



CS 412 Intro. to Data Mining

Chapter 9. Classification: Advanced Methods

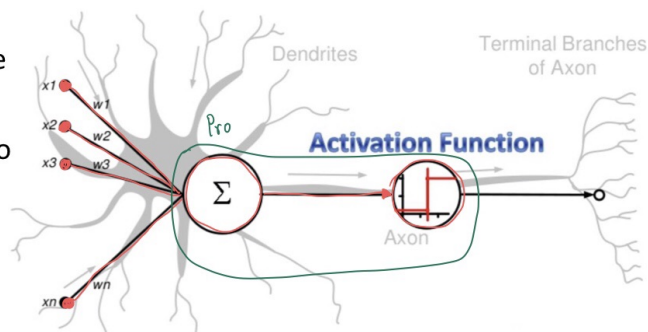
Jiawei Han, Computer Science, Univ. Illinois at Urbana-Champaign, 2017



Artificial +

Neural Network for Classification

- ❑ Started by psychologists and neurobiologists to develop and test computational analogues of neurons
- ❑ A neural network: A set of connected input/output units where each connection has a **weight** associated with it
- ❑ During the learning phase, the **network learns by adjusting the weights** so as to be able to predict the correct class label of the input tuples



Artificial Neural Networks as an analogy of Biological Neural Networks

26

Discussion on the k -NN Algorithm

- ❑ k -NN for real-valued prediction for a given unknown tuple
 - ❑ Returns the mean values of the k nearest neighbors
- ❑ Distance-weighted nearest neighbor algorithm
 - ❑ Weight the contribution of each of the k neighbors according to their distance to the query x_q
 - ❑ Give greater weight to closer neighbors
- ❑ Robust to noisy data by averaging k -nearest neighbors
- ❑ Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes
 - ❑ To overcome it, axes stretch or elimination of the least relevant attributes

1. กำหนดจำนวนเพื่อนบ้าน
2. วัดระยะเพื่อนบ้าน เลือกเพื่อนบ้านที่ใกล้
3. หาร = ผลรวม = ค่าเฉลี่ยเพื่อนบ้าน

$$w \equiv \frac{1}{d(x_q, x_i)^2}$$

ตารางที่ 6-17 อัลกอริทึมการเรียนรู้เพอร์เซปตรอน

Algorithm: Perceptron-Learning-Rule

1. Initialize weights w_i of the perceptron. *สำหรับ Data ใหม่*

2. **UNTIL** the termination condition is met **DO**

2.1 **FOR EACH** training example **DO**

- Input the example and compute the output.
- Change the weights if the output from the perceptron is not equal to the target output using the following rule.

$$w_i \leftarrow w_i + \Delta w_i$$

$\Delta w_i \leftarrow \alpha(t-o)x_i$

where t, o and α are the target output, the output from the perceptron and the learning rate, respectively.

การปรับน้ำหนักตามกฎการเรียนรู้เพอร์เซปตรอนโดยใช้อัตราการเรียนรู้ที่มีค่าน้อยเพียงพอ จะได้ระนาบหลายมิติที่จะเข้าสู่ระนาบหนึ่งที่สามารถแบ่งข้อมูลออกเป็นสองส่วน (ในกรณีที่ข้อมูลสามารถแบ่งได้) เพื่ออธิบายผลที่เกิดจากการปรับค่าน้ำหนัก เราจะลองพิจารณาพฤติกรรมของกฎการเรียนรู้ดูว่าทำไมการปรับน้ำหนักเช่นนี้จึงเข้าสู่ระนาบที่แบ่งข้อมูลได้อย่างถูกต้อง

- พิจารณากรณีที่เพอร์เซปตรอนแยกตัวอย่างสอนตัวหนึ่งที่ได้รับเข้ามาได้ถูกต้อง กรณีนี้จะพบว่า $(t-o)$ จะมีค่าเป็น 0 ดังนั้น Δw_i ไม่เปลี่ยนแปลงเพราะ $\Delta w_i = \alpha(t-o)x_i$
- พิจารณาในกรณีที่เพอร์เซปตรอนให้เอาต์พุตเป็น -1 แต่เอาต์พุตเป้าหมายหรือค่าที่แท้จริงเท่ากับ 1 ในกรณีนี้หมายความว่าค่าที่เราต้องการคือ 1 แต่ค่าน้ำหนักไม่เหมาะสม ดังนั้นเพื่อที่จะทำให้เพอร์เซปตรอนให้เอาต์พุตเป็น 1 น้ำหนักต้องถูกปรับให้สามารถเพิ่มค่าของ $w \cdot x$ ในกรณีนี้หมายความว่าผลรวมเชิงเส้นน้อยเกินไปและน้อยกว่า 0 จึงได้เอาต์พุตเป็น -1 ดังนั้นสิ่งที่เราต้องการคือการเพิ่มค่าผลรวมเชิงเส้นเพราะถ้าเราเพิ่มค่าได้เรื่อยๆ จนมากกว่า 0 เพอร์เซปตรอนจะให้เอาต์พุตเป็น 1 ซึ่งตรงกับที่เราต้องการ พิจารณาดูดังต่อไปนี้ว่าการปรับค่าโดยกฎการเรียนรู้ทำให้ผลรวมเชิงเส้นเพิ่มขึ้นได้อย่างไร กรณีนี้เราจะได้ว่า $(t-o)$ เท่ากับ $(1-(-1))$ มีค่าเป็น 2 และลองพิจารณาค่าของอินพุต x_i แยกกรณีดังนี้

