Understand the Softmax Function in minutes

Why is it called Softmax?

- It is an approximation of Max.
- It is a soft/smooth approximation of max. Notice how it approximates the sharp corner at 0 using a smooth curve.

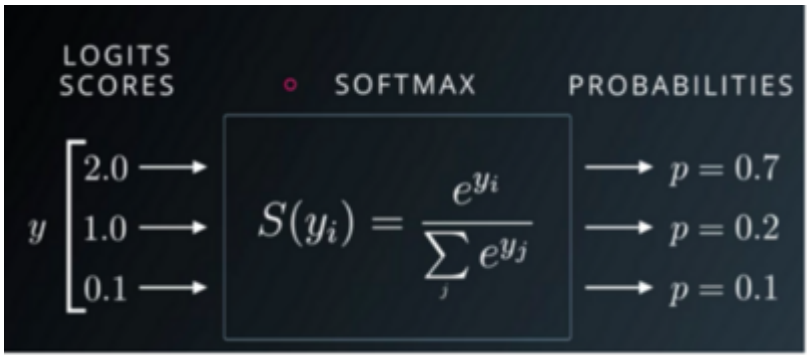
What is the purpose of Softmax?

Softmax gives us the differentiable approximation of a non- differentiable function max. It is important **for optimizing models**, including machine learning models, it is required that functions describing the model be differentiable. So if we want to optimize a model which uses the max function then we can do that by replacing the max with softmax.

Why is it useful ?

- The softmax activation is normally applied to t**he very last layer in a neural net**, instead of using ReLU, sigmoid, tanh, or another activation function. The reason why softmax is useful is because it converts the output of the last layer in your neural network into what is essentially a probability distribution. If you look at the origins of the cross-entropy loss function in information theory, you will know that it "expects" two probability distributions as input. That's why softmax output with cross entropy loss is very common.
- Other activation functions include RELU and Sigmoid. It is frequently used in classifications. Softmax output is large if the score (input called logit) is large. Its output is small if the score is small. The proportion is not uniform. **Softmax is exponential, can enlarge differences** - push one result closer to 1 while another closer to 0. It turns scores aka logits into probabilities. **Cross entropy (cost function) is often computed for output of softmax and true labels (encoded in one hot encoding).**
- .Softmax is **NOT a loss function**, but is used to make the output of a neural net more "compatible" with the cross entropy or negative log likelihood loss functions.

Softmax extends this idea into a multi-class world. That is, Softmax assigns decimal probabilities to each class in a multi-class problem. Those decimal probabilities must add up to 1.0. This additional constraint helps training converge more quickly than it otherwise would.



For a vector , softmax function is defined as:

$$S(y_i) = \frac{e^{y_i}}{\sum_{j=0} e^{y_j}}$$

Softmax function is nothing but a generalization of sigmoid function! Now, this softmax function computes the probability that the training sample belongs to class given the logits vector as:

$$P(y = j|Z_i) = [S(Z_i)]_j = \frac{e^{Z_{ij}}}{\sum_{p=0}^{k} e^{Z_{ip}}}$$

In vector form, we can simply write:

$$P(y|Z_i) = S(Z_i)$$

For simplicity, let denote the **softmax probability vector** for observation.

For example, returning to the image analysis, Softmax might produce the following likelihoods of an image belonging to a particular class:

Class	Probability
apple	0.001
bear	0.04
candy	0.008
dog	0.95
egg	0.001

Softmax is implemented through a neural network layer just before the output layer. The Softmax layer must have the same number of nodes as the output layer.

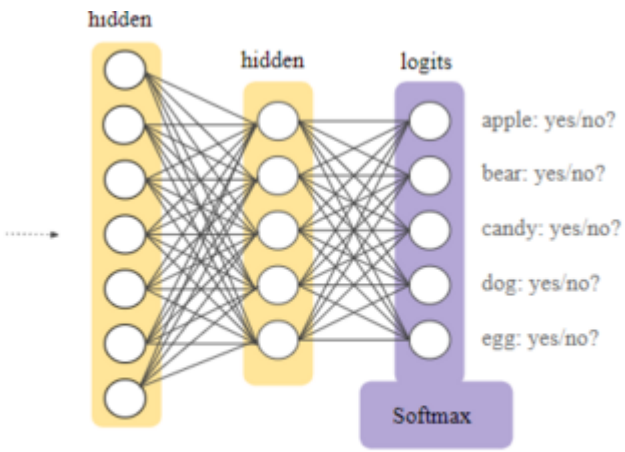


Figure 2. A Softmax layer within a neural network.

Softmax Options

Consider the following variants of Softmax:

- **Full Softmax** is the Softmax we've been discussing; that is, Softmax calculates a probability for every possible class.
- **Candidate sampling** means that Softmax calculates a probability for all the positive labels but only for a random sample of negative labels. For example, if we are interested in determining whether an input image is a beagle or a bloodhound, we don't have to provide probabilities for every non-doggy example.

Full Softmax is fairly cheap when the number of classes is small but becomes prohibitively expensive when the number of classes climbs. Candidate sampling can improve efficiency in problems having a large number of classes.

One Label vs. Many Labels

Softmax assumes that each example is a member of exactly one class. Some examples, however, can simultaneously be a member of multiple classes. For such examples:

- You may not use Softmax.
- You must rely on multiple logistic regressions.

For example, suppose your examples are images containing exactly one item—a piece of fruit. Softmax can determine the likelihood of that one item being a pear, an orange, an apple, and so on. If your examples are images containing all sorts of things—bowls of different kinds of fruit—then you'll have to use multiple logistic regressions instead.

In **deep learning** and neural networks, the final layer that data is passed through is called the “output layer”. Within this layer is an “activation function” that will determine the final output; for example, in a binary classification, the activation function could be a sigmoid function that will produce a 0 or a 1 based on a threshold to indicate which class the input belongs to.

In the case of multi-class classification problems, **the softmax function may be used as the activation function**. Based on the inputs, the softmax function returns a probability for each of the possible classes