Introduction to Python

Week 3 - Modules, matplotlib

TS Turing Students ROTTERDAM

Modules

Turing Students ROTTERDAM

When you have a large program, you may want to split its components into separate parts, and that's called a module.

You import a module by writing `import module_name` before you use it. After that, you can use variables and functions from the module by `module_name.variable` or `module_name.function()`

You can also give a shorter name for the module.

```
import math

print(math.pi)

import math as m

# Shorter!
print(m.pi)
```

Write your own module

Turing Students ROTTERDAM

```python
# Fibonacci numbers module
# return Fibonacci series up to n
def fib(n):
    result = []
    a, b = 0, 1
    while a < n:
        result.append(a)
        a, b = b, a+b
    return result
```

fibo.py

```python
import fibo
print(fibo.fib(1000))
```

main.py

Modules

Turing Students ROTTERDAM

- Python files run top-down
- When you import a file, it is executed (to load into memory)
- → Your module is run when you import it

- Solution?
  - Put everything in a function
  - `__main__` [1] [2]

```python
# ...after the fib function
# this will be run when imported
f = input("Input n: ")
print(fib(f))

# this will only be run when run
directly
if __name__ == "__main__":
    f = input("Input n: ")
    print(fib(f))
```

fibo.py

Modules

Turing Students ROTTERDAM

- Python files run top-down
- When you import a file, it is executed (to load into memory)
- → Your module is run when you import it

- Solution?
  - Put everything in a function
  - `__main__` [1] [2]

```python
# Even better, put it in a function
def main():
    f = input("Input n: ")
    print(fib(f))

# this will only be run when run
directly
if __name__ == "__main__":
    main()
```

fibo.py

Matplotlib

Turing Students ROTTERDAM

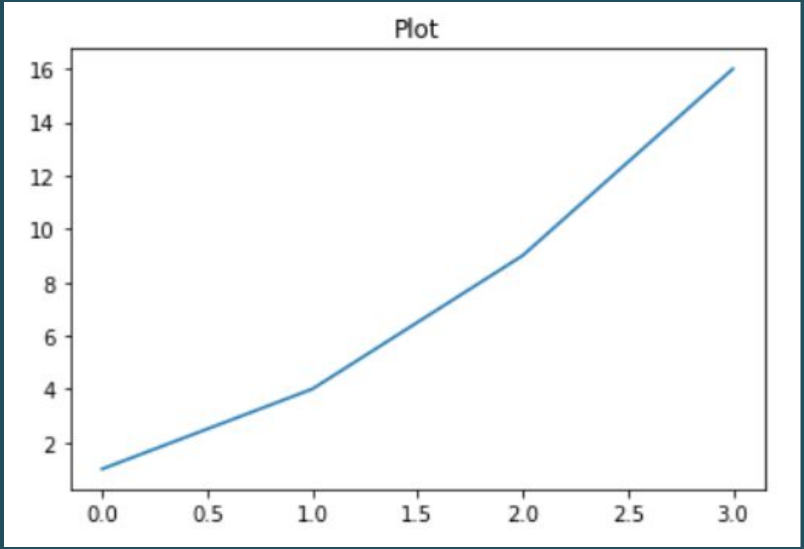
- Matplotlib is a python visualization package
- You can plot...
  - Line plot
  - Scatter plot
  - Contour plot
  - Surface plot
  - Bar chart
  - Histogram
  - Box plot
- If you want to have a look at visualization examples, http://matplotlib.org/gallery.html
- Install: `conda install -c anaconda matplotlib`

Line plot

TS Turing Students ROTTERDAM

- Connecting data points w/ a straight line
- Useful in understanding the trend over time

