Project 1: Basic plotting in Python 3 with matplotlib.pyplot

Before we start, please put your name and CUID in following format

: Firstname LASTNAME, #00000000 // e.g. Nianyi LI, #12345678

Your Answer:
Lakshmi Gayathri Rangaraju, #C81311556

General Rules of the Project Submission

Python 3 and Matplotlib will be used throughout the semseter, so it is important to be familiar with them. It is strongly suggested to go through Stanford CS231n and CS228 for more detailed Python and numpy tutorials if you haven't had used Python before.

In some cells and files you will see code blocks that look like this:

```
#######################################################################
########
#                     TODO: Write the equation for a line
#
#######################################################################
########
pass
#######################################################################
########
#                               END OF YOUR CODE
#
#######################################################################
########
```

You should replace the `pass` statement with your own code and leave the blocks intact, like this:

```
#######################################################################
########
#                     TODO: Write the equation for a line
#
#######################################################################
########
y = m * x + b
#######################################################################
########
#                               END OF YOUR CODE
#
```

```
################################################################################
```

When completing the notebook, please adhere to the following rules:

- Do not write or modify any code outside of code blocks
- Follow the instruction of the project description carefully
- Run all cells before submitting. **You will only get credit for code that has been run!**.

The last point is extremely important and bears repeating:

We will not re-run your notebook -- you will only get credit for cells that have been run

File name

Your Python program should be named **yourlastname_yourfirstname_P1.ipynb**, then zip it and upload to Canvas

Python 3

If you're unfamiliar with Python 3, here are some of the most common changes from Python 2 to look out for.

Print is a function

```
print("Hello!")
```

Without parentheses, printing will not work.

Floating point division by default

```
5 / 2
```

To do integer division, we use two backslashes:

```
5 // 2
```

No xrange

The xrange from Python 2 is now merged into "range" for Python 3 and there is no xrange in Python 3. In Python 3, range(3) does not create a list of 3 elements as it would in Python 2, rather just creates a more memory efficient iterator.

Hence, xrange in Python 3: Does not exist range in Python 3: Has very similar behavior to Python 2's xrange

```
for i in range(3):
    print(i)

range(3)

# If need be, can use the following to get a similar behavior to
Python 2's range:
print(list(range(3)))
```

Project Description

Data File

The first line of `IrisData.txt` contains two integers. The first represents how many lines of data are in the file. The second how many features are associated with each line. Each line after that contains four floating point values representing the sepal length, sepal width, petal length, and petal width of a type of iris flower, followed by the name of the iris type: either setosa, versicolor, or virginica. Values are tab separated.

