# Practical: 4A :Standardization

# Aim: Demonstrate the purpose of feature scaling and show that feature scaling does not effect the distribution of the data.

In [1]:
```
import pandas as pd
import numpy as np

df = pd.read_csv('Social_Network_Ads.csv', usecols=['Age','EstimatedSalary','P
df
```

Out[1]:

| | Age | EstimatedSalary | Purchased |
|---|---|---|---|
| **0** | 19 | 19000 | 0 |
| **1** | 35 | 20000 | 0 |
| **2** | 26 | 43000 | 0 |
| **3** | 27 | 57000 | 0 |
| **4** | 19 | 76000 | 0 |
| **...** | ... | ... | ... |
| **395** | 46 | 41000 | 1 |
| **396** | 51 | 23000 | 1 |
| **397** | 50 | 20000 | 1 |
| **398** | 36 | 33000 | 0 |
| **399** | 49 | 36000 | 1 |

400 rows × 3 columns

In [2]:
```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(df.drop('Purchased',axis=1),d
 test_size=0.3,random_state=0)

x_train.shape
```

Out[2]: (280, 2)

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

Out[3]: (120, 2)

In [4]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

scaler.fit(x_train)
```

Out[4]:
```
StandardScaler()
```

In [5]:
```
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.fit_transform(x_test)

scaler.mean_
```

Out[5]:
```
array([3.71666667e+01, 6.95916667e+04])
```

In [6]:
```
x_train
```

Out[6]:

| | Age | EstimatedSalary |
|---|---|---|
| **92** | 26 | 15000 |
| **223** | 60 | 102000 |
| **234** | 38 | 112000 |
| **232** | 40 | 107000 |
| **377** | 42 | 53000 |
| **...** | ... | ... |
| **323** | 48 | 30000 |
| **192** | 29 | 43000 |
| **117** | 36 | 52000 |
| **47** | 27 | 54000 |
| **172** | 26 | 118000 |

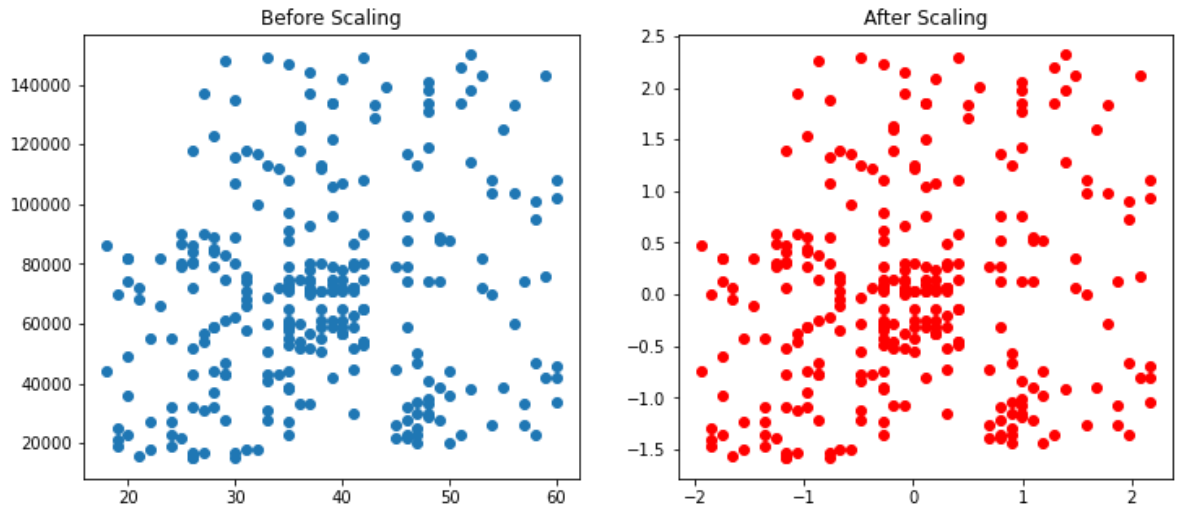
280 rows × 2 columns

In [7]:
```
x_train_scaled
```

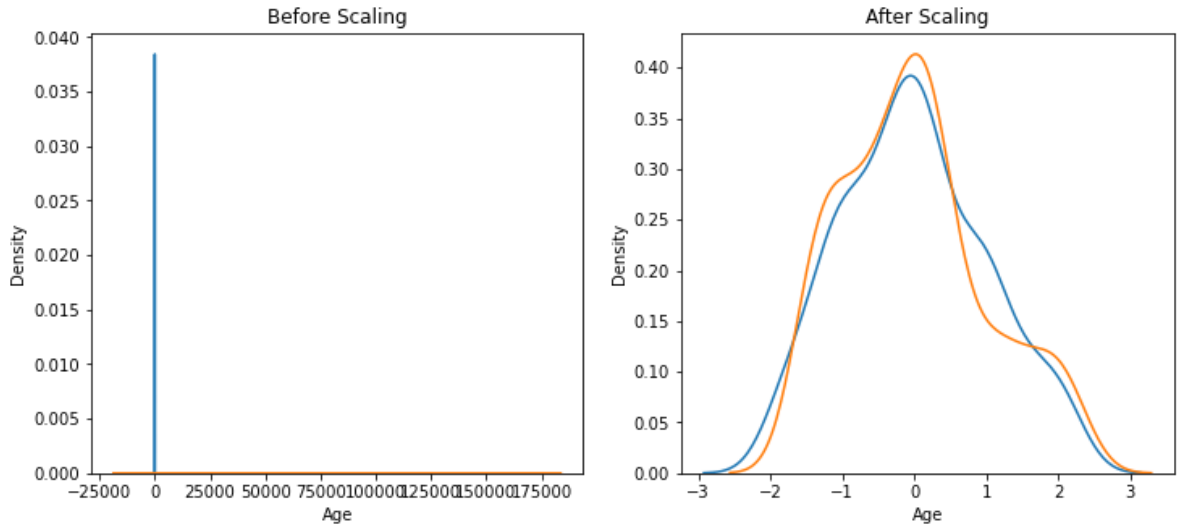
```
[ 2.17018137, -0.80415605],
[-1.35925203, -1.46929411],
[ 0.40546467,  2.2901819 ],
[ 0.79762394,  0.75747245],
[-0.96709276, -0.31253226],
[ 0.11134522,  0.75747245],
[-0.96709276,  0.55503912],
[ 0.30742485,  0.06341534],
[ 0.69958412, -1.26686079],
[-0.47689368, -0.0233418 ],
[-1.7514113 ,  0.3526058 ],
[-0.67297331,  0.12125343],
[ 0.40546467,  0.29476771],
[-0.28081405,  0.06341534],
[-0.47689368,  2.2901819 ],
[ 0.20938504,  0.03449629],
[ 1.28782302,  2.20342476],
[ 0.79762394,  0.26584866],
[-0.28081405,  0.15017248],
[ 0.0133054 , -0.54388463],
```

In [8]:
```
x_train_scaled = pd.DataFrame(x_train_scaled,columns=x_train.columns)
x_test_scaled = pd.DataFrame(x_test_scaled,columns=x_train.columns)
```

In [9]:
```
from matplotlib import pyplot as plt
fig, (ax1,ax2) = plt.subplots(ncols=2,figsize=(12,5))
ax1.scatter(x_train['Age'],x_train['EstimatedSalary'])
ax1.set_title('Before Scaling')
ax2.scatter(x_train_scaled['Age'],x_train_scaled['EstimatedSalary'],color='red
ax2.set_title('After Scaling')
plt.show()
```



In [11]:
```
import seaborn as sns
fig, (ax1,ax2) = plt.subplots(ncols=2,figsize=(12,5))
ax1.set_title('Before Scaling')
sns.kdeplot(x_train['Age'],ax =ax1)
sns.kdeplot(x_train['EstimatedSalary'],ax =ax1)
ax2.set_title('After Scaling')
sns.kdeplot(x_train_scaled['Age'],ax =ax2)
sns.kdeplot(x_train_scaled['EstimatedSalary'],ax =ax2)
plt.show()
```




In [ ]: