],
       [1.        , 0.0530303 , 0.        , 1.        , 0.        ],
       [0.92857143, 0.06060606, 0.        , 1.        , 0.        ],
       [0.46428571, 0.        , 0.        , 0.        , 1.        ]])
```

In [45]:

```
y_train
```

Out[45]:

```
array([0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0], dtype=int64)
```

In [46]:

```
y_train
```

Out[46]:

```
array([0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0], dtype=int64)
```

```
In [47]: y_test
```

```
Out[47]: array([0, 0, 0, 1], dtype=int64)
```

```
In [ ]:
```