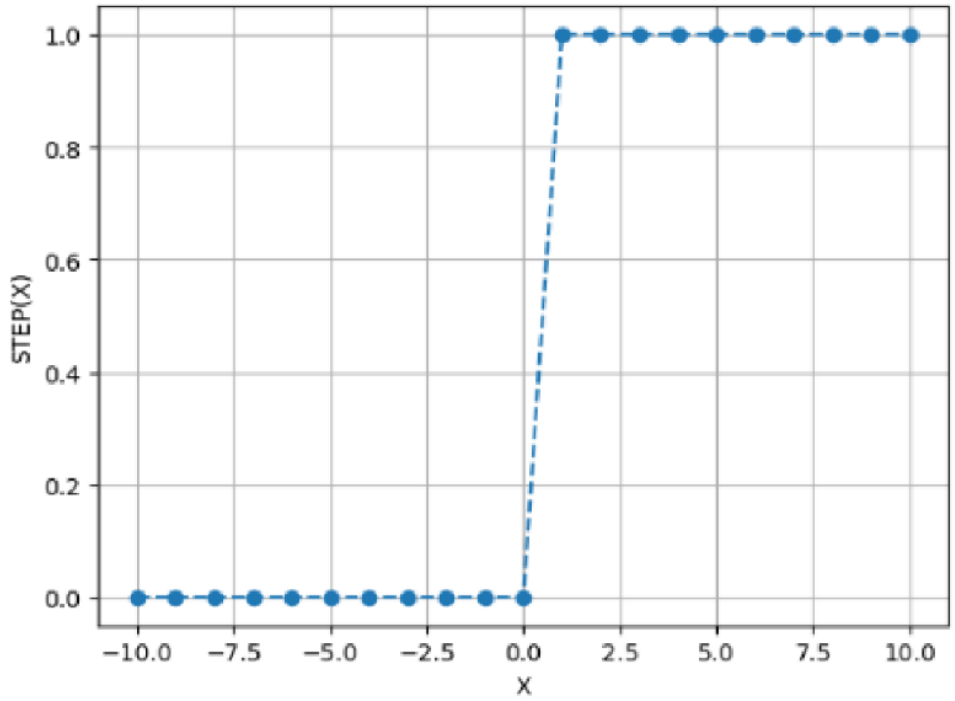
ACTIVATION FUNCTIONS

STEP FUNCTION

```
y = list(map(lambda n: 1 if n>0.5 else 0, x))
plot_graph(y,"STEP(X)")
```

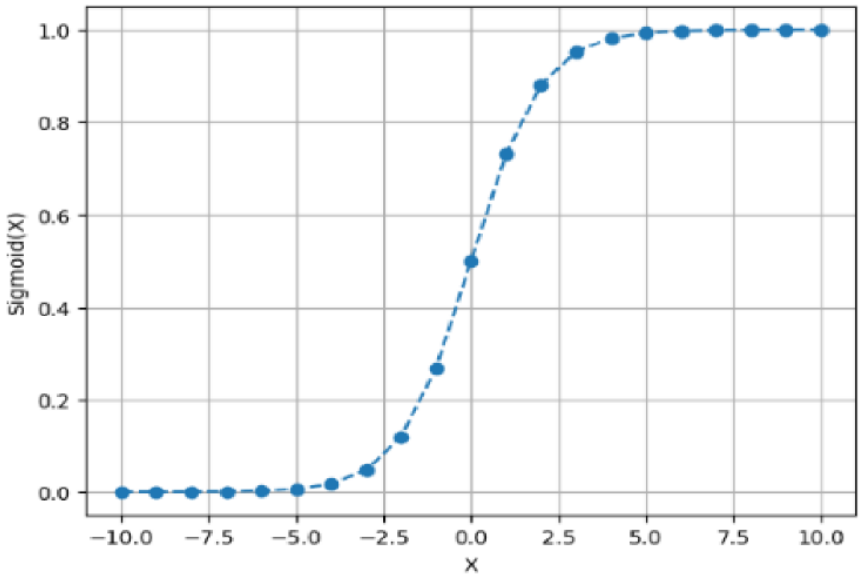


ADVANTAGES: Simple, easy to implement and interpret.

