# Exercise 2: Visual Exploratory Data Analysis

# Objective

To understand the role of visual exploratory data analysis (EDA) in identifying class separability and feature importance using paired plots, box plots, violin plots, and correlation heatmaps.

# Introduction

Visual EDA helps in understanding data distribution, relationships, and class separability. Before applying machine learning models, visualizations help identify:  
- Which features best separate the classes.  
- Which features have high within-class variability or overlap.  
- Correlation between features to avoid redundancy.  
  
Visualizations used:  
1. Pair Plots: Show pairwise relationships and class clusters.  
2. Box Plots: Display spread (IQR) and outliers.  
3. Violin Plots: Show distribution density and spread.  
4. Correlation Heatmaps: Summarize linear relationships between features.

# Dataset Used

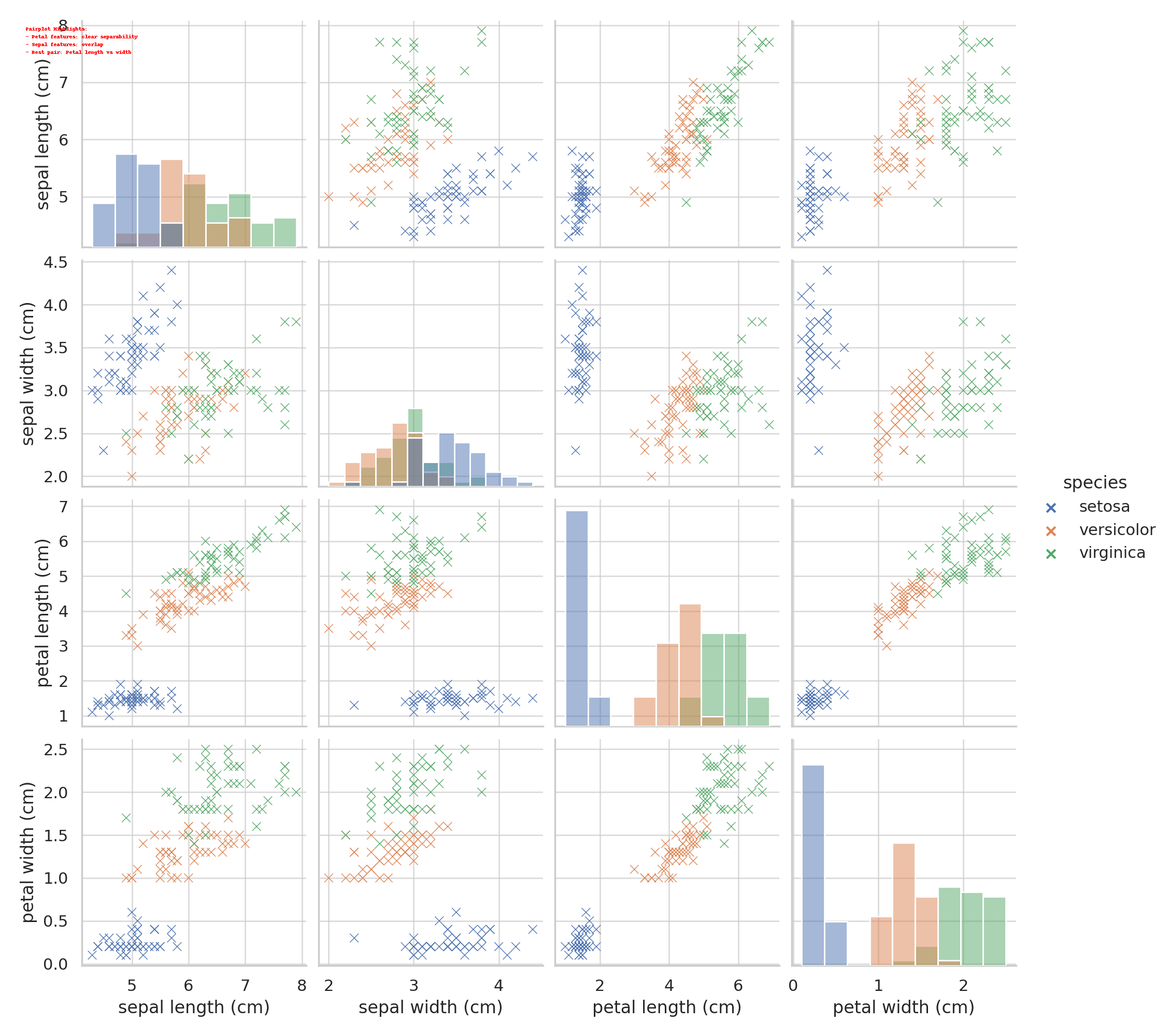
Dataset: Iris Dataset  
Features: Sepal length (cm), Sepal width (cm), Petal length (cm), Petal width (cm)  
Target: Species (Setosa, Versicolor, Virginica)

# Python Program

# Load Dataset  
from sklearn.datasets import load\_iris  
import pandas as pd  
import seaborn as sns  
import matplotlib.pyplot as plt  
  
iris = load\_iris()  
df = pd.DataFrame(iris.data, columns=iris.feature\_names)  
df["species"] = pd.Categorical.from\_codes(iris.target, iris.target\_names)  
  
# Pair Plot  
sns.pairplot(df, hue="species", diag\_kind="hist")  
plt.show()  
  
# Box Plots  
for col in iris.feature\_names:  
 sns.boxplot(x="species", y=col, data=df)  
 plt.title(f"Boxplot: {col}")  
 plt.show()  
  
