In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [3]:

```
col_name = ["sepal length","sepal width","petal length","petal width","class"]
```

In [4]:

```
df = pd.read_csv("iris.csv",header=None,names=col_name)
```

In [5]:

```
df
```

Out[5]:

	sepal length	sepal width	petal length	petal width	class
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
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

In [6]:

```
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   class         150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [7]:

```
df.describe()
```

Out[7]:

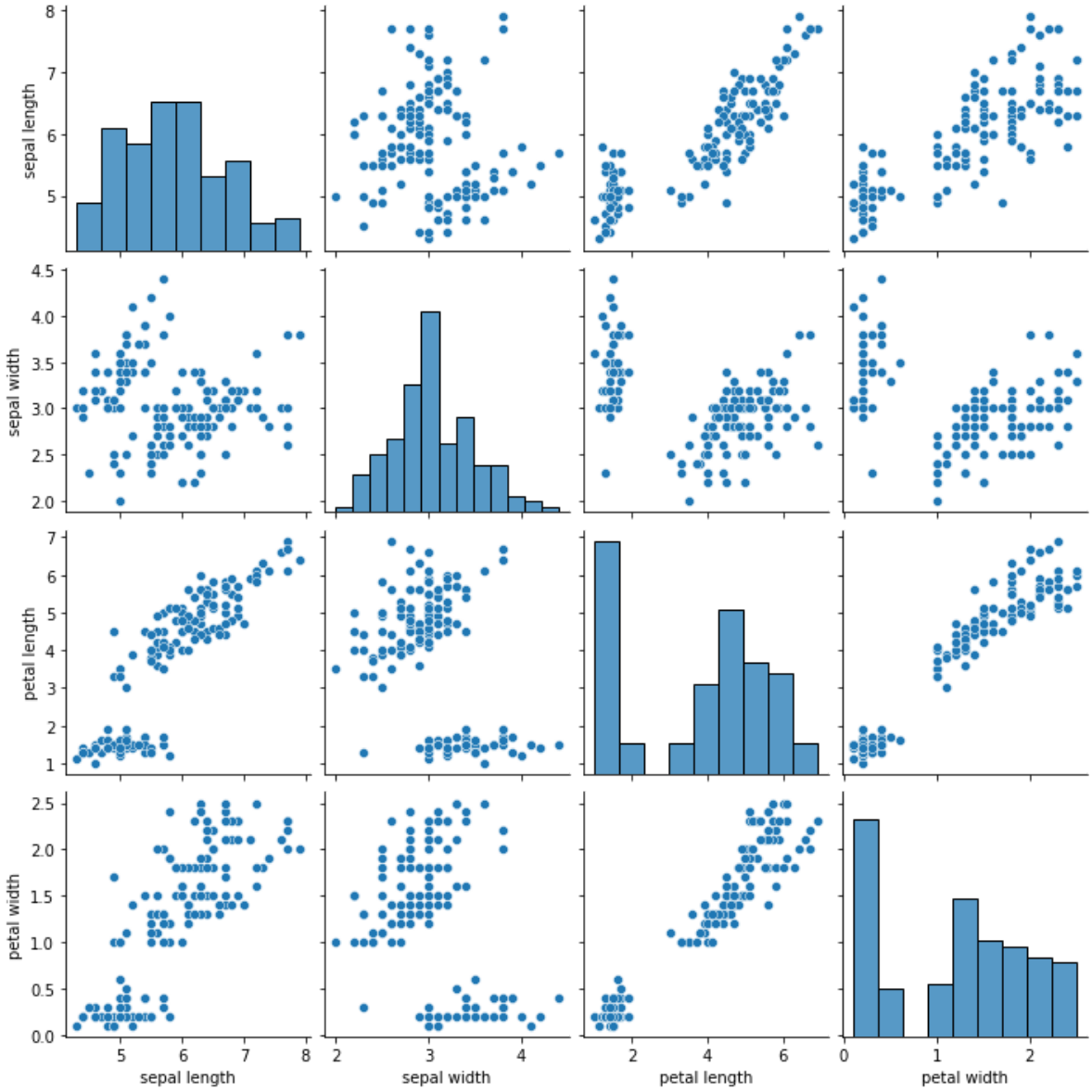
	sepal length	sepal width	petal length	petal width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000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

In [8]:

```
sns.pairplot(df)
```

Out[8]:

```
<seaborn.axisgrid.PairGrid at 0x2a2bee1f5e0>
```

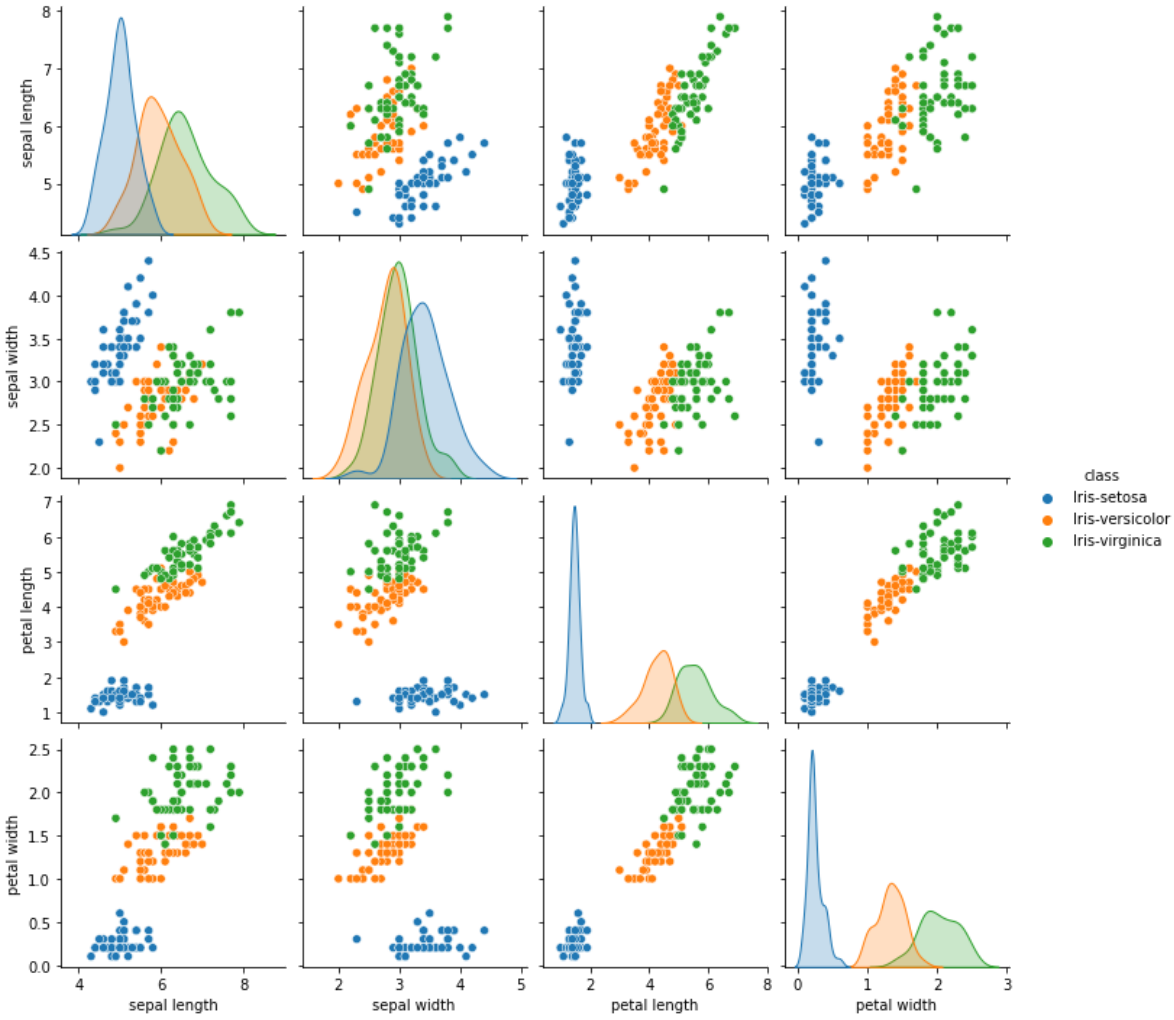


In [9]:

```
sns.pairplot(df,hue="class")
```

Out[9]:

```
<seaborn.axisgrid.PairGrid at 0x2a2c02b83d0>
```



In [11]:

```
df.groupby("class").size()
```

Out[11]:

```
class
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
dtype: int64
```

In [12]:

```
x = df.iloc[:,0:-1]
```

In [13]:

```
x
```

Out[13]:

	sepal length	sepal width	petal length	petal width
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
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns

In [14]:

```
y = df.iloc[:,-1]
```

In [15]:

```
y
```

Out[15]:

```
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: class, Length: 150, dtype: object
```

In [16]:

```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

y = le.fit_transform(y)
```

In [17]:

```
y
```

Out[17]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

In [18]:

```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size=0.3,random_state=1)
```

In [19]:

```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
```

In [20]:

```
pipe = Pipeline(steps=[("scaler",StandardScaler()),("svm",SVC())])
```

In [21]:

```
pipe.fit(xtrain,ytrain)
ypred = pipe.predict(xtest)
```

In [22]:

```
from sklearn.metrics import classification_report
```

In [23]:

```
cr = classification_report(ytest,ypred)
```

In [25]:

```
print(f"Classification Report:-\n{cr}")
```

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

           0       1.00      1.00      1.00        14
           1       0.94      0.94      0.94        18
           2       0.92      0.92      0.92        13

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

In [26]:

```
train = pipe.score(xtrain,ytrain)
test = pipe.score(xtest,ytest)

print(f"Training Score:-{train}\nTesting Score:-{test}")
```

```
Training Score:-0.9809523809523809
Testing Score:-0.9555555555555556
```

In [29]:

```
from sklearn.model_selection import GridSearchCV
```

In [31]:

```
parameter = {
            "C" : [0.1,1,10],
            "gamma" : [0.1,0.01,0.001],
            "kernel" : ["rbf"]
            }
```

In [33]:

```
grid = GridSearchCV(SVC(),parameter,verbose=2)
```

In [34]:

```
grid.fit(xtrain,ytrain)
```

```
Fitting 5 folds for each of 9 candidates, totalling 45 fits
[CV] END .....................C=0.1, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END .....................C=0.1, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END .....................C=0.1, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END .....................C=0.1, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END .....................C=0.1, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END ....................C=0.1, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END ....................C=0.1, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END ....................C=0.1, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END ....................C=0.1, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END ....................C=0.1, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END ...................C=0.1, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END ...................C=0.1, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END ...................C=0.1, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END ...................C=0.1, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END ...................C=0.1, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END .....................C=1, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END .....................C=1, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END .....................C=1, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END .....................C=1, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END .....................C=1, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END ....................C=1, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END ....................C=1, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END ....................C=1, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END ....................C=1, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END ....................C=1, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END ...................C=1, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END ...................C=1, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END ...................C=1, gamma=0.001, kernel=rbf; total time=
0.0s
```

```
[CV] END ......................C=1, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END ......................C=1, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END ......................C=10, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END ......................C=10, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END ......................C=10, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END ......................C=10, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END ......................C=10, gamma=0.1, kernel=rbf; total time=
0.0s
[CV] END .....................C=10, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END .....................C=10, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END .....................C=10, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END .....................C=10, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END .....................C=10, gamma=0.01, kernel=rbf; total time=
0.0s
[CV] END ....................C=10, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END ....................C=10, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END ....................C=10, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END ....................C=10, gamma=0.001, kernel=rbf; total time=
0.0s
[CV] END ....................C=10, gamma=0.001, kernel=rbf; total time=
0.0s
```

Out[34]:

```
GridSearchCV(estimator=SVC(),
             param_grid={'C': [0.1, 1, 10], 'gamma': [0.1, 0.01, 0.001],
                         'kernel': ['rbf']},
             verbose=2)
```

In [35]:

```
grid.best_params_
```

Out[35]:

```
{'C': 10, 'gamma': 0.1, 'kernel': 'rbf'}
```

In [37]:

```
grid.best_score_
```

Out[37]:

```
0.9714285714285715
```

In [38]:

```
grid.best_estimator_
```

Out[38]:

```
SVC(C=10, gamma=0.1)
```

In [39]:

```
svm = grid.best_estimator_

svm.fit(xtrain,ytrain)

ypred = svm.predict(xtest)
```

In [40]:

```
from sklearn.metrics import classification_report

cr = classification_report(ytest,ypred)

print(f"Classification Report:-\n {cr}")
```

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

           0       1.00      1.00      1.00        14
           1       1.00      1.00      1.00        18
           2       1.00      1.00      1.00        13

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

In [41]:

```
train = svm.score(xtrain,ytrain)
test = svm.score(xtest,ytest)

print(f"Training Accuracy :- {train}\nTesting Accuracy :- {test}")
```

```
Training Accuracy :- 0.9809523809523809
Testing Accuracy :- 1.0
```