Primary Keyword: Activation function

Secondary Keyword:

What is an activation function
Sigmoid activation function
ReLU activation function
What are the properties of activation function
Why is activation function important?
Linear vs. nonlinear activation function

URL: https://www.g2.com/categories/ai-machine-learning-operationalization

Articles to include:

- https://www.geeksforgeeks.org/activation-functions/
- https://www.v7labs.com/blog/neural-networks-activation-functions
- https://www.analyticsvidhya.com/blog/2021/04/activation-functions-and-their-derivatives-a-quick-complete-guide/
- https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/

Clearscope: https://www.clearscope.io/g2/reports/bff0fc61833de60c/editor

Target category: **Artificial neural network**

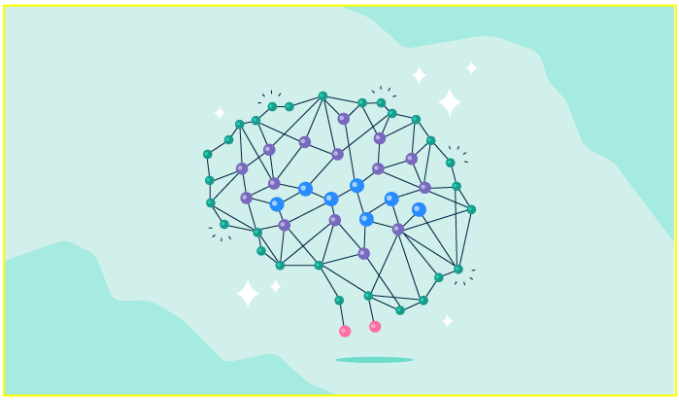
Category link: https://www.g2.com/categories/image-recognition

CTA on the side of the page: TBD

Meta Description: Activation function is a hidden layer of an artificial neural network that processes and classifies data for computers. Learn more.

Title options (minimum 5 options, list schedule score):

1. What is the activation function? How to compute your data correctly (69)
2. What is the activation function? How to set up for data accuracy (72)
3. How does the activation function lead to decision-making? (70)
4. What are activation functions? Types, examples and importance (73)



Raising the level of behavioral intelligence in computers

Humans don't get confused about making everyday life decisions.

They know exactly when to wake up, brew themselves a cup of coffee or leave for work. Their subconscious is strong enough to get them through their day without any conflict of interest.

What humans have acquired with evolution is being replicated in computers through an activation function. Mostly used in deep neural networks, the activation function creates a wise response to external user stimuli, just like the human nervous system.

In the business sphere, the activation function is a part of artificial neural network software that can be integrated into your business applications. It validates your business models and takes a rain check on data-centric matters. The more accurate your business model is, the deeper you will penetrate your target sales market.

The activation function is a core feature of deep learning introduced between the mid-2000s to 2010 (with theoretical proposals dating back to 1990). Deep learning is an artificial intelligence technique that works on large datasets and high graphical computational power (GPU) or CPU power. Because of this reason, it learns data patterns quickly, delivers accurate outcomes, and solves problems on a large scale.

What is an activation function?

The activation function, also known as the transfer function, is used to determine the output of an artificial neural network, which is a supervised deep learning method. The activation function decides the category of the input data by activating the right node. The node determines an output value that travels through to the output layer and exits the neural network.

