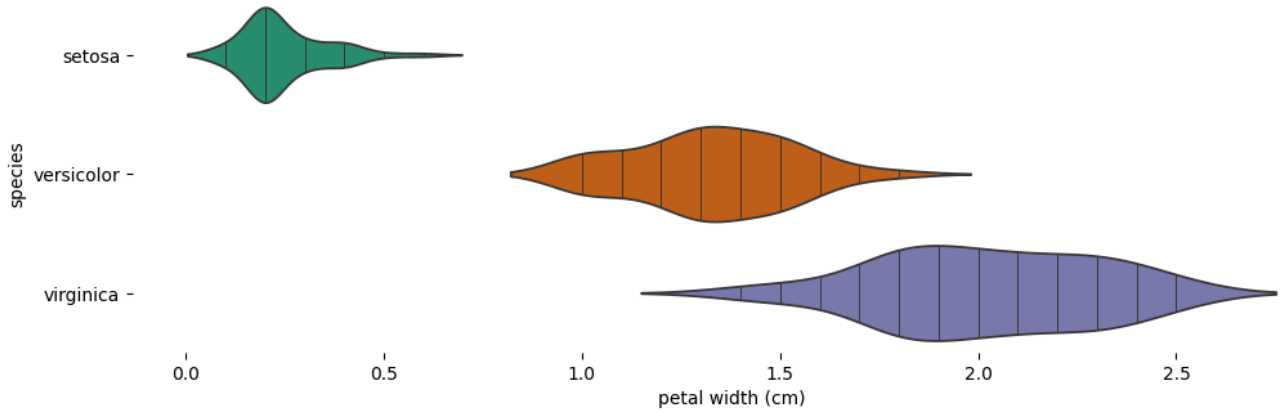
```
from sklearn.datasets import load_iris
import pandas as pd

# Load dataset
iris = load_iris()

# Create a DataFrame
df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
df['species'] = iris.target

# Map target values to species names
df['species'] = df['species'].map({i: species for i, species in enumerate(iris.target_names)})

df.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

species vs petal width (cm)

Show code

```
/tmp/ipython-input-3311808838.py:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `le

  sns.violinplot(df, x='petal width (cm)', y='species', inner='stick', palette='Dark2')
```



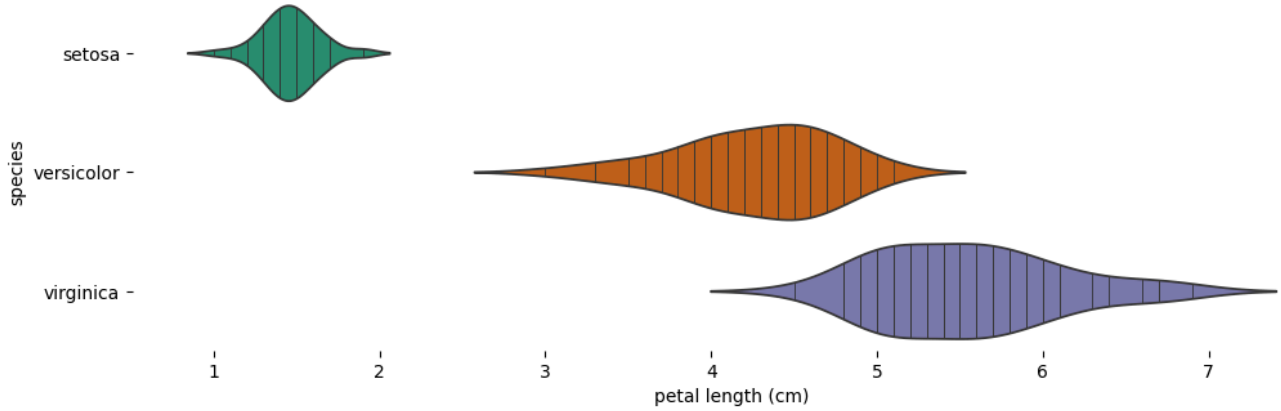
Start coding or generate with AI.

Start coding or generate with AI.

species vs petal length (cm)

Show code

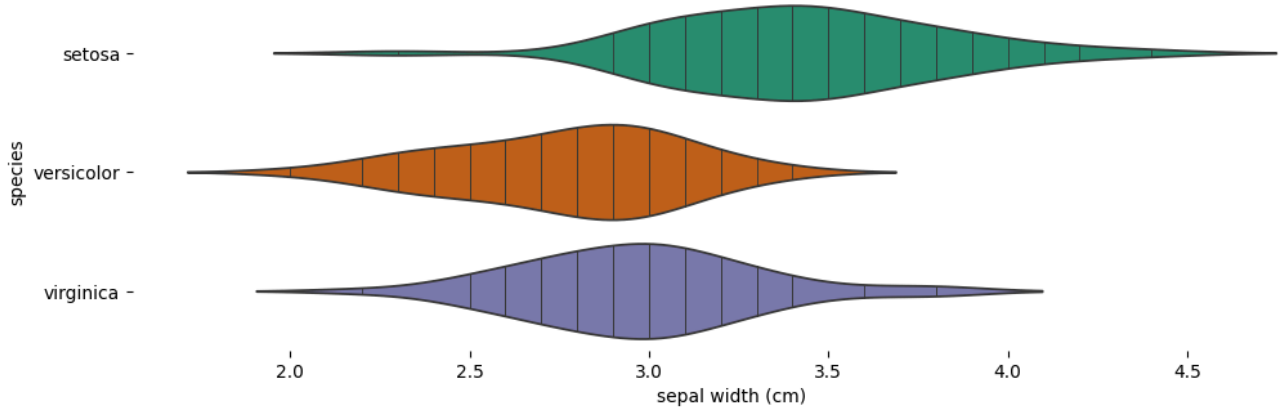
```
/tmp/ipython-input-4076210793.py:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `le