# Violin Plots  
for col in iris.feature\_names:  
 sns.violinplot(x="species", y=col, data=df)  
 plt.title(f"Violin Plot: {col}")  
 plt.show()  
  
# Correlation Heatmap  
sns.heatmap(df.corr(numeric\_only=True), annot=True, cmap="coolwarm")  
plt.title("Correlation Heatmap")  
plt.show()

# Annotated Visualizations

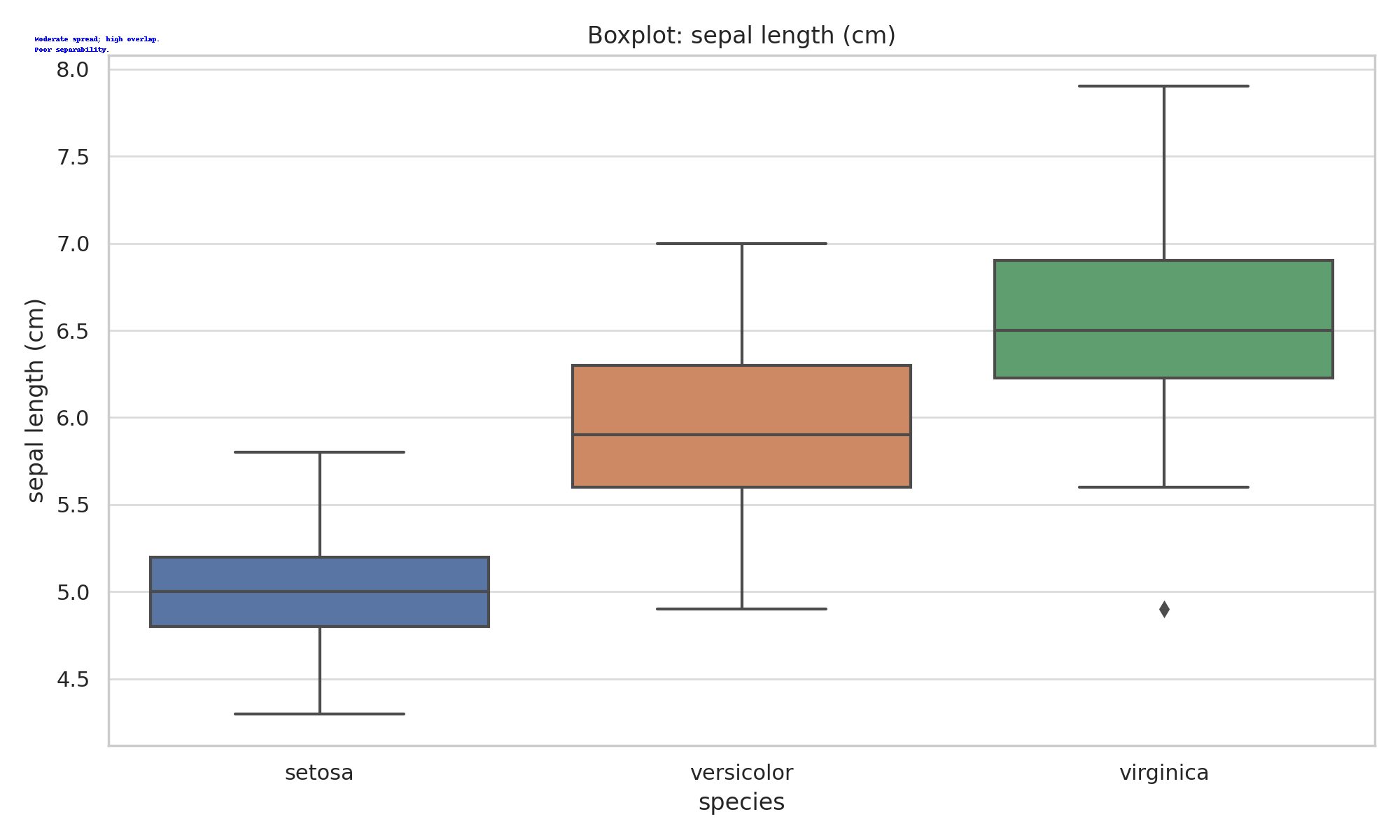
## Pair Plot (Annotated)



Highlights: Petal features show clear class separation; Sepal features overlap. Best pair: Petal length vs Petal width.