Python Program

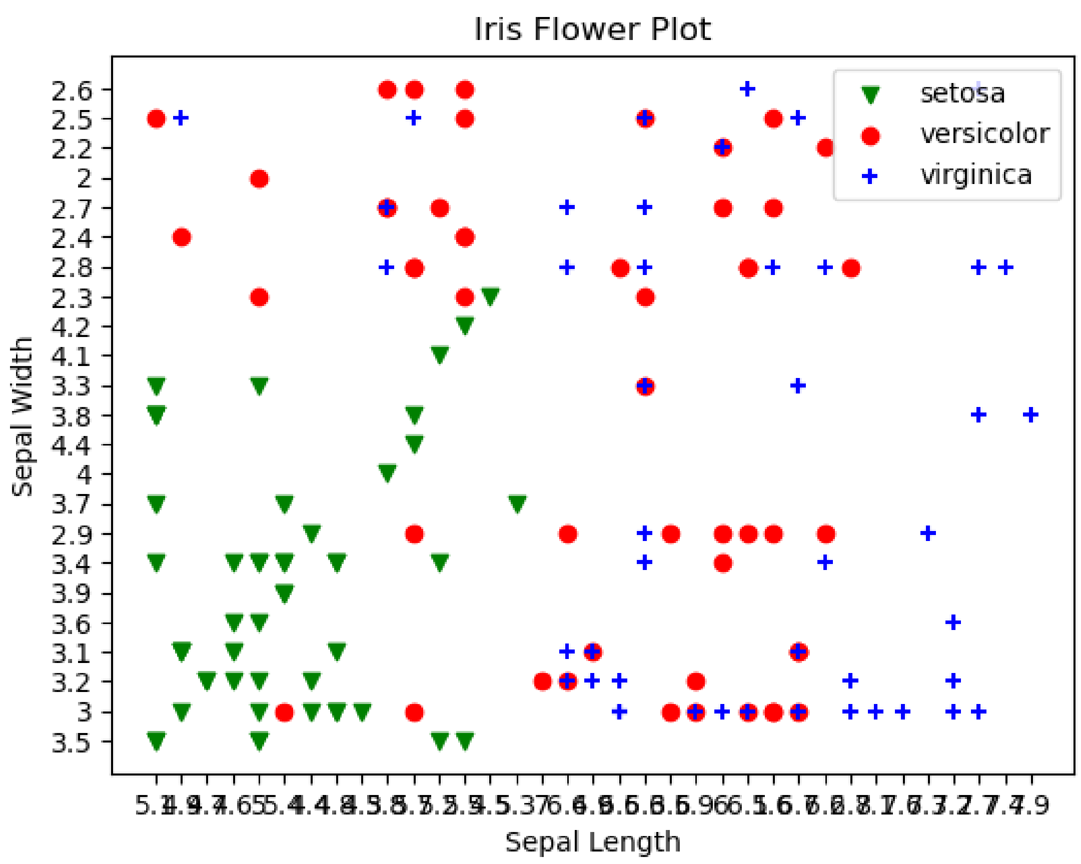
Write a Python program that will print out a **two-dimensional plot** of any two features for all three varieties of iris flowers using the features of `matplotlib.pyplot`. Your program should begin by asking for the name of the input file. For example:

```
Enter the name of your data file: Irisdata.txt

######################################################################
########
#         TODO: Write the code for reading data from file
#
######################################################################
########
import pandas as pd
import sys
import os

sys.path.append(os.path.abspath("/notebooks/ML-Imp&Eval/Project-1/"))

directory = "/home/seed/ML-Imp&Eval/Project-1/"
fileName = directory + input("Enter the name of your data file:")

dataset = []
with open(fileName, 'r') as f:
    lines = f.readlines()

# Reading the lines into dataset matrix.
```

```
for line in lines:
    line = line.strip()
    line = line.split("\t")
    if len(line)>2: # Excluding the first line
        dataset.append(line)
########################################################################
########
#                              END OF YOUR CODE
#
########################################################################
########

Enter the name of your data file:IrisData.txt
```

Then the program should prompt the user for two features to plot. For example, it could say:

```
You can do a plot of any two features of the Iris Data set The feature
codes are:
0 = sepal length
1 = sepal width
2 = petal length
3 = petal width

Enter feature code For the horizontal axis: 0
Enter feature code For the vertical axis: 1

########################################################################
########
#   TODO: Write the function for plotting using two features of the
data     #
########################################################################
########
import matplotlib.pyplot as plt


def plot_feature_map():
    print("You can do a plot of any two features of the Iris Data set
The feature codes are: \n 0 = sepal length \n 1 = sepal width \n 2 =
petal length \n 3 = petal width")
    feature_on_horizontal_axis = int(input("Enter feature code For the
horizontal axis:"))
    feature_on_vertical_axis = int(input("Enter feature code For the
vertical axis:"))

    for row in range(len(dataset)):
        if dataset[row][4] == 'setosa':
            x1 = dataset[row][feature_on_horizontal_axis]
            y1 = dataset[row][feature_on_vertical_axis]
            plt.scatter(x1, y1, c='green', label='setosa', marker='v')
        elif dataset[row][4] == 'versicolor':
            x2 = dataset[row][feature_on_horizontal_axis]
            y2 = dataset[row][feature_on_vertical_axis]
            plt.scatter(x2, y2, c='red', label='versicolor',
marker='o')
        elif dataset[row][4] == 'virginica':
            x3 = dataset[row][feature_on_horizontal_axis]
            y3 = dataset[row][feature_on_vertical_axis]
            plt.scatter(x3, y3, c='blue', label='virginica',
marker='+')


    labels = ['Sepal Length', 'Sepal Width', 'Petal Length', 'Petal
Width']
    plt.xlabel(labels[feature_on_horizontal_axis])
    plt.ylabel(labels[feature_on_vertical_axis])
    handles, labels = plt.gca().get_legend_handles_labels()
    labels_handles = dict(zip(labels, handles))
    plt.legend(labels_handles.values(), labels_handles.keys())
    plt.title("Iris Flower Plot")
    plt.show()



plot_feature_map()
######################################################################
########
#                               END OF YOUR CODE
#
######################################################################
########

You can do a plot of any two features of the Iris Data set The feature
codes are:
 0 = sepal length
 1 = sepal width
 2 = petal length
 3 = petal width
Enter feature code For the horizontal axis:0
Enter feature code For the vertical axis:1
```

