In [14]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [21]:
```
df=pd.read_csv("iris.csv")
df.head()
```

Out[21]:

	sepal_length	sepal_width	petal_length	petal_width	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 [22]:
```
df.info()
```

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

In [44]:
```
from sklearn.model_selection import train_test_split
x = df.drop('species', axis = 1)
y = df['species']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30)
```

In [45]:
```
x=df.iloc[:,:-2]
```

In [46]:
```
x
```

Out[46]:

	sepal_length	sepal_width	petal_length
0	5.1	3.5	1.4
1	4.9	3.0	1.4
2	4.7	3.2	1.3
3	4.6	3.1	1.5
4	5.0	3.6	1.4
...	...	...	...
145	6.7	3.0	5.2
146	6.3	2.5	5.0
147	6.5	3.0	5.2
148	6.2	3.4	5.4
149	5.9	3.0	5.1

150 rows × 3 columns

In [47]:
```
y
```

Out[47]:
```
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
Name: species, Length: 150, dtype: object
```

In [48]:
```
from sklearn.svm import SVC
svm = SVC()
svm.fit(X_train,y_train)
```

Out[48]:
```
SVC()
```

In [49]:
```
pred = svm.predict(X_test)
```

In [50]:
```
from sklearn.metrics import confusion_matrix
print(confusion_matrix(y_test, pred))
```

```
[[17  0  0]
 [ 0 12  0]
 [ 0  2 14]]
```

In [51]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test,pred))
```

```
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        17
Iris-versicolor       0.86      1.00      0.92        12
 Iris-virginica       1.00      0.88      0.93        16

       accuracy                           0.96        45
      macro avg       0.95      0.96      0.95        45
   weighted avg       0.96      0.96      0.96        45
```

In [52]:
```
svm.score(X_test,y_test)
```

Out[52]:
```
0.9555555555555556
```

In [53]:
```
from sklearn.model_selection import GridSearchCV
```

In [54]:
```
param_grid = {'C':[0.1,1,10,100], 'gamma':[1,0.1,0.01,0.001]}
```

In [55]:
```
grid = GridSearchCV(SVC(), param_grid, refit = True, verbose=3)
grid.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 16 candidates, totalling 80 fits
[CV 1/5] END ....................C=0.1, gamma=1;, score=0.952 total time=   0.0s
[CV 2/5] END ....................C=0.1, gamma=1;, score=1.000 total time=   0.0s
[CV 3/5] END ....................C=0.1, gamma=1;, score=1.000 total time=   0.0s
[CV 4/5] END ....................C=0.1, gamma=1;, score=0.810 total time=   0.0s
[CV 5/5] END ....................C=0.1, gamma=1;, score=0.952 total time=   0.0s
[CV 1/5] END ..................C=0.1, gamma=0.1;, score=0.905 total time=   0.0s
[CV 2/5] END ..................C=0.1, gamma=0.1;, score=0.857 total time=   0.0s
[CV 3/5] END ..................C=0.1, gamma=0.1;, score=0.905 total time=   0.0s
[CV 4/5] END ..................C=0.1, gamma=0.1;, score=0.810 total time=   0.0s
[CV 5/5] END ..................C=0.1, gamma=0.1;, score=0.857 total time=   0.0s
[CV 1/5] END .................C=0.1, gamma=0.01;, score=0.381 total time=   0.0s
[CV 2/5] END .................C=0.1, gamma=0.01;, score=0.381 total time=   0.0s
[CV 3/5] END .................C=0.1, gamma=0.01;, score=0.381 total time=   0.0s
[CV 4/5] END .................C=0.1, gamma=0.01;, score=0.333 total time=   0.0s
[CV 5/5] END .................C=0.1, gamma=0.01;, score=0.333 total time=   0.0s
[CV 1/5] END ................C=0.1, gamma=0.001;, score=0.381 total time=   0.0s
[CV 2/5] END ................C=0.1, gamma=0.001;, score=0.381 total time=   0.0s
[CV 3/5] END ................C=0.1, gamma=0.001;, score=0.381 total time=   0.0s
[CV 4/5] END ................C=0.1, gamma=0.001;, score=0.333 total time=   0.0s
[CV 5/5] END ................C=0.1, gamma=0.001;, score=0.333 total time=   0.0s
[CV 1/5] END ......................C=1, gamma=1;, score=1.000 total time=   0.0s
[CV 2/5] END ......................C=1, gamma=1;, score=1.000 total time=   0.0s
[CV 3/5] END ......................C=1, gamma=1;, score=1.000 total time=   0.0s
[CV 4/5] END ......................C=1, gamma=1;, score=0.810 total time=   0.0s
[CV 5/5] END ......................C=1, gamma=1;, score=0.952 total time=   0.0s
[CV 1/5] END ....................C=1, gamma=0.1;, score=0.952 total time=   0.0s
[CV 2/5] END ....................C=1, gamma=0.1;, score=1.000 total time=   0.0s
[CV 3/5] END ....................C=1, gamma=0.1;, score=1.000 total time=   0.0s
[CV 4/5] END ....................C=1, gamma=0.1;, score=0.810 total time=   0.0s
[CV 5/5] END ....................C=1, gamma=0.1;, score=0.952 total time=   0.0s
[CV 1/5] END ...................C=1, gamma=0.01;, score=0.905 total time=   0.0s
[CV 2/5] END ...................C=1, gamma=0.01;, score=0.952 total time=   0.0s
[CV 3/5] END ...................C=1, gamma=0.01;, score=1.000 total time=   0.0s
[CV 4/5] END ...................C=1, gamma=0.01;, score=0.810 total time=   0.0s
[CV 5/5] END ...................C=1, gamma=0.01;, score=0.952 total time=   0.0s
[CV 1/5] END ..................C=1, gamma=0.001;, score=0.381 total time=   0.0s
[CV 2/5] END ..................C=1, gamma=0.001;, score=0.381 total time=   0.0s
[CV 3/5] END ..................C=1, gamma=0.001;, score=0.381 total time=   0.0s
[CV 4/5] END ..................C=1, gamma=0.001;, score=0.333 total time=   0.0s
[CV 5/5] END ..................C=1, gamma=0.001;, score=0.333 total time=   0.0s
[CV 1/5] END .....................C=10, gamma=1;, score=0.952 total time=   0.0s
[CV 2/5] END .....................C=10, gamma=1;, score=1.000 total time=   0.0s
[CV 3/5] END .....................C=10, gamma=1;, score=0.952 total time=   0.0s
[CV 4/5] END .....................C=10, gamma=1;, score=0.762 total time=   0.0s
[CV 5/5] END .....................C=10, gamma=1;, score=0.952 total time=   0.0s
[CV 1/5] END ...................C=10, gamma=0.1;, score=0.952 total time=   0.0s
[CV 2/5] END ...................C=10, gamma=0.1;, score=1.000 total time=   0.0s
[CV 3/5] END ...................C=10, gamma=0.1;, score=1.000 total time=   0.0s
[CV 4/5] END ...................C=10, gamma=0.1;, score=0.810 total time=   0.0s
[CV 5/5] END ...................C=10, gamma=0.1;, score=0.952 total time=   0.0s
[CV 1/5] END ..................C=10, gamma=0.01;, score=0.952 total time=   0.0s
[CV 2/5] END ..................C=10, gamma=0.01;, score=1.000 total time=   0.0s
[CV 3/5] END ..................C=10, gamma=0.01;, score=1.000 total time=   0.0s
[CV 4/5] END ..................C=10, gamma=0.01;, score=0.810 total time=   0.0s
[CV 5/5] END ..................C=10, gamma=0.01;, score=0.952 total time=   0.0s
[CV 1/5] END .................C=10, gamma=0.001;, score=0.905 total time=   0.0s
[CV 2/5] END .................C=10, gamma=0.001;, score=0.952 total time=   0.0s
[CV 3/5] END .................C=10, gamma=0.001;, score=0.952 total time=   0.0s
[CV 4/5] END .................C=10, gamma=0.001;, score=0.810 total time=   0.0s
[CV 5/5] END .................C=10, gamma=0.001;, score=0.905 total time=   0.0s
[CV 1/5] END ....................C=100, gamma=1;, score=0.905 total time=   0.0s
[CV 2/5] END ....................C=100, gamma=1;, score=0.952 total time=   0.0s
[CV 3/5] END ....................C=100, gamma=1;, score=0.905 total time=   0.0s
[CV 4/5] END ....................C=100, gamma=1;, score=0.762 total time=   0.0s
[CV 5/5] END ....................C=100, gamma=1;, score=0.952 total time=   0.0s
[CV 1/5] END ..................C=100, gamma=0.1;, score=0.952 total time=   0.0s
[CV 2/5] END ..................C=100, gamma=0.1;, score=1.000 total time=   0.0s
[CV 3/5] END ..................C=100, gamma=0.1;, score=1.000 total time=   0.0s
[CV 4/5] END ..................C=100, gamma=0.1;, score=0.762 total time=   0.0s
[CV 5/5] END ..................C=100, gamma=0.1;, score=0.952 total time=   0.0s
[CV 1/5] END .................C=100, gamma=0.01;, score=0.952 total time=   0.0s
[CV 2/5] END .................C=100, gamma=0.01;, score=1.000 total time=   0.0s
[CV 3/5] END .................C=100, gamma=0.01;, score=0.952 total time=   0.0s
[CV 4/5] END .................C=100, gamma=0.01;, score=0.810 total time=   0.0s
[CV 5/5] END .................C=100, gamma=0.01;, score=0.952 total time=   0.0s
[CV 1/5] END ................C=100, gamma=0.001;, score=0.952 total time=   0.0s
[CV 2/5] END ................C=100, gamma=0.001;, score=1.000 total time=   0.0s
[CV 3/5] END ................C=100, gamma=0.001;, score=1.000 total time=   0.0s
[CV 4/5] END ................C=100, gamma=0.001;, score=0.810 total time=   0.0s
[CV 5/5] END ................C=100, gamma=0.001;, score=0.952 total time=   0.0s
```

Out[55]:
```
GridSearchCV(estimator=SVC(),
             param_grid={'C': [0.1, 1, 10, 100],
                         'gamma': [1, 0.1, 0.01, 0.001]},
             verbose=3)
```

In [56]:
```
pred_grid = grid.predict(X_test)
```

In [57]:
```
print(confusion_matrix(y_test, pred_grid))
```

```
[[17  0  0]
 [ 0 12  0]
 [ 0  0 16]]
```

In [58]:
```
print(classification_report(y_test, pred_grid))
```

```
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        17
Iris-versicolor       1.00      1.00      1.00        12
 Iris-virginica       1.00      1.00      1.00        16

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

In [59]:
```
hyperparameters = dict(C=C, penalty=penalty, solver=solver)
logistic = linear_model.LogisticRegression()
gridsearch = GridSearchCV(logistic, hyperparameters)
best_model_grid = gridsearch.fit(features, target)
print(best_model_grid.best_estimator_)
```

```
LogisticRegression(penalty='l1', solver='saga')
```

In [60]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [61]:
```
clf=RandomForestClassifier(n_estimators=100)
```

In [62]:
```
clf.fit(X_train,y_train)
```

Out[62]:
```
RandomForestClassifier()
```

In [63]:
```
y_pred=clf.predict(X_test)
```

In [64]:
```
from sklearn import metrics

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 1.0
```

In [ ]:
```

```