One-Layer Neural Network as a multi-class Classifier

(c) Marcin Sydow

Table of Contents

Limitations of a single perceptron

Single perceptron can be used as a classifier for maximum of 2 different classes.

Even for 2 classes there are cases that cannot be solved by a single perceptron.

In particular, only linearly separable regions in the attribute space can be distinguished.

Activation function of a neuron

The value of "activation": $\sum_i w_i x_i - \Theta$ (also called "net" value) is used as the argument in the **activation function** that decides the final otuput of the neuron.

There are many choices:

Concerning the neuron's output type:

- discrete (integer): it can be used for classification
- continuous (floating point number): it can be used for regression (or classification too)

Concerning the maximum (activation state) and minimum (non-activation state) values:

- unipolar (discrete: {0,1}, continuous: [0,1])
- bipolar (discrete: {-1,1}, continuous: [-1,1])

Concerning the "shape" of the activation function (step, linear, sigmoidal, etc.)

Examples of most important Activation Functions

Assume X is the input vector, denote $net = \sum_i w_i x_i - \Theta$, ($y$ is the output of neuron)

The most commonly used activation functions:

- sign function: $y = signum(net)$
- "step" function: $y = \lfloor x > 0 \rceil$
- sigmoid function: $y = \frac{1}{1+e^{-net}}$
- linear function ("raw" output): $y = net$

mini-test: which function corresponds to:

Examples of most important Activation Functions

Assume X is the input vector, denote $net = \sum_i w_i x_i - \Theta$, ($y$ is the output of neuron)

The most commonly used activation functions:

- sign function: $y = signum(net)$
- "step" function: $y = \lfloor x > 0 \rceil$
- sigmoid function: $y = \frac{1}{1+e^{-net}}$
- linear function ("raw" output): $y = net$

mini-test: which function corresponds to: continuous/discrete neuron?, unipolar/bipolar neuron?

Examples of most important Activation Functions

Assume X is the input vector, denote $net = \sum_i w_i x_i - \Theta$, ($y$ is the output of neuron)

The most commonly used activation functions:

- sign function: $y = signum(net)$
- "step" function: $y = \lfloor x > 0 \rceil$
- sigmoid function: $y = \frac{1}{1+e^{-net}}$
- linear function ("raw" output): $y = net$

mini-test: which function corresponds to: continuous/discrete neuron?, unipolar/bipolar neuron?

which function is good for classification/regression?

The Sigmoid Function

Unipolar variant: $y = \frac{1}{1+e^{-net}}$
Bipolar variant: $y = \frac{2}{1+e^{-net}} - 1$

The function may have a “steepness” parameter $\lambda \in (0, \infty)$:

$$y = \frac{1}{1 + e^{-\lambda \cdot net}}$$

(the higher the value of λ the “steeper” the function’s graph)

The sigmoid function has some important properties:

- is continuous and increasing
- “amplification” property
- has derivative and its derivative has a particularly convenient form (expressable in the terms of the function itself). This special property is very important for the back-propagation (BP) learning algorithm for multi-layer neural networks.

Multi-class classifier: one-layer NN

Each single perceptron can distinguish max. of 2 classes

In case of more than 2 classes, **a layer** of perceptrons can be used.

Typical architecture is as follows:

- each input is connected to each perceptron
- outputs of particular perceptrons are combined into the aggregated output of the network

Interpreting the output of multi-neuron network

2-class classifier consisting of a single perceptron has one output that could be easily interpreted:

- maximum activation means: “class 1”
- minimum activation means: “class 2”

In case of a multi-class classifier in the form of a layer of neurons, the network has multiple outputs. There are following ways for representing the class decision in such case:

- local
- global

“Local” representation of the output

The number of perceptrons is the same as the number of classes.

Each perceptron is trained to activate for exactly one class.

The “correct” output of such a network is: one perceptron is activated (what indicates the classification decision), other perceptrons are not activated.

“Global” representation of the output

There is no rule for the number of perceptrons (but is usually smaller than in the “local” one).

The classification decision of the network is derived from the outputs of all perceptrons.

Remark on the number of perceptrons in global architecture: since each separate perceptron can output 2 different values, the minimal number of perceptrons to deal with K categories is $log_2 K$.