$$f(\sum w_i x_i)$$

- ถ้า $x_i > 0$ จะได้ว่า Δw_i มากกว่า 0 เพราะว่า $\Delta w_i \leftarrow \alpha(t-o)x_i$ และ α มากกว่า 0, $(t-o) = 2$ และ $x_i > 0$ จากสมการการปรับน้ำหนัก $w_i \leftarrow w_i + \Delta w_i$ เมื่อ Δw_i มากกว่า 0 จะทำให้ w_i มีค่าเพิ่มขึ้นและ $\sum w_i x_i$ ก็จะมีค่าเพิ่มขึ้น เมื่อผลรวมมีค่ามากขึ้นแสดงว่าการปรับไปในทิศทางที่ถูกต้องคือเมื่อปรับไปจนกระทั่งได้ผลรวมมากกว่า 0 จะทำให้เพอร์เซปตรอนเอาต์พุตได้ถูกต้องยิ่งขึ้น
- ถ้า $x_i < 0$ เราจะได้ว่า $\alpha(t-o)x_i$ จะมีค่าน้อยกว่า 0 แสดงว่า w_i ตัวที่คูณกับ x_i ที่น้อยกว่า 0 จะลดลงทำให้ $\sum w_i x_i$ เพิ่มขึ้นเหมือนเดิม เพราะ x_i เป็นค่าลบและ w_i มีค่าลดลง ในที่สุดก็จะทำให้เพอร์เซปตรอนให้เอาต์พุตได้ถูกต้องยิ่งขึ้น
- ในกรณีที่เพอร์เซปตรอนให้เอาต์พุตเป็น 1 แต่เอาต์พุตเป้าหมายหรือค่าที่แท้จริงเท่ากับ -1 จะได้ว่า w_i ของ x_i ที่เป็นค่าบวกจะลดลง ส่วน w_i ของ x_i ที่เป็นค่าลบจะเพิ่มขึ้นและทำให้การปรับเป็นไปในทิศทางที่ถูกต้องเช่นเดียวกับในกรณีแรก

6.7.2 ตัวอย่างการเรียนรู้ฟังก์ชัน AND และ XOR ด้วยกฎการเรียนรู้เพอร์เซปตรอน

พิจารณาตัวอย่างการเรียนรู้ของเพอร์เซปตรอนโดยจะให้เรียนรู้ฟังก์ชัน 2 ฟังก์ชัน ฟังก์ชันแรกคือฟังก์ชัน AND แสดงในตารางที่ 6-18 ในกรณีนี้เราใช้ฟังก์ชันไบนารีเป็นฟังก์ชันกระตุ้น

ฟังก์ชัน X | Y label
ตารางที่ 6-18 ฟังก์ชัน AND(x1,x2)

x_1	x_2	เอาต์พุตเป้าหมาย
0	0	0
0	1	0
1	0	0
1	1	1

$$\begin{aligned}
 T \wedge T &\equiv T \\
 T \wedge F &\equiv F \\
 F \wedge T &\equiv F \\
 F \wedge F &\equiv F
 \end{aligned}$$

ฟังก์ชัน AND ตามตารางด้านบนนี้จะให้ค่าที่เป็นจริงก็ต่อเมื่อ x_1 และ x_2 เป็นจริงทั้งคู่ (ดูที่สดมภ์เอาต์พุตเป้าหมาย) ผลการใช้กฎการเรียนรู้เพอร์เซปตรอนกับฟังก์ชัน AND แสดงในตารางที่ 6-19