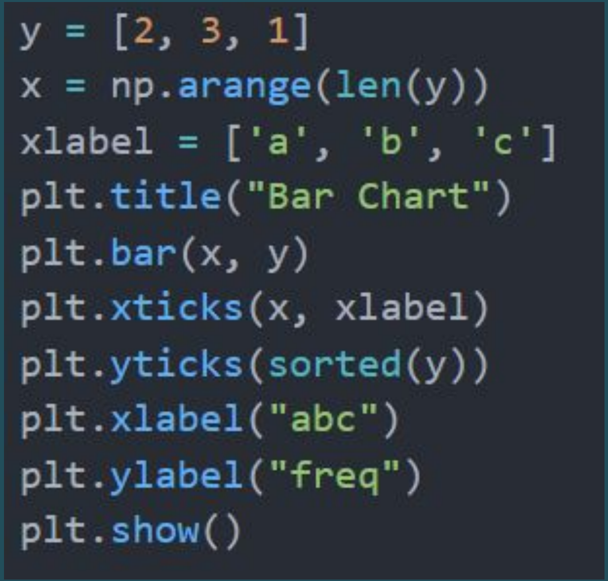
```
import matplotlib.pyplot as plt

plt.title("Plot")
plt.plot([1, 4, 9, 16])
plt.show()
```



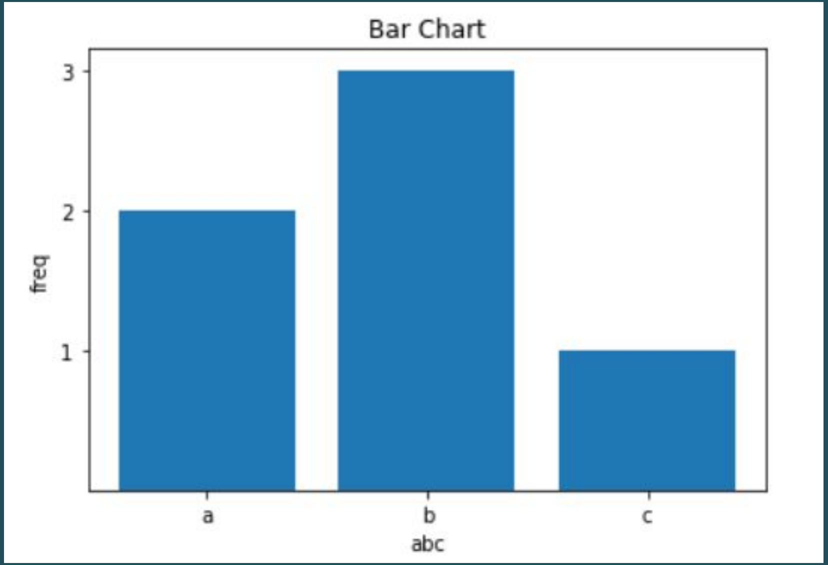
Bar chart

TS Turing Students ROTTERDAM

- Shows the distribution of data over several groups
- Helps in comparing multiple numeric values

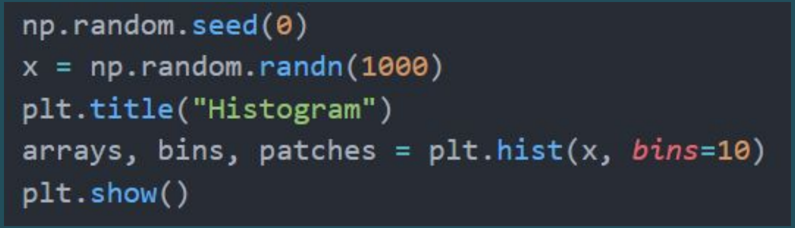


```python
y = [2, 3, 1]
x = np.arange(len(y))
xlabel = ['a', 'b', 'c']
plt.title("Bar Chart")
plt.bar(x, y)
plt.xticks(x, xlabel)
plt.yticks(sorted(y))
plt.xlabel("abc")
plt.ylabel("freq")
plt.show()
```

