Hyperiondev

TASK

Data Visualisation IV

Visit our website

Introduction

WELCOME TO THE DATA VISUALISATION IV TASK!

In this task, you will use another library called Matplotlib to visualise data.

Get in touch

Connect for support

Remember that with our courses, you're not alone! You can contact an expert code reviewer to get support on any aspect of your course.

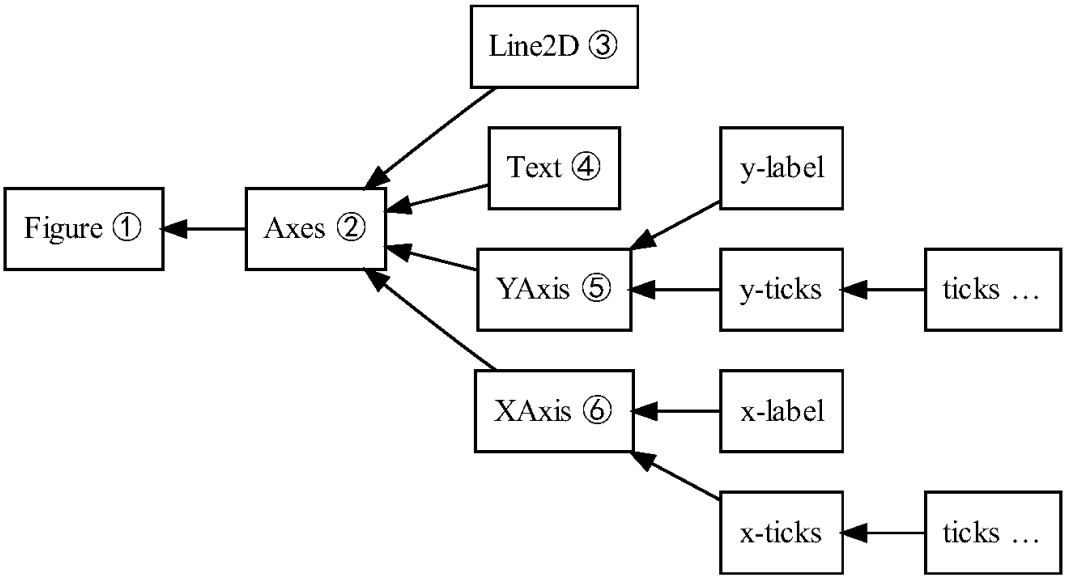
The best way to get help is to login to Discord at **https://discord.com/invite/hyperdev** where our specialist team is ready to support you.

Our team is happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

INTRODUCTION

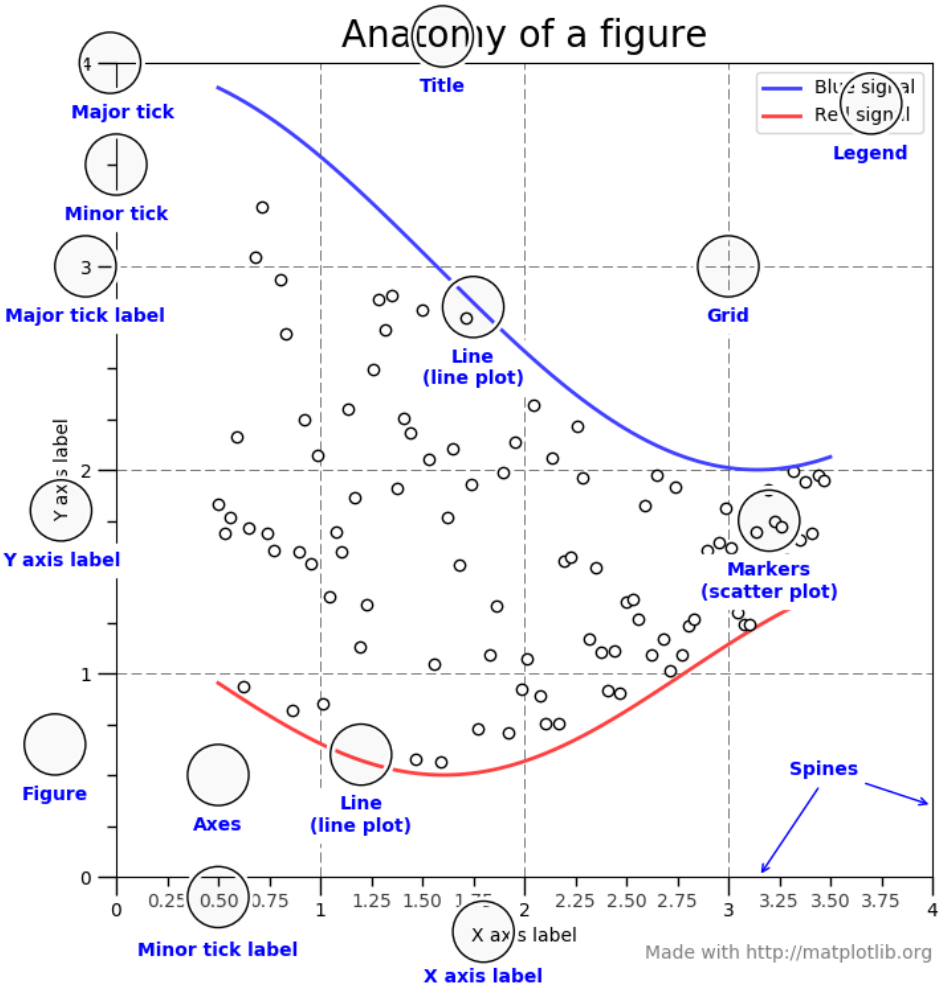
There are many packages in Python that allow one to perform powerful data analysis. **Matplotlib** is "... a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits" (Hunter, 2007). With Matplotlib, we are able to draw many different graphs in Python (documentation **here**).

