CS 021 –Computer Programming I: Python
Use Cases – Data Visualization

Overview

The following procedure outlines the process through which we can visualize data with Python. We will be using a package called Seaborn – a widely used, open source statistical data visualization library. Data visualization is used to communicate information clearly and efficiently with statistical graphics, plots, information graphics and other tools. Numerical data may be encoded using dots, lines, or bars, to visually communicate a quantitative message. Effective visualization helps users analyze and reason about data and evidence. It makes complex data more accessible, understandable and usable. In this lesson, you will learn how to turn data sets into beautiful intuitive graphs and charts using one of Pythons most widely used visualization libraries.

Procedure

Note: this tutorial is for Windows, if you are on a Mac, simply add the 'sudo' markup before each command

1. To start, we need to install Seaborn via **pip**, Python's package manager. Open up a command prompt window and navigate to your desktop. Once on the desktop, type the command below into command prompt, press enter and wait for the installation to complete.
    a. Commands
        i. '**pip install seaborn'**

2. Create file called **'visual.py'**
3. Once the file is created, we need to import a few external libraries to get up and running. To do this, add code below to your **visual.py** file

```
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.colors import ListedColormap
```

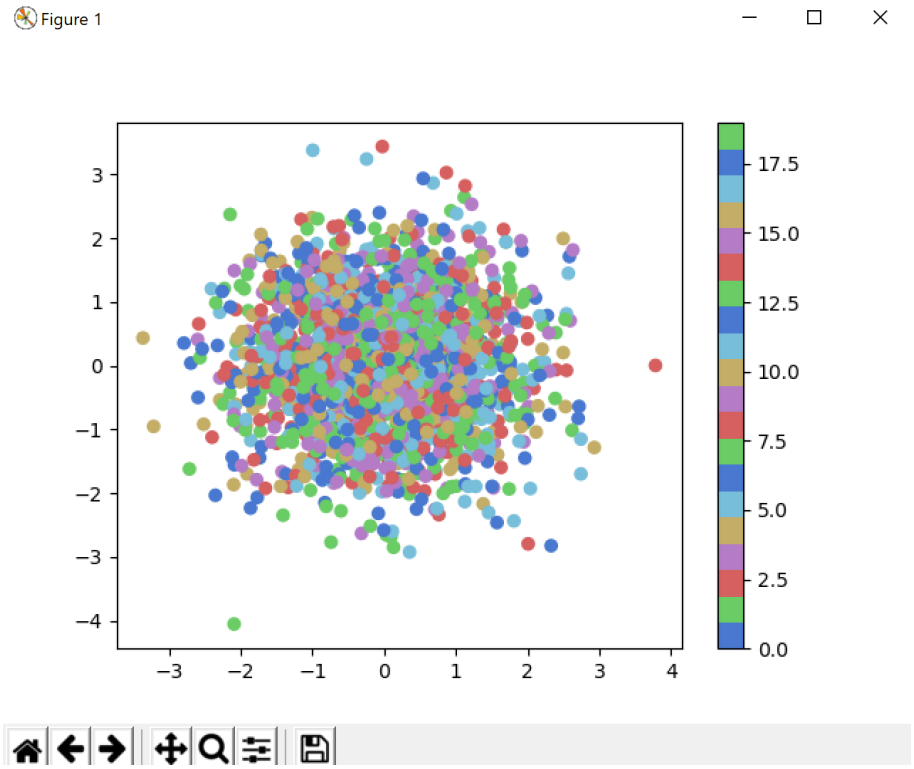
4. These import statements will allow us to easily create custom plots and generate data sets. After this, let's initialize our color scheme. Add the following code to your **visual.py** file.

```
N = 2000
current_palette = sns.color_palette("muted", n_colors=20)
cmap = ListedColormap(sns.color_palette(current_palette).as_hex())
```

5. This snippet creates a new seaborn color scheme and maps it to a ListedColormap object that we will use later to create the actual plot. Now, all we need to do is create a new plot and throw some data into it. To do this, add the following code into your **visual.py** file.

```
data1 = np.random.randn(N)
data2 = np.random.randn(N)
colors = np.random.randint(0,20,N)
plt.scatter(data1, data2, c=colors, cmap=cmap)
plt.colorbar()
plt.show()
```

6. Once this is complete, you will have a working data visualization model. Save code and run with Fn + F5. The output is shown below



7. The output above is a set of random data points, which isn't meaningful to us; however, to create a tailored data is quite simple. To accomplish this, all you need to do is replace the **data1** and **data2** variables with the datasets you want to see. Once that is complete, you will be able to configure your outputs to whatever you need.