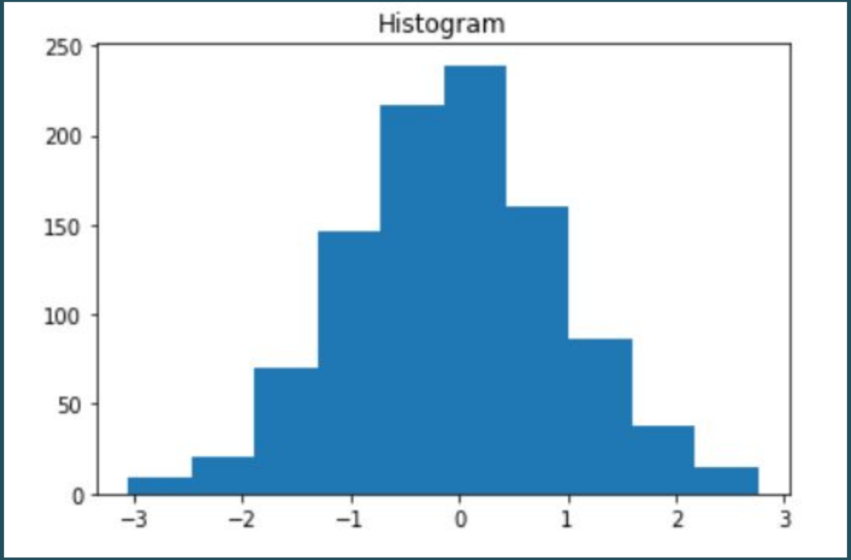


Histogram

- Plots the frequency data points in each bin
- Useful in understanding the count of data ranges



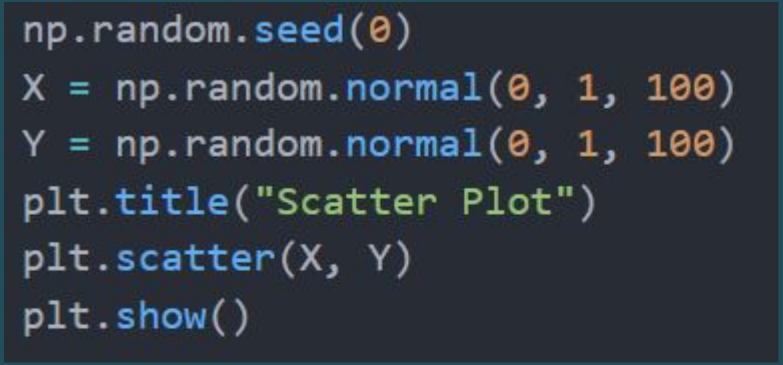
```
np.random.seed(0)
x = np.random.randn(1000)
plt.title("Histogram")
arrays, bins, patches = plt.hist(x, bins=10)
plt.show()
```



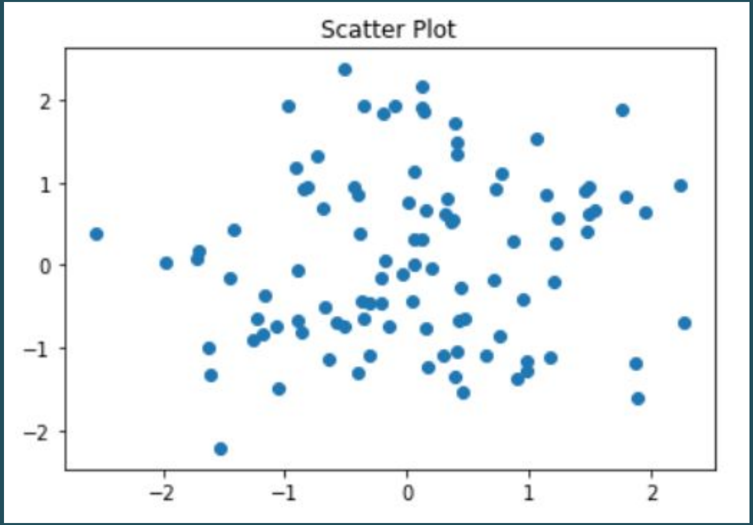
Scatter plot

TS Turing Students ROTTERDAM

- Helps in visualizing 2 numeric variables
- Helps in identifying the relationship of the data w/ each variable
- I.e. correlation or trend pattern



```
np.random.seed(0)
X = np.random.normal(0, 1, 100)
Y = np.random.normal(0, 1, 100)
plt.title("Scatter Plot")
plt.scatter(X, Y)
plt.show()
```



Some basic modules

Turing Students ROTTERDAM

- plt.plot()
  - Plotting line chart or plot other functions
- plt.xlabel, plt.ylabel
  - Labeling x and y-axis respectively
- plt.xticks, plt.yticks
  - Labeling x and y-axis observation tick points respectively
- plt.legend()
  - Signifying the observation variables
- plt.title()
  - Setting the title of the plot
- plt.show()
  - Displaying the plot

Data processing

- Data collection
- Preprocessing
  - Cleaning
- Exploration
- Analysis
- Output