Data preprocessing step

```
#importing libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

```
#importing datasets
data_set=pd.read_csv('/content/Iris dataset - Iris dataset.csv')
print(data_set)
```

```
     sepal_length  sepal_width  petal_length  petal_width    species
0             5.1          3.5           1.4          0.2     setosa
1             4.9          3.0           1.4          0.2     setosa
2             4.7          3.2           1.3          0.2     setosa
3             4.6          3.1           1.5          0.2     setosa
4             5.0          3.6           1.4          0.2     setosa
..            ...          ...           ...          ...        ...
145           6.7          3.0           5.2          2.3  virginica
146           6.3          2.5           5.0          1.9  virginica
147           6.5          3.0           5.2          2.0  virginica
148           6.2          3.4           5.4          2.3  virginica
149           5.9          3.0           5.1          1.8  virginica

[150 rows x 5 columns]
```

```
#extracting independent and dependent variables
x=data_set.iloc[:, [0,1,2,3]].values
y=data_set.iloc[:, 4].values
```

```
#splittig the data into training and testing
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=0)
```

```
#feature scaling
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
x_train= scaler.fit_transform(x_train)
x_test= scaler.transform(x_test)
print(x_train)
```

```
[[ 0.61303014  0.10850105  0.94751783  0.73603967]
 [-0.56776627 -0.12400121  0.38491447  0.34808318]
 [-0.80392556  1.03851009 -1.30289562 -1.3330616 ]
 [ 0.25879121 -0.12400121  0.60995581  0.73603967]
 [ 0.61303014 -0.58900572  1.00377816  1.25331499]
 [-0.80392556 -0.82150798  0.04735245  0.21876435]
 [-0.21352735  1.73601687 -1.19037495 -1.20374277]
 [ 0.14071157 -0.82150798  0.72247648  0.47740201]
 [ 0.02263193 -0.12400121  0.21613346  0.34808318]
 [-0.09544771 -1.05401024  0.10361279 -0.03987331]
 [ 1.0853487  -0.12400121  0.94751783  1.12399616]
 [-1.39432376  0.34100331 -1.41541629 -1.3330616 ]
 [ 1.20342834  0.10850105  0.72247648  1.38263382]
 [-1.04008484  1.03851009 -1.24663528 -0.81578628]
 [-0.56776627  1.50351461 -1.30289562 -1.3330616 ]
 [-1.04008484 -2.4490238  -0.1776889  -0.29851096]
 [ 0.73110978 -0.12400121  0.94751783  0.73603967]
 [ 0.96726906  0.57350557  1.0600385   1.64127148]
 [ 0.14071157 -1.98401928  0.66621615  0.34808318]
 [ 0.96726906 -1.2865125   1.11629884  0.73603967]
 [-0.33160699 -1.2865125   0.04735245 -0.16919214]
 [ 2.14806547 -0.12400121  1.28507985  1.38263382]
 [ 0.49495049  0.57350557  0.49743514  0.47740201]
 [-0.44968663 -1.51901476 -0.00890789 -0.16919214]
 [ 0.49495049 -0.82150798  0.60995581  0.73603967]
 [ 0.49495049 -0.58900572  0.72247648  0.34808318]
 [-1.15816448 -1.2865125   0.38491447  0.60672084]
 [ 0.49495049 -1.2865125   0.66621615  0.8653585 ]
 [ 1.32150798  0.34100331  0.49743514  0.21876435]
 [ 0.73110978 -0.12400121  0.77873682  0.99467733]
 [ 0.14071157  0.80600783  0.38491447  0.47740201]
 [-1.27624412  0.10850105 -1.24663528 -1.3330616 ]
 [-0.09544771 -0.82150798  0.72247648  0.8653585 ]
 [-0.33160699 -0.82150798  0.21613346  0.08944552]
 [-0.33160699 -0.35650346 -0.12142856  0.08944552]
 [-0.44968663 -1.2865125   0.10361279  0.08944552]
 [ 0.25879121 -0.12400121  0.4411748   0.21876435]
 [ 1.55766726  0.34100331  1.22881951  0.73603967]
 [-0.68584591  1.50351461 -1.30289562 -1.3330616 ]
```

```
[-1.86664232 -0.12400121 -1.52793696 -1.46238043]
[ 0.61303014 -0.82150798  0.83499716  0.8653585 ]
[-0.21352735 -0.12400121  0.21613346 -0.03987331]
[-0.56776627  0.80600783 -1.19037495 -1.3330616 ]
[-0.21352735  3.13103043 -1.30289562 -1.07442394]
[ 1.20342834  0.10850105  0.60995581  0.34808318]
[-1.5124034   0.10850105 -1.30289562 -1.3330616 ]
[ 0.02263193 -0.12400121  0.72247648  0.73603967]
[-0.9220052  -1.2865125  -0.45899058 -0.16919214]
[-1.5124034   0.80600783 -1.35915595 -1.20374277]
[ 0.37687085 -1.98401928  0.38491447  0.34808318]
[ 1.55766726  1.27101235  1.28507985  1.64127148]
[-0.21352735 -0.35650346  0.21613346  0.08944552]
[-1.27624412 -0.12400121 -1.35915595 -1.46238043]
[ 1.43958762 -0.12400121  1.17255917  1.12399616]
[ 1.20342834  0.34100331  1.0600385   1.38263382]
[ 0.73110978 -0.12400121  1.11629884  1.25331499]
[ 0.61303014 -0.58900572  1.00377816  1.12399616]
[-0.9220052   1.73601687 -1.24663528 -1.3330616 ]
```

```
#fitting KNN classifier to the training set
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2)
knn.fit(x_train, y_train)
```

```
KNeighborsClassifier
KNeighborsClassifier()
```

```
#predicting the test set result
y_pred=knn.predict(x_test)
print(y_pred)
```

```
['virginica' 'versicolor' 'setosa' 'virginica' 'setosa' 'virginica'
 'setosa' 'versicolor' 'versicolor' 'versicolor' 'virginica' 'versicolor'
 'versicolor' 'versicolor' 'versicolor' 'setosa' 'versicolor' 'versicolor'
 'setosa' 'setosa' 'virginica' 'versicolor' 'setosa' 'setosa' 'virginica'
 'setosa' 'setosa' 'versicolor' 'versicolor' 'setosa']
```

```
y_pred=knn.predict([[5,3.5,1.3,0.3]])
print(y_pred)
```

```
['virginica']
```

```
y_pred=knn.predict([[6,2.3,5,1]])
print(y_pred)
```

```
['virginica']
```

```
y_pred=knn.predict([[6,2.5,7,5]])
print(y_pred)
```

```
['virginica']
```