4. Machine Learning Overview

4.5 Neural network parameter optimization

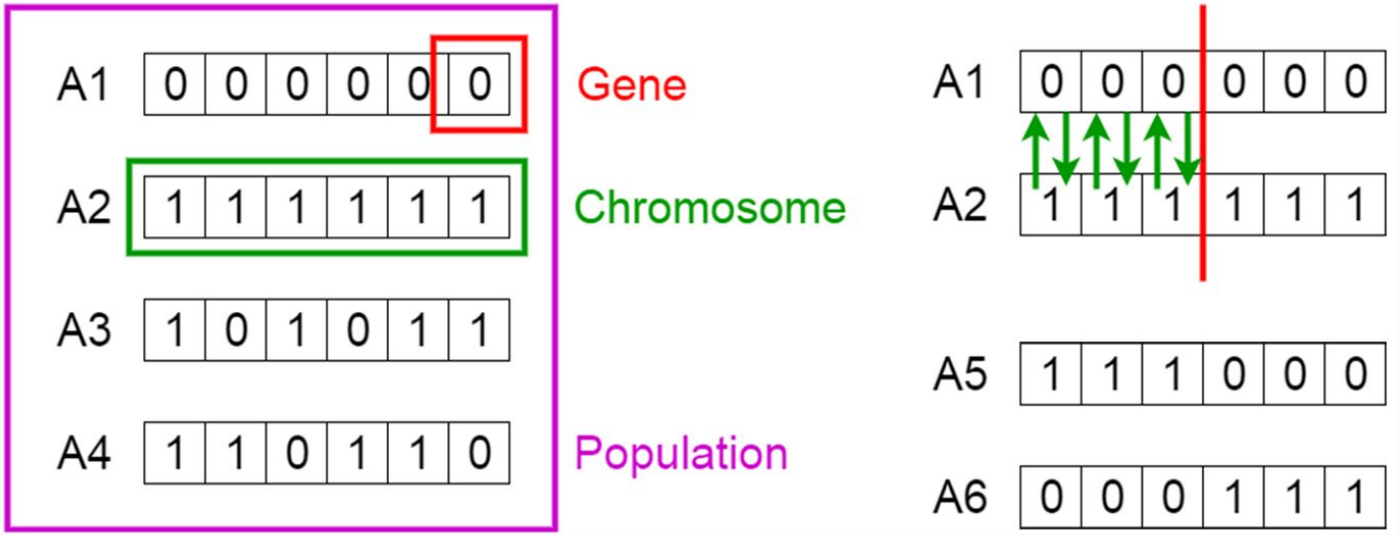
Ricardo Brauner

31 Julho, 2020

Instituto UFC Virtual

Evolutive Algorithms

- Genetic algorithms
- Map weights to chromosomes
- Implement population crossing to generate new solutions



Evolutive Algorithms

- Use error as fitness to implement
- Slower than gradient descent
- Less complex
- Less susceptible to local minima

Perceptron

Uses the current weights to calculate the perceptron output o.

