Task 1: Iris Classification

1Q) Iris flower has three species; setosa, versicolor, and virginica, which differs according to their measurements. Now assume that you have the measurements of the iris flowers according to their species, and here your task is to train a machine learning model that can learn from the measurements of the iris species and classify them.

In [1]:
```
# import all the required packages

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,precision_score,recall_score
from sklearn.metrics import confusion_matrix,ConfusionMatrixDisplay
```

In [2]:
```
df=pd.read_csv('Iris.csv') # read the data
df.head()
```

Out[2]:

	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

In [3]:
```
df.shape # the data has 150 rows and 6 columns
```

Out[3]:
```
(150, 6)
```

In [4]:
```
df.columns
```

Out[4]:
```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidth
Cm',
       'Species'],
      dtype='object')
```

In [5]:
```
df.dtypes
```

Out[5]:
```
Id                int64
SepalLengthCm    float64
SepalWidthCm     float64
PetalLengthCm    float64
PetalWidthCm     float64
Species           object
dtype: object
```

In [6]:
```
df.isnull().sum() # there are no null values in the data
```

Out[6]:
```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

In [7]:
```
df['Species'].unique()
```

Out[7]:
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

In [8]:
```
df['Species'].value_counts()
```

Out[8]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

In [9]:
```
df.drop('Id',axis=1,inplace=True) # dropping the id column from the dat
```

In [10]:
```
df.head()
```

Out[10]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In [11]:
```
df['Species'].unique()
```

Out[11]:
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

In [12]:
```
df['Species'].unique()[1]
```

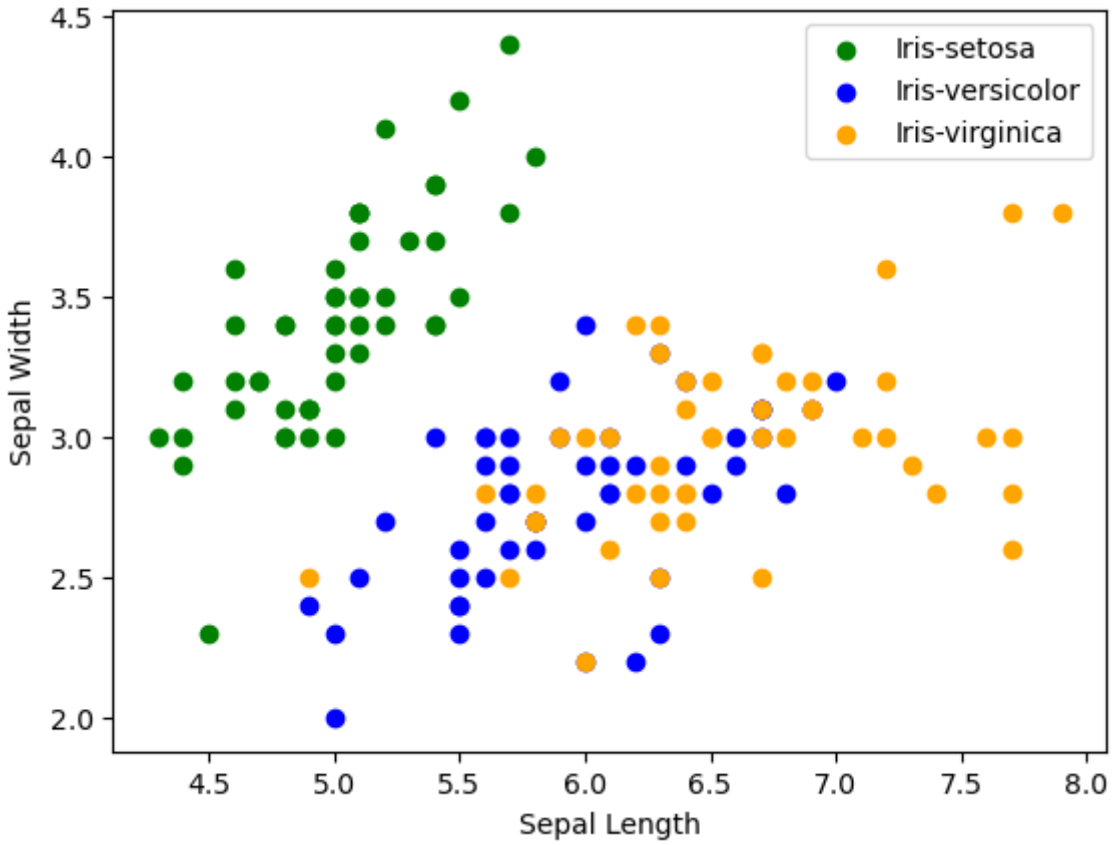
Out[12]:
```
'Iris-versicolor'
```

Data Visualisation

In [13]:

```
colors=['green', 'blue', 'orange']
for i in range(3):
    x=df[df['Species']==df['Species'].unique()[i]]
    plt.scatter(x['SepalLengthCm'], x['SepalWidthCm'], c=colors[i], lab
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
plt.legend()
```

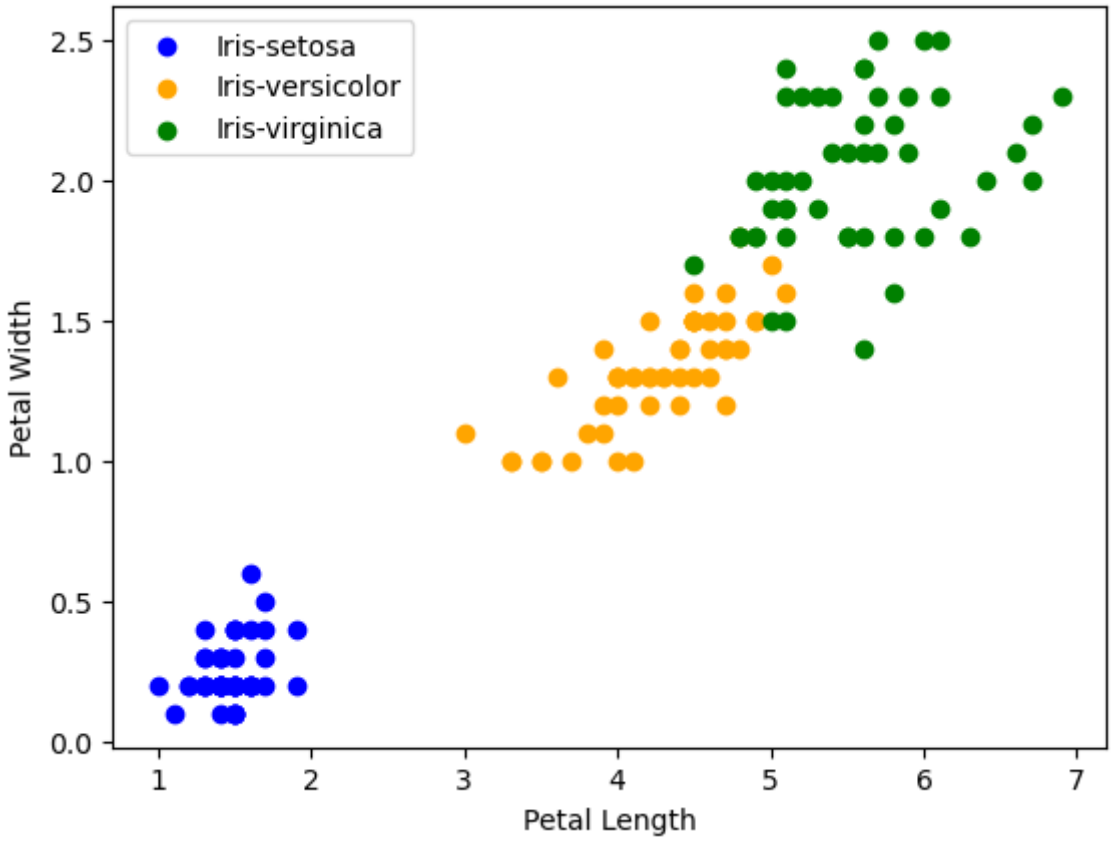
Out[13]: <matplotlib.legend.Legend at 0x1f85dd44710>



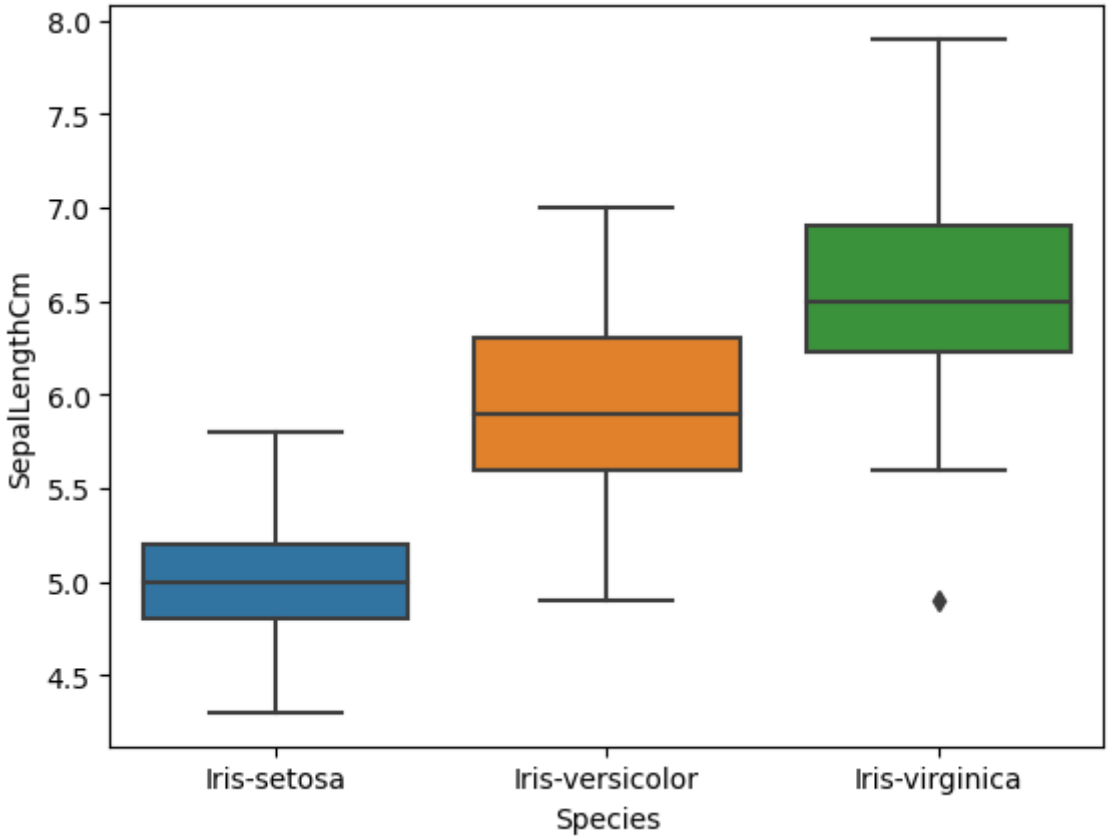
In [14]:

```
colors=['blue', 'orange', 'green']
for i in range(3):
    x=df[df['Species']==df['Species'].unique()[i]]
    plt.scatter(x['PetalLengthCm'], x['PetalWidthCm'], c=colors[i], lab
plt.xlabel('Petal Length')
plt.ylabel('Petal Width')
plt.legend()
```

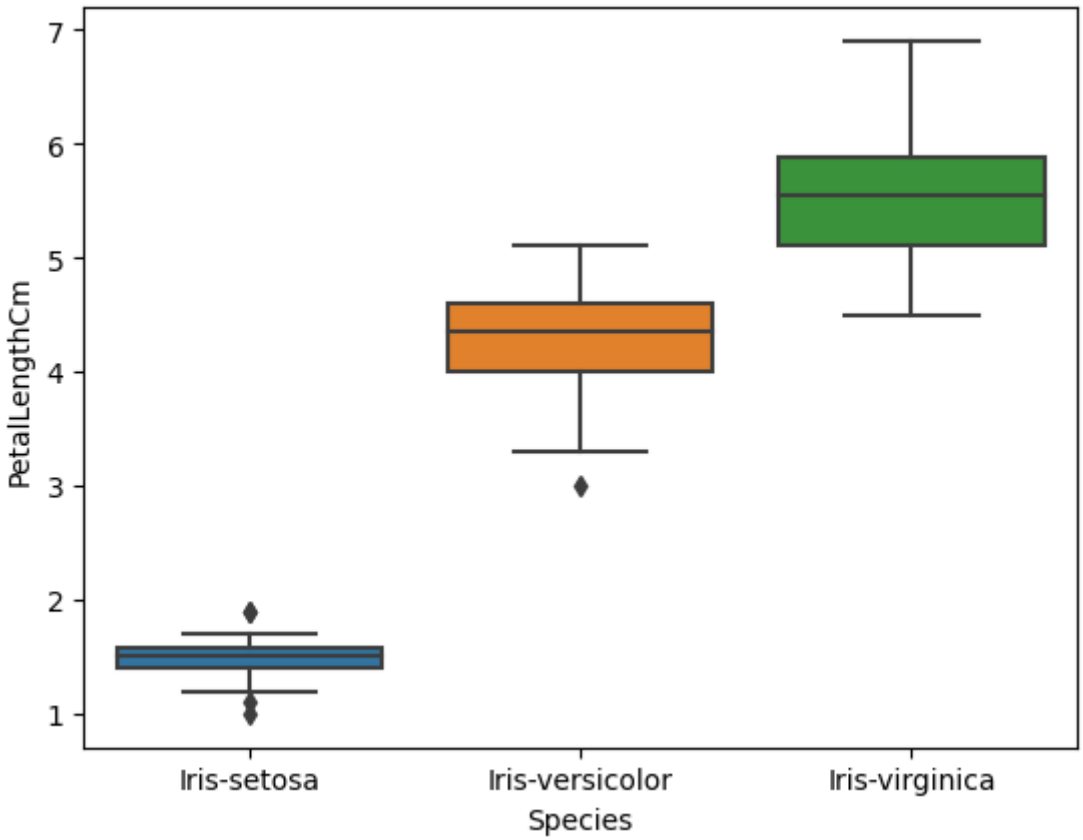
Out[14]: <matplotlib.legend.Legend at 0x1f85ddb6150>



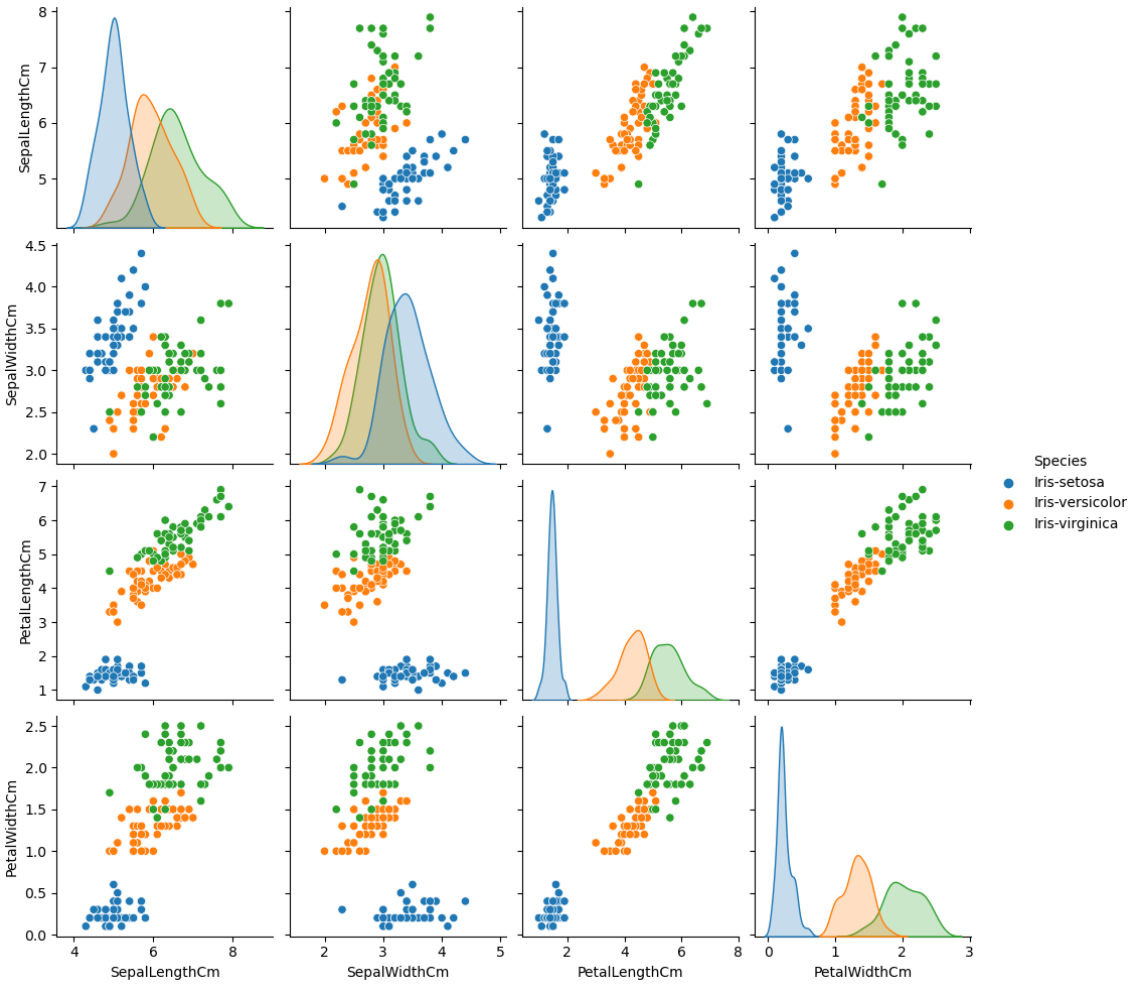
In [15]:
```
sns.boxplot(x='Species',y='SepalLengthCm',data=df)
plt.show()
```



In [16]:
```
sns.boxplot(x='Species',y='PetalLengthCm',data=df)
plt.show()
```



In [17]:
```
sns.pairplot(df, hue='Species')
plt.show()
```



Splitting the Dataset

In [18]:
```
df['Target']=df['Species'] # creating a new column 'Target'
df.head()
```

Out[18]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species	Target
0	5.1	3.5	1.4	0.2	Iris-setosa	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa	Iris-setosa

In [19]:
```
for col in df.select_dtypes('object').columns:
    print('{}: {}'.format(col,df[col].unique()))
```