ตารางที่ 6-19 ผลการเรียนรู้ฟังก์ชัน AND โดยกฎการเรียนรู้เพอร์เซปตรอน

Perceptron Learning Example - Function AND											
		Bias Input x0=+1									
Input	Input				Net Sum	Target	Actual	Alpha*	Weight Values		
x1	x2	1.0*w0	x1*w1	x2*w2	= Input	Output	Output	Error	w0	w1	w2
0	0	0.10	0.00	0.00	0.10	0	1	-0.50	0.1	0.1	0.1
0	1	-0.40	0.00	0.10	-0.30	0	0	0.00	-0.40	0.10	0.10
1	0	-0.40	0.10	0.00	-0.30	0	0	0.00	-0.40	0.10	0.10
1	1	-0.40	0.10	0.10	-0.20	1	0	0.50	0.10	0.60	0.60
0	0	0.10	0.00	0.00	0.10	0	1	-0.50	-0.40	0.60	0.60
0	1	-0.40	0.00	0.60	0.20	0	1	-0.50	-0.90	0.60	0.10
1	0	-0.90	0.60	0.00	-0.30	0	0	0.00	-0.90	0.60	0.10
1	1	-0.90	0.60	0.10	-0.20	1	0	0.50	-0.40	1.10	0.60
0	0	-0.40	0.00	0.00	-0.40	0	0	0.00	-0.40	1.10	0.60
0	1	-0.40	0.00	0.60	0.20	0	1	-0.50	-0.90	1.10	0.10
1	0	-0.90	1.10	0.00	0.20	0	1	-0.50	-1.40	0.60	0.10
1	1	-1.40	0.60	0.10	-0.70	1	0	0.50	-0.90	1.10	0.60
0	0	-0.90	0.00	0.00	-0.90	0	0	0.00	-0.90	1.10	0.60
0	1	-0.90	0.00	0.60	-0.30	0	0	0.00	-0.90	1.10	0.60
1	0	-0.90	1.10	0.00	0.20	0	1	-0.50	-1.40	0.60	0.60
1	1	-1.40	0.60	0.60	-0.20	1	0	0.50	-0.90	1.10	1.10
0	0	-0.90	0.00	0.00	-0.90	0	0	0.00	-0.90	1.10	1.10
0	1	-0.90	0.00	1.10	0.20	0	1	-0.50	-1.40	1.10	0.60
1	0	-1.40	1.10	0.00	-0.30	0	0	0.00	-1.40	1.10	0.60
1	1	-1.40	1.10	0.60	0.30	1	1	0.00	-1.40	1.10	0.60
0	0	-1.40	0.00	0.00	-1.40	0	0	0.00	-1.40	1.10	0.60
0	1	-1.40	0.00	0.60	-0.80	0	0	0.00	-1.40	1.10	0.60
1	0	-1.40	1.10	0.00	-0.30	0	0	0.00	-1.40	1.10	0.60
1	1	-1.40	1.10	0.60	0.30	1	1	0.00	-1.40	1.10	0.60

① กระจ่าง - กระจ่าง
- Target - Actual

② กระจ่าง
- กระจ่าง

0.1 + 0.5 (0 - 1) 0
= 0.10

0.1 + 0.5 (0 - 1) 1
= -0.40

0.1 + 0.5 (1 - 0) 1 = 0.60

-0.4 + 0.5 (1 - 0) 1
= 0.10

กระจ่าง!

ขั้นตอนแรกเริ่มจากการสุ่มค่า w_0 จนถึง w_2 ในที่นี้กำหนดให้เป็น 0.1 ทั้งสามตัว จากนั้นก็เริ่มป้อนตัวอย่างเข้าไป (ที่สีแดง) ตัวอย่างแรกได้ผลรวมเชิงเส้น (Net Sum) เป็น 0.10 ซึ่งมากกว่า 0 ดังนั้นเปอร์เซปตรอนจะให้เอาต์พุตจริง (Actual Output) ออกมาเป็น 1 ซึ่งผิดเพราะเอาต์พุตเป้าหมาย (Target Output) จะต้องได้เป็น 0 ทำให้อัตราการเรียนรู้คูณค่าผิดพลาด (Alpha x Error) ได้ -0.50 หลังจากนั้นก็นำไปปรับน้ำหนักตาม $w_i \leftarrow w_i + \Delta w_i$ และ $\Delta w_i \leftarrow \alpha(t-o)x_i$ ดังนั้นจะได้เป็น $w_0 \leftarrow w_0 + \alpha(t-o)x_0 = w_0 + 0.50(-1) \times 1 = 0.10 + (-0.5) = -0.4$ ต่อไปก็ปรับค่า w_1 ในทำนองเดียวกัน $w_1 \leftarrow w_1 + \alpha(t-o)x_1 = w_1 + 0.50(-1) \times 0$ ดังนั้น w_1 จะเท่ากับ 0.10 คือไม่เปลี่ยนแปลง เช่นเดียวกับ w_2 ที่ไม่เปลี่ยนแปลง จะเห็นได้ว่าแม้มีค่าผิดพลาดแต่ไม่มีการปรับค่า w_1 และ w_2 เนื่องจากอินพุตที่ใส่เข้าไปเป็น 0 ทำ