  sns.violinplot(df, x='petal length (cm)', y='species', inner='stick', palette='Dark2')
```



species vs sepal width (cm)

Show code

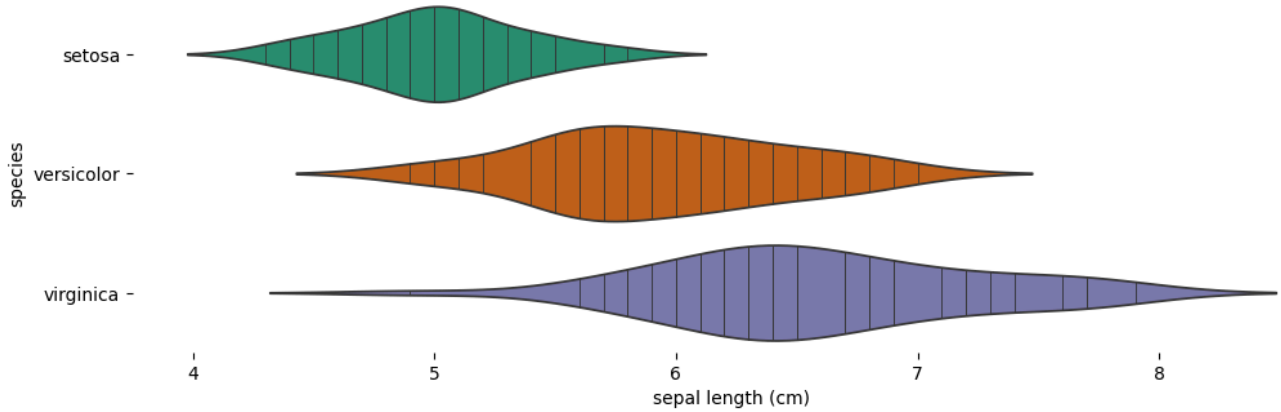
```
/tmp/ipython-input-2833929885.py:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `le

  sns.violinplot(df, x='sepal width (cm)', y='species', inner='stick', palette='Dark2')
```



species vs sepal length (cm)

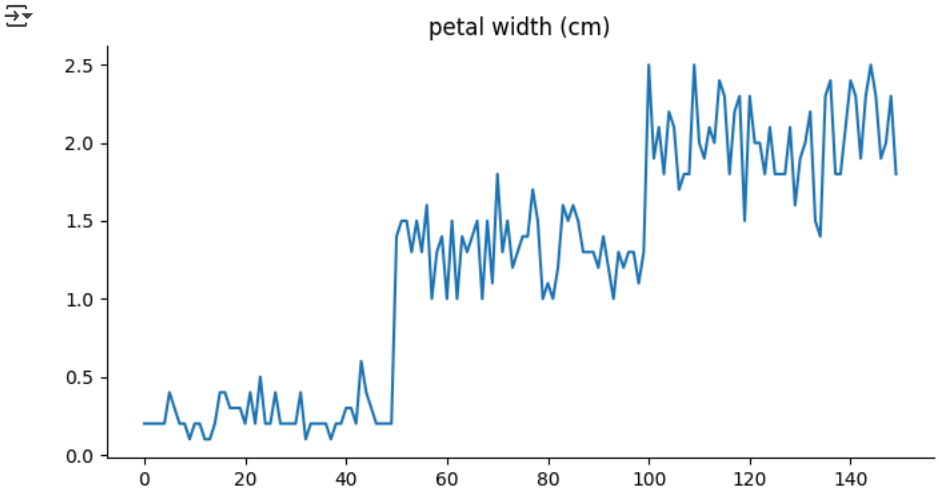
Show code

