NEURALNET - Problem Solving Session II

FER

December, 2021

Administrative information about the final exam

- Thursday, January 26^{th}, 2022
- 11:30 - 14:00
- 30 points
- covers **BOTH** the first and the second half of the course materials
- other information regarding the exam can be found on the Neural Net website

Exam preparation resources

- all lecture notes that cover topics from the first and the second cycle
- the chapters of the textbook "Neural Networks - A Comprehensive Foundation" by Simon A. Haykin available on the Neural Net website
- Jupyter notebooks for hands-on sessions (exercise 1/2/3/4)
- problem solving sessions

Hopfield neural network I

General

- works in the same way as associative memory
- has ability to reconstruct the whole memorized pattern when only its fragment or its noisy version is presented to the network

Properties

- recurrent network consisting of only one layer of input-output nodes
- states of discrete Hopfield networks can achieve values $\{-1, 1\}$ or $\{0, 1\}$
- connection strengths or weights are symmetrical
- no self-connections
- states can be updated **asynchronously** and **synchronously**

Hopfield neural network II

Learning phase

Let us assume we want to store a set of p N-dimensional vectors: $\{\xi_m | m = 1, ..., p\}$

$$W = \frac{1}{N}\sum_{m=1}^{p} \xi_m \xi_m^T - \frac{p}{N} I \tag{1}$$

Retrieval phase

- A probe vector x is imposed on the discrete HN as its state.
- The network operates using an updating rule
- Eventually, the network will converge to the stable state which satisfies the following alignment (stability) condition:

$$y = sgn(Wy - \Theta) \tag{2}$$

Hopfield neural network - Example

Consider a Hopfield network made up of four neurons which is required to store the following three fundamental memories:

$$s_1 = [1, 1, -1, -1]^T$$
$$s_2 = [1, -1, 1, -1]^T$$
$$s_3 = [1, 1, 1, -1]^T$$

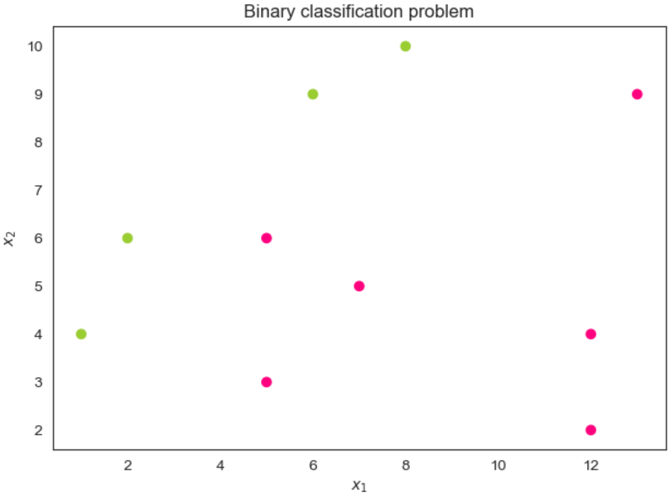
Let threshold vector Θ be zero.

1. Determine synaptic weight matrix W.
2. Show that the states s_2 and s_3 are stable.
3. Find a stable state on which the state s_2 converges by using asynchronous state update.
4. Find a stable state on which the state s_2 converges by using synchronous state update.

AdaBoost algorithm

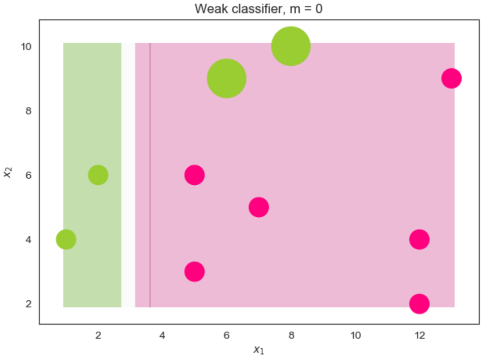
1. Data: $D = \{x, y\}_{i=1}^{N}$
2. Define the initial uniform pdf: $D_1(i) = \frac{1}{N}$ for $i = 1, ...N$
3. Train a classifier $h_t(x)$ using the pdf D_t
4. Calculate estimation error: $\epsilon_t = \sum D_t(i)$ for $i : h_t(x^i) \neq y^i$
5. Calculate α factor: $\alpha_t = \frac{1}{2} ln(\frac{1 - \epsilon_t}{\epsilon_t})$
6. Define new pdf $D_{t+1}(i)$:
$$D_{t+1}(i) = \frac{D_t(i)}{Z_t}. \exp(-\alpha_t) \text{ for } h_t(x^i) == y^i$$
$$D_{t+1}(i) = \frac{D_t(i)}{Z_t}. \exp(\alpha_t) \text{ for } h_t(x^i) \neq y^i$$
7. Repeat steps $3 - 6$ for each classifier
8. Result: $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$

