## IMPORTING LIBRARIES

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
```

2.Load the dataset into the Google Colab

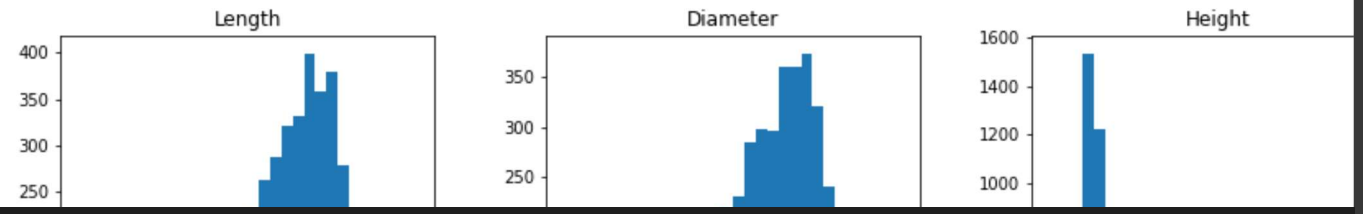
```
df=pd.read_csv("abalone.csv")
```

```
df['age'] = df['Rings']+1.5
df = df.drop('Rings', axis = 1)
```

3.UNIVARIATE ANALYSIS

```
df.hist(figsize=(20,10), grid=False, layout=(2, 4), bins = 30)
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f70473e9250>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f7048ae63d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f7047379d50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f704733e390>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7f70472f5990>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f70472aaf90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f704726b650>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f7047224b90>]],
      dtype=object)
```




```
df.groupby('Sex')[['Length', 'Diameter', 'Height', 'Whole weight', 'Shucked weight',
       'Viscera weight', 'Shell weight', 'age']].mean().sort_values('age')
```

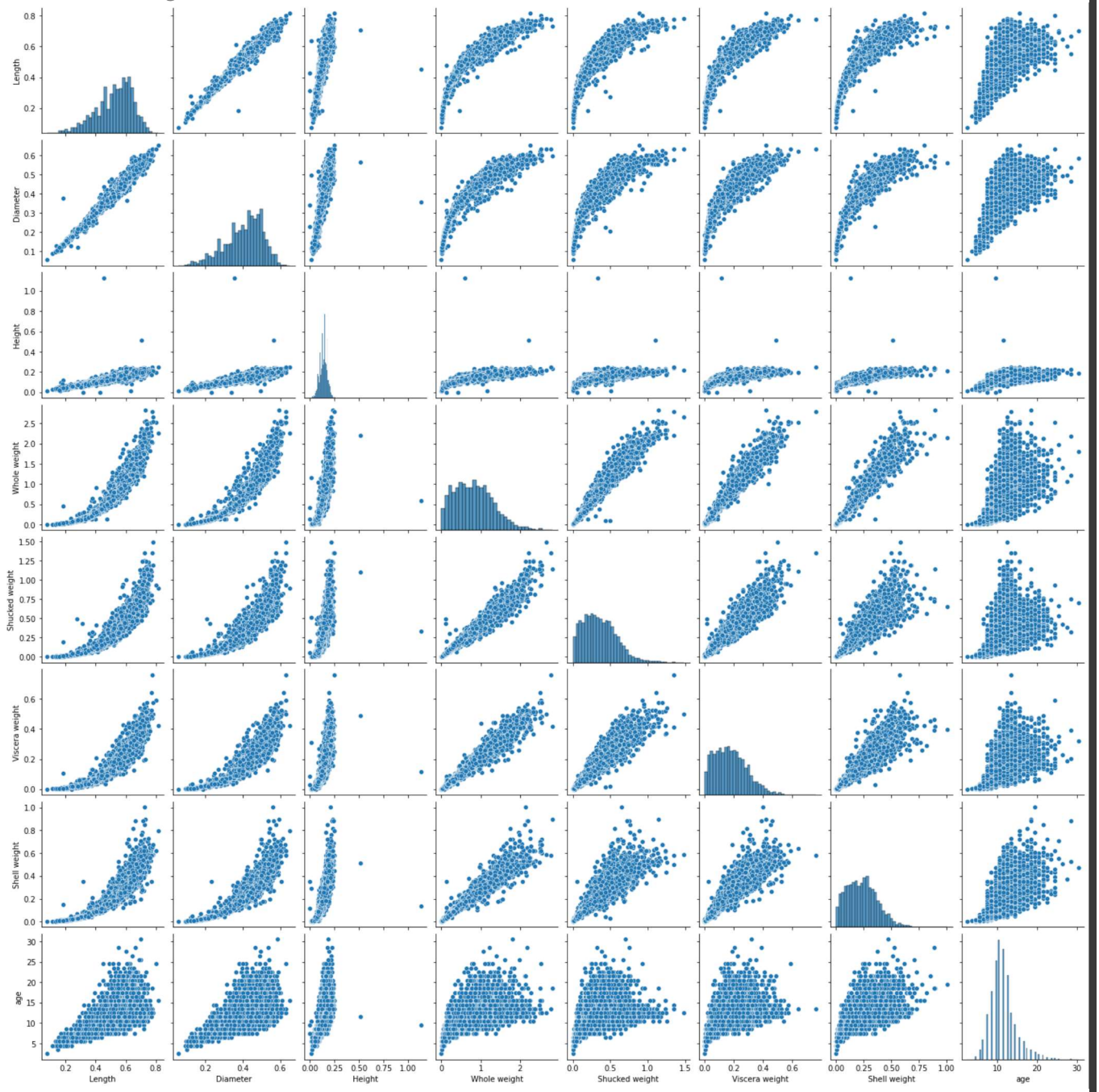
| Sex | Length | Diameter | Height | Whole weight | Shucked weight | Viscera weight | Shell w |
|---|---|---|---|---|---|---|---|
| I | 0.427746 | 0.326494 | 0.107996 | 0.431363 | 0.191035 | 0.092010 | 0.1 |
| M | 0.561391 | 0.439287 | 0.151381 | 0.991459 | 0.432946 | 0.215545 | 0.2 |
| F | 0.579093 | 0.454732 | 0.158011 | 1.046532 | 0.446188 | 0.230689 | 0.3 |


## BIVARIATE ANALYSIS,MULTIVARIATE ANALYSIS