```
/tmp/ipython-input-3903507029.py:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `le

  sns.violinplot(df, x='sepal length (cm)', y='species', inner='stick', palette='Dark2')
```



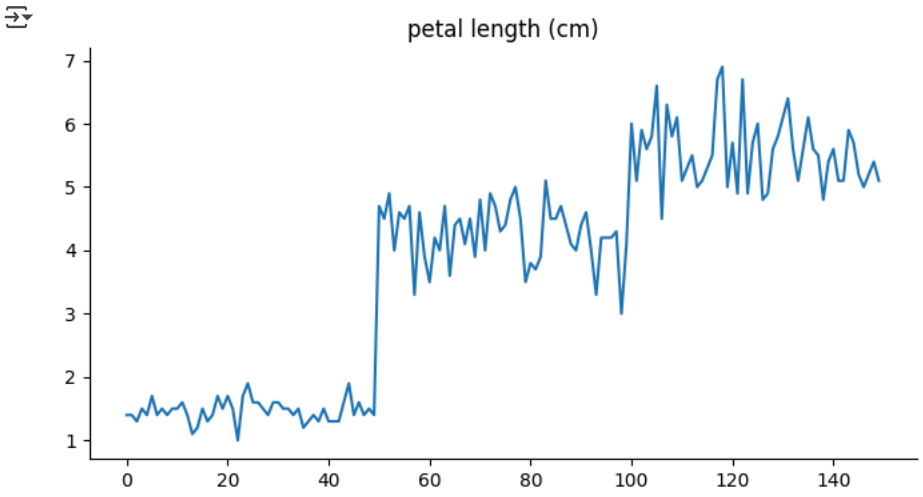
› petal width (cm)

Show code



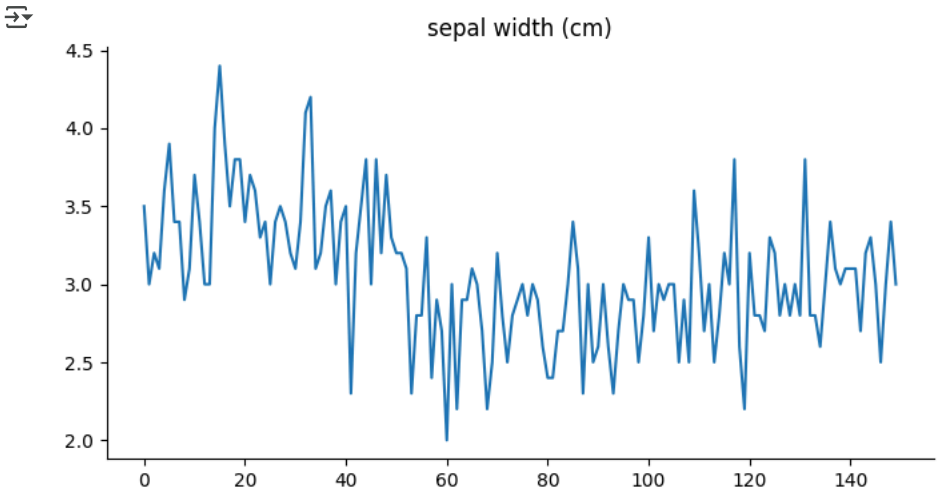
› petal length (cm)

Show code



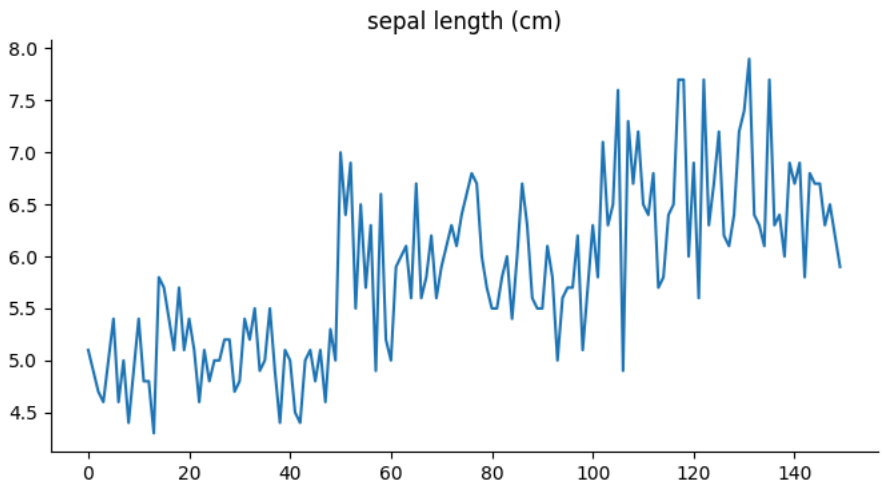
› sepal width (cm)

Show code



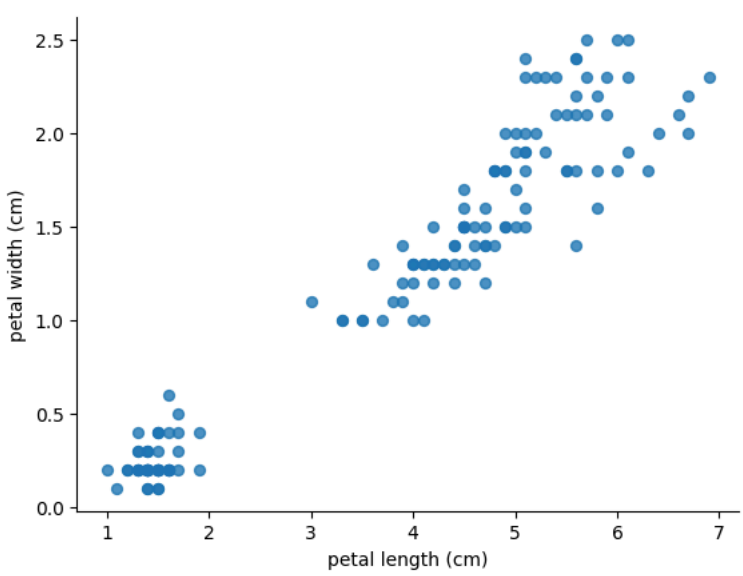
› sepal length (cm)

Show code



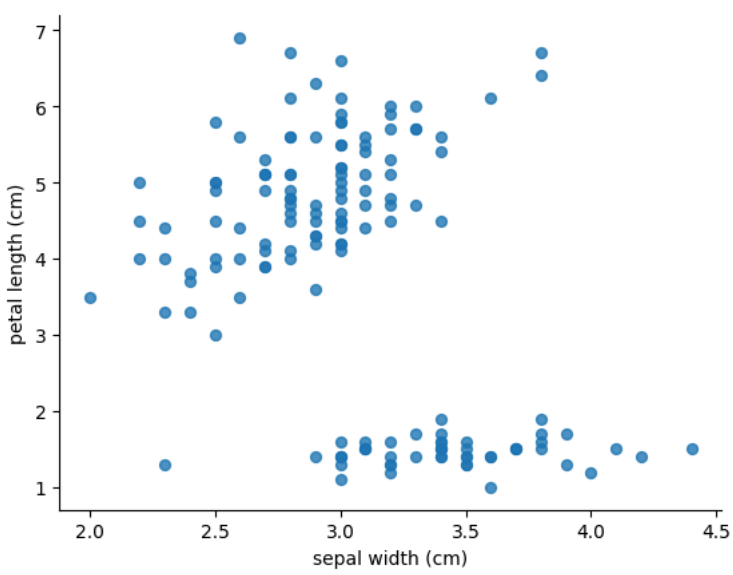