AdaBoost example - binary classification problem



AdaBoost example - weak classifier 1

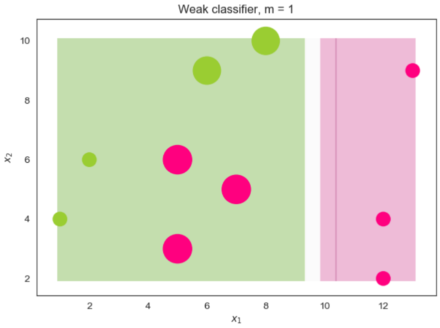
i	$D_1(i)$	$D_2(i)$	$D_2(i)$
1	0.1	0.05	0.0625
2	0.1	0.05	0.0625
3	0.1	0.2	0.25
4	0.1	0.2	0.25
5	0.1	0.05	0.0625
6	0.1	0.05	0.0625
7	0.1	0.05	0.0625
8	0.1	0.05	0.0625
9	0.1	0.05	0.0625
10	0.1	0.05	0.0625
sum	1	0.8	1



$$\alpha_1 = 0.69314$$

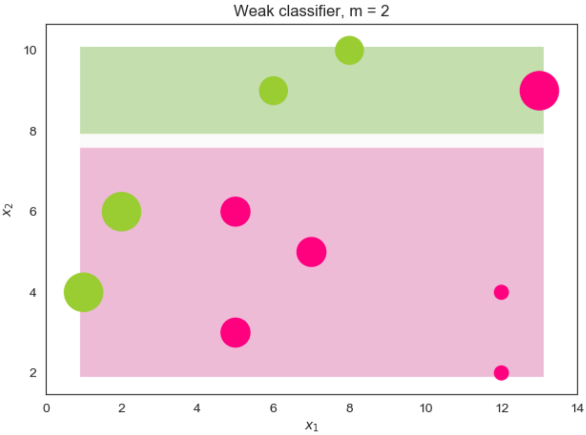
AdaBoost example - weak classifier 2

i	$D_2(i)$	$D_3(i)$	$D_3(i)$
1	0.0625	0.03	0.0385
2	0.0625	0.03	0.0385
3	0.25	0.12	0.159
4	0.25	0.12	0.159
5	0.0625	0.13	0.167
6	0.0625	0.13	0.167
7	0.0625	0.13	0.167
8	0.0625	0.03	0.0385
9	0.0625	0.03	0.0385
10	0.0625	0.03	0.0385
sum	1	0.78	1



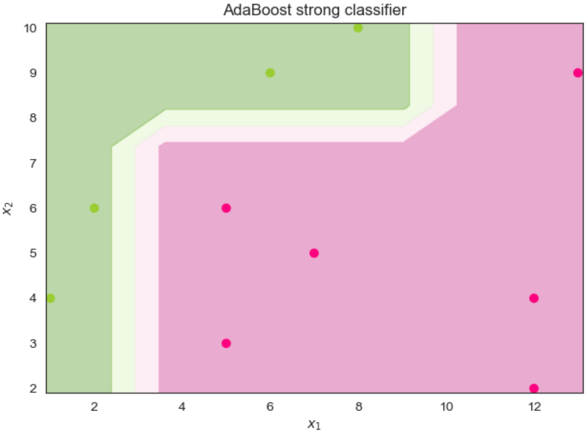
$$\alpha_2 = 0.73317$$

AdaBoost example - weak classifier 3



$$\alpha_3 = 1.0184$$

AdaBoost example - strong classifier



$$H(x) = sign(0.6931 h_1(x) + 0.73317 h_2(x) + 1.0184 h_3(x))$$

PCA - Principal Component Analysis, example

Let $T = \{(x^{(i)}, d^{(i)}), i = 1...N\}$ be the set of datapoints used for learning. $x^{(i)}$ is the feature vector and $d^{(i)}$ is the class label.

$$T = \{([0,0]^T, 0), ([1,1]^T, 0), ([0,2]^T, 0), ([2,0]^T, 0), ([7,5]^T, 1), ([6,6]^T, 1), ([5,7]^T, 1)\}.$$

1. Calculate the covariance matrix of the input data.
2. Reduce the dimensionality of the input data to one dimension by using PCA.