```
numerical_features = df.select_dtypes(include = [np.number]).columns
sns.pairplot(df[numerical_features])
```

```
<seaborn.axisgrid.PairGrid at 0x7f7046ddc490>
```




## 4. Descriptive statistics

```
df.describe()
```

| | Length | Diameter | Height | Whole weight | Shucked weight | Viscera weigh |
|---|---|---|---|---|---|---|
| count | 4177.000000 | 4177.000000 | 4177.000000 | 4177.000000 | 4177.000000 | 4177.00000 |
| mean | 0.523992 | 0.407881 | 0.139516 | 0.828742 | 0.359367 | 0.18059 |
| std | 0.120093 | 0.099240 | 0.041827 | 0.490389 | 0.221963 | 0.10961 |
| min | 0.075000 | 0.055000 | 0.000000 | 0.002000 | 0.001000 | 0.00050 |
| 25% | 0.450000 | 0.350000 | 0.115000 | 0.441500 | 0.186000 | 0.09350 |
| 50% | 0.545000 | 0.425000 | 0.140000 | 0.799500 | 0.336000 | 0.17100 |
| 75% | 0.615000 | 0.480000 | 0.165000 | 1.153000 | 0.502000 | 0.25300 |
| max | 0.815000 | 0.650000 | 1.130000 | 2.825500 | 1.488000 | 0.76000 |

5.Check for Missing Values

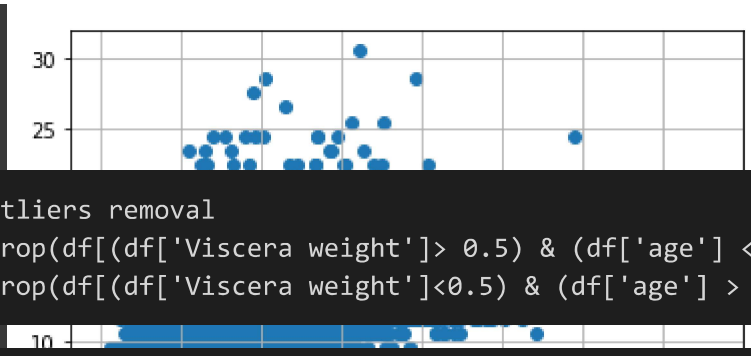
```
df.isnull().sum()
```

```
Sex               0
Length            0
Diameter          0
Height            0
Whole weight      0
Shucked weight    0
Viscera weight    0
Shell weight      0
age               0
dtype: int64
```

6. OUTLIER HANDLING

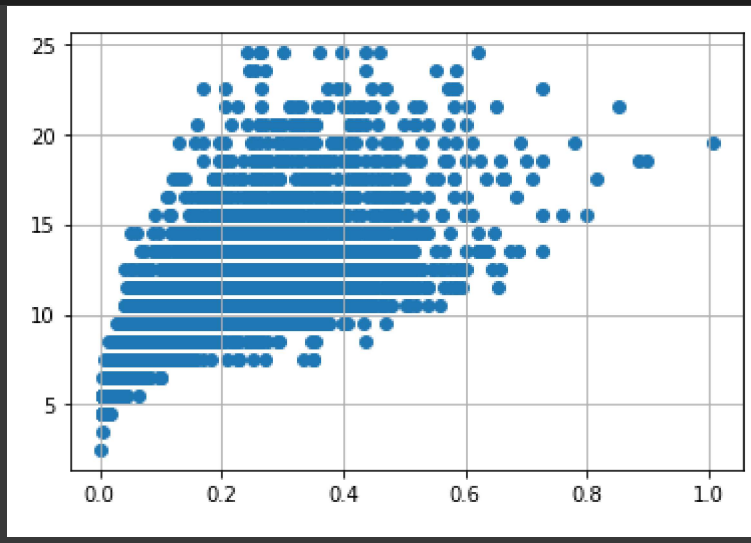
```
df = pd.get_dummies(df)
dummy_data = df.copy()
```

```
var = 'Viscera weight'
plt.scatter(x = df[var], y = df['age'],)
plt.grid(True)
```



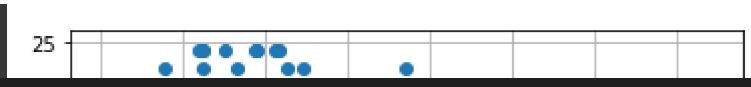

```
# outliers removal
df.drop(df[(df['Viscera weight']> 0.5) & (df['age'] < 20)].index, inplace=True)
df.drop(df[(df['Viscera weight']<0.5) & (df['age'] > 25)].index, inplace=True)
```

```
var = 'Shell weight'
plt.scatter(x = df[var], y = df['age'],)
plt.grid(True)
#Outliers removal
df.drop(df[(df['Shell weight']> 0.6) & (df['age'] < 25)].index, inplace=True)
df.drop(df[(df['Shell weight']<0.8) & (df['age'] > 25)].index, inplace=True)
```



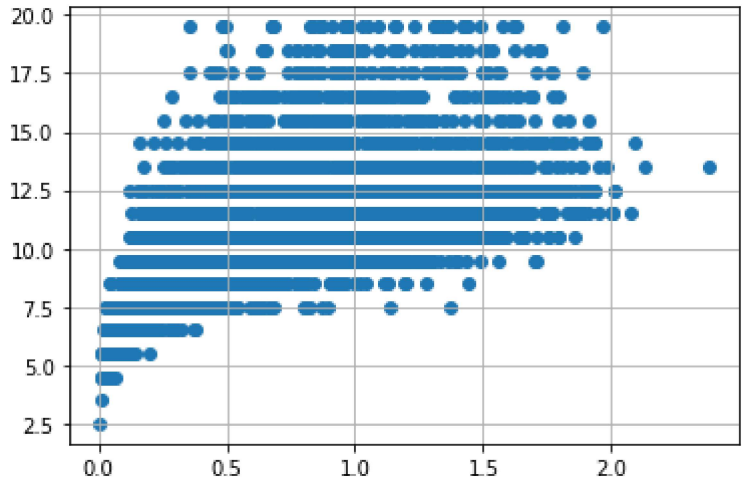

