```
# Importing the packages
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import sklearn
```

```
# Importing the dataset
dataset = pd.read_csv('/Fish.csv')
dataset.tail(5)
```

	Species	Weight	Length1	Length2	Length3	Height	Width
154	Smelt	12.2	11.5	12.2	13.4	2.0904	1.3936
155	Smelt	13.4	11.7	12.4	13.5	2.4300	1.2690
156	Smelt	12.2	12.1	13.0	13.8	2.2770	1.2558
157	Smelt	19.7	13.2	14.3	15.2	2.8728	2.0672
158	Smelt	19.9	13.8	15.0	16.2	2.9322	1.8792

```
# if your dataset contains missing value, check which column has missing values
dataset.isnull().sum()
```

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

```
## encoding the categorical features
from sklearn import preprocessing

col_cat = ['Species']

lab_en= preprocessing.LabelEncoder()

for c in col_cat:
    dataset[c]= lab_en.fit_transform(dataset[c])

dataset.head()
```

	Species	Weight	Length1	Length2	Length3	Height	Width
0	0	242.0	23.2	25.4	30.0	11.5200	4.0200
1	0	290.0	24.0	26.3	31.2	12.4800	4.3056
2	0	340.0	23.9	26.5	31.1	12.3778	4.6961
3	0	363.0	26.3	29.0	33.5	12.7300	4.4555

```
## correlation values of features with target label
corr_col = np.abs(dataset.corr()['Weight']).sort_values(ascending=False)
corr_col = corr_col.rename_axis('Col').reset_index(name='Correlation')
corr_col
```

	Col	Correlation
0	Weight	1.000000
1	Length3	0.923044
2	Length2	0.918618
3	Length1	0.915712
4	Width	0.886507
5	Height	0.724345
6	Species	0.312960

```
## features and target label
X = dataset.iloc[:, [2,3,4,5,6]].values
y = dataset.iloc[:, 1].values
```

```
X[:3]
```

```
array([[23.2   , 25.4   , 30.    , 11.52  ,  4.02  ],
       [24.    , 26.3   , 31.2   , 12.48  ,  4.3056],
       [23.9   , 26.5   , 31.1   , 12.3778,  4.6961]])
```

```
## train-test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0,
```

```
##scaling the features
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_squared_error
```

```
reg = MLPRegressor(hidden_layer_sizes=(256, 256),activation="relu" ,random_state=1, max_iter=
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/_multilayer_perceptron.py
  ConvergenceWarning,
```

```
y_pred  =  reg.predict(X_test)
y_pred, y_test
```

```
(array([ 392.59626657,  116.54948231,  158.15496037,  164.38019259,
         591.06154954,  895.3854849 ,  721.39537114,  339.94954464,
        1035.81400622,  109.33138073,  252.81448635,  511.78826973,
         846.9549219 , 1133.81779565,   74.53420828,   77.76858553,
         143.98943808, 1509.30170913,  125.66372784,  760.59654259,
          81.28947564,  403.21615419,  127.53209769, 1707.64715833,
          93.06454816,  451.96978945,  710.8529888 ,  262.77453033,
         901.4542246 ,   10.56573481,  634.64629063,  143.71288469]),
 array([ 390. ,    0. ,  170. ,  160. ,  556. ,  900. ,  800. ,  300. ,
         975. ,  115. ,  200. ,  456. , 1000. , 1000. ,   60. ,   78. ,
         145. , 1600. ,  130. ,  720. ,   55. ,  390. ,  120. , 1650. ,
          90. ,  450. ,  700. ,  270. ,  850. ,    9.7,  650. ,  110. ]))
```

```
print("Error rate ", mean_squared_error(y_pred, y_test))
```

```
Error rate  2884.1377617119733
```

✓ 0s completed at 10:57 AM