DISADVANTAGES: Output is not continuous and can't be differentiated. Limit to binary classification problem.

SIGMOID FUNCTION

[5]:
```
y = 1 / (1 + np.exp(-x))
plot_graph(y, "Sigmoid(X)")
```

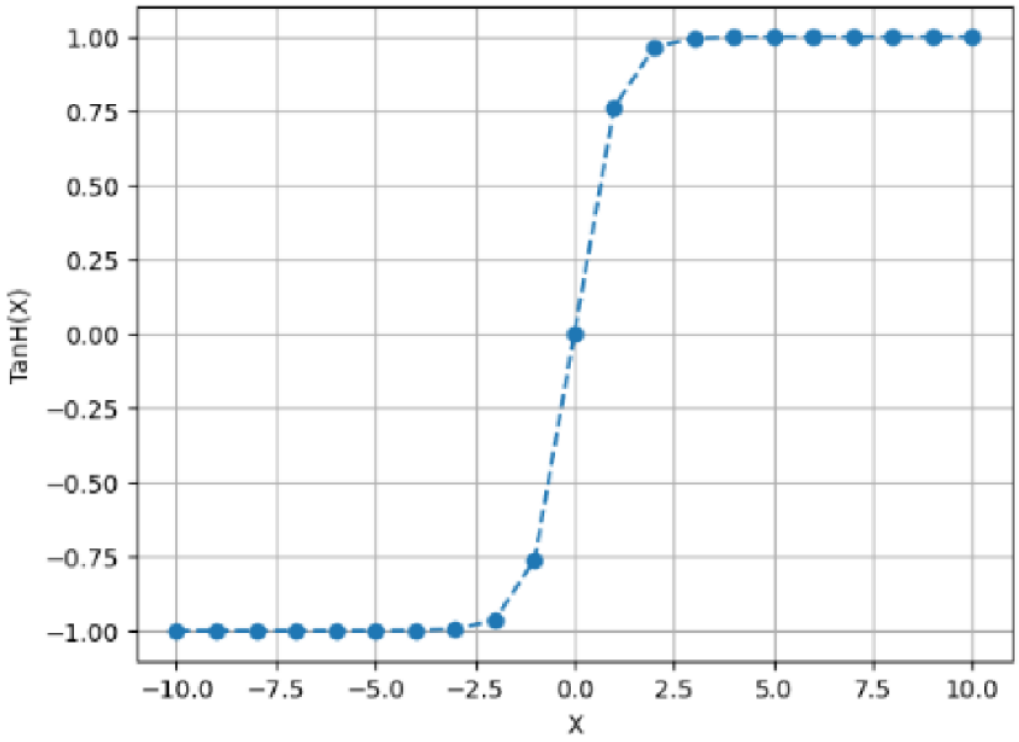


ADVANTAGES: Output is bounded between 0 & 1 which makes it suitable for binary classification problem. It is differentiable and use in backpropagation.

DISADVANTAGES: Can suffer from vanishing gradient and prone to saturation for extreme values. Not zero-centered.

TANH FUNCTION

```
y = (np.exp(2*x) - 1) / (np.exp(2*x) + 1)
plot_graph(y, "TanH(X)")
```