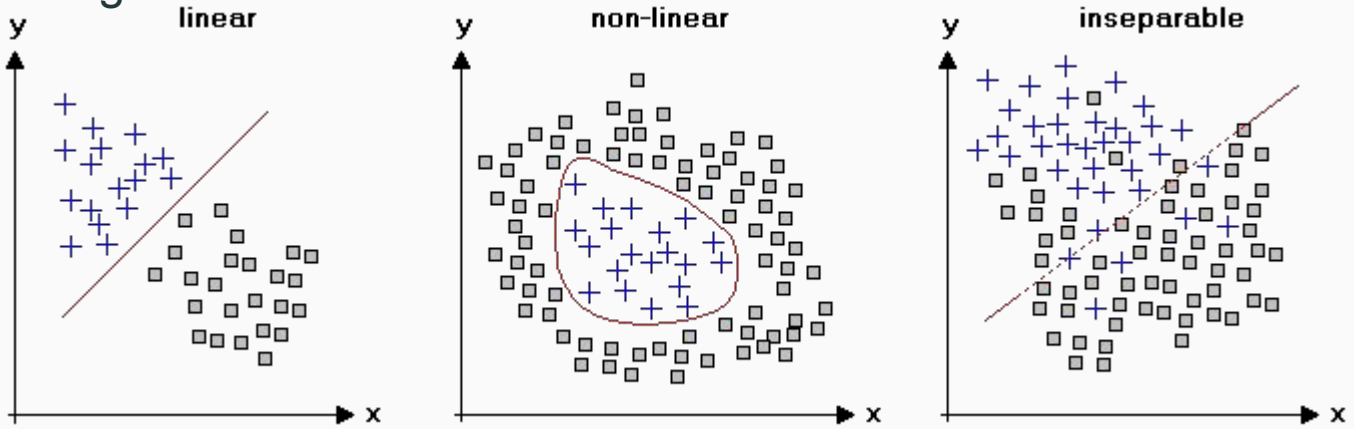
$\omega i \leftarrow \omega i + \Delta\omega i$

$\Delta\omega i = \eta[t - o]xi$

Where X is the input vector, t is the target value, o is the output under the current weights, η is the learning rate, xi and ωi are the i-th elements of vectors X and W.

Perceptron

- When the training sample is linearly separable, the perceptron converges to a classifier that can correctly classify all training samples.
- When the training sample is not linearly separable, the training may be unable to converge.

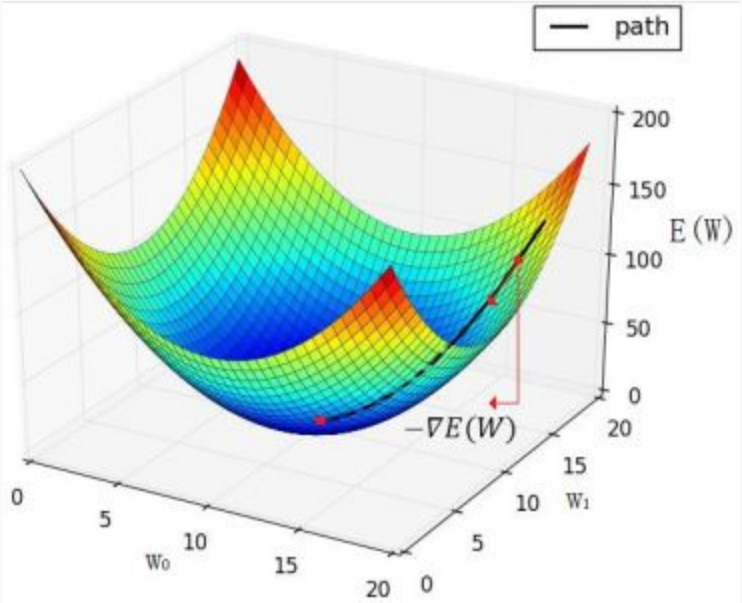


Steepest descent

- The negative gradient vector points to the steepest descent direction.
- When we cannot perfectly classify training samples we can classify them approximately.
- To minimize the errors we need a loss function (error function).
- The function reflects the error between the target output and the actual output of the perceptron.