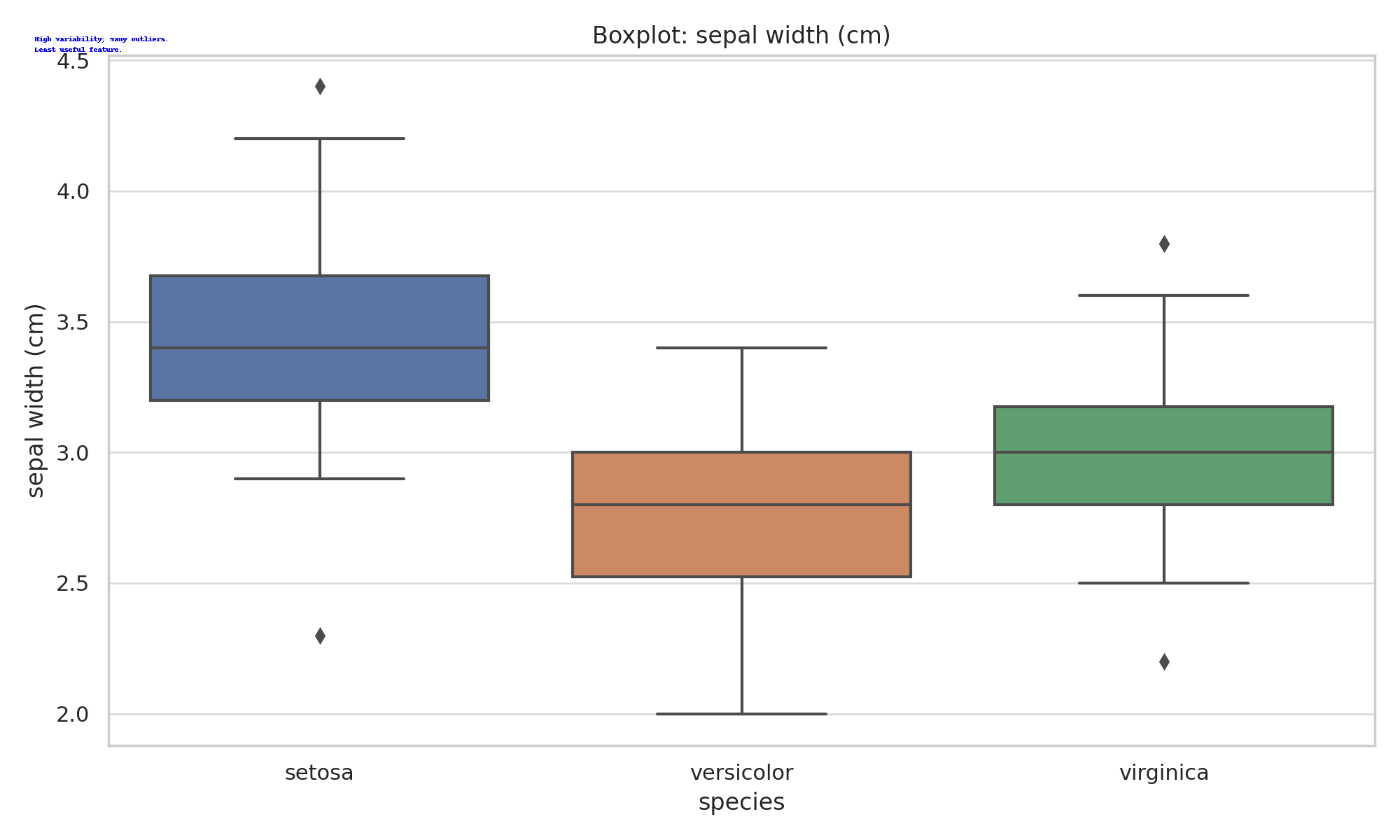
# Box Plot Analysis (Annotated)

## Sepal Length (Cm)



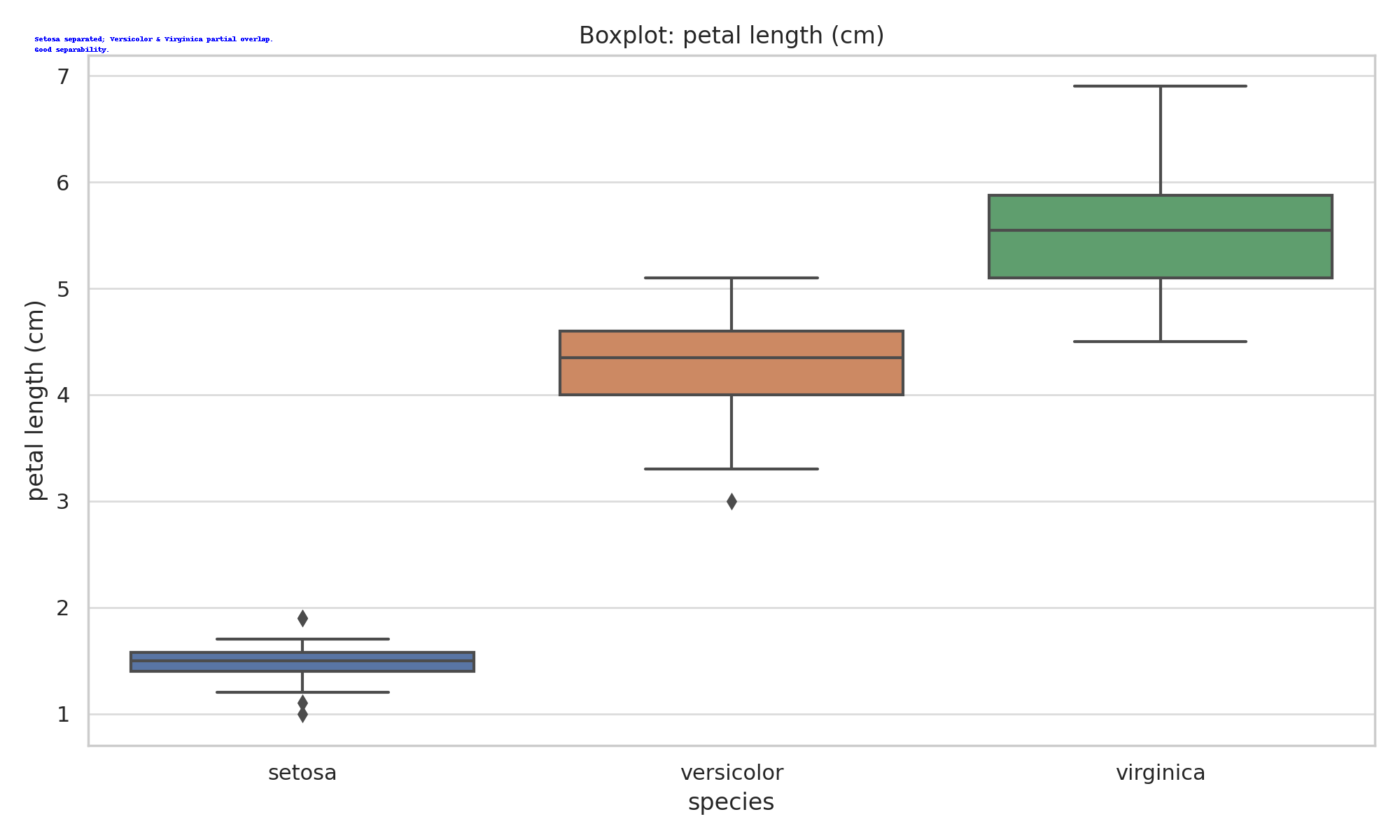
Moderate spread; high overlap.  
Poor separability.

