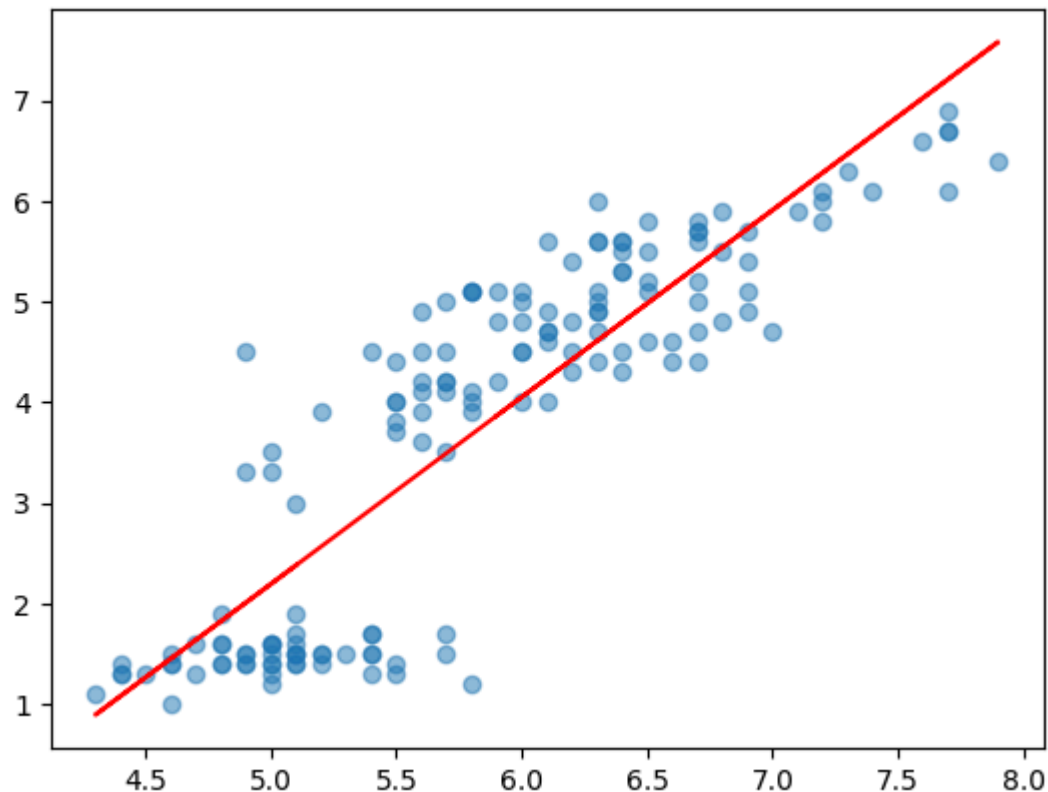


```
In [1]: import pandas as pd
        from sklearn.datasets import load_iris
        import matplotlib.pyplot as plt

        iris=load_iris()
        iris=pd.DataFrame(iris.data,columns=iris.feature_names)
        iris['class']=load_iris().target
        iris['class']=iris['class'].map({0:'Setosa',1:'Versicolour',2:'Virginica'})
```

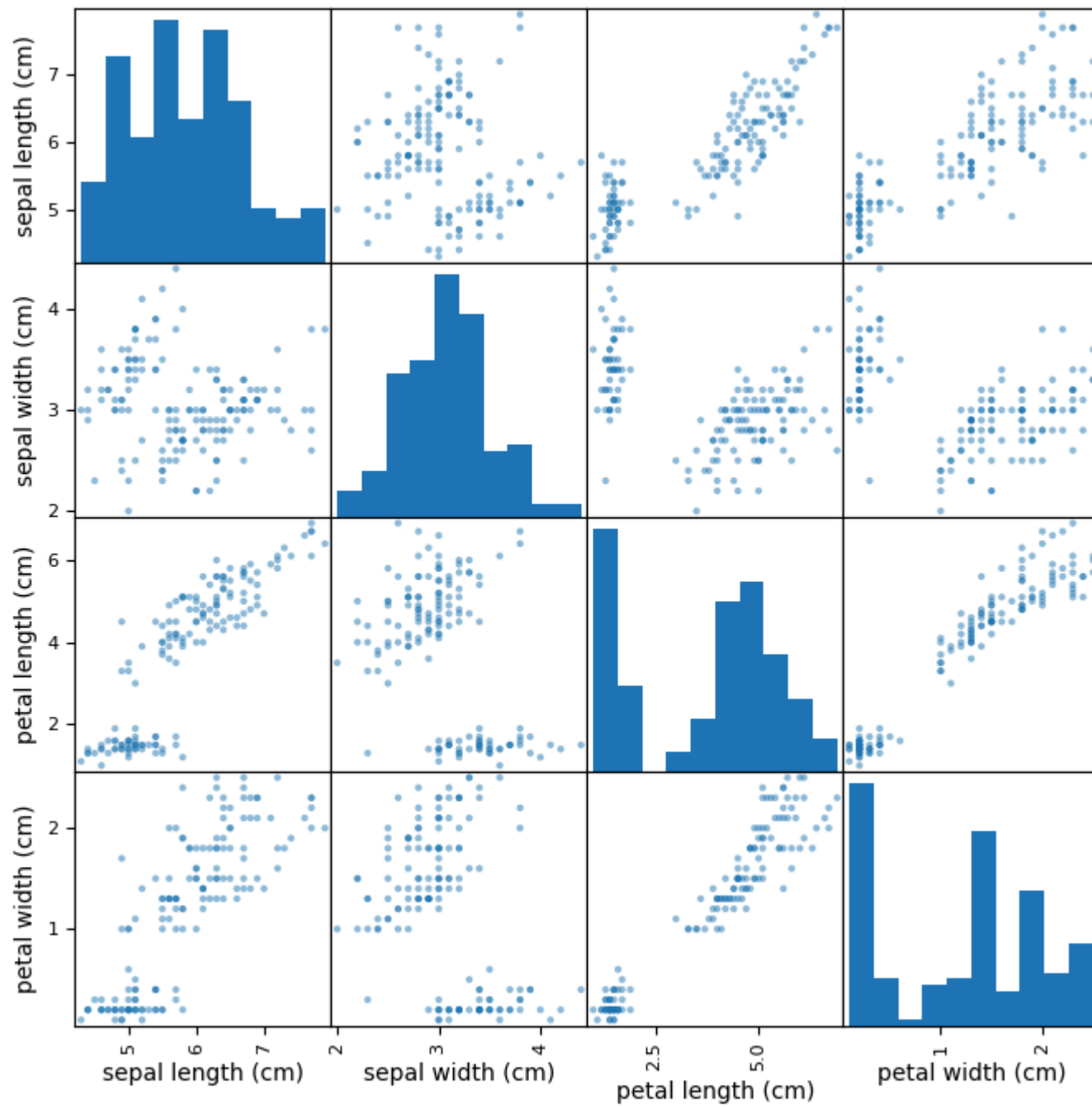
```
In [2]: import numpy as np

        X,Y=iris['sepal length (cm)'],iris['petal length (cm)']
        b1,b0 = np.polyfit(X,Y,1)
        plt.scatter(x=X,y=Y,alpha=0.5)
        plt.plot(X,b1*X+b0,color='r')
        plt.show()
```



```
In [3]: from pandas.plotting import scatter_matrix
        scatter_matrix(iris,alpha=0.5,figsize=(8,8),diagonal='hist')
```

```
plt.show()
```



2. 수식

KDE의 추정식은 다음과 같습니다.

$$\hat{f}_h(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right)$$

- n : 데이터 개수
 - h : 밴드위스(**bandwidth**) — 곡선의 폭을 조절하는 매개변수
 - $K(\cdot)$: 커널 함수 (예: Gaussian, Epanechnikov, Uniform 등)
 - x_i : 데이터 포인트
-

📌 공통점

- 둘 다 데이터 분포를 부드럽게 근사한다.
- 결과물로 **연속적인 확률밀도 함수(PDF)**를 얻을 수 있다.
- 초매개변수(커널 폭 h 또는 가우시안 개수 k) 선택이 성능에 매우 중요하다.

📌 차이점

| 구분 | KDE | GMM |
|---------|-----------------------------------|--------------------------------|
| 모델 유형 | 비모수적 (모양 가정 없음) | 모수적 (가우시안 혼합 가정) |
| 기본 아이디어 | 각 데이터 포인트마다 하나의 커널을 씌워서 합친다 | "데이터는 k 개의 가우시안 분포 혼합"이라고 가정 |
| 커널 개수 | 데이터 개수(n) = 커널 개수 | 가우시안 개수(k)는 모델에서 설정 |
| 학습 방식 | 별도 학습 과정 없음, bandwidth(h)만 선택 | EM 알고리즘으로 평균·분산·가중치 추정 |
| 유연성 | 분포 모양이 자유롭지만, 대역폭 선택에 민감 | 분포 모양은 가우시안 혼합으로 제한 |
| 복잡도 | $O(n)$ 평가 (데이터 많으면 느림) | $O(k)$ 평가 (k 는 보통 작음) |


📌 KDE에서 "여러 커널"이란?

- KDE는 데이터 포인트마다 하나씩 커널(예: 가우시안)을 씌움
- 이때 커널 종류를 여러 개 시도할 수 있지만, 보통은 가우시안 하나를 쓰고
밴드위스(h)만 여러 후보로 시도해서 최적값을 선택
- 커널 종류보다 밴드위스가 분포 부드러움에 훨씬 큰 영향을 줌
(→ 마치 GMM에서 k 조정이 중요한 것처럼 KDE에서는 h 조정이 핵심)

📌 그림으로 차이 감

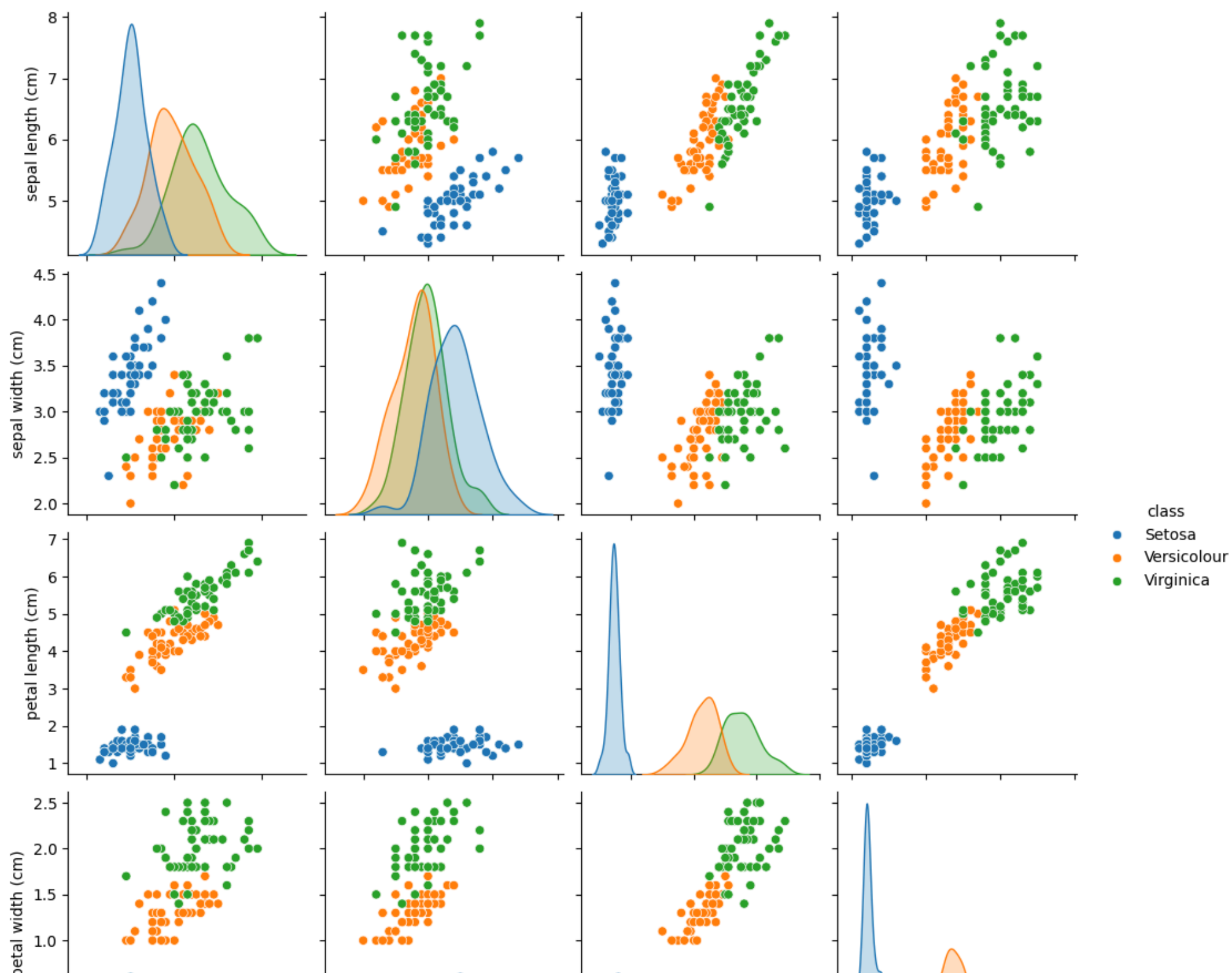
- **KDE** : 모든 데이터에 얇은 종모양(또는 다른 커널) 곡선을 얹어서 전부 합침
- **GMM** : "데이터는 k개의 가우시안 클러스터"라고 가정하고 그 가우시안들의 합으로 분포를 표현

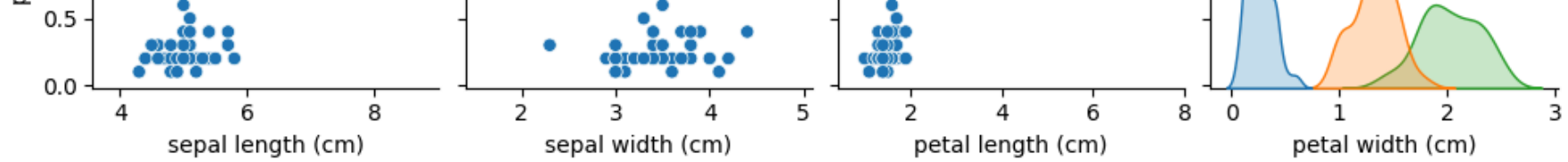
💡 요약

- KDE = "데이터 포인트 = 커널 중심" → 전체 데이터를  써서 부드러운 분포를 만들고, 모양 가정이 없음
- GMM = "분포 = k개의 가우시안 혼합" → 데이터 개수와 무관하게 k개의 가우시안만 사용, 모양이 제한됨

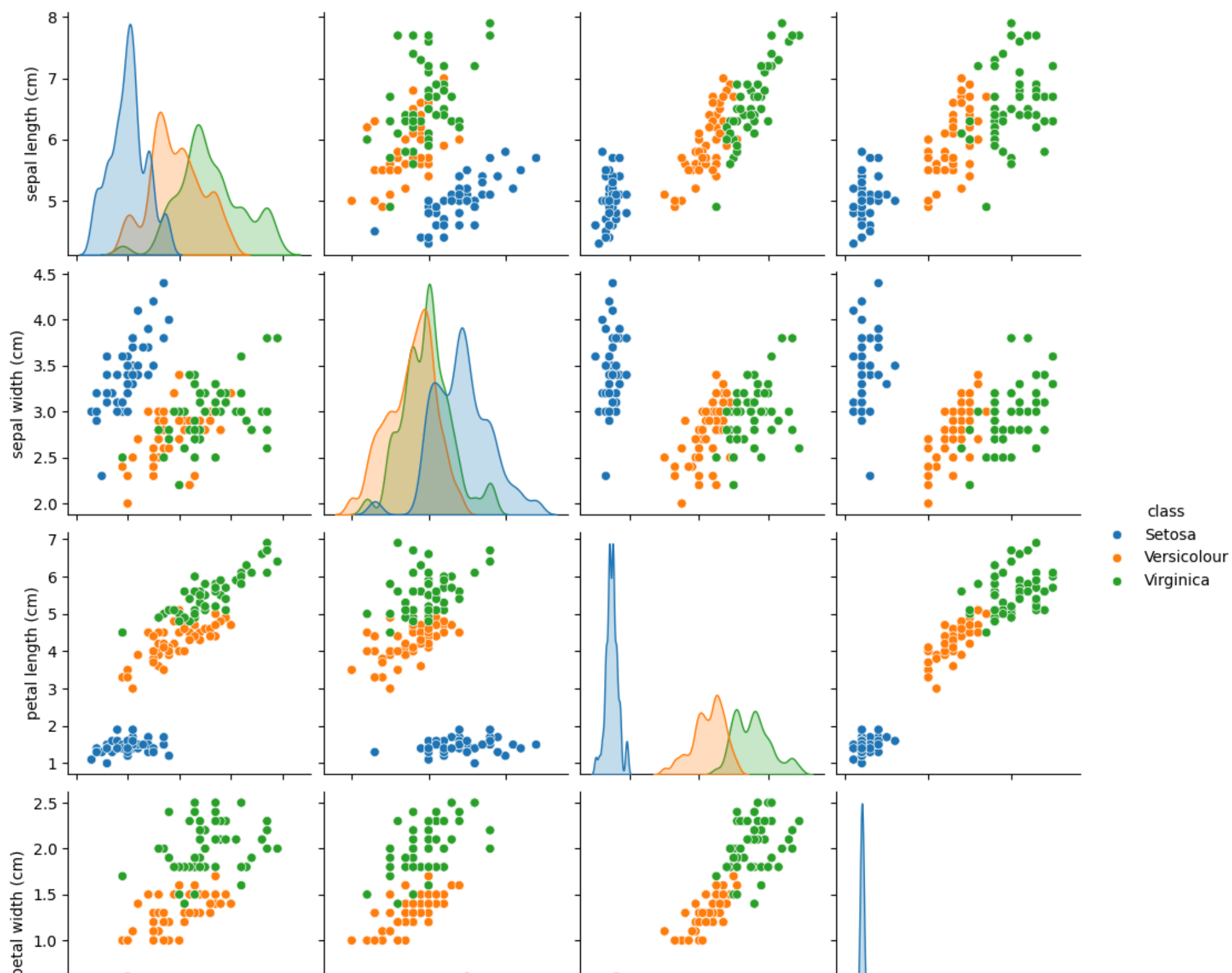
In [4]: `import seaborn as sns`

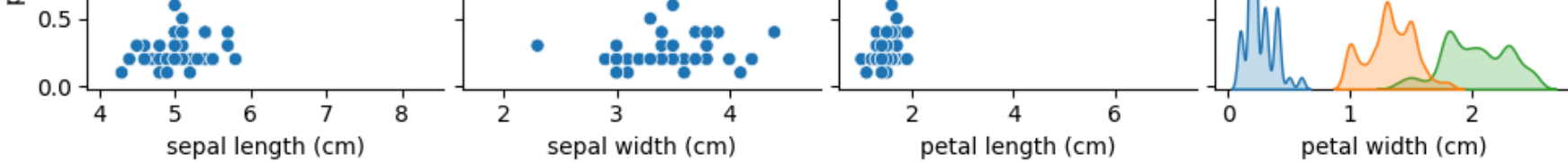
```
sns.pairplot(iris,diag_kind='kde',hue='class')  
plt.show()
```



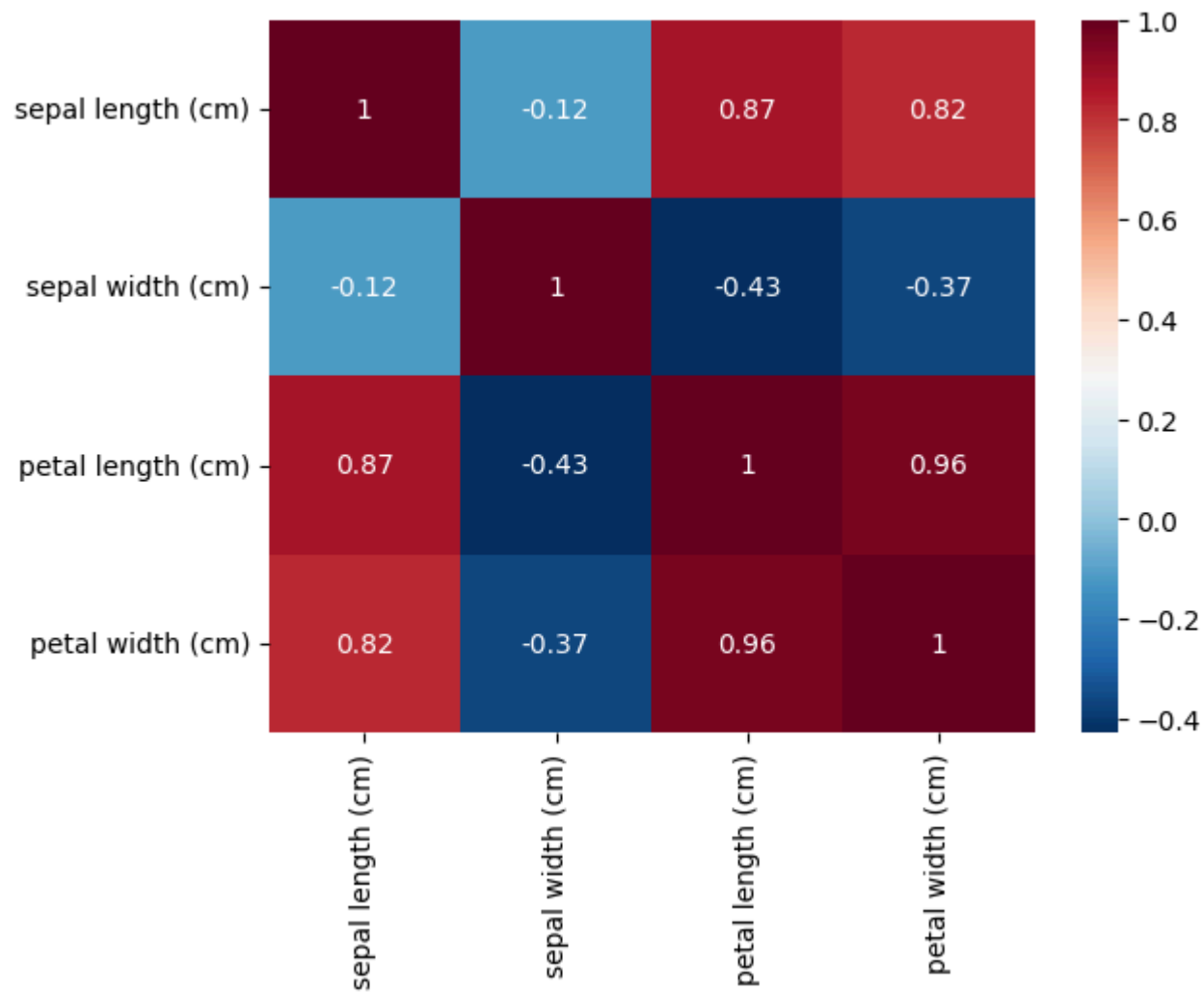


```
In [5]: sns.pairplot(iris,diag_kind='kde',hue='class',diag_kws={'bw_adjust': 0.5} ) # bandwidth 조절)
plt.show()
```





```
In [6]: iris_corr=iris.drop(columns='class').corr(method='pearson')
sns.heatmap(iris_corr,xticklabels=iris_corr.columns,yticklabels=iris_corr.columns,cmap='RdBu_r',annot=True)
plt.show()
```



```
In [ ]: !pip install ydata_profiling
!pip install pydantic pydantic-settings
```

```
In [8]: from ydata_profiling import ProfileReport
```

```
ProfileReport(iris)
```

[Upgrade to ydata-sdk](#)

Improve your data and profiling with ydata-sdk, featuring data quality scoring, redundancy detection, outlier identification, text validation, and synthetic data generation.

```
Summarize dataset:  0%|          | 0/5 [00:00<?, ?it/s]
```

```
100%|██████████| 5/5 [00:00<00:00, 29.71it/s]
```

```
Generate report structure:  0%|          | 0/1 [00:00<?, ?it/s]
```

```
Render HTML:  0%|          | 0/1 [00:00<?, ?it/s]
```

Overview

Brought to you by [YData](#)

Overview

Alerts 6

Reproduction

Dataset statistics

| | |
|-------------------------------|---------|
| Number of variables | 5 |
| Number of observations | 150 |
| Missing cells | 0 |
| Missing cells (%) | 0.0% |
| Duplicate rows | 1 |
| Duplicate rows (%) | 0.7% |
| Total size in memory | 6.0 KiB |
| Average record size in memory | 40.9 B |

Variable types

| | |
|-------------|---|
| Numeric | 4 |
| Categorical | 1 |

Variables

Select Columns ▾

Out[8]:

In []: