Math notation: Cheat sheet

Matrix notation

Bold face lower case, as in $\mathbf{a}$, represents a vector; most of the time this represents a column vector.

Bold face upper case, as in $\mathbf{A}$, represents a matrix.

T - usually means transpose, as in $\mathbf{w}^T$.

Probability

$P(X = x)$ - probability that random variable X takes on a value x; a single value between 0 and 1.

$p(x)$ - probability distribution, a function that assigns probabilities to all possible values x can take; sums up to 1.

$p(x|y)$ - probability distribution of attribute x given the value of y is known

Input

x - single attribute input.

$\mathbf{x}$ - an $M \times 1$ input vector consisting of M attributes.

x_j - j^{th} component of the input vector; j^{th} attribute of the input.

$\mathbf{x}_i$ - i^{th} input vector from the training set.

x_{ji} - j^{th} component of the i^{th} input vector.

y - single value output of a computational model; output of the model in response to input x.

Output

$\mathbf{y}$ - an $K \times 1$ output vector consisting of K values; output of the model in response to vector input $\mathbf{x}$.

y_j - j^{th} component of the output vector;j^{th} value of the output.

$\mathbf{y}_i$ - i^{th} output vector; output of the model in response to input vecotor $\mathbf{x}_i$.

y_{ji} - j^{th} value of the i^{th} output vector.

Desired output

$\tilde{y}$ - single value desired output; the value that we want the computational model to output.

$\tilde{\mathbf{y}}$ - an $K \times 1$ desired output vector consisting of K values; desired output of the model in response to vector input $\mathbf{x}$.

$\tilde{y}_j$ - j^{th} component of the desired output vector; j^{th} value of the desired output.

$\tilde{\mathbf{y}}_i$ - i^{th} desired output vector; desired output of the model in response to input vecotor $\mathbf{x}_i$.

$\tilde{y}_{ji}$ - j^{th} value of the i^{th} desired output vector.

Computational model

$\mathbf{w}$ - set of parameters of a computational model; it is a set of all the parameters that model needs; occasionally, in the context of neural networks, it might represent the weight vector of the single output neuron (should be obvious from the context).

w_j - j^{th} parameter of a computational model; occasionally, in the context of neural networks, it might represent the weight on the j^{th} input of a single neuron (should be obvious from the context).

$f(\mathbf{x}, \mathbf{w})$ - hypothesis function that takes input vector of M attributes and is parametrised by the set of U parameters $\mathbf{w}$; produces output vector $\mathbf{y}$ of K values.

M - number of input attributes, dimensionality of the input vector.

K - number of output values, dimensionality of the output vector.

U - number of parameters in the model.

N - number of training samples; number of input with the desired output pairs.

T_t - temperature of the simulated annealing algorithm at time t.

Optimisation

$\Delta\mathbf{w}$ - change in model parameters; occasionally, in the context of neural networks, it might represent the change of weight vector of the single output neuron (should be obvious from the context).

Δw_j - change of the j^{th} parameter; occasionally, in the context of neural networks, it might represent the change of the weight on the j^{th} input of a single neuron (should be obvious from the context).

α - learning rate parameter that controls the magnitude of the change in parameters; typically some value between 0 and 1.

J - the cost/performance of the model for given choice of parameters and evaluated on the training set.

e - depending on the context, might represent number $e = 2.718...$ where something raises that number to some power, or an error that is the different between actual output of a model and desired output (should be obvious from the context).

exp - number $e = 2.718...$ for exponential function - used whenever e in the context is supposed to correspond to error.

e_i - error in the i^{th} output of the model.

$\frac{\partial J}{\partial w_j}$ - partial derivative of the cost with respect to j^{th} parameter of the model; it's essentially a formula for how w_j affects the cost J, which can be used to figure out what the update Δw_j should be in order to change the output to reduce the cost; this is a partial derivative, because it treats w_j as a variable, and all w_k where $k \neq j$ as constants.

Neural networks

L - number of layers in the networks

l - indexes layer in the network

U_l - number of neurons in layer l.

$\mathbf{W}_l$ - $U_{l-1} \times U_l$ matrix of all the weights in layer l;number of rows correspond to number of inputs to the layer, number of columns correspond to number of neurons in the layer .

$\mathbf{b}_l$ - $U_l \times 1$ vector of all biases in layer l.

$\mathbf{v}_l$ - $U_l \times 1$ vector of activities of all neurons in layer l.

$\mathbf{y}_l$ - $U_l \times 1$ vector of outputs from all neurons in layer l.

$w_{ij}^{[l]}$ - weight on the connection between neuron i from layer $l-1$ and neuron j from layer l.

$b_j^{[l]}$ - bias input to neuron j in layer l.

$v_j^{[l]}$ - activity of neuron j in layer l.

$y_j^{[l]}$ - output of neuron j in layer l.

$\Delta w_{ij}^{[l]}$ - change in weight on the connection between neuron i from layer $l-1$ and neuron j from layer l.

$\Delta b_j^{[l]}$ - change in bias of neuron j in layer l.

$\delta J_j^{[l]}$ - blame on neuron j in layer l for network producing wrong output with respect to some cost J.

$\frac{dy_j^{[l]}}{dv_j^{[l]}}$ - derivative of the output of j^{th} neuron in layer l with respect to its activity; it's a derivative of the activation function used on that neuron.