## Sepal Width (Cm)



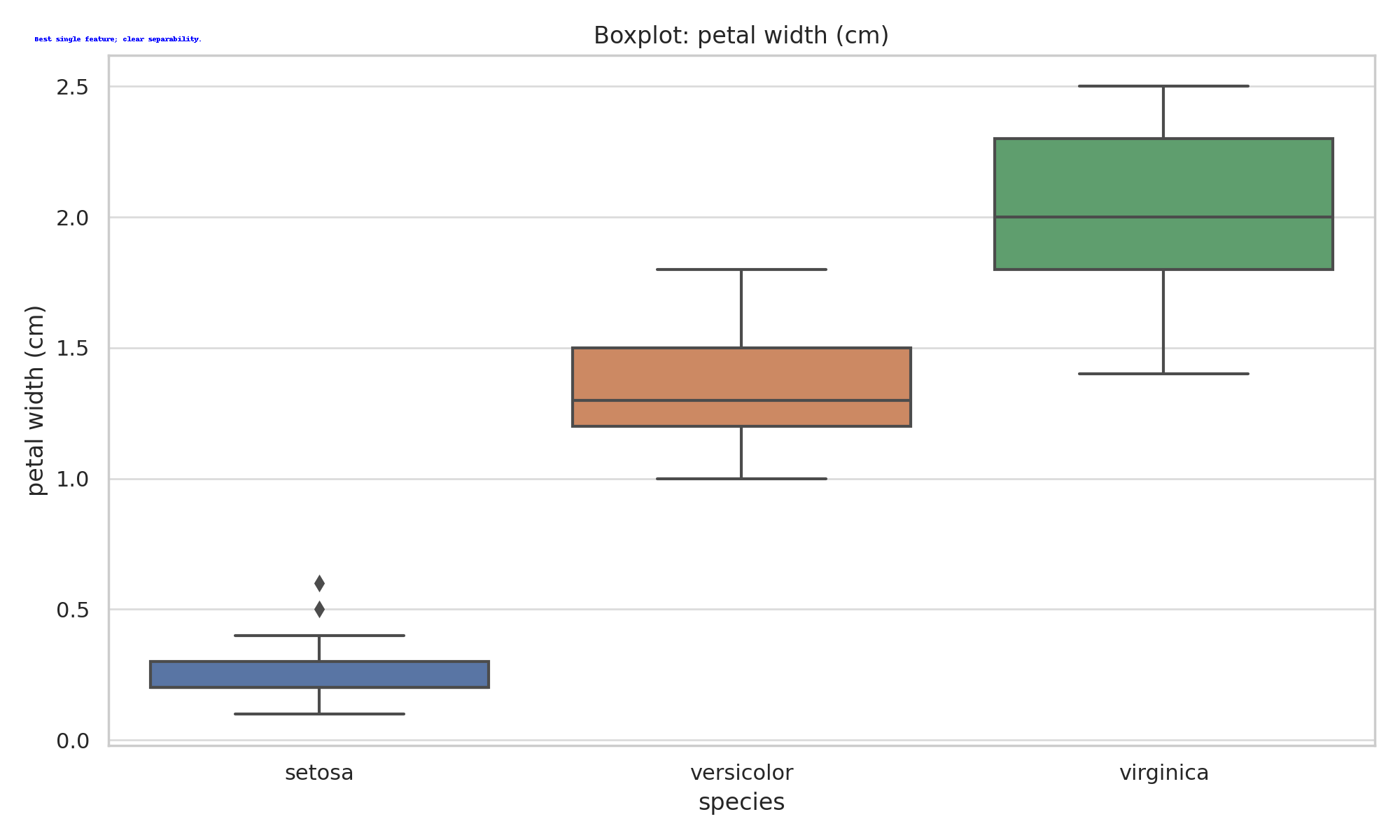
High variability; many outliers.  
Least useful feature.

## Petal Length (Cm)



Setosa separated; Versicolor & Virginica partial overlap.  
Good separability.

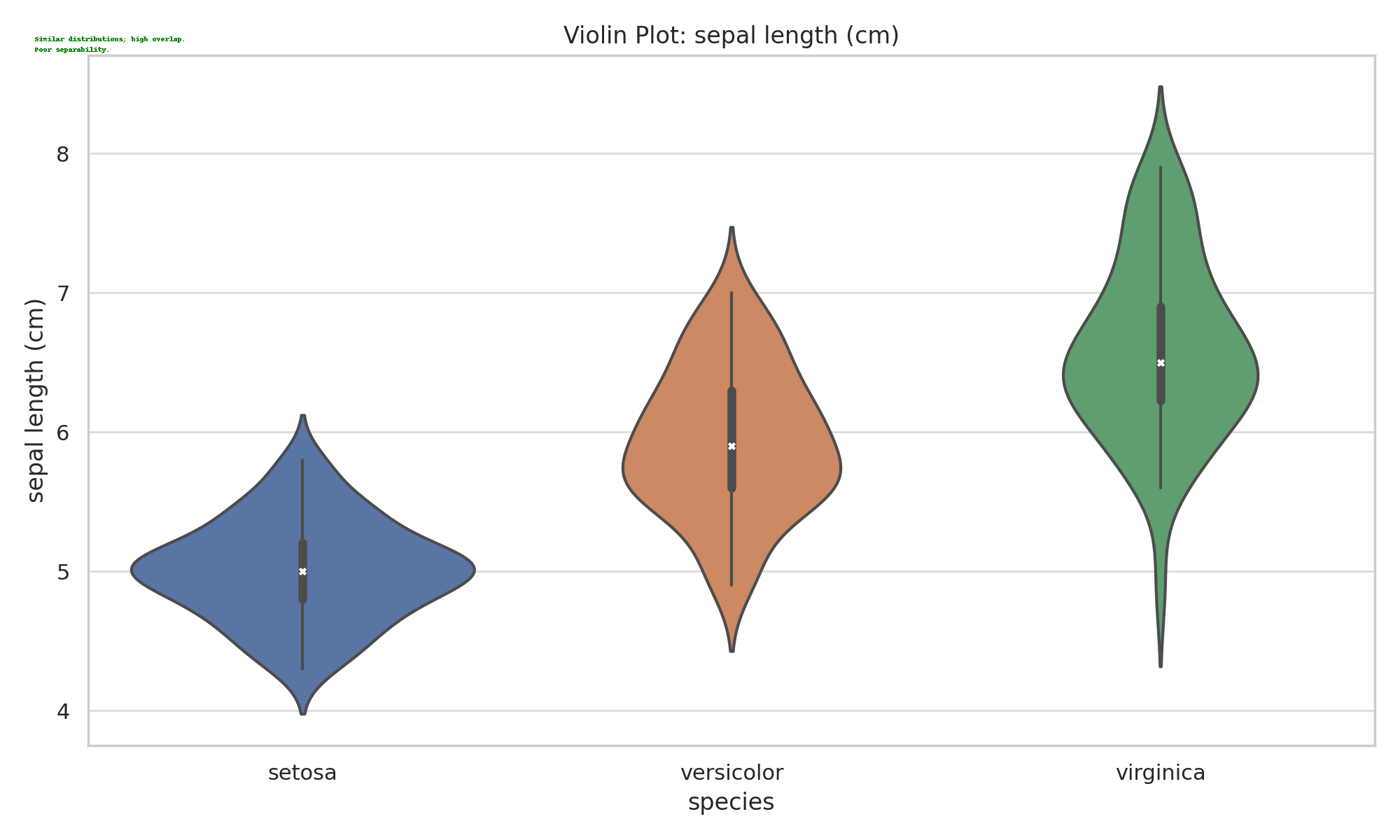
## Petal Width (Cm)



Best single feature; clear separability.