```
var = 'Shucked weight'
plt.scatter(x = df[var], y = df['age'],)
plt.grid(True)

#Outlier removal
df.drop(df[(df['Shucked weight']>= 1) & (df['age'] < 20)].index, inplace=True)
df.drop(df[(df['Shucked weight']<1) & (df['age'] > 20)].index, inplace=True)
```



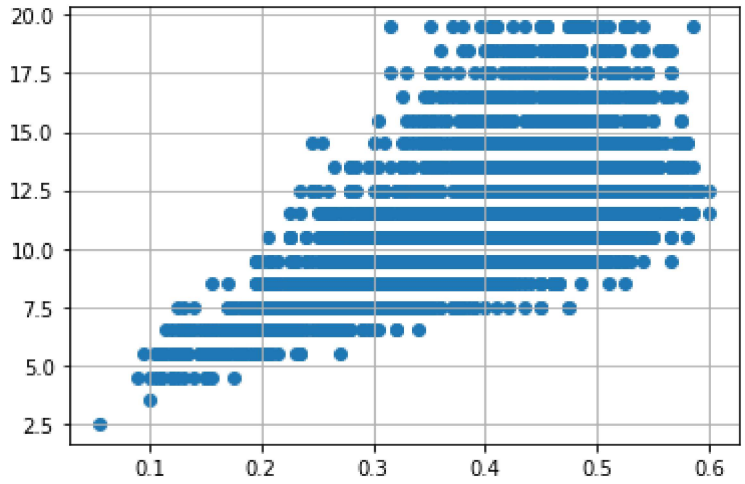

```
var = 'Whole weight'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)

df.drop(df[(df['Whole weight'] >= 2.5) &
          (df['age'] < 25)].index, inplace = True)
df.drop(df[(df['Whole weight']<2.5) & (
df['age'] > 25)].index, inplace = True)
```

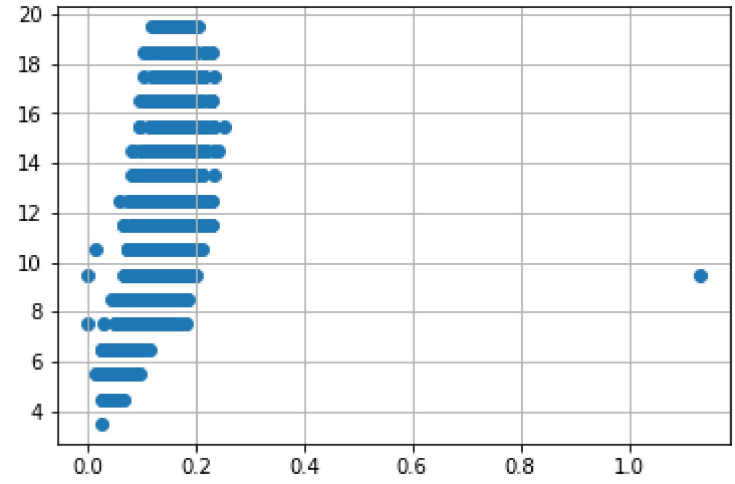