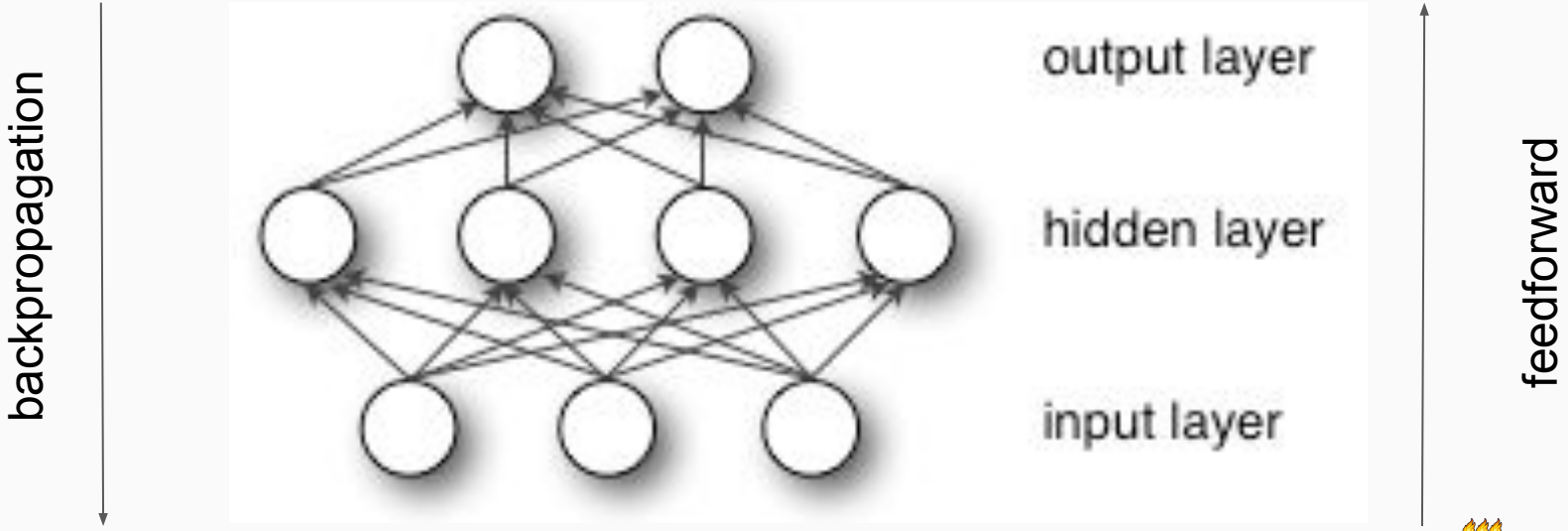
$$E(w) = \frac{1}{2}\sum_{d \in D}(t_d - o_d)^2$$

Steepest descent(2)



Backpropagation

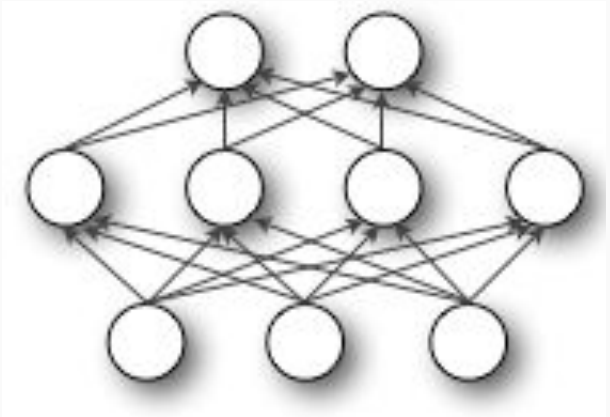
- Estimate error at the output
- Map error to previous layer



Backpropagation

$$\Delta w_{ji} = \eta \delta_j x_{ji}$$

$$\delta_j = \begin{cases} o_j(1-o_j)\,(t_j - o_j), & j \in outputs \\ o_j(1-o_j) \sum_{k \in DS(j)} \delta_k w_{kj}, & otherwise \end{cases}$$



$$o_j = \sigma(net_j) \qquad net_j = \sum_i w_{ji} x_{ji}$$

w_{ji} weight from layer i to j

x_{ji} output from layer i to j

Stop Criteria

- Fixed number of iterations
- Norm of error gradient
- Variation of the quadratic error
- Mean quadratic error

Avoid Overfitting

- Reduce generalization errors
- Prevent overfitting due to diverse parameters
- Constraints to parameters such as norms
- Expanding the training set by adding noise and transformations
- Dropout
- Parameter penalty to objective function

Different algorithms

- ADADELTA
- ADAGRAD
- ADAM
- NESTEROVS
- NONE
- RMSPROP
- SGD
- CONJUGATE GRADIENT
- HESSIAN FREE
- LBFGS
- LINE GRADIENT DESCENT