Matplotlib is organised into a hierarchy. At the top of the hierarchy, there is a library called pyplot (`import matplotlib.pyplot as plt`). We create a pyplot to create a figure. The figure keeps track of all the child Axes, artists (titles, figure legends, etc.), and the canvas.



(Hunter, 2007)

Artists are defined as follows: "Basically everything you can see on the figure is an artist (even the Figure, Axes, and Axis objects). This includes Text objects, Line2D objects, collection objects, Patch objects ... (you get the idea). When the figure is rendered, all of the artists are drawn to the canvas. Most Artists are tied to an Axes; such an Artist cannot be shared by multiple Axes or moved from one to another" (Hunter, 2007).



(Hunter, 2007)

All of the plotting functions expect `np.array` or `np.ma.masked_array` as input so it is best to convert any pandas (or similar array-like data structures) to arrays before using them with Matplotlib.

INSTALL MATPLOTLIB

Open up your terminal and input the following, and then hit enter:

```
python -m pip install -U pip
python -m pip install -U matplotlib
```

For further guidelines for installing Matplotlib, see the official documentation **here**.

CREATING A FIGURE WITH MATPLOTLIB

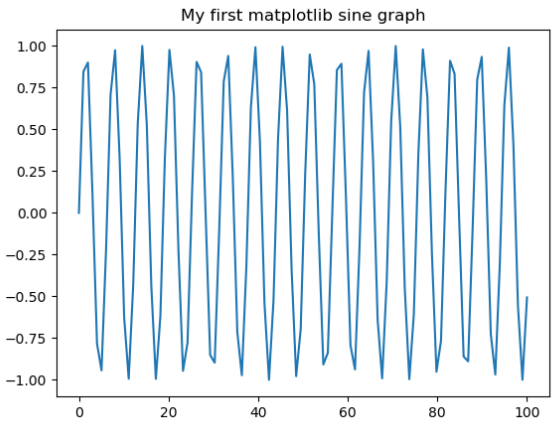
To use packages or libraries in Python, you will need to import them at the top of your python file:

```
import matplotlib.pyplot as plt
import numpy as np
```

We can create a quick dataset using numpy and visualise the data using Matplotlib:

```
import matplotlib.pyplot as plt
import numpy as np

#Prepare the data
x = np.linspace(0, 100, 100) #x axis
y = np.sin(x) # y values
#Plot the data
plt.plot(x, y, label="sine")
#Create a title
plt.title("My first matplotlib sine graph")
#Show the plot
plt.show()
```

And if you run the program, you will get something like this:



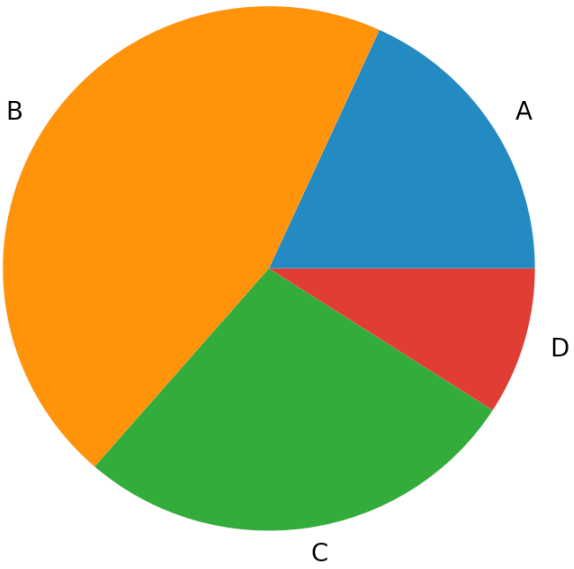
Try to play around with it to get a better understanding. You can also see the documentation to matplotlib **here**.

