iris

July 5, 2023

[1]:
```
#import requried libraries
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
```

[2]:
```
#loading dataset
df = pd.read_csv("/content/Iris.csv")
```

[3]:
```
#Display first 8 and last 6 rows of dataset
print("First eight rows of dataset are: ")
print(df.head(8))
print("\n")
print("Last six rows of dataset are: ")
print(df.tail(6))
```

```
First eight rows of dataset are:
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
0   1            5.1           3.5            1.4           0.2  Iris-setosa
1   2            4.9           3.0            1.4           0.2  Iris-setosa
2   3            4.7           3.2            1.3           0.2  Iris-setosa
3   4            4.6           3.1            1.5           0.2  Iris-setosa
4   5            5.0           3.6            1.4           0.2  Iris-setosa
5   6            5.4           3.9            1.7           0.4  Iris-setosa
6   7            4.6           3.4            1.4           0.3  Iris-setosa
7   8            5.0           3.4            1.5           0.2  Iris-setosa


Last six rows of dataset are:
      Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
144  145            6.7           3.3            5.7           2.5
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
```

```
144    Iris-virginica
145    Iris-virginica
146    Iris-virginica
147    Iris-virginica
148    Iris-virginica
149    Iris-virginica
```

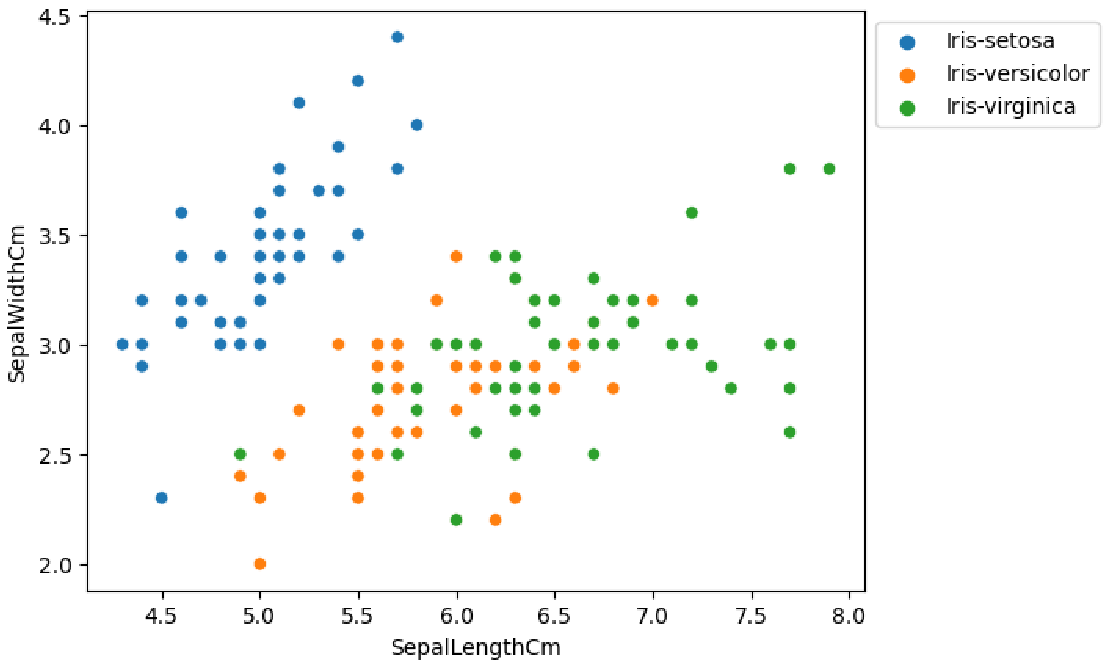
[4]:
```
#Summary statistics of Iris dataset
df.describe()
```

[4]:
```
               Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
count  150.000000     150.000000    150.000000     150.000000    150.000000
mean    75.500000       5.843333      3.054000       3.758667      1.198667
std     43.445368       0.828066      0.433594       1.764420      0.763161
min      1.000000       4.300000      2.000000       1.000000      0.100000
25%     38.250000       5.100000      2.800000       1.600000      0.300000
50%     75.500000       5.800000      3.000000       4.350000      1.300000
75%    112.750000       6.400000      3.300000       5.100000      1.800000
max    150.000000       7.900000      4.400000       6.900000      2.500000
```

