# Getting Started with Scikit-Learn

## Overview

In this lab, you’ll load a simple dataset into a pandas DataFrame and use some of the capabilities of the seaborn library to visualise the data in various ways.

## Source folders

Student folder: PythonML\Student\02-ScikitLearnGettingStarted

Solution folder: PythonML\Solutions\02-ScikitLearnGettingStarted

## Roadmap

1. Working with a pandas DataFrame
2. Plotting datapoints as scatterplot
3. Plotting all combinations of pairs of data in a scatterplot
4. (If time permits) Using other seaborn visualizations

## Exercise 1: Working with a pandas DataFrame

Open your favourite text editor and create a new Python script file in the *student* folder. Give the file a name such as loadAndVisualizeData.py.

*Note*: To try out your code at any stage, open a Command Prompt window and run python loadAndVisualizeData.py.

Now in the script file, add code to do the following:

* Import the seaborn module.
* Call the seaborn function load\_dataset() to load the 'iris' dataset. This function loads the data into a pandas DataFrame object – store it in a variable named iris.
* Call iris.head() and print the results. This displays the first 5 rows in the dataset, by default. Note you can pass a parameter into the head() function to display a different number of rows; e.g. iris.head(20) would print the first 20 rows.
* Call iris.info() and print the results. This displays useful information about the dataset, such as the name and type of each column, and the number of rows that have a value for each column.

You can also get all the values for a particular column. For example, the following command prints the value for the ‘species’ column for all the rows:

print(iris['species'])

In the above statement, iris['species'] returns a pandas Series object, which is like a one-dimensional vector. The Series object has various methods that describe the data in the series. For example, the following command prints the repeat-count for all the different species:

print(iris['species'].value\_counts())**Exercise 2: Plotting datapoints as scatterplot**

In this exercise you’ll plot some of the iris data in a seaborn scatterplot. The first step is to add the following statement at the top of your script file, to import the matplotlib.pyplot module:

import matplotlib.pyplot as plt

Then at the bottom of your script file, add the following code:

sns.scatterplot(x='petal\_length',y='petal\_width',data=iris)

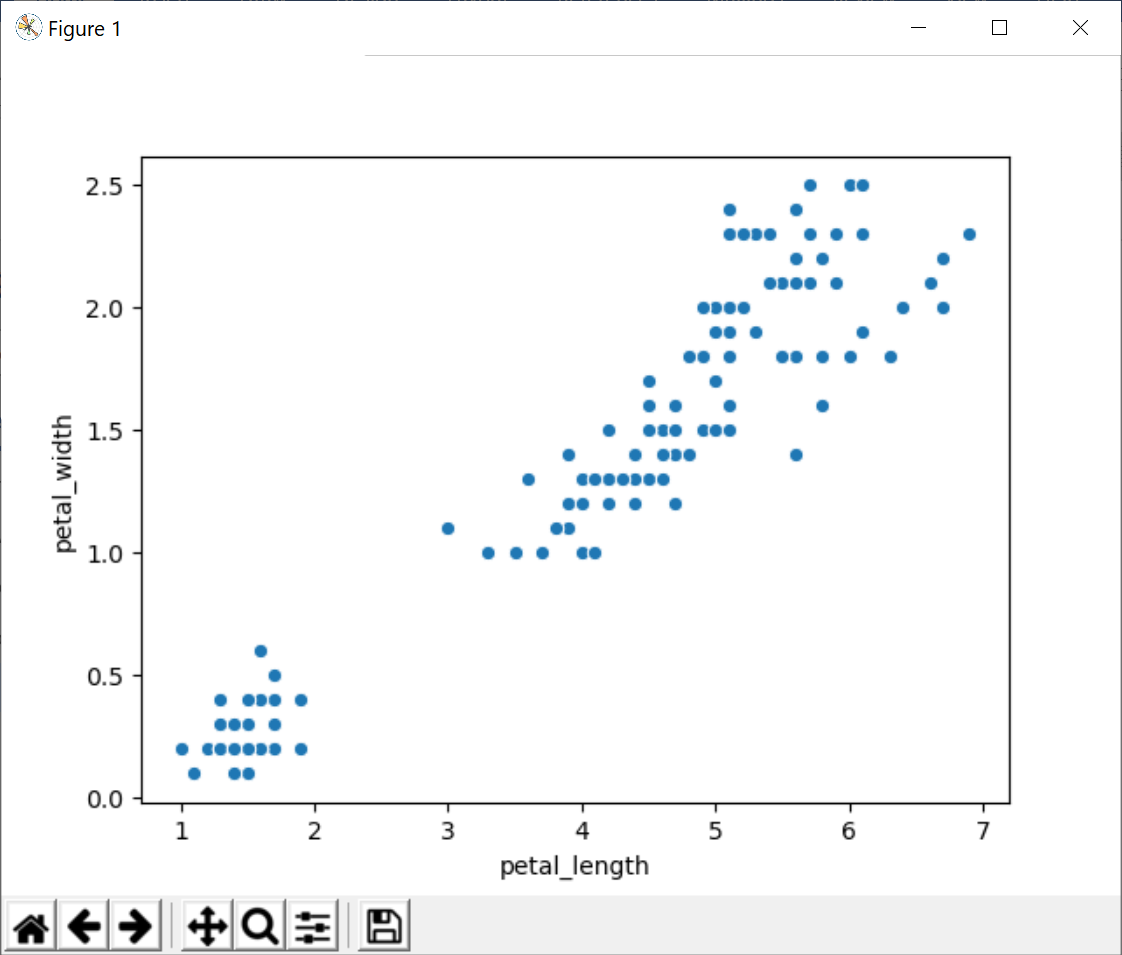
Note the following points in this code:

* We’ve assumed sns is the alias for the seaborn module.
* The scatterplot() function tells seaborn you’re interested in drawing a scatterplot graph. The x and y parameters specify what data values to display on each axis, and the data property is the pandas DataFrame that contains the data.