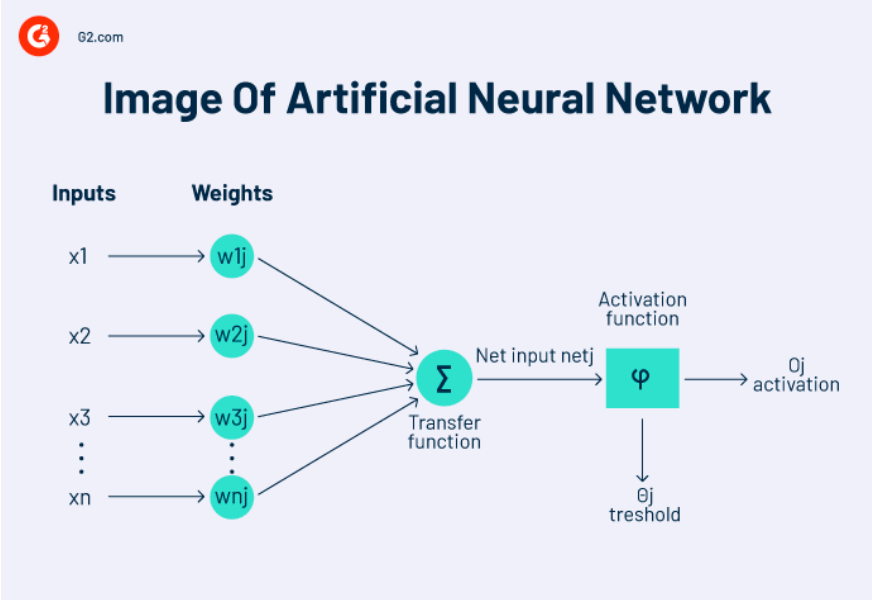
ANN uses training data to perform categorical analysis of the incoming test data. The test data evaluates the accuracy of the trained algorithm in different ways.

Categories, in general, are quantitative variables that data can be grouped into. A common example will be "whether a person is obese or non-obese, male or female, tall or short, and so on. This categorical analysis of data using artificial neural networks is known as data classification.

What is an artificial neural network?

If you have read about artificial neural networks, you might be aware of their tiered structure. A neural network consists of a minimum of three layers: the input layer,

the hidden layer, and the output layer. However, it can go up to 7 layers. There is no consensus on how many layers a neural network can have.



The input layers accept data from the user, along with weights and added bias. The hidden layer, or middle layer, consists of an activation function that processes information and rounds up the output value. Finally, the output layer displays the result.

Weights and bias are externally added to the overall net input value. You often run neural networks iteratively, and the weights get updated with each new row of data. While processing data, weights go up and down quite often whenever the actual class matches the predicted class. The process gets iterated multiple times until you obtain a certain degree of accuracy.

An artificial neural network fires its decision-making nodes like the human brain fires electrical impulses.

For any kind of neural network, the range of an activation function fluctuates between -infinity to +infinity. The weight parameters are often randomly defined as decimals, like 0.01, 0.05, 0.1, and so on.

Under all circumstances, the activation function remains non-linear and applicable to any data input.

Why is activation function important?

The activation function aims to raise a machine's consciousness and make them aware of external objects. In terms of structure, it mimics the human brain's central nervous system.

In the business sphere, activation functions can be integrated into ERPs to label your company data. Every stakeholder can use AI automation to review their reports, handle projects or even study consumer data points.

It also detects real-life objects through sonics for a blind man to walk freely. It can be used in metal scanners to detect unwanted objects at the airport. Some entities use this technology in surveillance and security systems for facial recognition. Smart vacuums, pods, and glass cleaners based on object detection have also been launched to ease the lives of people.

The non-linear activation function is the most important element of a neural network. It runs the chores, computes values, triggers decisions, and displays the required outcome.

The formula of activation function

Activation function is the main processing layer deep neural network. While the activation algorithm strives for an 80% and above accuracy, there are sometimes issues with data. Dirty data, outliers, multicollinearity, or confounding variables impede the accuracy of data.

Without an activation function, a neural network is a linear regression model which doesn't work on more complex datasets. The coefficients of each data variable

would be linear and static, and the success of predicting the class for it would be less.

The purpose of the activation function is also to reduce the small error value along with data categories. It carries out all the operations, from accepting input to performing the hypothesis and predicting the correct object category. Inputs at different stages are collated using the below mathematical approach for the activation function:

Formula for activation function:

y = ∑(weights*input + bias).

The range of "y" in this equation is -infinity to infinity. Note that the value of y needs to be bounded to a whole number for accurate prediction. However, "y" may or may not be binomial. You can use ANN for classification such as "cancerous" or "non-cancerous" (Like in nanobots).

Along with "y," you also get a residue value. This value can be fed in the next cycle of data entry, also known as backpropagation.

What is backpropagation?

Backpropagation reduces the error residue of a neural network. It transmits the output again to the input layer to stabilize numbers. The whole process iterates backward to avoid back-and-forth between the layers. It's fast and simple.

The algorithmic flow chart of backpropagation flows involuntarily in the following manner:

1. The output travels back to the input layer of the neuron.

2. The value is added to the new inputs, along with their weights.
3. The output of each layer, from input to hidden to output, is calculated.

Backpropagation error = Actual output - desired output

4. The weights are again adjusted to reduce error scope.
5. The process is repeated till the user receives desired output.

Did you know? Backpropagation models are used to train feedforward neural networks to avoid decision loops. This type of artificial neural network steads forward and not backward. During data flow, input nodes receive data that travels through hidden layers and exits through the output layer.

What are gradients?

In the activation function, gradients are the slope of the function curve that is evaluated by its partial derivatives at a given point in space. The higher the gradient, the steeper the slope and the faster a model will train. It simply measures the change of weight components during the insertion of data rows with respect to the output error value.

What are batch-size?

The batch size is set between 0 and n. It defines the number of training data samples an ANN model sees before updating the weights. So, during every update, the batch size is 1. But, if you set the batch size to 50, the network won't be activated till the model runs on 50 training samples. Batch size is an efficient technique, as updating weights manually gets very expensive.

Linear activation function vs. Non-Linear activation function

Activation function can be linear or nonlinear. A linear function is plotted on a straight line and has an infinite output range. It is used to make simple data-related observations. A linear model is monotonic (either increasing, decreasing, or flat) and is a supervised form of machine learning that can train specific datasets.

Linear activation functions are used within normal data distribution and are quantitative. Nominal or ordinal variables are converted into dummy variables to run the linear algorithm. Although a linear activation function can be used for labeling data, it is not ideally recommended. The monotony of a linear activation function makes it unsuitable for accurate prediction.

In contrast, a non-linear function has a specific range of values of the desired output. It tests the scope of a problem much faster than a linear function. A non-linear function is used to classify complicated data types, including images, speech, video, audio, etc.

8 Types of activation functions in the artificial neural network

The current popular convolutional neural networks like R-CNN, Mask R-CNN, or YOLO V5 use non-linear activation functions to categorize objects. Let's understand the types of non-linear functions in detail:

Logistic sigmoid function

The sigmoid function is a special function that looks like an “S” shaped graph and ranges between 0 and 1. It’s used for binary classifications, where you can have two possible outcomes for the predicted variable. Examples are “died” or “survived,” “malignant,” or “non-malignant”.

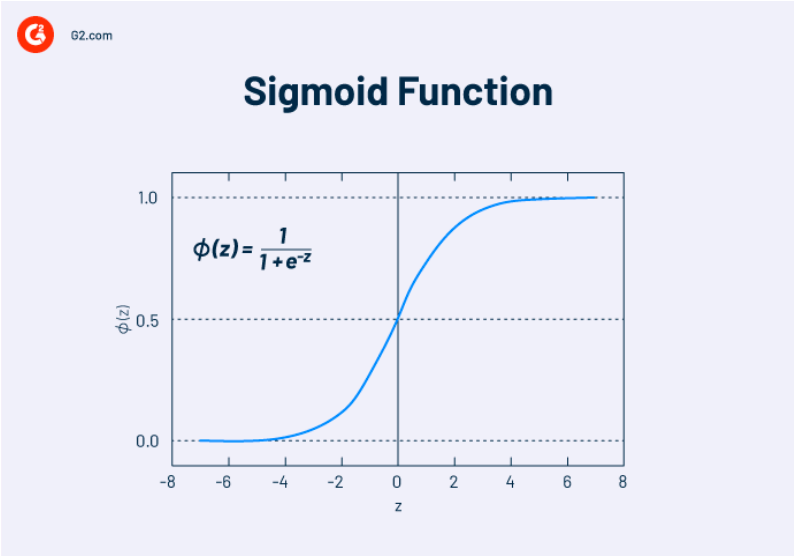
For example, if you want to classify an airplane in an image, the sigmoid activation function will deliver the output as “yes”. It suggests that a given image contains an airplane.