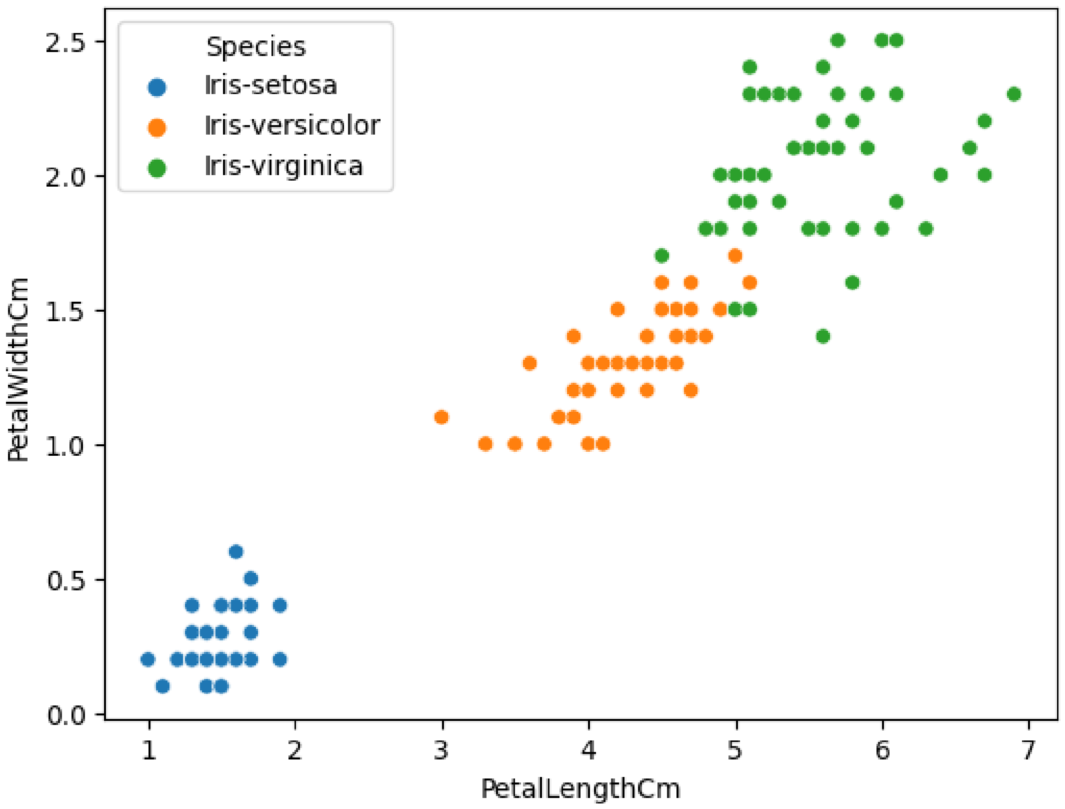
[5]:
```
#info of dataset
df.info()
#To get no. of rows and columns in the dataset
print("Dimensions of dataset are: ",df.shape)
#To get category of each type of species
print("No.of flowers in each species",df.value_counts("Species"))
```

```
<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
Dimensions of dataset are:  (150, 6)
No.of flowers in each species Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
dtype: int64
```

```
[6]: #We will see the plot between sepallength and sepal width by plotting a Scatter␣
      ↪plot between them.
     sns.scatterplot(x='SepalLengthCm', y='SepalWidthCm', hue='Species', data=df, )
     # Placing Legend outside the Figure
     plt.legend(bbox_to_anchor=(1, 1), loc=2)
     #To display the plot
     plt.show()
     #We will see the plot between petal length and petal width by plotting a␣
      ↪Scatter plot between them.
     sns.scatterplot(x='PetalLengthCm', y='PetalWidthCm', hue='Species', data=df, )
```

