Sigmoid Activation Function in Python

Define the function in python as:

```
import numpy as np
def sig(x):
 return 1/(1 + np.exp(-x))
```

inputs.

```
import numpy as np
def sig(x):
 return 1/(1 + np.exp(-x))



x = 1.0
print('Applying Sigmoid Activation on (%.1f) gives %.1f' % (x, sig(x)))

x = -10.0
print('Applying Sigmoid Activation on (%.1f) gives %.1f' % (x, sig(x)))

x = 0.0
print('Applying Sigmoid Activation on (%.1f) gives %.1f' % (x, sig(x)))

x = 15.0
print('Applying Sigmoid Activation on (%.1f) gives %.1f' % (x, sig(x)))

x = -2.0
print('Applying Sigmoid Activation on (%.1f) gives %.1f' % (x, sig(x)))
```

output:

Applying Sigmoid Activation on (1.0) gives 0.7

Applying Sigmoid Activation on (-10.0) gives 0.0

Applying Sigmoid Activation on (0.0) gives 0.5

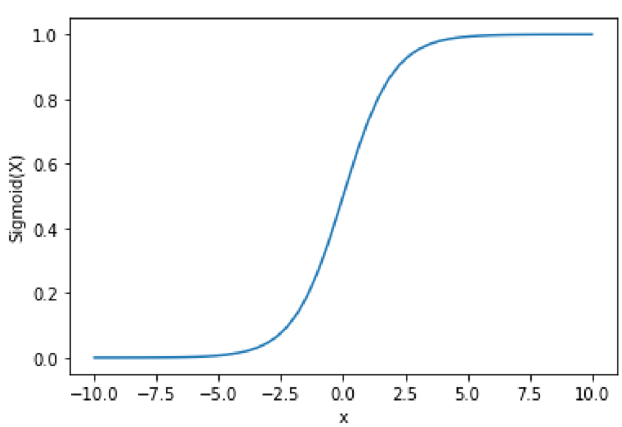
Applying Sigmoid Activation on (15.0) gives 1.0

Applying Sigmoid Activation on (-2.0) gives 0.1

Plotting Sigmoid Activation using Python

```
import numpy as np

import matplotlib.pyplot as plt

x = np.linspace(-10, 10, 50)

p = sig(x)

plt.xlabel("x")

plt.ylabel("Sigmoid(x)")

plt.plot(x, p)

plt.show()
```

Output:



Sigmoid

```python
import numpy as np
from matplotlib import pyplot as plt

# Hyperbolic Tangent
def hyp(x):
  return (np.exp(x) - np.exp(-x))/(np.exp(x) +
np.exp(-x))

# Hyperbolic derivative
def dhyp(x):
  return 1 - hyp(x) * hyp(x)

# Generating data to plot
x_data = np.linspace(-6,6,100)
y_data = hyp(x_data)
dy_data = dhyp(x_data)

# Plotting
plt.plot(x_data, y_data, x_data, dy_data)
plt.title('Hyperbolic Tangent & Derivative')
plt.legend(['f(x)','f\'(x)'])
plt.grid()
plt.show()
```

Python Tangent Hyperbolic Output