› petal length (cm) vs petal width (cm)

Show code



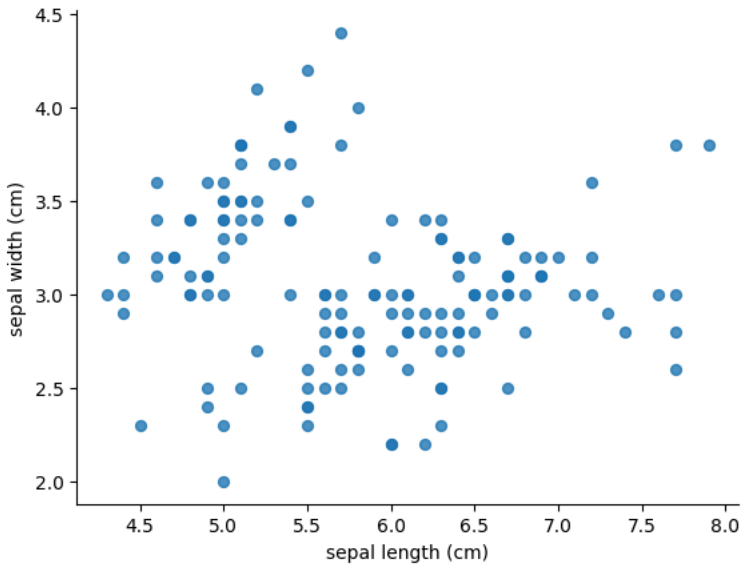
› sepal width (cm) vs petal length (cm)

Show code



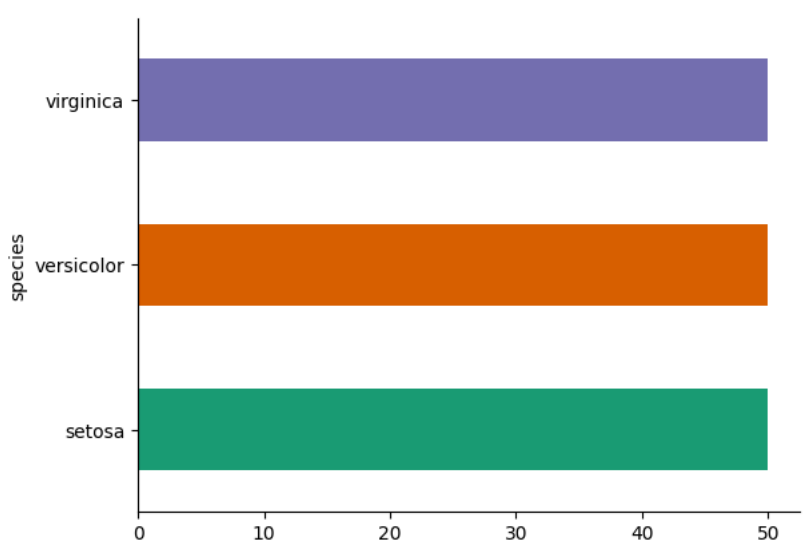
› sepal length (cm) vs sepal width (cm)

Show code



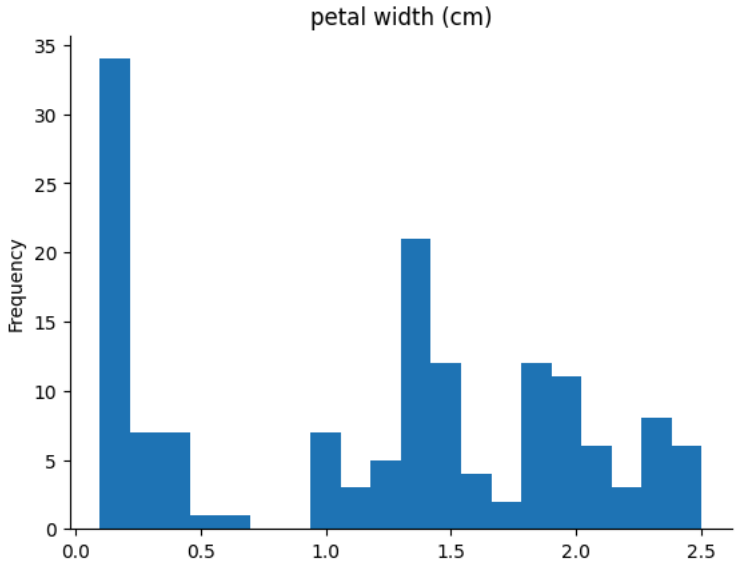
› species

Show code



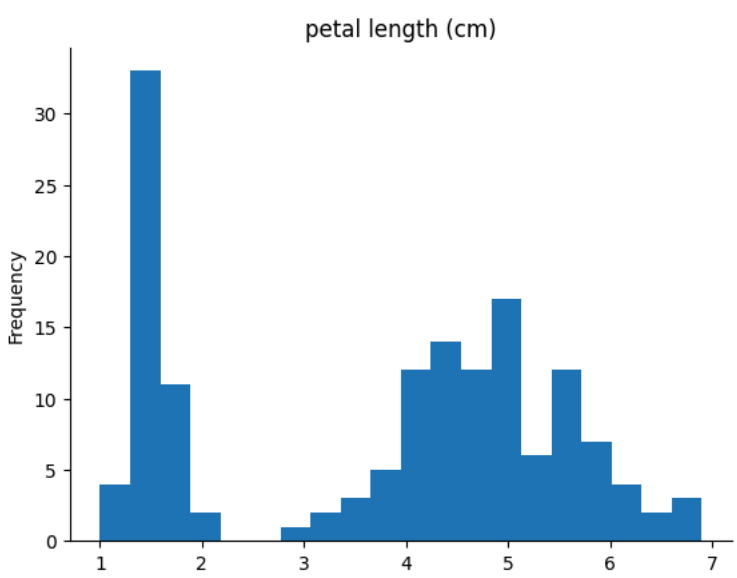
› petal width (cm)

Show code



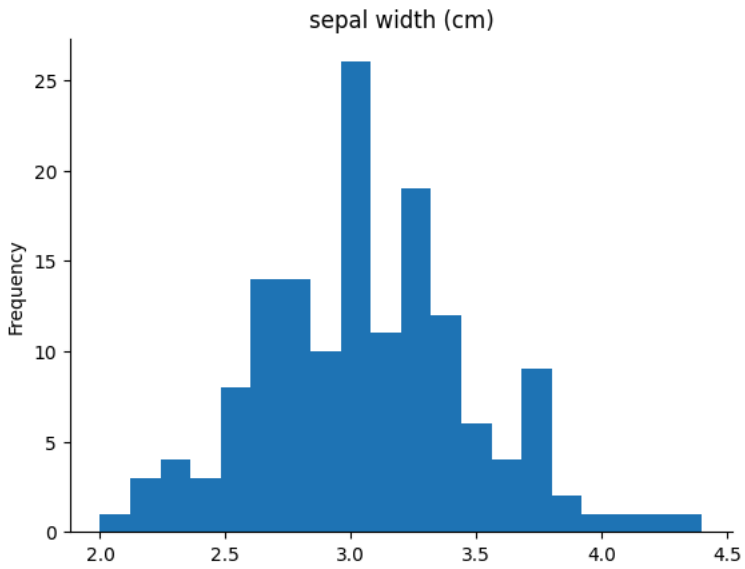
petal length (cm)

Show code

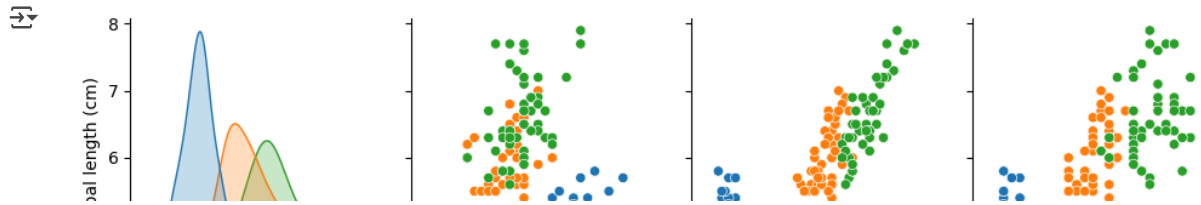