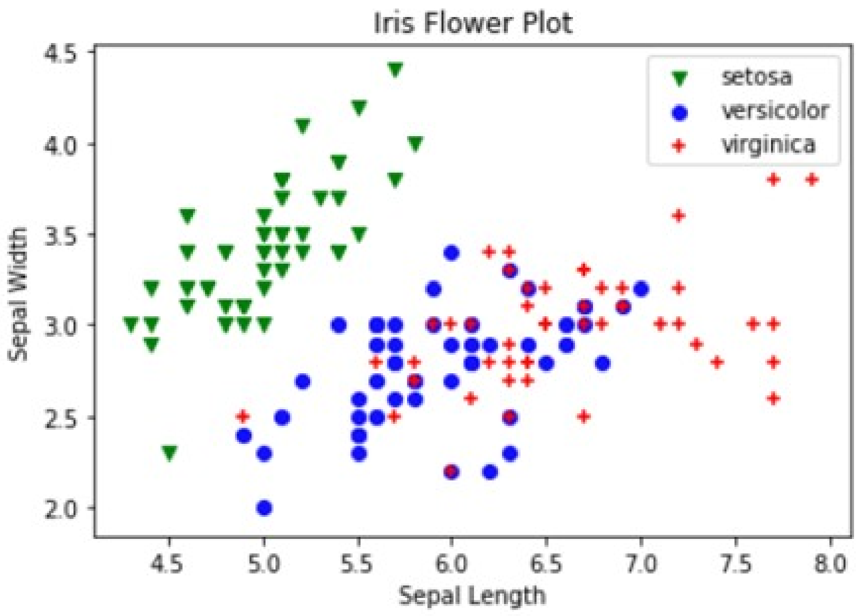


Once the program does a plot it should ask the user if they want to another plot. For example:

```
Would you like to do another plot? (y/n) y
```

If the answer is `y`, then the user should be prompted for new features to plot on the horizontal and vertical axes as before and a new plot generated. `n` should end the program.

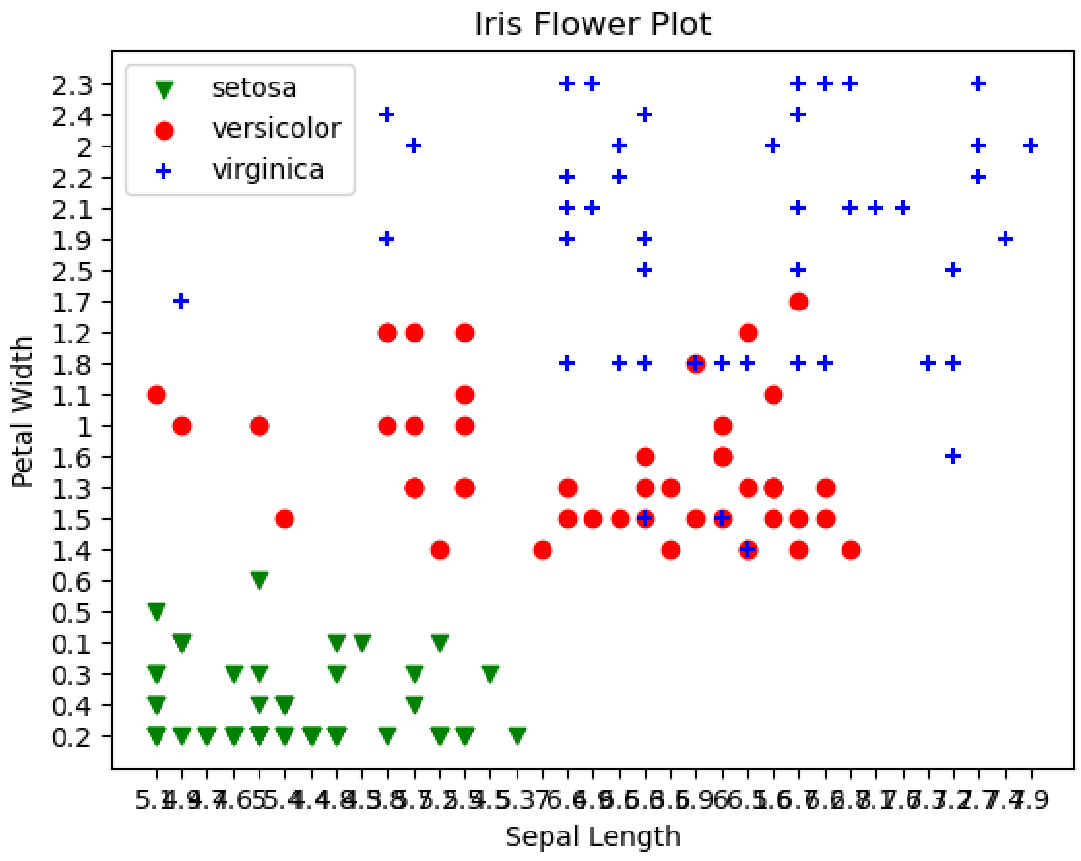
Plots should be labelled with a *title, axes should be labelled*, there should be a *legend* and all three varieties should be **color coded** and have different symbols. An example is shown below.



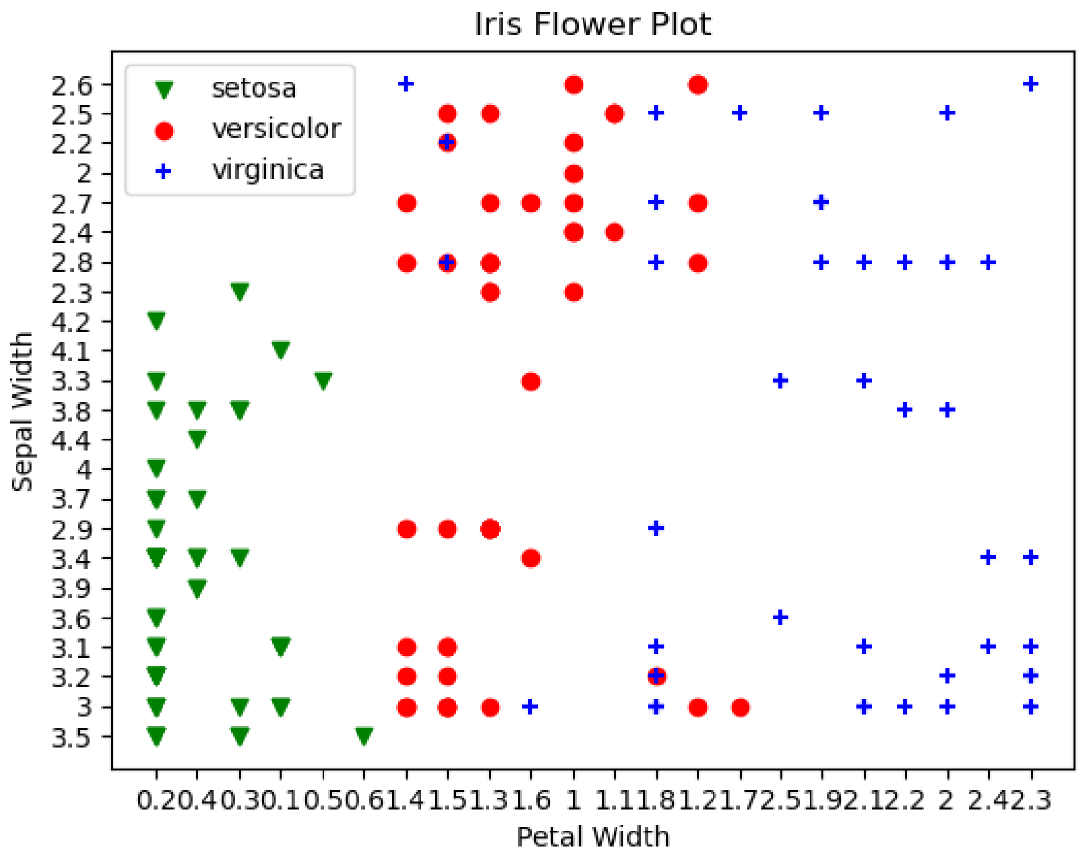
You need to try **at least 3 plots with different x and y labels combination**.

```
########################################################################
#          TODO: Use the loop function for plotting figures
#
########################################################################
answer = input("Would you like to do another plot? (y/n)")
while answer == 'y':
    plot_feature_map()
    answer = input("Would you like to do another plot? (y/n)")
########################################################################
#                               END OF YOUR CODE
#
########################################################################
```

