Knowledge Discovery & Data Mining Lab-04

Name: Gurvinder Kaur Matharu
PRN: 20190802077

Steps:

1. Importing the libraries
2. Importing Datasets
3. Handling of missing values
4. Handling Categorical data
5. Splitting the datset into training and testing datasets
6. Feature Scaling

```
# importing the libraries
import numpy as np
import pandas as pd
from sklearn.impute import SimpleImputer # for handling missing data
from sklearn.preprocessing import LabelEncoder # for encoding categorical data
from sklearn.model_selection import train_test_split # for splitting the dataset into training and testing dataset
from sklearn.preprocessing import StandardScaler # for feature scaling
```

```
# importing dataset
df=pd.read_csv('Fish_data.csv')
```

```
df.head()
```

	Species	Weight	Length1	Length2	Length3	Height	Width
0	Bream	242.0	23.2	25.4	30.0	NaN	4.0200
1	Bream	290.0	24.0	26.3	31.2	12.4800	4.3056
2	Bream	340.0	23.9	26.5	31.1	12.3778	4.6961
3	Bream	363.0	26.3	29.0	33.5	12.7300	4.4555
4	Bream	430.0	26.5	29.0	34.0	12.4440	5.1340

```
df.shape
```

```
(159, 7)
```

```
df.isnull().sum() # checking for missing values
```

```
Species    0
Weight     0
Length1    0
Length2    0
Length3    0
Height     2
Width      0
dtype: int64
```

```
X = df.iloc[:,1:].values
y = df.iloc[:,0].values
```

```
#X
```

```
# handling missing values
imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
imputer = imputer.fit(X[:,1:])
X[:,1:] = imputer.transform(X[:,1:])
#print(X)
```

```
# handling categorical data
labelencoder_y = LabelEncoder()
y = labelencoder_y.fit_transform(y)
print(y)
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 4
 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 6 6 6 6 6 6 1 1 1 1 1 1 1 1 1 1 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 5 5 5
 5 5 5 5 5 5 5 5 5 5]
```

```
# splitting the dataset into training and testing datasets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

```
# feature scaling
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

The dataset can now be fed to a machine learning algorithm.

```
# importing the 2nd dataset
data = pd.read_csv('hr_data.csv')
```

```
data.head()
```

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	work_accic
0	0.38	0.53	2	157.0	3.0	
1	0.80	0.86	5	262.0	6.0	
2	0.11	0.88	7	272.0	4.0	
3	0.72	0.87	5	223.0	5.0	
4	0.37	0.52	2	NaN	NaN	

```
data.isna().sum() # checking for missing values
```

```
satisfaction_level        0
last_evaluation           0
number_project            0
average_montly_hours    368
time_spend_company      151
work_accident             0
left                      0
promotion_last_5years     0
department                0
salary                    0
dtype: int64
```

```
X1 = data.iloc[:,:-1].values
y1 = data.iloc[:,-1].values

y1
```

```
array(['low', 'medium', 'medium', ..., 'low', 'low', 'low'], dtype=object)
```

```
# encoding the categorical data
labelencoder_x1 = LabelEncoder()
X1[:,6] = labelencoder_x1.fit_transform(X1[:,6])
X1[:,8] = labelencoder_x1.fit_transform(X1[:,8])

labelencoder_y1 = LabelEncoder()
y1 = labelencoder_y.fit_transform(y1)
print(y1)
```

```
[1 2 2 ... 1 1 1]
```

```
# handling the missing values
imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
imputer = imputer.fit(X1[:,1:])
X1[:,1:] = imputer.transform(X1[:,1:])
```

```
# splitting the dataset into training and testing datasets
X_train1, X_test1, y_train1, y_test1 = train_test_split(X1, y1, test_size=0.2, random_state=0)
```

```
# feature scaling
sc_X1 = StandardScaler()
X_train1 = sc_X.fit_transform(X_train1)
X_test1 = sc_X.transform(X_test1)
```

This dataset can now be fed to a machine learning algorithm.