sepal width (cm)

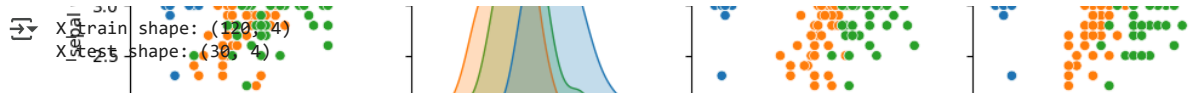
```
# @title sepal width (cm)

from matplotlib import pyplot as plt
df['sepal width (cm)'].plot(kind='hist', bins=20, title='sepal width (cm)')
plt.gca().spines[['top', 'right',]].set_visible(False)
```




```
import seaborn as sns
import matplotlib.pyplot as plt

# Pairplot for visualizing relationships
sns.pairplot(df, hue='species')
plt.show()

# Histograms
df.hist(figsize=(10, 6))
plt.tight_layout()
plt.show()
```



```
from sklearn.model_selection import train_test_split

X = df[iris.feature_names]
y = df['species']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
```



```
X_train shape: (120, 4)
X_test shape: (30, 4)
```

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
print(X_train[:2])
```

```
[[-1.47393679  1.20365799 -1.56253475 -1.31260282]
 [-0.13307079  2.99237573 -1.27600637 -1.04563275]]
```

```
from sklearn.neighbors import KNeighborsClassifier

model = KNeighborsClassifier(n_neighbors=3)
model.fit(X_train, y_train)
```