```
var = 'Diameter'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)

df.drop(df[(df['Diameter'] <0.1) &
          (df['age'] < 5)].index, inplace = True)
df.drop(df[(df['Diameter']<0.6) & (
df['age'] > 25)].index, inplace = True)
df.drop(df[(df['Diameter']>=0.6) & (
df['age'] < 25)].index, inplace = True)
```

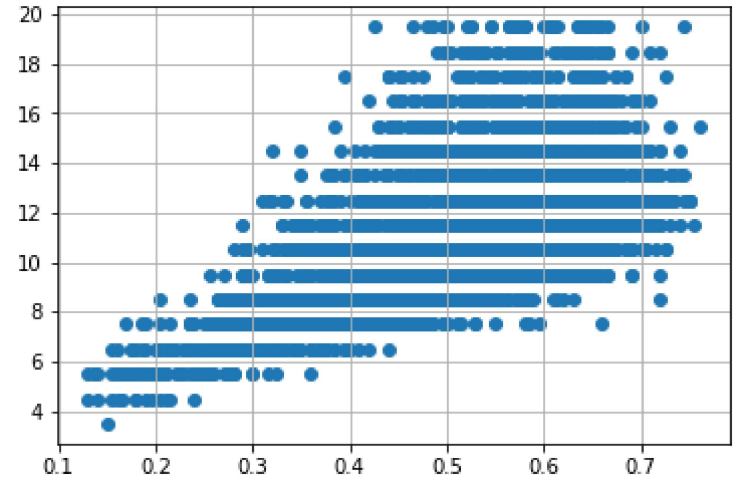


```
var = 'Height'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)
df.drop(df[(df['Height'] > 0.4) &
          (df['age'] < 15)].index, inplace = True)
df.drop(df[(df['Height']<0.4) & (
df['age'] > 25)].index, inplace = True)
```




```
var = 'Length'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)

df.drop(df[(df['Length'] <0.1) &
          (df['age'] < 5)].index, inplace = True)
df.drop(df[(df['Length']<0.8) & (
df['age'] > 25)].index, inplace = True)
df.drop(df[(df['Length']>=0.8) & (
df['age'] < 25)].index, inplace = True)
```



## 7.Categorical columns

```
numerical_features = df.select_dtypes(include = [np.number]).columns
categorical_features = df.select_dtypes(include = [np.object]).columns
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: DeprecationWarning: `np
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/relea
```

```
numerical_features
```

```
Index(['Length', 'Diameter', 'Height', 'Whole weight', 'Shucked weight',
       'Viscera weight', 'Shell weight', 'age', 'Sex_F', 'Sex_I', 'Sex_M'],
      dtype='object')
```

```
categorical_features
```

```
Index([], dtype='object')
```

## ENCODING

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
print(df.Sex_F.value_counts())
print(df.Sex_M.value_counts())
print(df.Sex_I.value_counts())
```

```
0    2768
1    1227
Name: Sex_F, dtype: int64
0    2561
1    1434
Name: Sex_M, dtype: int64
0    2661
1    1334
Name: Sex_I, dtype: int64
```

## 8.Split the dependent and independent variables

```
x=df.iloc[:,:5]
x
```

| | Length | Diameter | Height | Whole weight | Shucked weight |
|---|---|---|---|---|---|
| **0** | 0.455 | 0.365 | 0.095 | 0.5140 | 0.2245 |
| **1** | 0.350 | 0.265 | 0.090 | 0.2255 | 0.0995 |
| **2** | 0.530 | 0.420 | 0.135 | 0.6770 | 0.2565 |
| **3** | 0.440 | 0.365 | 0.125 | 0.5160 | 0.2155 |
| **4** | 0.330 | 0.255 | 0.080 | 0.2050 | 0.0895 |
| **...** | ... | ... | ... | ... | ... |
| **4172** | 0.565 | 0.450 | 0.165 | 0.8870 | 0.3700 |
| **4173** | 0.590 | 0.440 | 0.135 | 0.9660 | 0.4390 |
| **4174** | 0.600 | 0.475 | 0.205 | 1.1760 | 0.5255 |
| **4175** | 0.625 | 0.485 | 0.150 | 1.0945 | 0.5310 |

```
y=df.iloc[:,5:]
y
```

