

Dropout in neural networks

Description

- solution to overfitting the model
- randomly eliminate p nodes in each hidden neural network layer during training
- train the model using the reduced list of nodes

Implementation of inverted dropout

- set the probability that any given node in a layer will be kept

$$keepprob = 0.8$$

- randomly choose which nodes to drop in layer l

$$todrop^{[l]} = np.random.rand(a^{[l].shape[0]}, a^{[l].shape[1]}) < keepprob$$

- drop chosen nodes

$$a^{[l]} = np.multiply(a^{[l]}, todrop^{[l]})$$

- scale the activation values $a^{[l]}$ (invert the dropout) in order to not reduce the expected values of $w^{[l+1]}$

$$a^{[l]} = \frac{a^{[l]}}{keepprob}$$

Hyperparameters

- $keepprob$