Practical-3

Regression

Problem statement - To predict the age based on other variables

Libraries used - a) Pandas
b) collections
c) sklearn

dataset - abalone
Plots - None
Metrics - accuracy score, mean absolute error.
~~St~~ Model - Regression & naive Bayes
NB - failed
hence ↓ Regression

$$P(H/E) = \frac{P(E/H) * P(H)}{P(E)}$$

train to test split ratio - 75:25

~~Train accuracy~~

classmate

Practical No.3

Data Science and Visualization (Honors Course)

Name:Manjunath GB

PRN: 72018269H

Class: TE ENTC 'B'

To determine the age of abalone on the basis of its physical measurements

In [19]:

```
import pandas as pd
```

In [20]:

```
col = ['sex', 'length', 'diameter', 'height', 'weight', 'sweight', 'vweight', 'shweight',
'rings']
df=pd.read_csv('abalone.csv')
```

In [21]:

```
df.head()
```

Out[21]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

In [22]:

```
df.describe()
```

Out[22]:

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
count	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000
mean	0.523992	0.407881	0.139516	0.828742	0.359367	0.180594	0.238831	9.933684
std	0.120093	0.099240	0.041827	0.490389	0.221963	0.109614	0.139203	3.224169
min	0.075000	0.055000	0.000000	0.002000	0.001000	0.000500	0.001500	1.000000
25%	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	0.130000	8.000000
50%	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	0.234000	9.000000
75%	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	0.329000	11.000000
max	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	1.005000	29.000000

We can say the dataset here is already cleaned because there are no null values.

In [24]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4177 entries, 0 to 4176
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Sex             4177 non-null   object 
 1   Length          4177 non-null   float64
 2   Diameter        4177 non-null   float64
 3   Height          4177 non-null   float64
 4   Whole weight    4177 non-null   float64
 5   Shucked weight  4177 non-null   float64
 6   Viscera weight  4177 non-null   float64
 7   Shell weight    4177 non-null   float64
 8   Rings           4177 non-null   int64  
dtypes: float64(7), int64(1), object(1)
memory usage: 293.8+ KB
```

In [12]:

```
X = df.drop('Rings' , axis=1) #Input
y = df['Rings'] #Output
```

In [13]:

```
X.head()
```

Out[13]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055

In [14]:

```
from collections import Counter
Counter(y)
```

Out[14]:

```
Counter({15: 103,
         7: 391,
         9: 689,
         10: 634,
         8: 568,
         20: 26,
         16: 67,
         19: 32,
         14: 126,
         11: 487,
         12: 267,
         18: 42,
         13: 203,
         5: 115,
         4: 57,
         6: 259,
         21: 14,
         17: 58,
         22: 6,
         1: 1,
         3: 15,
         26: 1,
         23: 9,
         29: 1,
```

```
         2: 1,
         27: 2,
         25: 1,
         24: 2})
```

In [17]:

```
set(X['Sex']) #Displaying unique entries
```

Out[17]:

```
{'F', 'I', 'M'}
```

In [26]:

```
from sklearn.preprocessing import LabelEncoder
enc=LabelEncoder()
X['Sex']=enc.fit_transform(X['Sex'])
```

In [27]:

```
set(X['Sex'])
```

Out[27]:

```
{0, 1, 2}
```

In [28]:

```
df.head()
```

Out[28]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

In [29]:

```
from sklearn.model_selection import train_test_split
```

In [34]:

```
X_train,X_test,y_train,y_test = train_test_split(X,y,random_state=0,test_size=0.25)

#Splitting the dataset
```

In [33]:

```
len(X_train)
```

Out[33]:

```
3132
```

In [35]:

```
len(X_test)
```

Out[35]:

```
1045
```

In [36]:

```
X_train.head()
```

```
X_train.head()
```

Out[36]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight
940	1	0.460	0.345	0.105	0.4490	0.1960	0.0945	0.1265
2688	2	0.630	0.465	0.150	1.0270	0.5370	0.1880	0.1760
1948	2	0.635	0.515	0.165	1.2290	0.5055	0.2975	0.3535
713	2	0.355	0.265	0.085	0.2010	0.0690	0.0530	0.0695
3743	0	0.705	0.555	0.195	1.7525	0.7105	0.4215	0.5160

Prediction

In [37]:

```
from sklearn.naive_bayes import GaussianNB
```

In [38]:

```
clf = GaussianNB()
```

In [39]:

```
#train
clf.fit(X_train,y_train)
```

Out[39]:

```
GaussianNB()
```

In [40]:

```
y_pred=clf.predict(X_test)
```

In [42]:

```
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
```

In [43]:

```
accuracy_score(y_test,y_pred)*100
```

Out[43]:

```
26.02870813397129
```

The accuracy score is low due to presence of multiple classes.

Regression

precision=TP/TP+FP

recall=TP/TP+FN

f1-score=2PR/P+R

Support is the number of actual occurences of class in a specified dataset.

In [44]:

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support
```

```
           3       0.50      1.00      0.67         7
           4       0.30      0.62      0.40        13
           5       0.27      0.42      0.33        40
           6       0.32      0.43      0.36        63
           7       0.26      0.36      0.30       114
           8       0.27      0.29      0.28       139
           9       0.25      0.30      0.27       152
          10       0.21      0.24      0.23       139
          11       0.26      0.42      0.32       121
          12       0.50      0.01      0.02        93
          13       0.00      0.00      0.00        51
          14       0.00      0.00      0.00        32
          15       0.00      0.00      0.00        22
          16       0.00      0.00      0.00        16
          17       0.00      0.00      0.00        12
          18       0.00      0.00      0.00         6
          19       0.00      0.00      0.00        10
          20       0.00      0.00      0.00         8
          21       0.00      0.00      0.00         2
          22       0.00      0.00      0.00         1
          23       0.00      0.00      0.00         2
          24       0.00      0.00      0.00         1
          27       0.00      0.00      0.00         0
          29       0.00      0.00      0.00         1

    accuracy                           0.26      1045
   macro avg       0.13      0.17      0.13      1045
weighted avg       0.24      0.26      0.22      1045
```

```
C:\Users\HP\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1245: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\HP\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1245: UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\HP\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1245: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\HP\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1245: UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\HP\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1245: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\HP\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1245: UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [45]:

```
from sklearn.linear_model import LinearRegression
```

In [46]:

```
reg=LinearRegression()
```

In [47]:

```
reg.fit(X_train,y_train)
```

Out[47]:

```
LinearRegression()
```

In [48]:

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

In [49]:

```
y_pred
```

Out[49]:

```
array([13.10451425,  9.66747548, 10.35605247, ...,  9.95962005,
       12.59111443, 12.18516586])
```

In [50]:

```
from sklearn.metrics import mean_absolute_error
```

In [51]:

```
mean_absolute_error(y_test,y_pred) #summation of (|y_pred-y_train|/no.of entries)
```

Out[51]:

```
1.5955158378194019
```

In [52]:

```
from sklearn.metrics import r2_score
```

In [53]:

```
r2_score(y_test,y_pred)   #r2_score = 1-(summation of (y_pred-y_train)^2 / summation of (
mean of y_train - y_train)^2)
```

Out[53]:

```
0.5354158501894077
```

In this case we can say that Regression outperforms GaussianNB in terms of accuracy. ***(due to the dataset)***

In [ ]:

```

```