# Task-14

## GO_STP_5247

We have a dataset that contains multiple user's information through the social network who are interested in buying SUV Car or not.

```
import pandas as pd
import sklearn

from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
```

[2]
```
data=pd.read_csv('/content/User_Data.csv')
```

[3]
```
data.head()
```

| | User ID | Gender | Age | EstimatedSalary | Purchased |
|---|---|---|---|---|---|
| 0 | 15624510 | Male | 19 | 19000 | 0 |
| 1 | 15810944 | Male | 35 | 20000 | 0 |
| 2 | 15668575 | Female | 26 | 43000 | 0 |
| 3 | 15603246 | Female | 27 | 57000 | 0 |

✓ 0s completed at 5:25 PM

[4]
```
data.shape
```

```
(400, 5)
```

[5]
```
le = LabelEncoder()
data['Gender'] = le.fit_transform(data['Gender'])
data.head()
```

| | User ID | Gender | Age | EstimatedSalary | Purchased |
|---|---|---|---|---|---|
| 0 | 15624510 | 1 | 19 | 19000 | 0 |
| 1 | 15810944 | 1 | 35 | 20000 | 0 |
| 2 | 15668575 | 0 | 26 | 43000 | 0 |
| 3 | 15603246 | 0 | 27 | 57000 | 0 |
| 4 | 15804002 | 1 | 19 | 76000 | 0 |

[6]
```
data.drop(['User ID'],axis='columns',inplace=True)
```

[7]
```
y=data['Purchased']
X=data.drop(['Purchased'],axis='columns')
```

[8]
```
scaler = StandardScaler()
scaler.fit(X)
X_scaled=scaler.transform(X)
```

[9]
```
print(X_scaled)
```

```
[9] print(X_scaled)
```

```
[[ 1.02020406 -1.78179743 -1.49004624]
 [ 1.02020406 -0.25358736 -1.46068138]
 [-0.98019606 -1.11320552 -0.78528968]
 ...
 [-0.98019606  1.17910958 -1.46068138]
 [ 1.02020406 -0.15807423 -1.07893824]
 [-0.98019606  1.08359645 -0.99084367]]
```

```
[10] X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.20, random_state=42)
```

```
[11] knn=KNeighborsClassifier(n_neighbors=5)
```

```
[12] knn.fit(X_train,y_train)
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='uniform')
```

```
[15] y_pred=knn.predict(X_test)
     cm=confusion_matrix(y_test,y_pred)

     print(cm)
     print("correct predicition",accuracy_score(y_test,y_pred))
     print("wrong predicition",(1-accuracy_score(y_test,y_pred)))
```

```
[[48  4]
 [ 2 26]]
correct predicition 0.925
wrong predicition 0.07499999999999996
```