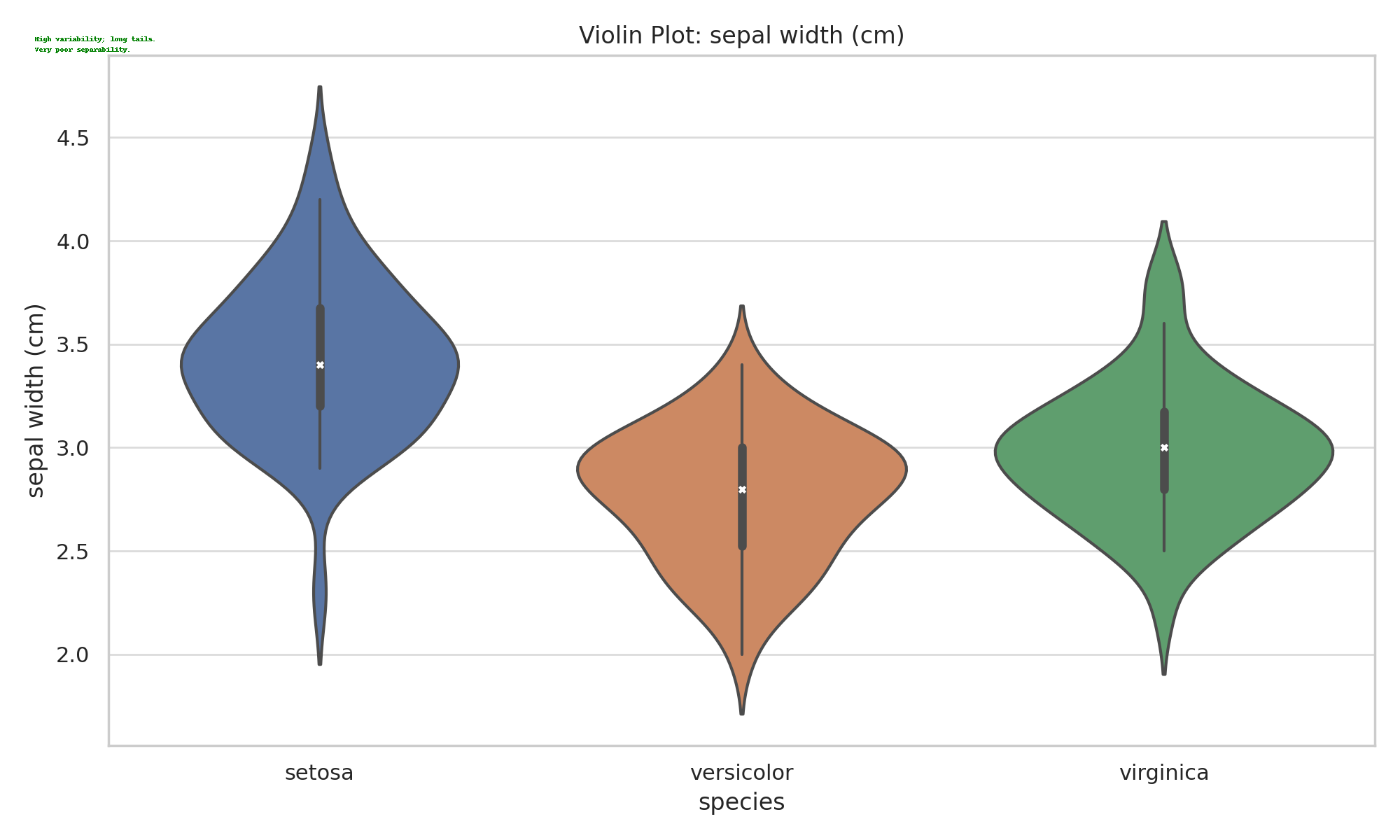
# Violin Plot Analysis (Annotated)

## Sepal Length (Cm)



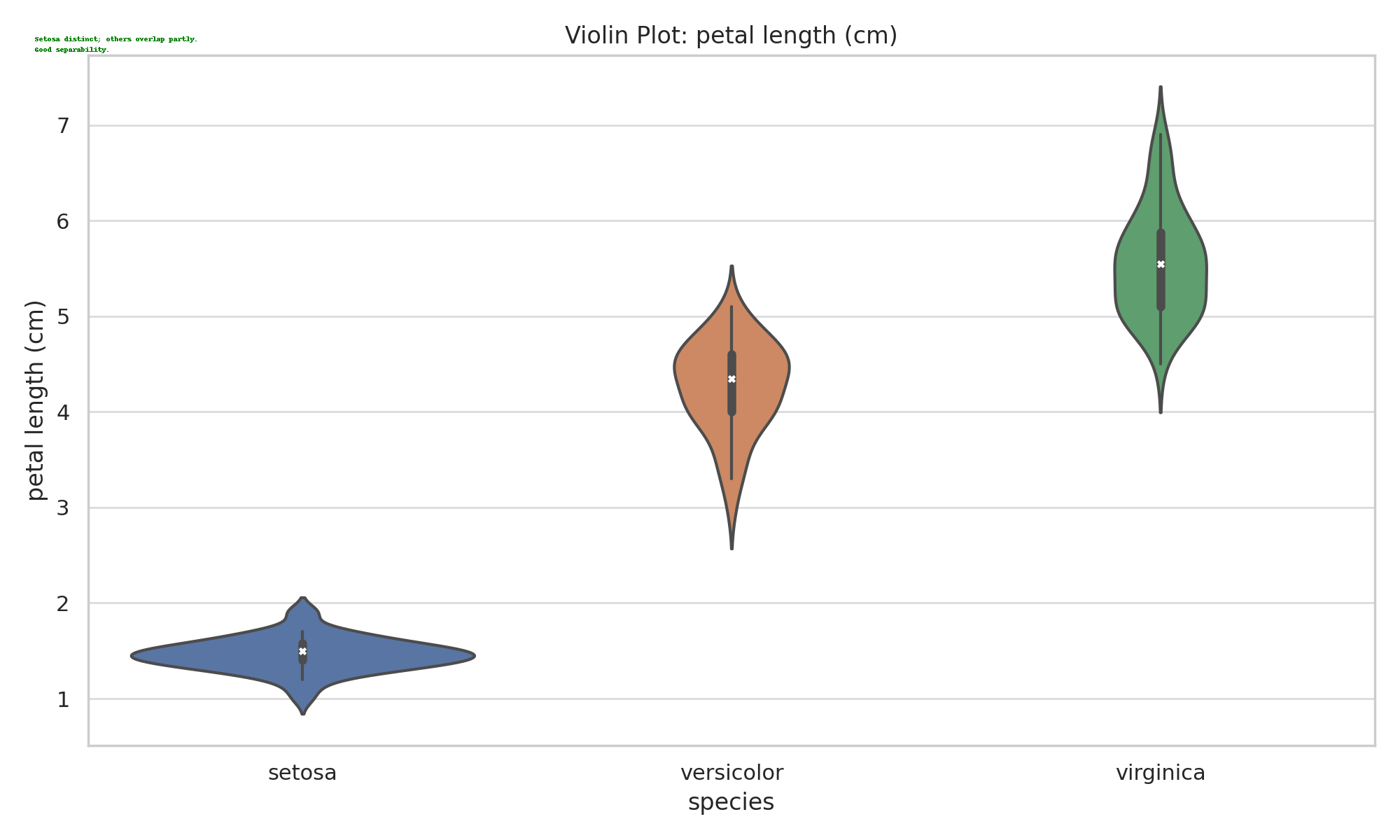
Similar distributions; high overlap.  
Poor separability.

## Sepal Width (Cm)



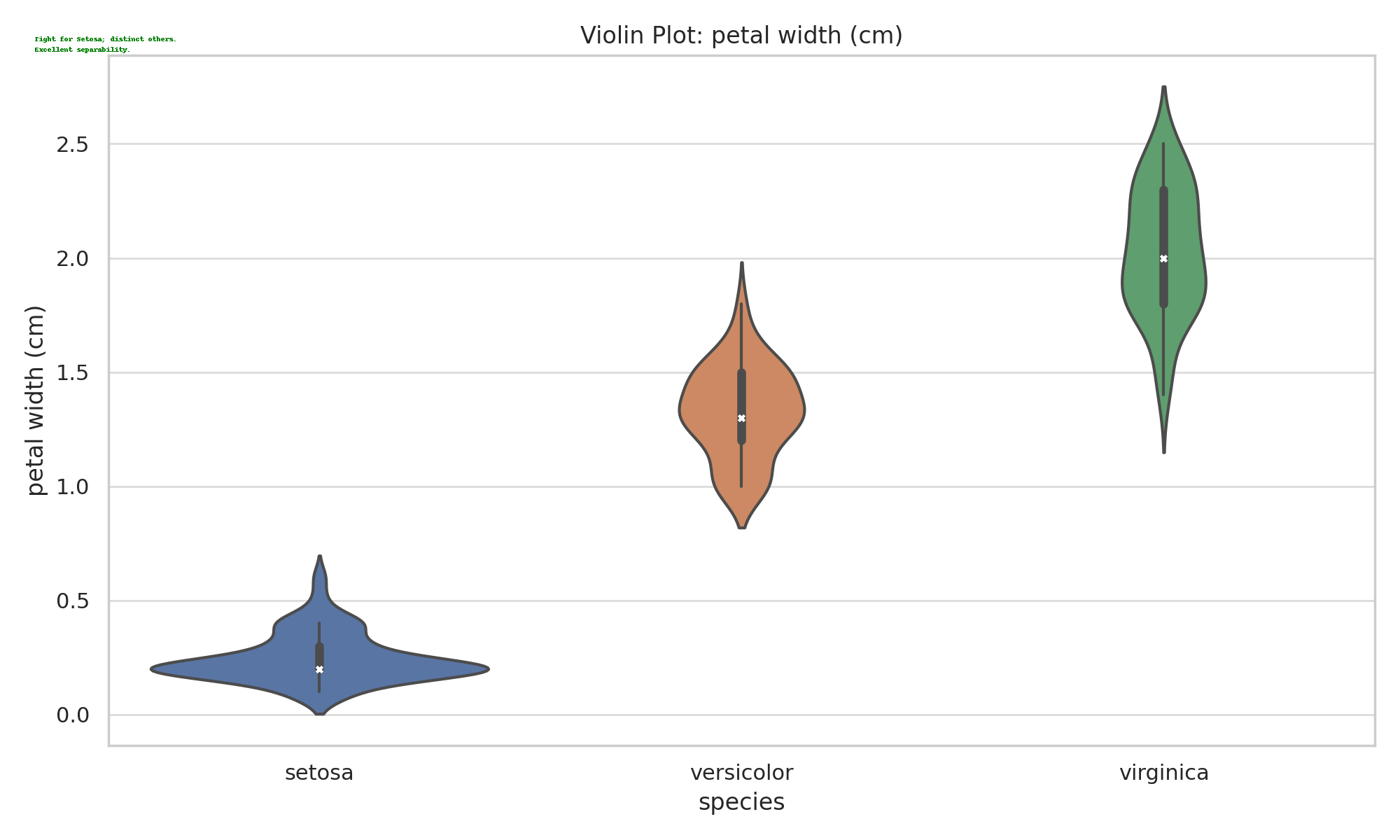
High variability; long tails.  
Very poor separability.