| | Viscera weight | Shell weight | age | Sex_F | Sex_I | Sex_M |
|---|---|---|---|---|---|---|
| **0** | 0.1010 | 0.1500 | 16.5 | 0 | 0 | 1 |
| **1** | 0.0485 | 0.0700 | 8.5 | 0 | 0 | 1 |
| **2** | 0.1415 | 0.2100 | 10.5 | 1 | 0 | 0 |
| **3** | 0.1140 | 0.1550 | 11.5 | 0 | 0 | 1 |
| **4** | 0.0395 | 0.0550 | 8.5 | 0 | 1 | 0 |
| **...** | ... | ... | ... | ... | ... | ... |
| **4172** | 0.2390 | 0.2490 | 12.5 | 1 | 0 | 0 |
| **4173** | 0.2145 | 0.2605 | 11.5 | 0 | 0 | 1 |
| **4174** | 0.2875 | 0.3080 | 10.5 | 0 | 0 | 1 |
| **4175** | 0.2610 | 0.2960 | 11.5 | 1 | 0 | 0 |
| **4176** | 0.3765 | 0.4950 | 13.5 | 0 | 0 | 1 |

3995 rows × 6 columns

## 10.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=0.2)
```

## 11.Model building

```
from sklearn.linear_model import LinearRegression
mlr=LinearRegression()
mlr.fit(x_train,y_train)
```

```
LinearRegression()
```

## 12 , 13 Train and Test the model

```
x_test[0:5]
```

| | Length | Diameter | Height | Whole weight | Shucked weight |
|---|---|---|---|---|---|
| **3864** | 0.320 | 0.235 | 0.065 | 0.1385 | 0.0580 |
| **504** | 0.625 | 0.485 | 0.190 | 1.1745 | 0.4385 |
| **1729** | 0.665 | 0.525 | 0.175 | 1.4430 | 0.6635 |
| **1871** | 0.530 | 0.430 | 0.160 | 0.7245 | 0.3210 |
| **3491** | 0.550 | 0.440 | 0.160 | 0.9850 | 0.4645 |

```
y_test[0:5]
```

| | Viscera weight | Shell weight | age | Sex_F | Sex_I | Sex_M |
|---|---|---|---|---|---|---|
| **3864** | 0.0225 | 0.050 | 6.5 | 0 | 1 | 0 |
| **504** | 0.2305 | 0.420 | 18.5 | 1 | 0 | 0 |
| **1729** | 0.3845 | 0.353 | 12.5 | 0 | 0 | 1 |
| **1871** | 0.1275 | 0.240 | 10.5 | 0 | 1 | 0 |
| **3491** | 0.2010 | 0.270 | 9.5 | 0 | 0 | 1 |

## 9.Feature Scaling

```
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()
x_train=ss.fit_transform(x_train)
```

```
mlrpred=mlr.predict(x_test[0:9])
```

```
mlrpred
```

```
array([[ 0.02956588,  0.03874939,  7.89084417,  0.0272362 ,  0.79781766,
         0.17494614],
       [ 0.26738808,  0.3647421 , 14.55216195,  0.52727525,  0.02227635,
         0.45044841],
       [ 0.31302449,  0.38491237, 12.37974513,  0.4834383 , -0.02082715,
         0.53738885],
       [ 0.15627382,  0.22358855, 11.91867117,  0.35173995,  0.27909171,
         0.36916834],
       [ 0.21094798,  0.26926134, 11.29519549,  0.36151335,  0.20069245,
         0.4377942 ],
       [ 0.04272371,  0.07381584,  9.32869064,  0.11975987,  0.66890248,
         0.21133765],
       [ 0.15909885,  0.22694516, 12.23159617,  0.34491907,  0.31851923,
         0.3365617 ],
       [ 0.22391056,  0.28712688, 11.70535044,  0.38340791,  0.18426127,
         0.43233082],
       [ 0.25827407,  0.33317179, 12.55202186,  0.44309759,  0.10567532,
         0.45122709]])
```

14. Measure the performance using metrics

```
from sklearn.metrics import r2_score
r2_score(mlr.predict(x_test),y_test)
```

```
-2.8282903669244077
```