devl4a

November 13, 2024

[1]:
```
import pandas as pd
import numpy as np
```

[3]:
```
df=pd.read_csv("/content/Iris.csv")
df
```

[3]:
```
       Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
0       1            5.1           3.5            1.4           0.2
1       2            4.9           3.0            1.4           0.2
2       3            4.7           3.2            1.3           0.2
3       4            4.6           3.1            1.5           0.2
4       5            5.0           3.6            1.4           0.2
..    ...            ...           ...            ...           ...
145   146            6.7           3.0            5.2           2.3
146   147            6.3           2.5            5.0           1.9
147   148            6.5           3.0            5.2           2.0
148   149            6.2           3.4            5.4           2.3
149   150            5.9           3.0            5.1           1.8

            Species
0       Iris-setosa
1       Iris-setosa
2       Iris-setosa
3       Iris-setosa
4       Iris-setosa
..              ...
145  Iris-virginica
146  Iris-virginica
147  Iris-virginica
148  Iris-virginica
149  Iris-virginica

[150 rows x 6 columns]
```

[4]:
```
df.info()
df.drop("Id",axis=1,inplace=True)
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

[5]:
```
df.describe()
```

[5]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

[6]:
```
df.dtypes
```

[6]:
```
SepalLengthCm    float64
SepalWidthCm     float64
PetalLengthCm    float64
PetalWidthCm     float64
Species           object
dtype: object
```

[7]:
```
df.isnull().sum()
```

[7]:
```
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

[8]:
```
# Determine X and Y
x=df.iloc[:,0:4].values
y=df.iloc[:,4].values
```

[9]:
```
#Label Encoding
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
y=le.fit_transform(y)
```

[11]:
```
#Accuracy Prediction
from sklearn.metrics import␣
 ↪classification_report,confusion_matrix,accuracy_score,precision_score,recall_score,f1_score
#Model Selection
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
```

[14]:
```
#Training and Testing the model
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=0)
gaussian=GaussianNB()
gaussian.fit(x_train,y_train)
y_pred=gaussian.predict(x_test)
accuracyscore=round(accuracy_score(y_test,y_pred)*100,2)
print(accuracyscore)
precisionscore=precision_score(y_test,y_pred,average='weighted')*100
print(precisionscore)
recallscore=recall_score(y_test,y_pred,average='weighted')*100
print(recallscore)
f1score=f1_score(y_test,y_pred,average='weighted')*100
print(f1score)
conf_mat=confusion_matrix(y_test,y_pred)
print(conf_mat)
class_report=classification_report(y_test,y_pred)
print(class_report)
```

```
100.0
100.0
100.0
100.0
[[16  0  0]
 [ 0 18  0]
 [ 0  0 11]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        16
           1       1.00      1.00      1.00        18
           2       1.00      1.00      1.00        11

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
weighted avg       1.00      1.00      1.00        45
```

[ ]: