# Python Tutorial – Part I: Introduction

Mark A. Austin

University of Maryland

*austin@umd.edu*
*ENCE 201, Fall Semester 2025*

September 2, 2025

# Overview


1. What is Python?
   - Origins, Features, Framework for Scientific Computing
2. Program Development with Python
   - Working with the Terminal
   - Integrated Development Environments
3. Elements of Python Programming
   - Data Types, Variables, Arithmetic Expressions, Strings, Program Control, and Functions
4. First Program (Evaluate and Plot Sigmoid Function)
5. Builtin Collections (Lists, Dictionaries, and Sets)
6. Numerical Python (NumPy)

# First Program

## (Evaluate and Plot Sigmoid Function)

# Problem Desription

## Problem Description

In neural network models, the sigmoid function:

$$\sigma(x) = \left[\frac{1}{1+e^{-x}}\right]. \tag{3}$$

serves as an activation. Our first program evaluates and plots $\sigma(x)$ over the range $x \in [-10, 10]$.

## Running the Program

From the terminal window, simply type:

```
prompt >> python3 TestSigmoidFunction.py
```

# Evaluate and Plot Sigmoid Function

The Python interpreter/compiler will complain if one or more of the required packages (e.g., matplotlib) are missing.

**Use pip to install missing Python Packages**

The standard package-management system used to install and manage software packages is called pip (or pip3).

**Example:** And installation is easy!

```
prompt >> pip3 install numpy
prompt >> pip3 install matplotlib
```

To get a list of installed packages:

```
prompt >> pip3 list
```

# Evaluate and Plot Sigmoid Function

Abbreviated Output:

```
Package                    Version
---------------------- ---------
....
jupyter                    1.0.0
Keras                      2.4.3
....
matplotlib                 3.4.1
....
numpy                      1.19.5
....
pandas                     1.1.5
....
scikit-learn               0.24.2
scipy                      1.6.2
....
sklearn                    0.0
```

# Program Source Code in Visual Studio Code

TestSigmoidFunction.py

Get Started | TestSigmoidFunction.py

Users > austin > ence688p.d > python-code.d > neural > TestSigmoidFunction.py > ...

```python
# ==============================================================
# TestSigmoidFunction.py:  Evaluate and plot sigmoid function.
#
# Written by: Mark Austin                          September, 2020
# ==============================================================

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

# define sigmoid function ...

def sigmoid (x):
    return 1/(1 + math.exp(-x))

# main method ...

def main():
    print("--- Enter TestSigmoidFunction.main()     ... ");
    print("--- ===================================== ... ");

    # Part 1: evaluate and print values of sigmoid function ...

    xvalues = list( np.arange( -10.0, 10.0, 0.5 ) );
    for x in xvalues:
       print ("--- sigmoid({:6.2f}) --> {:14.10f}".format(x, sigmoid(x)));

    # Part 2: Create list of sigmoid(x) values ...

    yvalues = []
    for x in xvalues:
       yvalues.append( sigmoid(x) );

    # Part 3: Organize and display plot ...

    fig, ax = plt.subplots()
    ax.plot( xvalues, yvalues )
    ax.set(xlabel='x', ylabel='sigmoid(x)', title='Plot sigmoid(x) vs x')
    ax.grid()

    # display plot ...