The sigmoid activation function is calculated with this formula:

f(z) = 1.0 / (1.0 + e-z),

Where “e” is a mathematical constant that would either increase or decrease based on the size of input data.

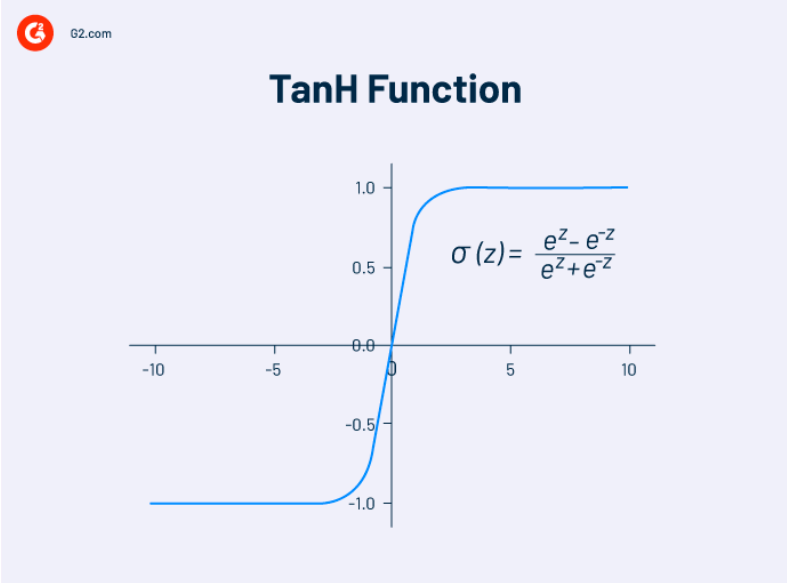
Because of the constrained graphical area, the output sometimes gets stuck in a loop. To overcome the issue, the TanH function can be used.



Tip: While using an activation function, use “Xavier normal” or “Xavier uniform” weight initialization to declare inputs. This technique scales your answers between “yes” or “no.”

Hyperbolic Tangent Function (Tanh)

TanH is similar to sigmoid but a bit better in terms of output accuracy. It ranges between -1 to +1 and also includes negative output values.



In the TanH function, the larger the input value (more positive), the closer the output would be to 1. Similarly, the smaller the input value, the closer the output would be to -1.

The TanH activation function can be derived from this formula:

TanH (x) = (e^x – e^-x) / (e^x + e^-x).

In the above-mentioned formula, e stands for exponential constant. As TanH is a non-linear function, it has area gradients that can be easily computed within the range of (0,1) and (1,1). The diverse nature of the TanH function makes it more accurate for business processes.

You can create a TanH function using the below code. It is written and compiled in integrated data environments like Jupyter or Spyder.

```
# example plot for the tanh activation function
from math import exp
from matplotlib import pyplot

# tanh activation function
def tanh(x):
 return (exp(x) - exp(-x)) / (exp(x) + exp(-x))

# define input data
inputs = [x for x in range(-10, 10)]
# calculate outputs
outputs = [tanh(x) for x in inputs]
# plot inputs vs outputs
pyplot.plot(inputs, outputs)
pyplot.show()
```

Source: *machinelearningmastery.com*

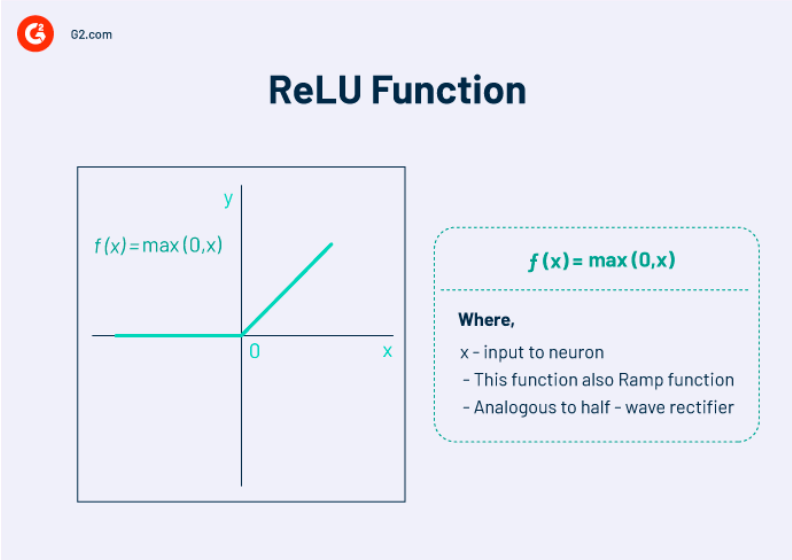
ReLu function

The rectified linear activation function is a non-linear or piecewise linear function that would return either the same value or 0.

The following formula denotes it:

$f(x) = \max(x)$; if $x>0$,

$f(x) = 0$; if $x<0$



If the input value is positive, the output of the activation function will be the same as the input. But if the value is negative, the output will be zero. ReLu function has one of the highest accuracies and is used in all the latest classifiers like CNN, R-CNN, multilayer perceptrons, self-organizing maps, and generative adversarial networks.

At an initial glance, ReLu seems linear because it is a steeply increasing function and equalizes y to x. However, it is a non-linear function that especially returns "0" for negative input.

For inputs less than 0, ReLU deactivates the decision-making neurons to produce "zero" as an output. In other words, it is known as the dying ReLU problem. To return a positive value, the ReLU function needs to be backed up with something. Otherwise, the processing comes to a halt.

To define the ReLu activation function, here is a relevant code snippet.

#to define ReLu

```
def ReLU(x):
 if x>0:
          return x
 else:
  return 0

#to enter and store a set of input values and plot them

from matplotlib import pyplot
from matplotlib import numpy
def relu(z):
          return max(0.0, z)
input = [z for z in range(-1, 10)]
# run ReLu on every input value
output = [relu (z) for z in input ]
# plot our graph
pyplot.plot(series_in, series_out)
pyplot.show()
```

In terms of output accuracy, ReLU is better than TanH and Sigmoid and has a lesser error value. The specificity of the ReLU function depends on the kind of pre-trained weights being added along with the input data.

Tip: Specificity is the true negative rate which describes the presence or absence of an output category. It is calculated using the formula:

Specificity = True Negative /(True Negative + False Positive)

Here, negative or positive refers to the certainty of a category.

Leaky ReLu function

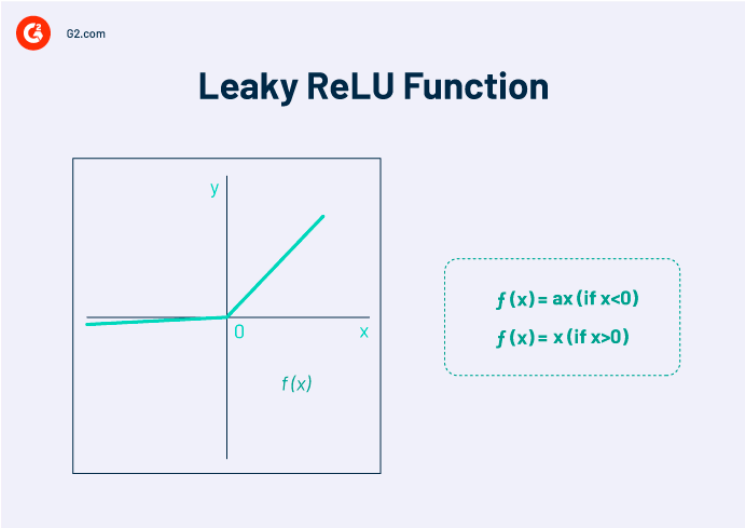
Leaky ReLu function aims to solve the inconsistencies of the ReLU function and process things faster. It is an approved version of the ReLU function and is non-linear and differentiable in nature.

Instead of defining the function as zero for negative values of x, Leaky ReLu simply multiplies x by 0.01. So, whenever f(x)<0, the graph remains moving upwards and doesn't die down. Hence, the graph of the original ReLU function changes a bit to accommodate the additional value of x.

```
F(x) = max (0.01*x, x)   /*for all values of x*/
```

This function returns x, for positive data values, and a decimal value of x, for a negative value.

Thus, the neurons stay activated even when the input is negative.



Softmax Activation Function

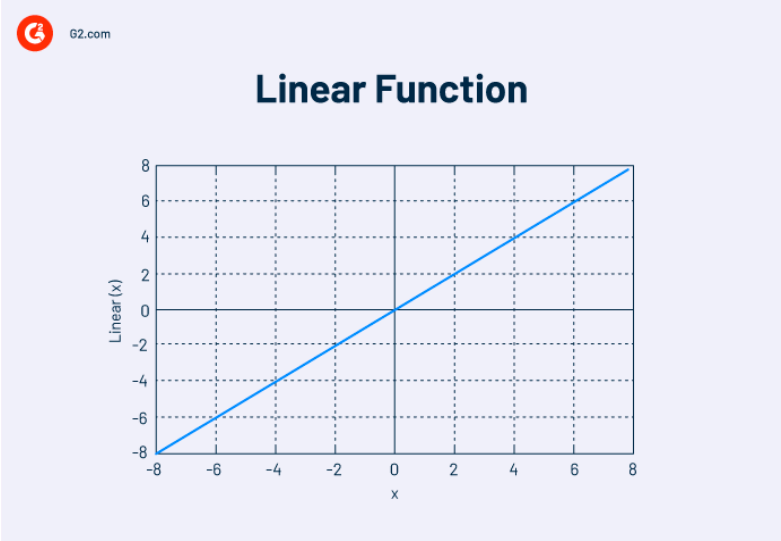
Softmax layer is the last layer of a deep neural network, which returns the class and location of the input. It also returns the value of location coordinates, axial parameters, and bounding boxes. It is also known as the "BB regressor," which comprises a support vector machine that classifies objects.

The softmax layer converts a set of input values into a singular output vector (say v), which is fed into a support vector machine (SVM). The results are twofold: the class of the object and utility. One of the best examples of this is object recognition.

The term softmax depicts the "smooth functioning" of an activation function. This layer has been recently added to modern neural networks to process data in less time.

Linear Function

linear activation, also known as no activation, is a linear model that displays a direct correlation between input and output. The data you feed to the input node and the resultant matrix would be identical. Your input variables will be your predictor variables without any difference between them.



As linear function is a strictly increasing or decreasing function, it has 2 major drawbacks:

- Due to the absence of error, you cannot use backpropagation in this type of neural network.
- A linear activation function will club all the layers of a neural network into one single layer.

Mathematically, it is denoted by:

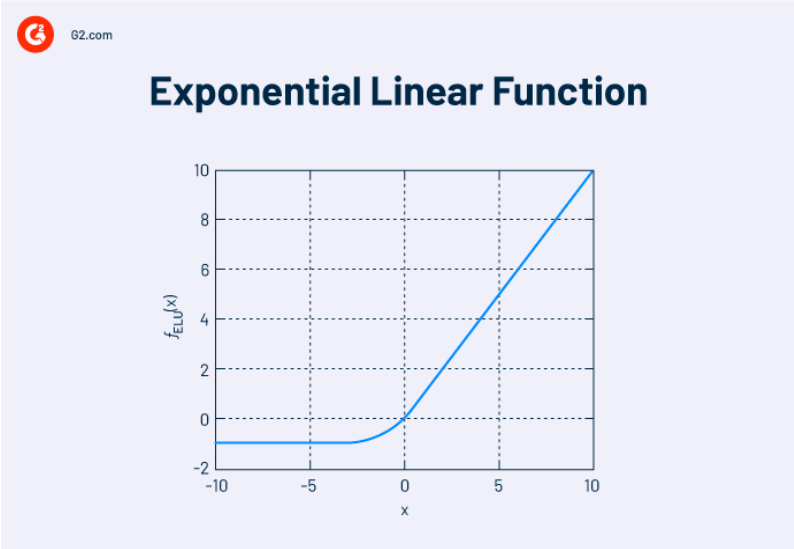
$f(x) = x$.

The one-dimensional nature of a linear activation function doesn't make it suitable for classifying large volumes of data.

Exponential linear units (ELUs)

Exponential linear unit is different from other linear activation functions. This algorithm processes negative data elements into useful output classes. If the input

value decreases, the output is capped at -1, as the exp (x) limit goes to -infinity. ELU activation function calculates the real value of the input, regardless of how low it can be. The range can go up from -infinity to +infinity.



If the input values are really small, there will be no information transmission from one node to the other node of the neural network. And the output would be the same as the input.

The difference between ELU and other functions is that it gives negative outputs. A negative output would mean the neural network acquires natural gradients. The features extracted from natural gradients make the output more accurate.

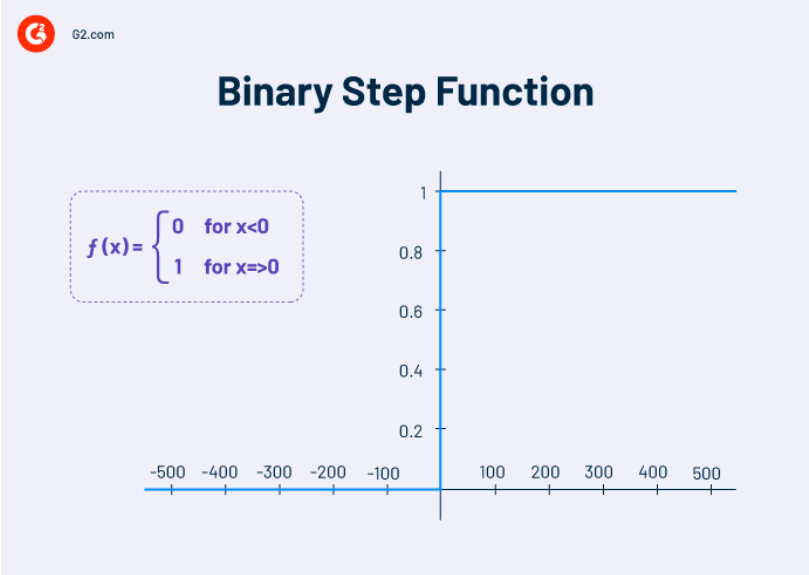
Tip: A derivative is a slope of a tangent line. If the derivative it positive, it means the function is increasing. When it is negative, it means the function is

decreasing. The function may have achieved a maximum or minimum if it's zero.

Binary Step Function

is a non-linear activation function that sets a threshold for the input values. If the input values surpass that threshold, the neurons activate and return the value 1. If not, it returns the value as 0.

This activation function is mainly used to create a binary classifier (yes or no). But it might not be helpful if your database has multiple categories, classes, or column headers.



For example, suppose you have a supplier database consisting of transportation, warehouse, raw material, and assembly costs and want to calculate a competitive supplier cost. You must co-relate between your input and predicted variables for each expected cost. In that case, the binary step function will not be suitable.

This is a simple activation function that can be called with a simple if-else

```
condition in python:

def bin(x):      /function declaration and body
if x<0:
   return 0
else:
   return 1


bin(5), bin(-1)    /function call



Output: (1,0)
```

Pros and Cons of Activation Functions

Not every activation function is cross-usable in nature. Its actual use case is to label your business data in a way that makes sense. They have a defined range of outputs and added errors, which can change your predicted outcomes. The neural network results in the same net input without an activation function. Discrepant data metrics won't lead your analytics team anywhere.

To decide which activation function can ease your work, check out the pros and cons:

- **Linear Activation Function:** It gives a range of similar outcomes, not necessarily in a binary (yes/no) fashion. It connects a few neurons together, takes a call on the best-suited class, and executes it.

  However, it is a constant gradient, whether it increases or decreases. So, the concept of using backpropagation for error reduction fails.

- **Logistics Sigmoid:** It is a non-linear function that is applicable to complex commercial datasets. It gives an analog, human-readable output.

  However, the function traverses between a specific range. Sigmoids saturate and kill area gradients after some time. This results in vanishing gradients, which makes your network slow and error-prone.

- **Tanh:** The gradients are strong for TanH since it accepts negative input values. Negative input values have deep derivatives, which result in accurate output.
  However, it also has a vanishing gradient problem.

- **ReLU:** ReLU is an inexpensive way of processing data. Mostly because it uses simple mathematical expressions in nodes of a neural network. It avoids and corrects the vanishing gradient problem.

  If the input surpasses the required range threshold, the function becomes dead. The burden of pre-trained weights might not activate the neurons ever. This results in the dying ReLU problem.

- **Leaky ReLU:** It solves the inconsistencies of the ReLU function by multiplying the input variable (x) with a delta component (0.1x).

  As it possesses a little linearity, it is unsuitable for complex classifications, unlike Sigmoid or TanH.

- **ELU:** Unlike ReLU, ELU also produces negative and accurate outputs.

  On the flip side, input values greater than zero triggers the activation to the output range of 0 to infinity.

How to choose the right activation function

While each one of the above functions seems like a good fit, you need to analyze the problem at hand first. You need to check the layer location, optimizers, and other parameters of your neural network. It is recommended to choose an activation function based on the error margin of different models.

The output of each one of them differs by a huge margin, and so choosing the wrong one might result in an inadvertent loss for your company.

If you're a data scientist trying to optimize your client requirements using artificial neural networks, keep the following things in mind.

1. Begin with the ReLU function first, then move to other activation functions if it doesn't create a good fit model.

2. Logistic sigmoid and TanH are restrictive functions that won't cater to every input variable. Refrain from using these in the hidden layers of ANN.

3. If your neural network has more than 50 layers, use the swish function.

Choosing the right activation function also depends on the type of data analysis you need to perform.

1. **Linear regression** = Linear activation function.

2. **Multiclass classification** = softmax activation function.

3. **Multilabel classification** = logistic sigmoid activation function.

4. **Convolutional neural network** = ReLU activation function.

5. **Recurrent Neural Network, Convolutional Neural Network** = ReLU activation function

Creatures from deep

Artificial intelligence is an ocean of pearls you can dive deep into and extract benefits. It is a roadmap for end-to-end business automation. Better automation will result in a better customer experience, which automatically translates into more sales. Deep learning helps you navigate the ups and downs of an uncertain market by molding your data in different ways, where each way brings in the cash.

Be more data opportunistic and realistic! Harness the power of AI and MLops software to scale your brand vision beyond your limits.