Usually the local representation is preferred to the global one as it is easier to train (in most cases) and simpler to interprets

Continuous neurons used for classification

Instead of using a layer of discrete neurons with local representation for the classification task, continuous neurons can be used too.

The most common approach is to, similarly as in the discrete case:

- each continuous neuron is trained to maximally activate for "its" class
- the **maximum selector** is used to select the category decision

Such approach is more robust than the discrete, local, since (almost) any output can be interpreted.
(question: what kind of output cannot be interpreted in this approach?)

Evaluation of a Classifier

The following measures of classification performance are used:

- Accuracy
- Precision and Recall (2-class case only)
- F-measure (2-class case only)
- Confusion Matrix (multi-class cases)

Accuracy and its limitations

The simplest mesure of the performance of a classifier is "Accuracy" i.e. the percentage of correct answers on the test set.

Problem: Imagine 2 classes A and B, where A concerns 99% of cases. A *fake* classifier that "classifies" each case always as A has 99% correctness ratio (but actually is useless, e.g. consider a fire detection system)

In addition, if we have a multi-class classification problem we need more perfect tool for measuring or visualising performance.

Confusion matrix

It is a square matrix $K \times K$, where K is the number of classes.

Each row corresponds to one of the *actual categories* of the cases.

Each column corresponds to one of the categories to which an element *is classified by the system* (may be incorrectly).

Each cell (i, j) of the matrix contains the number of cases from the evaluation (or testing) set that actually belong to the category i and were classified as category j.

Example:

classified as ->	a	b	c
a = Iris-setosa	50	0	0
b = Iris-versicolor	0	44	6
c = Iris-virginica	0	5	45

Question: What would be the confusion matrix of an ideal classification?

Confusion matrix

It is a square matrix $K \times K$, where K is the number of classes.

Each row corresponds to one of the *actual categories* of the cases.

Each column corresponds to one of the categories to which an element *is classified by the system* (may be incorrectly).

Each cell (i, j) of the matrix contains the number of cases from the evaluation (or testing) set that actually belong to the category i and were classified as category j.

Example:

classified as ->	a	b	c
a = Iris-setosa	50	0	0
b = Iris-versicolor	0	44	6
c = Iris-virginica	0	5	45

Question: What would be the confusion matrix of an ideal classification? (all the elements on the diagonal)

Evaluation of a 2-category classfier: Precision and Recall

If we have only 2 categories (for now, call them “positive” and “negative”) the two most basic evaluation measures for a classifier are **precision** and **recall** (in short: P and R). These measures come from information retrieval (IR).

Definition

Precision is the ratio of the cases that are actually positive *among* all the cases that were classified as positive by the classifier.

Definition

Recall is the ratio of all the positive cases correctly classified by the classifier to all actually positive cases in the evaluation (or training) set.

Values of P and R are between 0 and 1 (the higher the better).

F-measure

Of course, the higher values of P and R (max. of 1) the better is the classifier.

In practice, usually if we tune the parameters of the classifier to increase precision, recall drops (and the other way round).

Because it is hard to maximise both P and R, there is introduced another evaluation measure that is an aggregation of P and R:
F-measure:

Definition

$F = \frac{2 \cdot P \cdot R}{P + R}$ (harmonic mean of P and R)

Intuitively: it is defined so that if F-measure has “high” value, both P and R cannot be “too low” (i.e. it is very sensitive for low values of P or R)

Example

Consider the following confusion matrix:

classified as $\rightarrow$	positive	negative
positive	40	5
negative	10	45

Precision: $P = \frac{40}{(40+10)} = \frac{4}{5}$

Recall: $R = \frac{40}{(40+5)} = \frac{8}{9}$

F-measure: $F = \frac{2 \cdot \frac{4}{5} \cdot \frac{8}{9}}{\frac{4}{5} + \frac{8}{9}} = \frac{64}{76} = \frac{16}{19}$

Questions/Problems:

- activation functions
- limitations of single Perceptron as a classifier
- multi-class categorization with 1-layer Neural Network
- limitations of 1-layer Neural Network
- evaluation measures for classification
  - confusion matrix
  - Precision, Recall and F-measure

Thank you for attention