In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

In [ ]:
```
df = pd.read_csv("Data.csv")
df
```

Out[ ]:

	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 [ ]:
```
X = df.iloc[:,:3].values    # independent variable
y = df.iloc[:,-1].values    # dependent variable
```

In [ ]:
```
print(X)
```

```
[['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]]
```

In [ ]:
```
print(y)
```

```
['No' 'Yes' 'No' 'No' 'Yes' 'Yes' 'No' 'Yes' 'No' 'Yes']
```

In [ ]:
```
# replace missing value with mean
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(missing_values=np.nan, strategy="mean")
X[:,1:3] = imputer.fit_transform(X[:,1:3]).round(2)
```

In [ ]:
```
print(X)
```

```
[['France' 44.0 72000.0]
 ['Spain' 27.0 48000.0]
 ['Germany' 30.0 54000.0]
 ['Spain' 38.0 61000.0]
 ['Germany' 40.0 63777.78]
 ['France' 35.0 58000.0]
 ['Spain' 38.78 52000.0]
 ['France' 48.0 79000.0]
 ['Germany' 50.0 83000.0]
 ['France' 37.0 67000.0]]
```

In [ ]:
```
# encoding categorical independent feature
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
ct = ColumnTransformer(transformers=[("encoder", OneHotEncoder(), [0])],remainder="passthrough")
X = np.array(ct.fit_transform(X))
```

In [ ]:
```
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]
 [0.0 1.0 0.0 40.0 63777.78]
 [1.0 0.0 0.0 35.0 58000.0]
 [0.0 0.0 1.0 38.78 52000.0]
 [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]]
```

In [ ]:
```
# encoding categorical dependent feature
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)
```

In [ ]:
```
print(y)
```

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

In [ ]:
```
# split training set dan test set
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 [ ]:
```
print(X_train)
```

```
[[0.0 0.0 1.0 38.78 52000.0]
 [0.0 1.0 0.0 40.0 63777.78]
 [1.0 0.0 0.0 44.0 72000.0]
 [0.0 0.0 1.0 38.0 61000.0]
 [0.0 0.0 1.0 27.0 48000.0]
 [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 35.0 58000.0]]
```

In [ ]:
```
print(X_test)
```

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

In [ ]:
```
print(y_train)
```

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

In [ ]:
```
print(y_test)
```

```
[0 1]
```

In [ ]:
```
# feature scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train[:,3:] = sc.fit_transform(X_train[:,3:]).round(2)
X_test[:,3:] = sc.transform(X_test[:,3:]).round(2)
```

In [ ]:
```
print(X_train)
```

```
[[0.0 0.0 1.0 -0.19 -1.08]
 [0.0 1.0 0.0 -0.01 -0.07]
 [1.0 0.0 0.0 0.57 0.63]
 [0.0 0.0 1.0 -0.3 -0.31]
 [0.0 0.0 1.0 -1.9 -1.42]
 [1.0 0.0 0.0 1.15 1.23]
 [0.0 1.0 0.0 1.44 1.57]
 [1.0 0.0 0.0 -0.74 -0.56]]
```

In [ ]:
```
print(X_test)
```

```
[[0.0 1.0 0.0 -1.47 -0.91]
 [1.0 0.0 0.0 -0.45 0.21]]
```