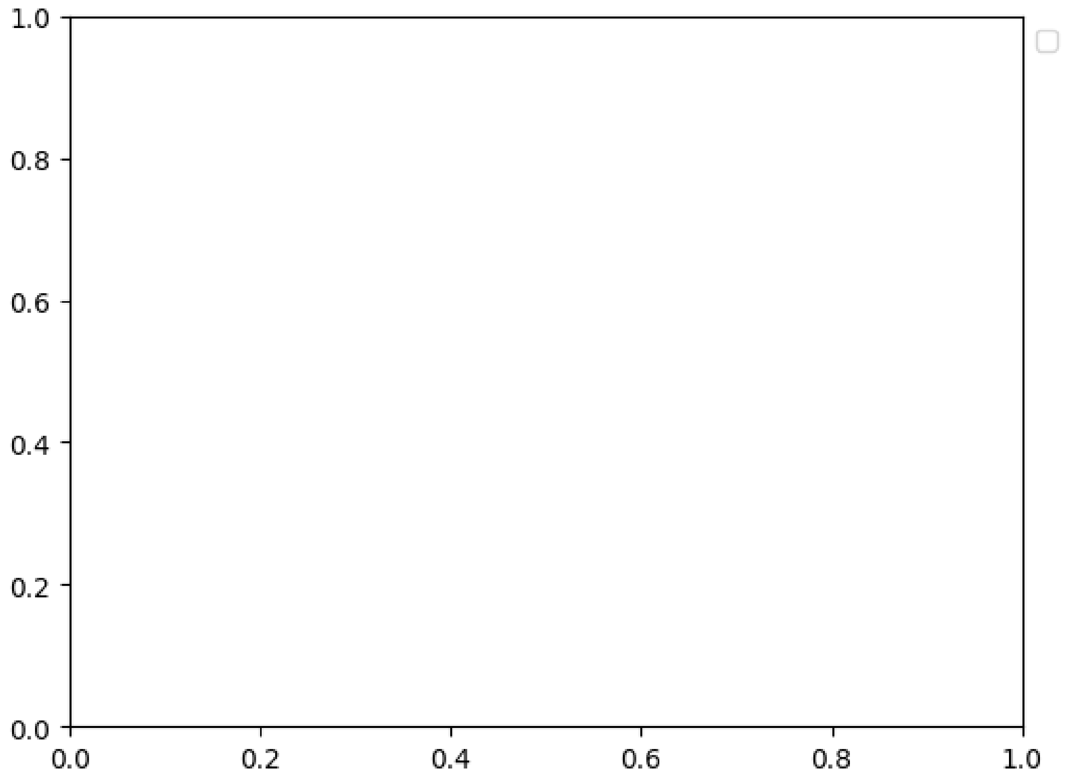


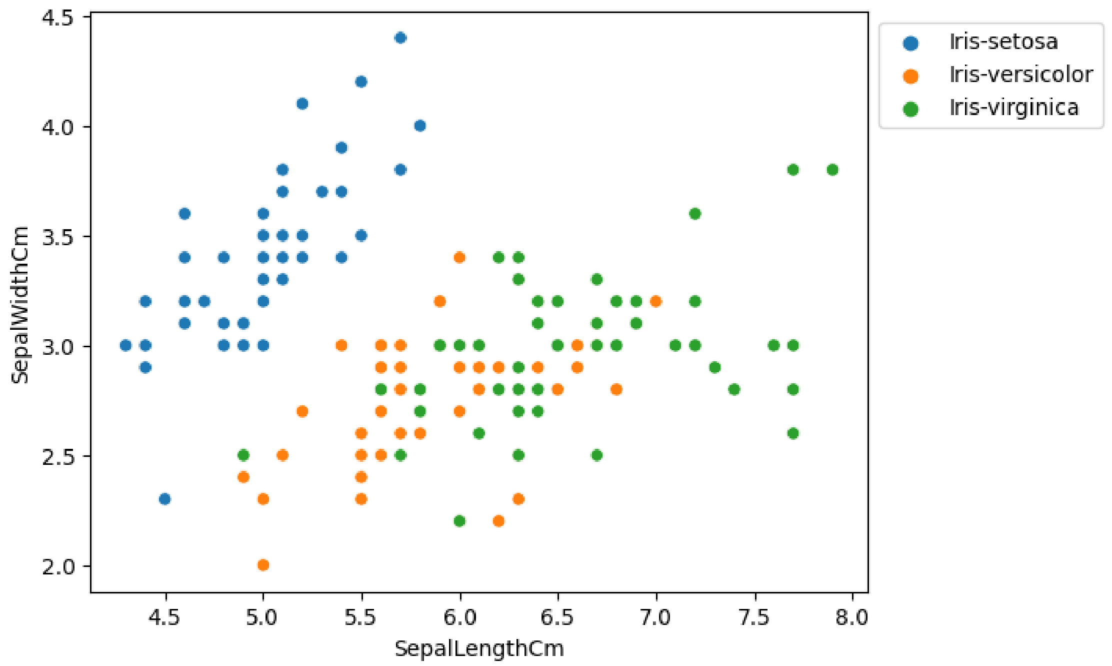
```
[6]: <Axes: xlabel='PetalLengthCm', ylabel='PetalWidthCm'>
```