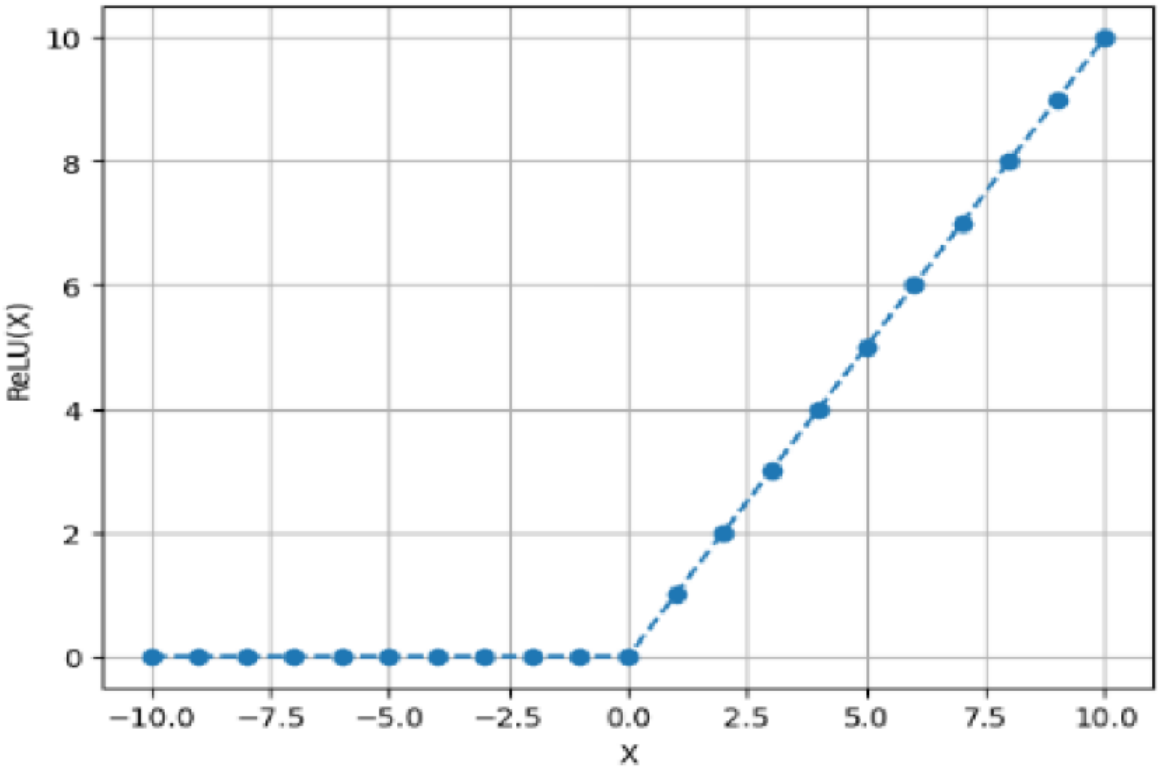


ADVANTAGES: Same to sigmoid but zero-centered. Outputs are bound between -1 & 1.

DISADVANTAGES: Like sigmoid, it can suffer from vanishing gradients.

ReLU FUNCTION

```
y = list(map(lambda a: a if a>=0 else 0, x))
plot_graph(y,"ReLU(X)")
```



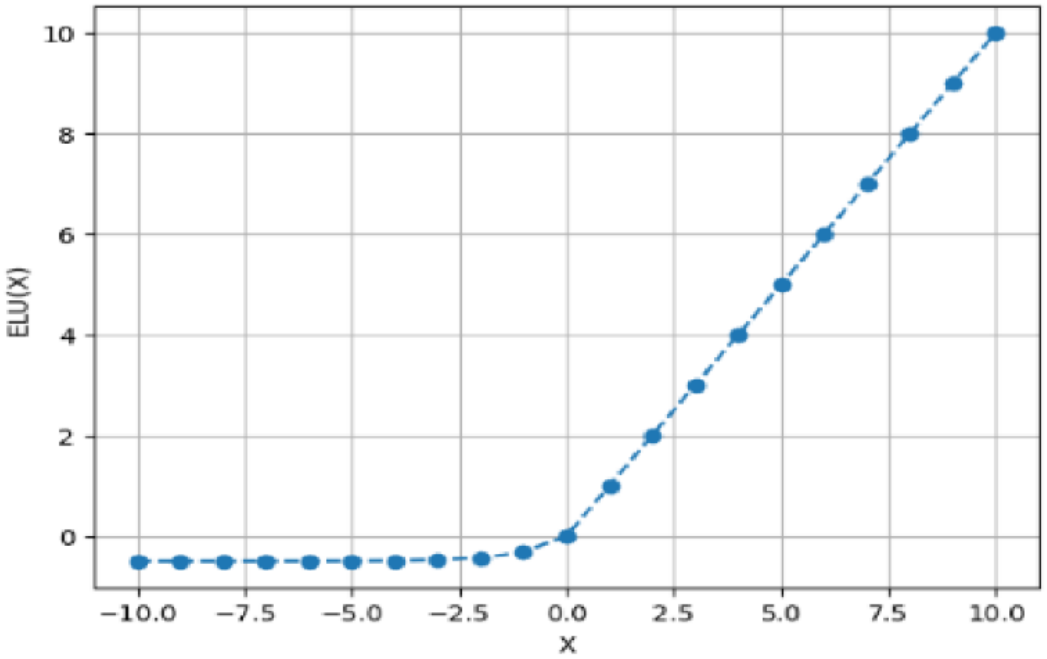
ADVANTAGES: Simple, computationally efficient and has been shown to work well in practice. It avoids the vanishing gradient problem.