## Petal Length (Cm)



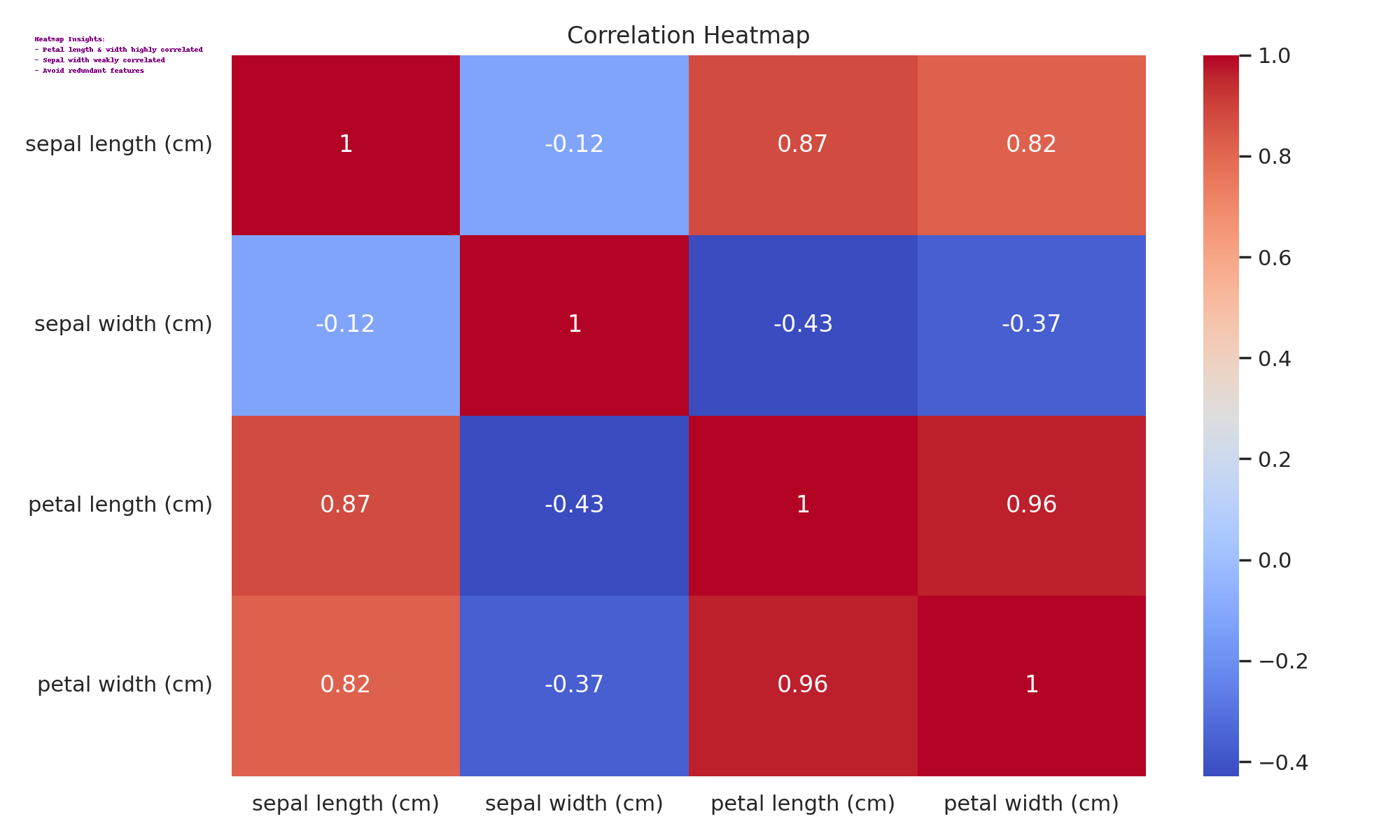
Setosa distinct; others overlap partly.  
Good separability.

## Petal Width (Cm)



Tight for Setosa; distinct others.  
Excellent separability.

# Correlation Heatmap (Annotated)



Petal length and petal width are highly correlated; sepal width shows weak correlation. Remove redundant features if necessary.

1. **Task 1** [CO1] [BTL 2] [3 marks]  
   Implement the above task in colab and share the link here (give public access). Go through the libraries and the functions called specifically for different plots. In the Iris dataset, which features are most discriminative and why?

**Solution :**

<https://colab.research.google.com/drive/1okJBo9qpsFcIQbk3LVEnWwT5KCkpnAe2?usp=sharing>

The most discriminative features are **Petal Width** and **Petal Length** because they show clear class separation in pair plots, violin plots, and box plots. Sepal features have high overlap and variability, hence are less useful for classification.

1. **Task 2** [CO1] [BTL 3] [1 marks]

Take any mobile app classification dataset having 10 features. Concisely explain how would you proceed to find the best feature combinations to yield highest classification performance.

For task 2 I have selected dataset: <https://www.kaggle.com/datasets/mhamidasn/user-feedback-data-from-the-top-15-mobile-apps>

**Data Understanding and Cleaning:**

Remove irrelevant identifiers (review\_id, app\_id) since they don’t help classification.

Handle missing values in RC\_ver.

Decide the target variable (eg score high vs low rating).

**Feature Engineering**

Text Features (content): Convert reviews into numerical features using TF-IDF, Bag of Words, or embeddings.

Numeric Features: Use score (target), TU\_count (activity measure).

Categorical Features: app\_name, RC\_ver (encode with one-hot or embeddings).

**Exploratory Data Analysis (EDA)**

Box/violin plots: Compare TU\_count across score groups.

Correlation heatmap: Check relationships between TU\_count and score.

Word clouds / token frequency: Identify discriminative words for high vs low ratings.

**Feature Selection**

Filter methods: Correlation, Chi-square test between features (e.g., TU\_count vs score).

Wrapper methods: Recursive Feature Elimination (RFE) on text+numeric features.

Embedded methods: Feature importance from models like Random Forest or XGBoost.

**Model Training & Evaluation**

Train classification models (Logistic Regression, Random Forest, SVM) using different feature subsets.

Use cross-validation to compare models.

The best feature combination = text embeddings (content) + numeric signals (TU\_count) + categorical (RC\_ver).