[7]:
```
# Placing Legend outside the Figure
plt.legend(bbox_to_anchor=(1, 1), loc=2)
#To display the plot
plt.show()
#We will see the plot between sepallength and petal width by plotting a Scatter␣
↪plot between them.
sns.scatterplot(x='SepalLengthCm', y='SepalWidthCm', hue='Species', data=df, )
# Placing Legend outside the Figure
plt.legend(bbox_to_anchor=(1, 1), loc=2)
#To display the plot
plt.show()
#Multivariate analysis
#Pairplot which shows the pair-wise relation between every attributes
sns.pairplot(df,hue='Species', height=2)
df.corr()
```

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note
that artists whose label start with an underscore are ignored when legend() is
called with no argument.
```

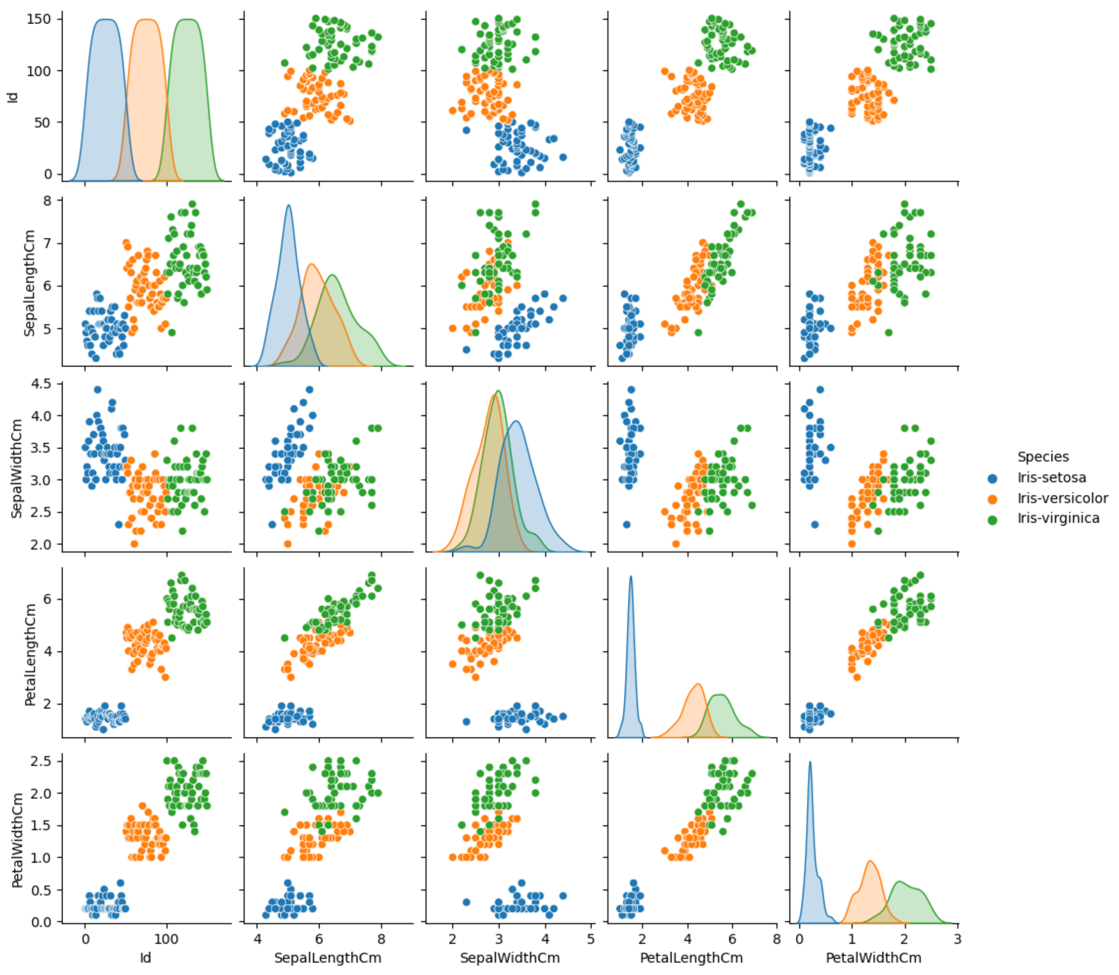

1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


Iris-setosa
Iris-versicolor
Iris-virginica
SepalWidthCm
SepalLengthCm