```

⊗ 0 ⚠ 0 Ln 1, Col 1 Spaces: 4 UTF-8 LF Python 3.8.2 64-bit

# Program Source Code + Output in Visual Studio Code

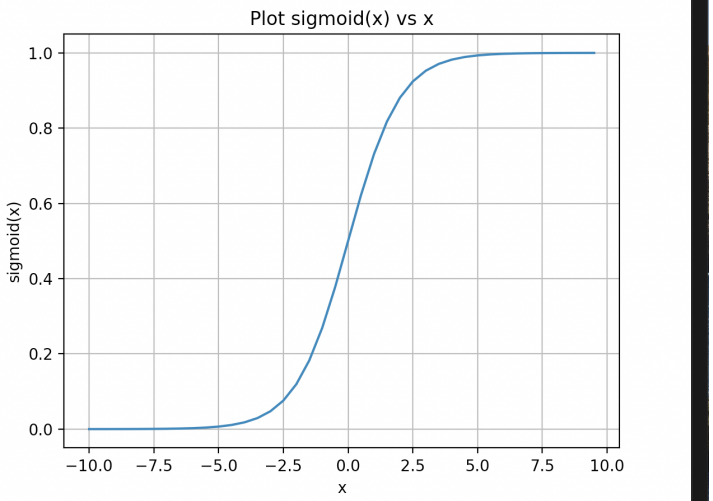
TestSigmoidFunction.py

Get Started | TestSigmoidFunction.py

Users > austin > ence688p.d > python-code.d > neural > TestSigmoidFunction.py > ...

```python
# ==========================================================
# TestSigmoidFunction.py:  Evaluate and plot sigmoid function.
#
# Written by: Mark Austin                          September, 2020
# ==========================================================

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

# define sigmoid function ...

def sigmoid (x):
    return 1/(1 + math.exp(-x))

# main method ...

def main():
    print("--- Enter TestSigmoidFunction.main()     ... ");
    print("--- ==================================== ... ");

    # Part 1: evaluate and print values of sigmoid function ...

    xvalues = list( np.arange( -10.0, 10.0, 0.5 ) );
    for x in xvalues:
       print ("--- sigmoid({:6.2f}) --> {:14.10f}".format(x, sigmoid(x)));

    # Part 2: Create list of sigmoid(x) values ...
```

Figure 1




PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

Python - austin

```
--- sigmoid(  3.00) -->   0.9525741268
--- sigmoid(  3.50) -->   0.9706877692
--- sigmoid(  4.00) -->   0.9820137900
--- sigmoid(  4.50) -->   0.9890130574
--- sigmoid(  5.00) -->   0.9933071491
--- sigmoid(  5.50) -->   0.9959298623
--- sigmoid(  6.00) -->   0.9975273768
--- sigmoid(  6.50) -->   0.9984988177
--- sigmoid(  7.00) -->   0.9990889488
--- sigmoid(  7.50) -->   0.9994472214
--- sigmoid(  8.00) -->   0.9996646499
--- sigmoid(  8.50) -->   0.9997965730
--- sigmoid(  9.00) -->   0.9998766054
--- sigmoid(  9.50) -->   0.9999251538
```

⊗ 0 ⚠ 0 Ln 1, Col 1 Spaces: 4 UTF-8 LF Python 3.8.2 64-bit

# Program Source Code

```
# ==========================================================
# TestSigmoidFunction.py: Evaluate/plot sigmoid function.
#
# Written by: Mark Austin                    September, 2020
# ==========================================================

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

# define sigmoid function ...

def sigmoid (x):
    return 1/(1 + math.exp(-x))

# main method ...

def main():
    print("--- Enter TestSigmoidFunction.main() ...");
    print("--- ================================ ...");

    # Part 1: Evaluate and print sigmoid function

    xvalues = list( np.arange( -10.0, 10.0, 0.5 ) );
    for x in xvalues:
       print ("--- sigmoid({:6.2f}) --> {:14.10f}".format(x, sigmoid(x)));

    # Part 2: Create list of sigmoid(x) values ...
```

# Program Source Code

```
    # Part 2: Create list of sigmoid(x) values ...

    yvalues = []
    for x in xvalues:
       yvalues.append( sigmoid(x) );

    # Part 3: Organize and display plot ...

    fig, ax = plt.subplots()
    ax.plot( xvalues, yvalues )
    ax.set(xlabel='x', ylabel='sigmoid(x)',
           title='Plot sigmoid(x) vs x')
    ax.grid()

    # display and save plot ...

    plt.show()

    fig.savefig("sigmoid-plot.jpg")

    print("--- ================================ ...");
    print("--- Leave TestSigmoidFunction.main() ...");

# call the main method ...

main()
```

# Program Source Code

**Points to Note**:

- Line comment statements begin with the # character.
- Lines 7-10 import the math, matplotlib, matplotlib.pyplot and numpy modules to the program, and make the functions therein available.
- Functions are the primary method of code organization and reuse in Python.
- User-defined functions are declared with the `def` keyword. A function contains a block of code with an optional `return` keyword.
- Lines 13-14 evaluate and return the sigmoid function.
- Use of the second function, `main()`, is a carry over from programming with C, where all programs begin their execution in `main()`. Its use in Python is optional.

# Program Source Code

**Points to Note** (continued):

- Line 25 creates a list of x coordinates. The numpy function `np.arange()` covers $[-10, 10]$ in increments of 0.5.
- Lines 26-27 systematically traverse the elements of `xvalues`, and compute and print the corresponding values of the `sigmoid()` function.
- Line 27 formats and prints the output. The specification `{:6.2}f` means that the output should be printed as a floating point number, six characters wide, and with two decimal places of accuracy to the right of the decimal point.
- Lines 31-33 traverse the elements of `xvalues`, and systematically assemble a list of sigmoid function `yvalues`.
- Lines 37-47 format a plot of `yvalues` vs `xvalues`, and save to `sigmoid-plot.jpg`.