IMPORTING DATA

You can also load different data files and create visualisations using them. For example, if you have your own .csv file, create a simple pie chart as below:

```
import matplotlib.pyplot as plt
import numpy as np
import csv

#open csv file
outfile = open("piechartdata.csv","r")

#let the program know it is a csv
file = csv.reader(outfile)

#skip the header ('letter','value')
next(file, None)

#create arrays to store the values
category = []
value = []

#iterates through the file
for row in file:
     category.append(row[0]) #appends 'letter' into category
     value.append(row[1]) #appends 'value' into value

#appends the array values into pie chart
plt.pie(value, labels=category)

#show
plt.show()
```

You will create something like this:



Remember that your dataset should also be in the same folder as your Python file. Alternatively, if it is not in the same folder, you can specify the path:

```
#specific csv file path
outfile = open("/users/admin/...../piechartdata.csv","r")
```

Extra resource

*For more information about working with Matplotlib, please consult the fourth chapter ("**Visualization with Matplotlib**") in the book entitled, "**Python Data Science Handbook**" by Jake VanderPlas.*

WORKING WITH SEABORN

Seaborn is a data visualisation library that has been built on top of Matplotlib. While Matplotlib provides basic graphs, such as line and pie charts, Seaborn can provide a bit more in terms of graphing. In addition, it integrates quite well with Pandas.

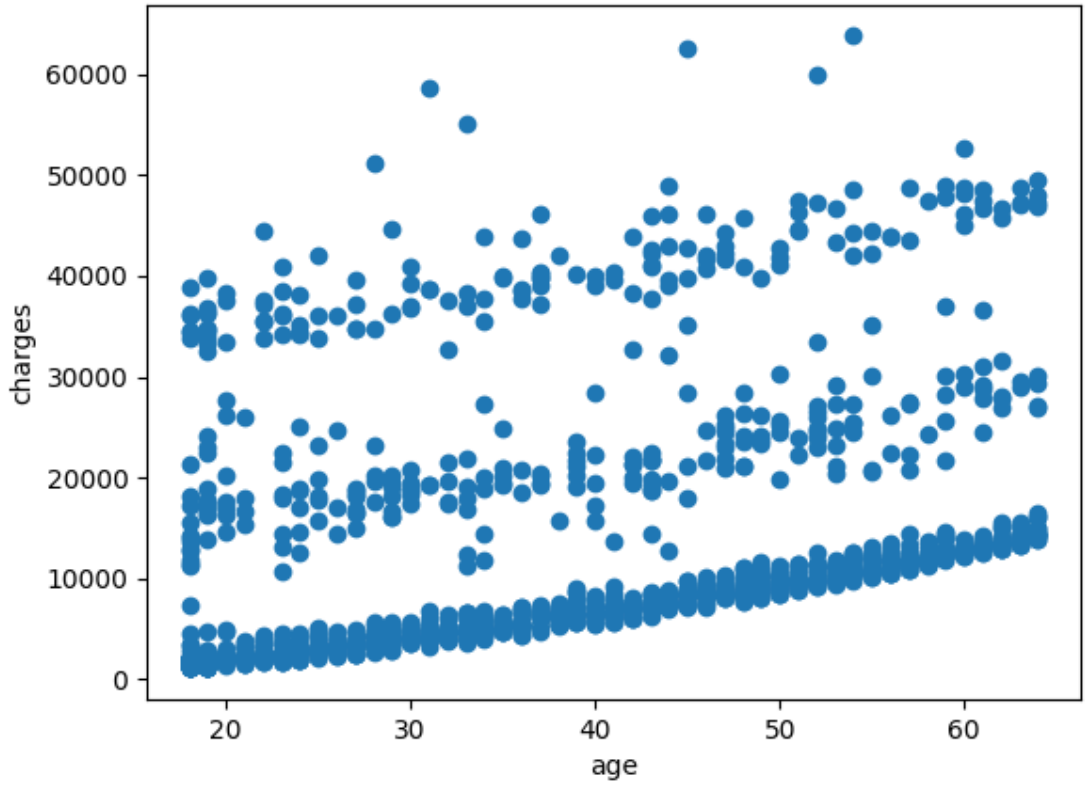
Let's say that we are reading insurance data that contains a column for age and a column for the insurance charge. We would like to understand the relationship existing between these two columns:

```
ins_df = pd.read_csv('insurance.csv')
```

In Matplotlib, the scatter() method would be most appropriate. This can be done using:

```
plt.figure()
plt.scatter(ins_df['age'], ins_df['charges'])
plt.xlabel("age")
plt.ylabel('charges')
plt.show()
plt.close()
```

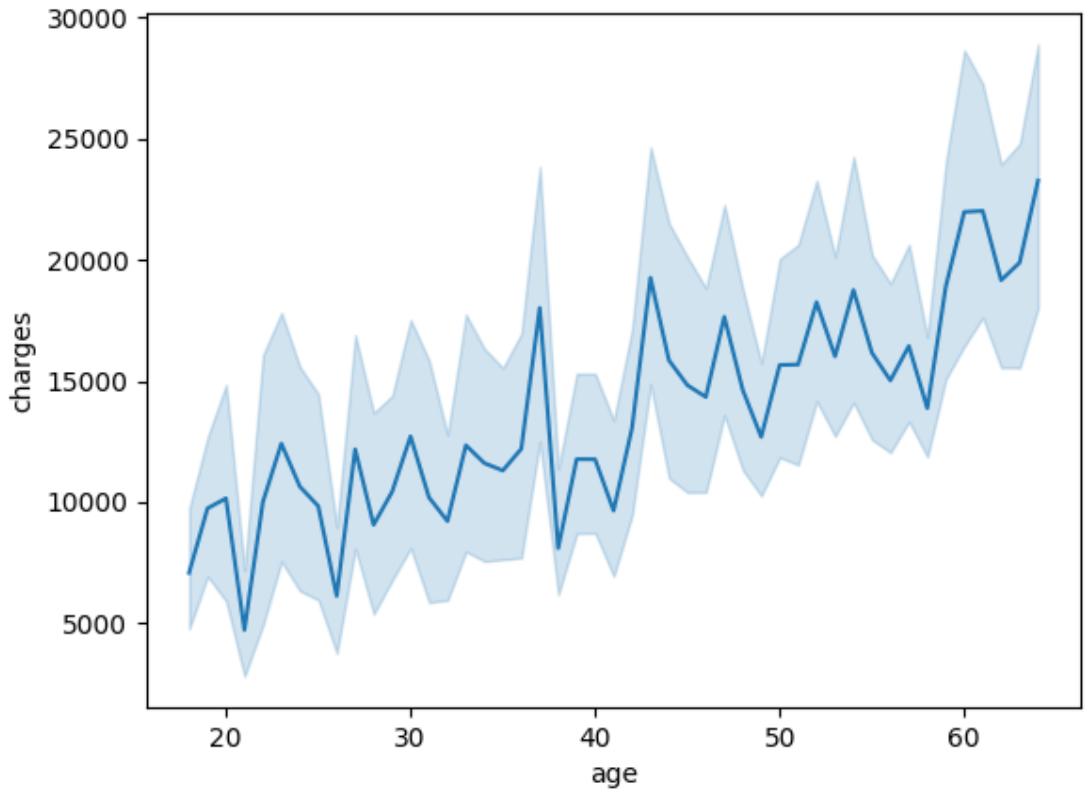
And would look something like this:



You can get a general sense of this data, but it may be difficult to understand the overall trend immediately. In Seaborn, there is a lineplot() method that automatically plots averages and standard deviations for ease of reading. To do this, you can simply:

```
plt.figure()
sns.lineplot(x='age', y='charges', data=ins_df)
plt.savefig('sns.png')
plt.close()
```

And will end up with something that looks like this:



This makes it a lot easier to see the overall trend existing in the data: the higher your age, the larger your insurance charges.

Some commonly-used Seaborn plots include:

- histplot()
- barplot()
- boxplot()

Among many others! You will get accustomed to many of these methods in the course of this bootcamp.

Compulsory Task 1

Follow these steps:

- Please read through and interact with the jupyter notebook attached to this task.
- Generate the following graphs from the dataset in this notebook. Answer the accompanying questions in markdown cells in the notebook:
    - A box plot for the revs per mile for the Audi, Hyundai, Suzuki and Toyota car manufacturers. Which of these manufacturers has the car with the highest revs per mile?
    - A histogram of MPG in the city. On the same axis, show a histogram of MPG on the highway. Is it generally more fuel efficient to drive in the city or the highway?
    - A lineplot showing the relationship between the Wheelbase and turning circle. What is this relationship? What happens when the wheelbase gets larger?
    - A bar plot showing the mean horsepower for each car Type (Small, Midsize, etc.). Does a larger car mean more horsepower?
    - A pie chart for the respective engine sizes. Which engine size seems to be the most popular?

If you are having any difficulties, please feel free to contact our specialist team **on Discord** for support.

Completed the task(s)?

Ask an expert to review your work!

Review work

Rate us

Share your thoughts

HyperionDev strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think that the content of this task, or this course as a whole, can be improved? Do you think we've done a good job?

Click here to share your thoughts anonymously.

References:

Hunter, J.D., "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.