```
<ipython-input-7-1475378a0238>:14: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  df.corr()
```

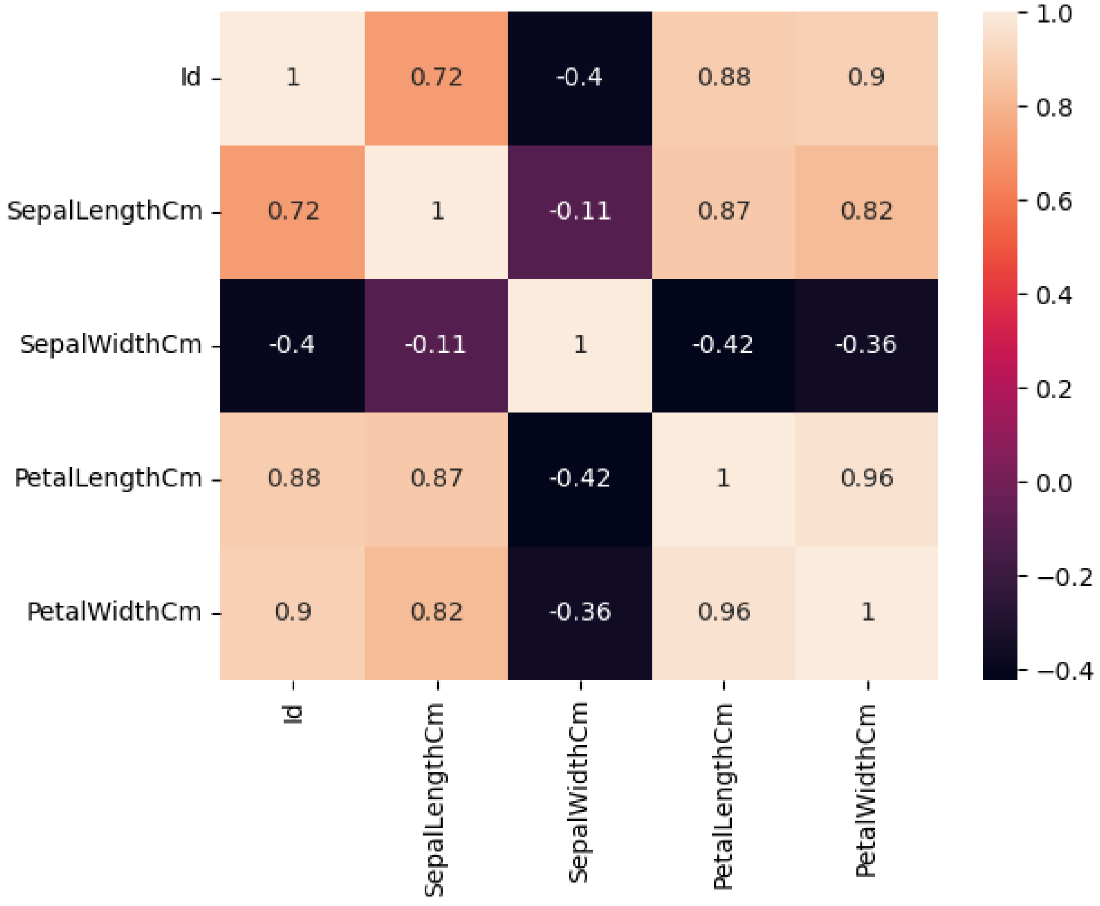
[7]:
```
                       Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  \
Id               1.000000       0.716676     -0.397729       0.882747
SepalLengthCm    0.716676       1.000000     -0.109369       0.871754
SepalWidthCm    -0.397729      -0.109369      1.000000      -0.420516
PetalLengthCm    0.882747       0.871754     -0.420516       1.000000
PetalWidthCm     0.899759       0.817954     -0.356544       0.962757