```
Species: ['Iris-setosa' 'Iris-versicolor' 'Iris-virginica']
Target: ['Iris-setosa' 'Iris-versicolor' 'Iris-virginica']
```

In [20]:
```
# converting the string value of target columns to numerical columns

LE=LabelEncoder()
LE.fit(df['Target'].unique())
df['Target']=LE.fit_transform(df[col])
print('{}: {}'.format(col,df['Target'].unique()))
```

```
Target: [0 1 2]
```

In [21]:
```
df
```

Out[21]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species	Target
0	5.1	3.5	1.4	0.2	Iris-setosa	0
1	4.9	3.0	1.4	0.2	Iris-setosa	0
2	4.7	3.2	1.3	0.2	Iris-setosa	0
3	4.6	3.1	1.5	0.2	Iris-setosa	0
4	5.0	3.6	1.4	0.2	Iris-setosa	0
...	...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica	2
146	6.3	2.5	5.0	1.9	Iris-virginica	2
147	6.5	3.0	5.2	2.0	Iris-virginica	2
148	6.2	3.4	5.4	2.3	Iris-virginica	2
149	5.9	3.0	5.1	1.8	Iris-virginica	2

150 rows × 6 columns

In [22]:
```
X=df.drop(columns=['Species', 'Target'],axis=1) # dropping 'species' ar
X.head()
```

Out[22]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

In [23]:
```
y=df['Species'] # separating the 'species' column as output column
y.head()
```

Out[23]:
```
0    Iris-setosa
1    Iris-setosa
2    Iris-setosa
3    Iris-setosa
4    Iris-setosa
Name: Species, dtype: object
```

In [24]:
```
# splitting the input and output data into train and test data with tes

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,randc
```

In [25]:
```
X_train.head()
```

Out[25]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
61	5.9	3.0	4.2	1.5
92	5.8	2.6	4.0	1.2
112	6.8	3.0	5.5	2.1
2	4.7	3.2	1.3	0.2
141	6.9	3.1	5.1	2.3

In [26]:
```
X_train.shape
```

Out[26]:
```
(112, 4)
```

In [27]:
```
X_test.head()
```

Out[27]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
114	5.8	2.8	5.1	2.4
62	6.0	2.2	4.0	1.0
33	5.5	4.2	1.4	0.2
107	7.3	2.9	6.3	1.8
7	5.0	3.4	1.5	0.2

In [28]:
```
X_test.shape
```

Out[28]:
```
(38, 4)
```

Training the model

In [29]:
```
# from sklearn import support vector machine algorithm

from sklearn.svm import SVC
model=SVC(kernel='rbf')
```

In [30]:
```
model.fit(X_train,y_train) # we fit the model here with train data
```

Out[30]:
```
SVC()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [31]:
```
model.predict(X_test) # we predict the output of the train data
```

Out[31]:
```
array(['Iris-virginica', 'Iris-versicolor', 'Iris-setosa',
       'Iris-virginica', 'Iris-setosa', 'Iris-virginica', 'Iris-setosa',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-virginica', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-setosa',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-setosa', 'Iris-setosa',
       'Iris-virginica', 'Iris-versicolor', 'Iris-setosa', 'Iris-setosa',
       'Iris-virginica', 'Iris-setosa', 'Iris-setosa', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-setosa', 'Iris-virginica',
       'Iris-versicolor', 'Iris-setosa', 'Iris-virginica',
       'Iris-virginica', 'Iris-versicolor', 'Iris-setosa',
       'Iris-virginica'], dtype=object)
```

In [32]:
```
model.score(X_test,y_test) # measures accuracy score of predicted data
```

Out[32]:
```
0.9736842105263158
```

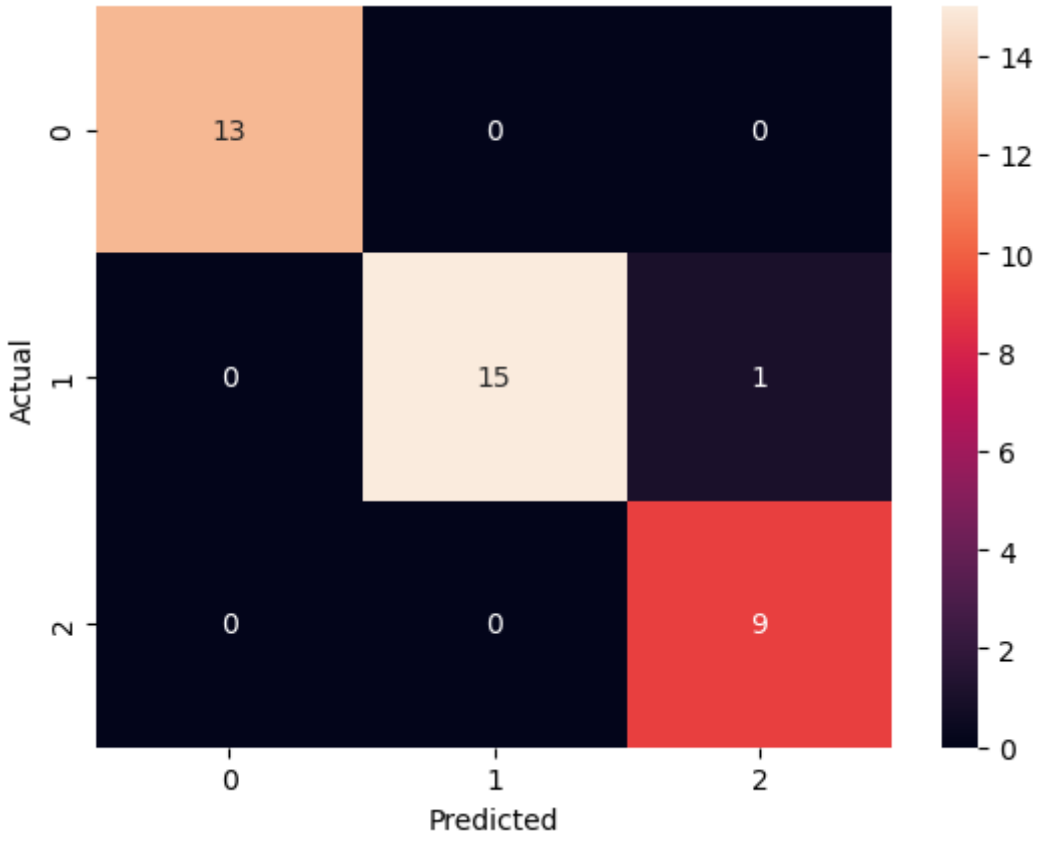
Confusion Matrix

In [33]:
```
# creating a confusion matrix

pred=model.predict(X_test)
CM=confusion_matrix(y_test,pred)
CM
```

Out[33]:
```
array([[13,  0,  0],
       [ 0, 15,  1],
       [ 0,  0,  9]], dtype=int64)
```

In [34]:
```
sns.heatmap(CM,annot=True)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```



In [35]:
```
# analysing metrics to measure performance

print("Classification Report: ")
print(classification_report(y_test,pred))
```

```
Classification Report:
                precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        13
Iris-versicolor       1.00      0.94      0.97        16
 Iris-virginica       0.90      1.00      0.95         9

       accuracy                           0.97        38
      macro avg       0.97      0.98      0.97        38
   weighted avg       0.98      0.97      0.97        38
```

In [36]:
```
model.predict([[4.7, 3.2, 1.3, 0.2]]) # predicting output with custom i
```

```
C:\Users\hp\anaconda3\Lib\site-packages\sklearn\base.py:439: UserWarning:
X does not have valid feature names, but SVC was fitted with feature names
  warnings.warn(
```

Out[36]:
```
array(['Iris-setosa'], dtype=object)
```

In [37]:
```
1 model.predict([[7.1, 3.0, 5.9, 1.5]]) # predicting output with custom i
```

```
C:\Users\hp\anaconda3\Lib\site-packages\sklearn\base.py:439: UserWarning:
X does not have valid feature names, but SVC was fitted with feature names
  warnings.warn(
```

Out[37]:
```
array(['Iris-virginica'], dtype=object)
```

In [ ]:
```
1
```