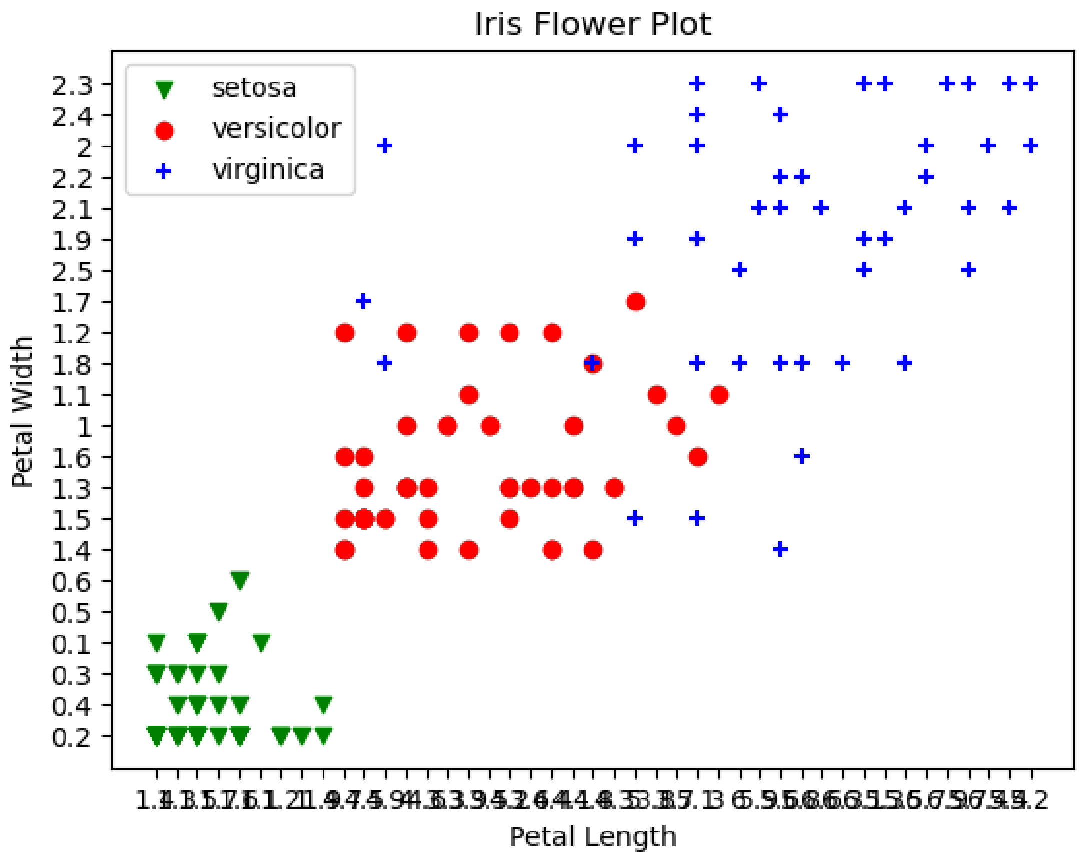
```
Would you like to do another plot? (y/n)y
You can do a plot of any two features of the Iris Data set The feature
codes are:
 0 = sepal length
 1 = sepal width
 2 = petal length
 3 = petal width
Enter feature code For the horizontal axis:0
Enter feature code For the vertical axis:3
```



```
Would you like to do another plot? (y/n)y
You can do a plot of any two features of the Iris Data set The feature
codes are:
 0 = sepal length
 1 = sepal width
 2 = petal length
 3 = petal width
Enter feature code For the horizontal axis:3
Enter feature code For the vertical axis:1
```



```
Would you like to do another plot? (y/n)y
You can do a plot of any two features of the Iris Data set The feature
codes are:
 0 = sepal length
 1 = sepal width
 2 = petal length
 3 = petal width
Enter feature code For the horizontal axis:2
Enter feature code For the vertical axis:3
```



```
Would you like to do another plot? (y/n)n
```

Conclusion

Briefly discuss **which feature combination is the best** for classify the iris data.

Your Answer:

By seeing the above plots, I believe that by using "Petal Width" and "Petal length" we can classify the Iris dataset with high accuracy. This is because, from the last plot where I plotted Petal Length vs Petal Width, we can see that the features are separatable using a decision boundary in higher dimensional space. Hence considering only these two features will improve the model performance in classifying the unseen data.