In [1]:
```
import pandas as pd
```

In [2]:
```
import numpy as np
```

In [3]:
```
df=pd.read_csv(r'C:\Users\user\OneDrive\Desktop\cpp\Iris.csv')
```

In [4]:
```
x=df[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]
```

In [5]:
```
y=df['Species'].values
```

In [6]:
```
from sklearn.model_selection import train_test_split
```

In [7]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   SepalLengthCm  150 non-null    float64
 1   SepalWidthCm   150 non-null    float64
 2   PetalLengthCm  150 non-null    float64
 3   PetalWidthCm   150 non-null    float64
 4   Species        150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [9]:
```
df.shape
```

Out[9]:
```
(150, 5)
```

In [10]:
```
df.shape
```

Out[10]:
```
(150, 5)
```

In [11]:
```
df.columns
```

Out[11]:
```
Index(['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

In [12]:
```
from sklearn.preprocessing import StandardScaler
```

In [13]:
```
sc=StandardScaler()
```

```
In [14]: x_train=sc.fit_transform(x_train)
```

```
In [15]: x_test=sc.transform(x_test)
```

```
In [16]: from sklearn.naive_bayes import GaussianNB
```

```
In [17]: nb=GaussianNB()
```

```
In [18]: nb.fit(x_train,y_train)

Out[18]: GaussianNB()
```

```
In [19]: y_pred=nb.predict(x_test)
```

```
In [20]: y_pred.shape

Out[20]: (30,)
```

```
In [21]: y_pred

Out[21]: array(['Iris-versicolor', 'Iris-versicolor', 'Iris-virginica',
                'Iris-setosa', 'Iris-setosa', 'Iris-versicolor', 'Iris-setosa',
                'Iris-versicolor', 'Iris-virginica', 'Iris-versicolor',
                'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
                'Iris-virginica', 'Iris-setosa', 'Iris-versicolor',
                'Iris-versicolor', 'Iris-virginica', 'Iris-versicolor',
                'Iris-setosa', 'Iris-versicolor', 'Iris-versicolor',
                'Iris-virginica', 'Iris-setosa', 'Iris-virginica',
                'Iris-virginica', 'Iris-virginica', 'Iris-versicolor',
                'Iris-versicolor', 'Iris-virginica'], dtype='<U15')
```

```
In [22]: from sklearn.metrics import confusion_matrix,accuracy_score,precision_score,recal
```

```
In [23]: cm=confusion_matrix(y_test,y_pred)
```

```
In [24]: accuracy_score(y_test,y_pred)

Out[24]: 0.9333333333333333
```

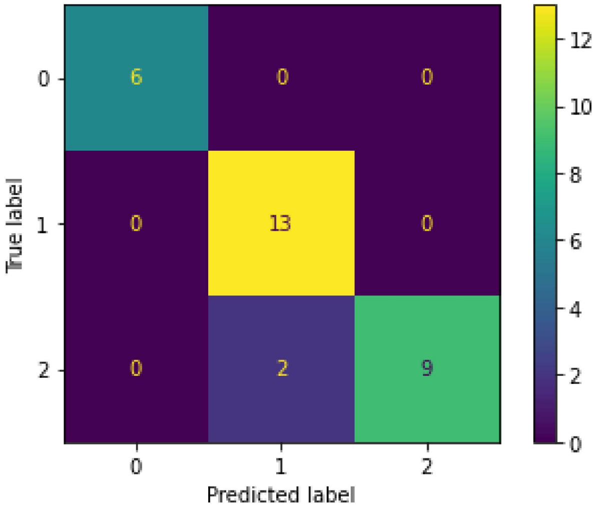
```
In [25]: cm

Out[25]: array([[ 6,  0,  0],
                [ 0, 13,  0],
                [ 0,  2,  9]], dtype=int64)
```

cm[0][1]

In [26]:
```
cm[0][1]
```

Out[26]:
```
0
```

In [27]:
```
cm_display=ConfusionMatrixDisplay(cm).plot()
```



In [28]:
```
y_test.shape,x_test.shape
```

Out[28]:
```
((30,), (30, 4))
```

In [29]:
```
cm=np.array(cm)
```

In [30]:
```
cm
```

Out[30]:
```
array([[ 6,  0,  0],
       [ 0, 13,  0],
       [ 0,  2,  9]], dtype=int64)
```

In [35]:
```
recall = np.diag(cm) / np.sum(cm, axis = 1)
precision = np.diag(cm) / np.sum(cm, axis = 1)
```

In [36]:
```
print(np.mean(recall))
print(np.mean(precision))
```

```
0.9393939393939394
0.9393939393939394
```

In [ ]:
```

```