               PetalWidthCm
Id                 0.899759
SepalLengthCm      0.817954
SepalWidthCm      -0.356544
PetalLengthCm      0.962757
PetalWidthCm       1.000000
```



```
[8]: # Visualizing the correlation between the columns using heatmap.
     sns.heatmap(df.corr(), annot = True)
     plt.show()
     from sklearn.model_selection import train_test_split
     x = df.drop(columns = ['Species'])
     y = df['Species']
     x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = 0.4)
     y_test
```

```
<ipython-input-8-a5cdf70f9b35>:2: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  sns.heatmap(df.corr(), annot = True)
```

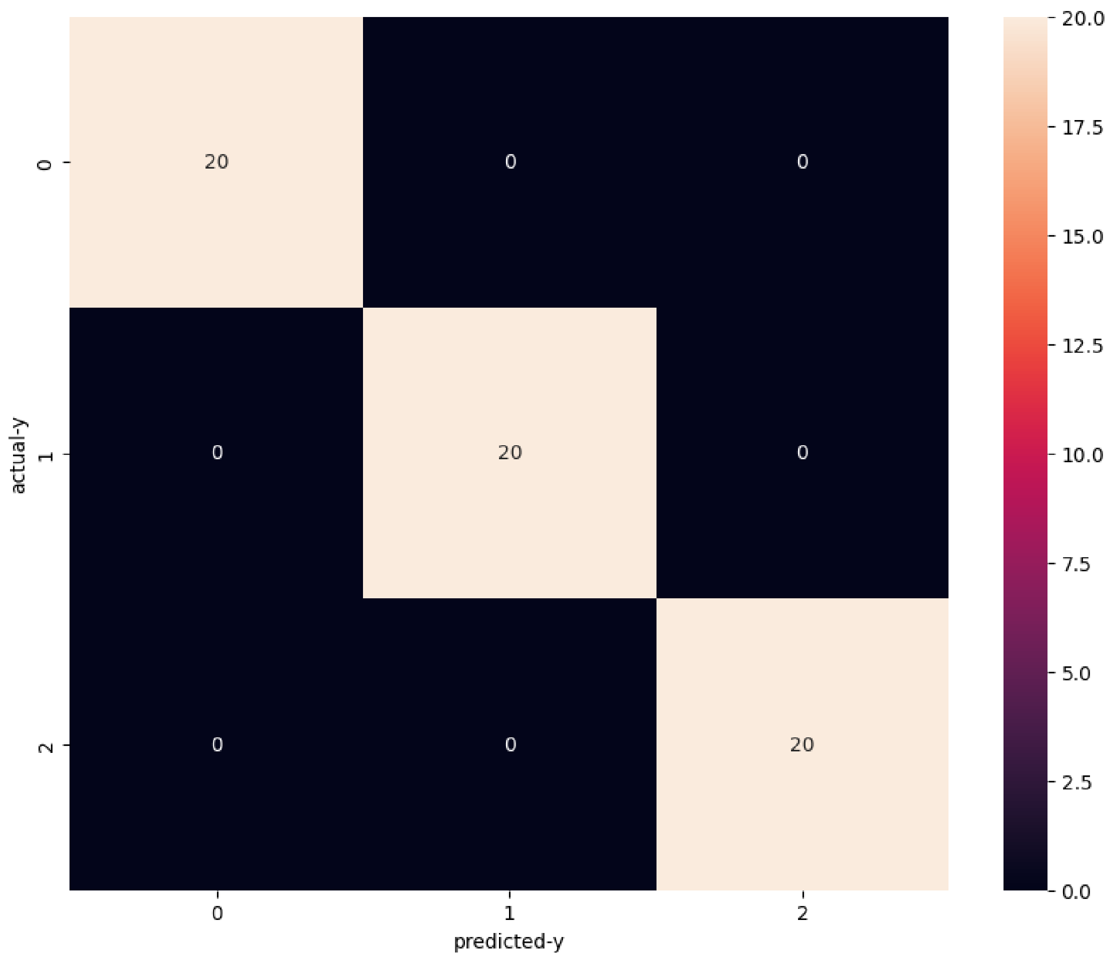


[8]:
```
137     Iris-virginica
143     Iris-virginica
59     Iris-versicolor
22         Iris-setosa
93     Iris-versicolor
118     Iris-virginica
145     Iris-virginica
74     Iris-versicolor
87     Iris-versicolor
122     Iris-virginica
21         Iris-setosa
14         Iris-setosa
102     Iris-virginica
46         Iris-setosa
70     Iris-versicolor
108     Iris-virginica
51     Iris-versicolor
75     Iris-versicolor
72     Iris-versicolor
120     Iris-virginica
86     Iris-versicolor
55     Iris-versicolor
6          Iris-setosa
107     Iris-virginica
18         Iris-setosa
148     Iris-virginica
41         Iris-setosa
132     Iris-virginica
106     Iris-virginica
147     Iris-virginica
45         Iris-setosa
82     Iris-versicolor
2          Iris-setosa
136     Iris-virginica
129     Iris-virginica
35         Iris-setosa
101     Iris-virginica
39         Iris-setosa
23         Iris-setosa
19         Iris-setosa
60     Iris-versicolor
144     Iris-virginica
113     Iris-virginica
97     Iris-versicolor
36         Iris-setosa
56     Iris-versicolor
61     Iris-versicolor
```

```
134     Iris-virginica
99     Iris-versicolor
68     Iris-versicolor
27         Iris-setosa
20         Iris-setosa
95     Iris-versicolor
9          Iris-setosa
53     Iris-versicolor
117     Iris-virginica
24         Iris-setosa
63     Iris-versicolor
7          Iris-setosa
12         Iris-setosa
Name: Species, dtype: object
```

[9]:
```
#Importing library foe decision classifier
from sklearn.tree import DecisionTreeClassifier
id3=DecisionTreeClassifier(criterion='entropy')
#Fit the data
k=id3.fit(x_train,y_train)
#predict the data
y_pred=id3.predict(x_test)
print(y_pred)
#Confusion matrix
from sklearn.metrics import␣
 ↪confusion_matrix,accuracy_score,classification_report
cm=confusion_matrix(y_pred,y_test)
plt.figure(figsize=(10,8))
sns.heatmap(cm,annot=True)
plt.xlabel('predicted-y')
plt.ylabel('actual-y')
plt.show()
```

```
['Iris-virginica' 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa'
 'Iris-versicolor' 'Iris-virginica' 'Iris-virginica' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor' 'Iris-virginica'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa' 'Iris-virginica'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor'
 'Iris-setosa' 'Iris-virginica' 'Iris-virginica' 'Iris-setosa'
 'Iris-virginica' 'Iris-setosa' 'Iris-setosa' 'Iris-setosa'
 'Iris-versicolor' 'Iris-virginica' 'Iris-virginica' 'Iris-versicolor'
 'Iris-setosa' 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-versicolor' 'Iris-setosa' 'Iris-versicolor' 'Iris-virginica'
 'Iris-setosa' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa']
```



[10]:
```
#Accuracy score and model score
print(classification_report(y_pred,y_test))
print('accuracy-score',accuracy_score(y_pred,y_test))
print('Model score',id3.score(x_test,y_test))
X_new = np.array([[7,3, 2, 1, 0.2], [8,4.9, 2.2, 3.8, 1.1], [9,5.3, 2.5, 4.6, 1.
 ↪9]])
#Prediction of the species
prediction = id3.predict(X_new)
print("Prediction of Species: {}".format(prediction))
```

```
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        20
Iris-versicolor       1.00      1.00      1.00        20
 Iris-virginica       1.00      1.00      1.00        20
```

```
    accuracy                           1.00        60
   macro avg       1.00      1.00      1.00        60
weighted avg       1.00      1.00      1.00        60

accuracy-score 1.0
Model score 1.0
Prediction of Species: ['Iris-setosa' 'Iris-setosa' 'Iris-setosa']
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does
not have valid feature names, but DecisionTreeClassifier was fitted with feature
names
  warnings.warn(
```