DISADVANTAGES: Not zero-centered and can suffer from the “dying ReLU” problem where the gradient becomes zero for negative inputs and stop training.

ELU FUNCTION

```
from math import exp
alpha = 0.5

elu = lambda x, alpha: x if x > 0 else alpha * (np.exp(x) - 1)
y = [elu(x_i, alpha) for x_i in x]
plot_graph(y,"ELU(X)")
```



ADVANTAGES: Avoids the "dying ReLU" issue and is able to generate negative outputs. Moreover, it is differentiable for all input and has been demonstrated to perform better than ReLU.

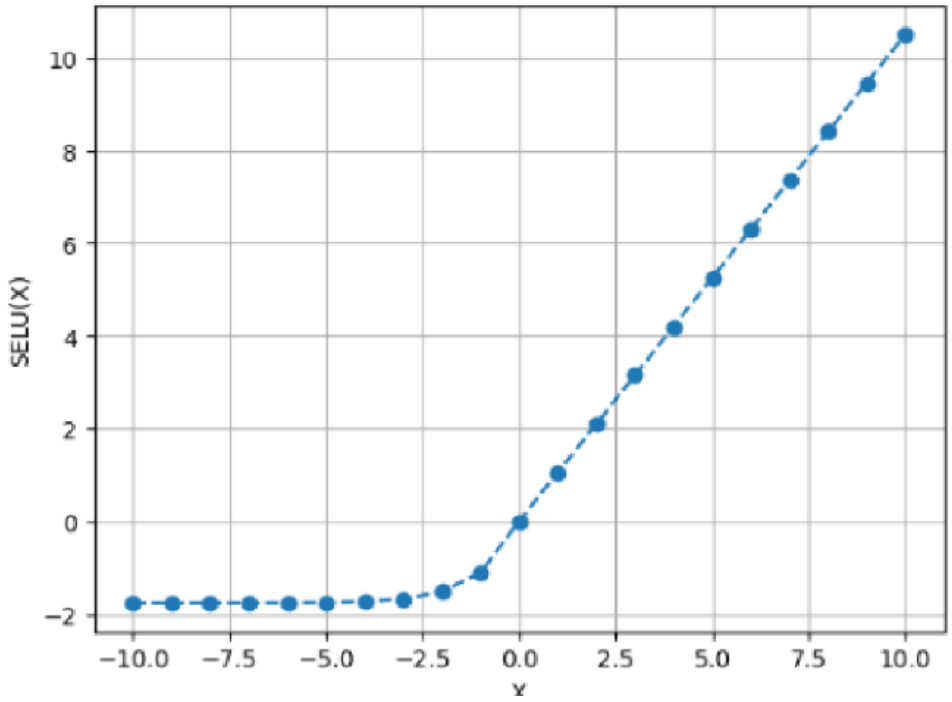
DISADVANTAGES: Somewhat more expensive to compute than ReLU.

SELU FUNCTION

```
alpha = 1.67326
scale = 1.0507

selu = lambda x, alpha, scale: scale * (x if x > 0 else alpha * (np.exp(x) - 1))
y = [selu(x_i, alpha, scale) for x_i in x]

plot_graph(y,"SELU(X)")
```



ADVANTAGES: Intended to achieve self-normalization in deep neural networks, resulting in more rapid convergence and enhanced performance. Having a zero-centered and smooth curve.

DISADVANTAGES: To achieve self-normalization, weights and biases must be properly initialized.

CONCLUSION

Hence, there is no universally superior activation function for all types of problems. Best practice is to experiment with different activation functions and choose the one that works best for your particular problem and architecture. But, ReLU is a good default choice for many problems, and ELU and SELU are good alternatives when negative inputs are present or when deeper architecture is used.