The scatterplot() function doesn’t actually show the graph. To show the graph, add the following statement at the end of your script file:

plt.show()

Save your script file and run it from the Command Prompt window. All being well, you should see the following graph displayed:



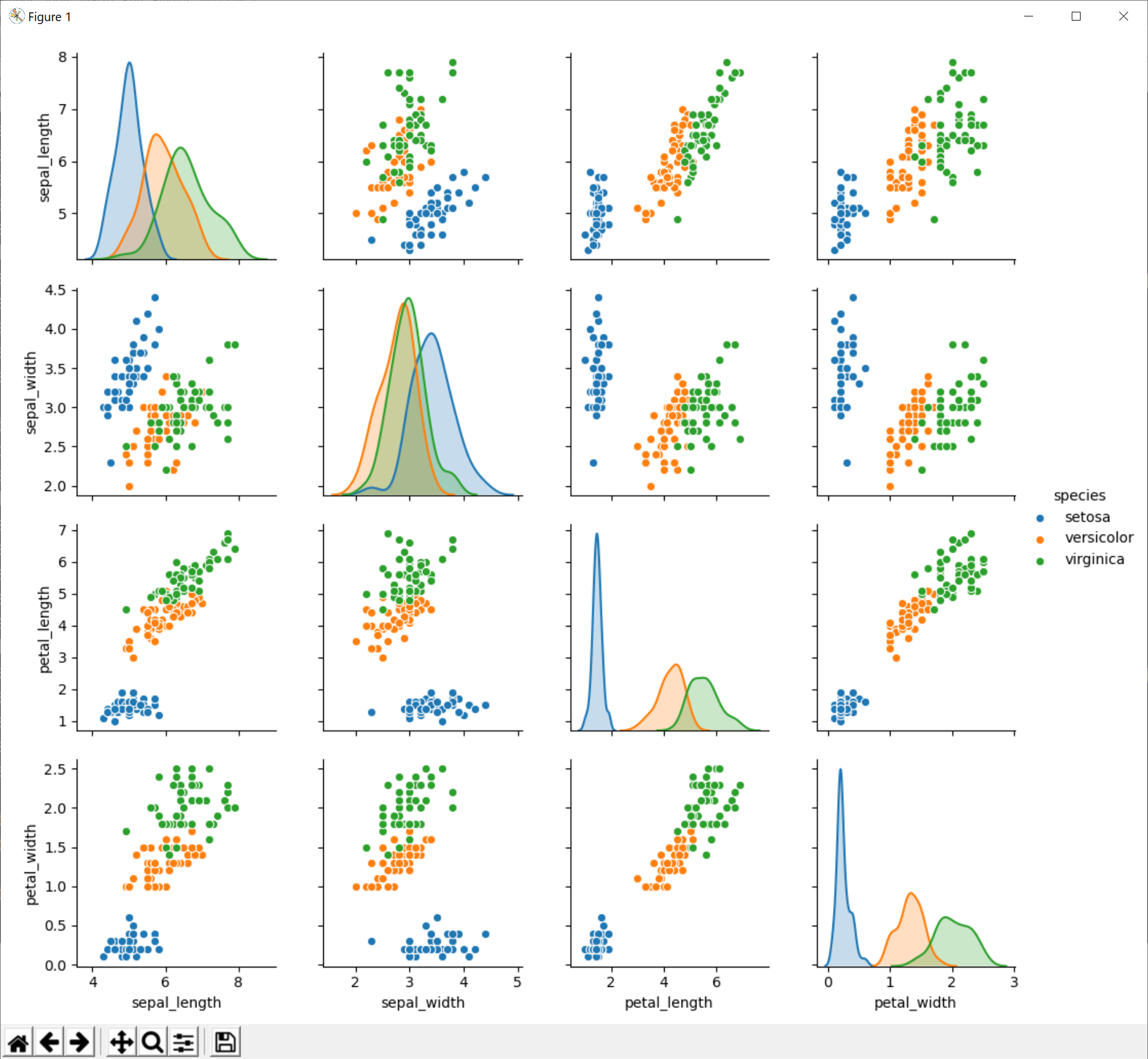
**Exercise 3: Plotting all combinations of pairs of data in a scatterplot**

In the previous exercise, you created a scatterplot graph with the petal\_length feature on the x axis and the petal\_width feature on the y axis.

Seaborn also has a pairplot() function that plots multiple graphs - it plots a separate graph for each combination of pairs of features. The following statement shows how to use the pairplot() function to plot all combinations of pairs of features in the iris dataset, in scatterplot graphs:

sns.pairplot(data=iris, hue='species', kind='scatter')

Modify your script file to call pairplot() as shown above. Save your file and re-run it. It should display the following graphs (note that the graphs on the diagonal are histograms showing the distribution of each variable):



**Exercise 4 (If time permits): Using other seaborn visualizations**

Seaborn has a rich set of visualizations available, so that you can plot datasets in various types of graphs. For example, try out the following statements (we have solutions for each of these suggestions, see the solution files named Ex4a.py, Ex4b.py, Ex4c.py):

1. sns.jointplot(x='petal\_length', y='petal\_width', data=iris)
2. sns.boxplot(x='species', y='petal\_length', data=iris)
3. sns.violinplot(x='species', y='petal\_length', data=iris)