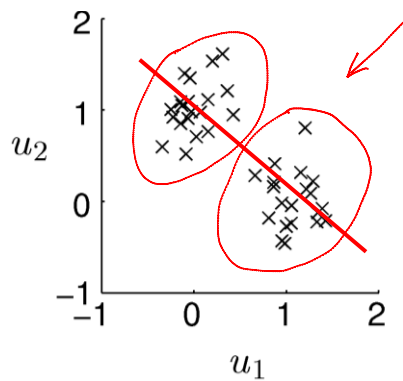


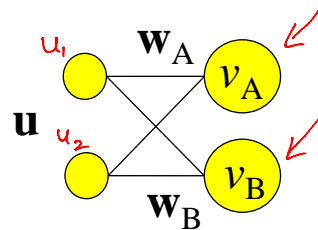
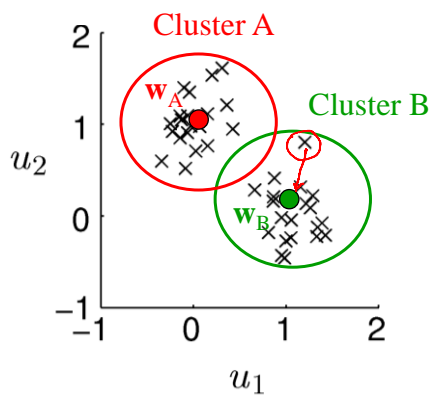
Introduction to Unsupervised Learning



Input data seems to be made up of two clusters of points
Can neurons learn such clusters?

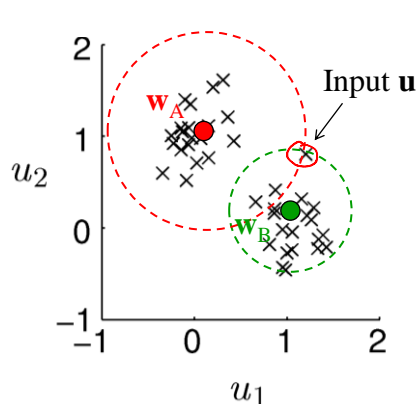
Image Source: Dayan & Abbott textbook

Using Neurons to represent Clusters



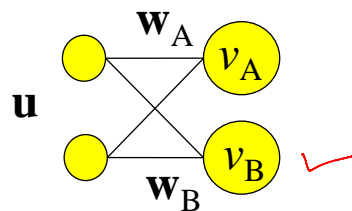
$$v_i = \mathbf{w}_i \cdot \mathbf{u} = \mathbf{w}_i^T \mathbf{u} = \mathbf{u}^T \mathbf{w}_i$$

Most active neuron is the one whose weight vector is closest to an input



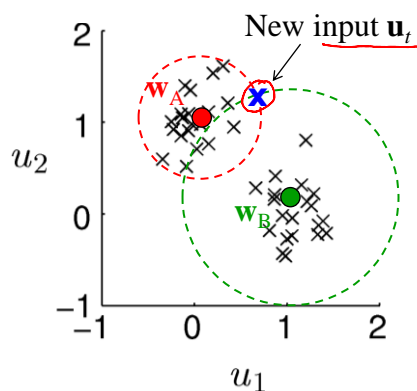
$$\|u\| \approx 1$$

$$\|w_i\| \approx 1$$



$$\begin{aligned} \|u - w_i\|^2 &= (u - w_i)^T (u - w_i) \\ &= u^T u + w_i^T w_i - u^T w_i - w_i^T u \\ &= \|u\|^2 + \|w_i\|^2 - 2v_i \\ \arg \min_i \|u - w_i\|^2 &= \arg \max_i v_i \end{aligned}$$

Updating the Weights given a New Input

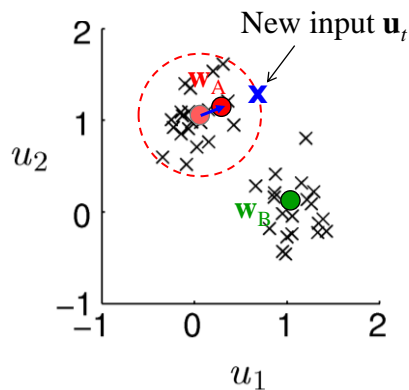


- Given new input, pick a cluster (weight vector closest to input)
- Set weight vector to running average of all inputs u_i in that cluster

$$\begin{aligned} w_A(t) &= \left(\sum_{i=1}^t u_i \right) / t = \left(\sum_{i=1}^{t-1} u_i + u_t \right) / t \\ &= \frac{(t-1)}{t} w_A(t-1) + \frac{u_t}{t} \\ &= w_A(t-1) + (1/t)(u_t - w_A(t-1)) \end{aligned}$$

$$\Delta w_A = \varepsilon \cdot (u_t - w_A)$$

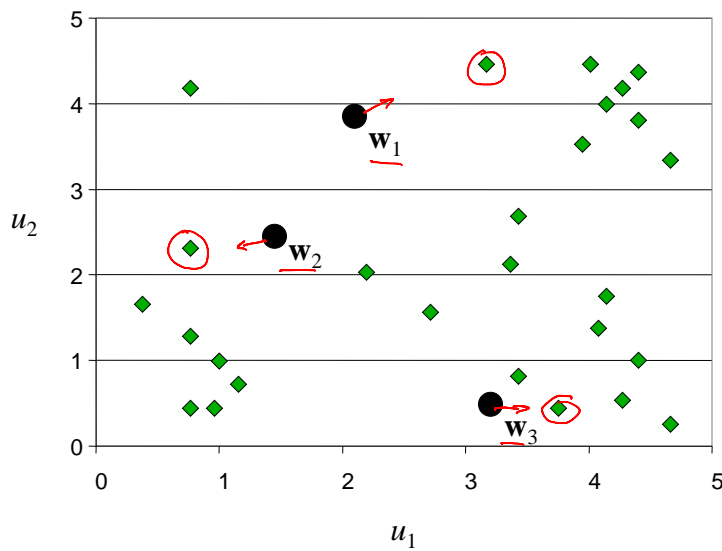
Competitive Learning



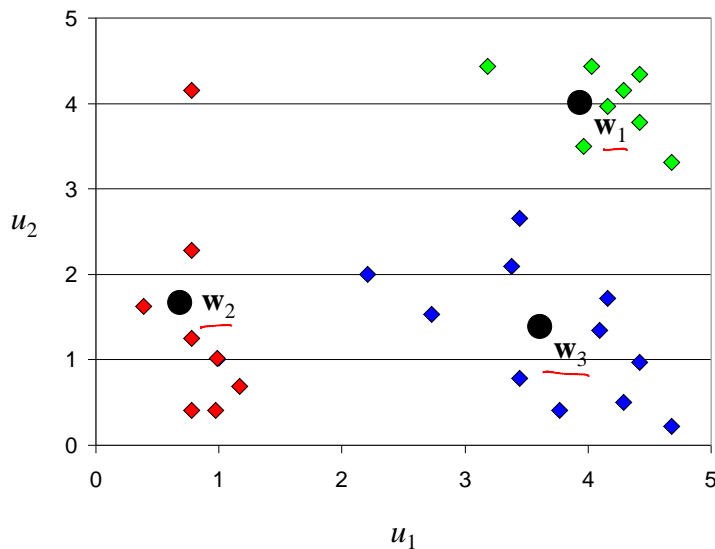
- Given a new input, pick the most active neuron (“winner-takes-all”)
 \Rightarrow One whose weights are closest to new input
- Update weight vector for that neuron

$$\Delta \mathbf{w} = \varepsilon \cdot (\mathbf{u}_t - \mathbf{w})$$

Competitive Learning Example: Initial Weights



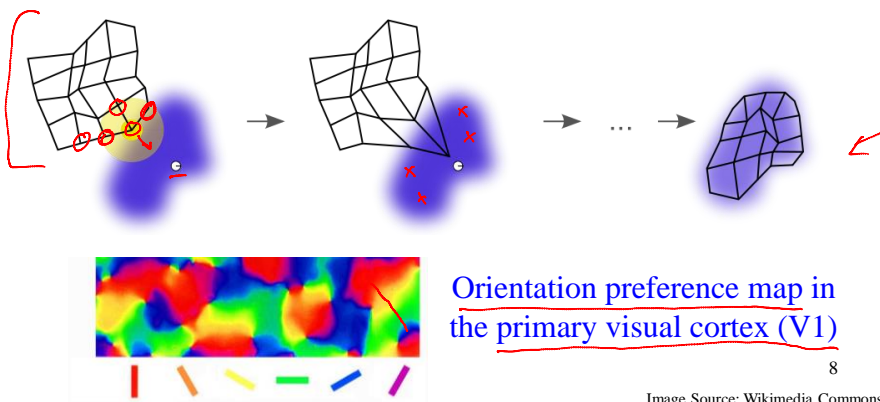
Competitive Learning Example: After Updates



7

Competitive Learning and Self Organizing Maps (a.k.a. Kohonen Maps)

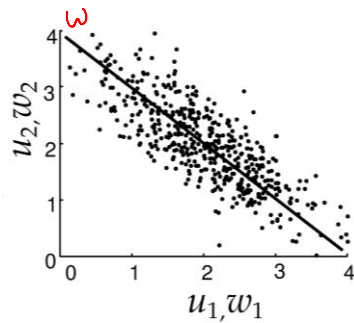
- Given an input, pick the winning neuron
- Update weights for that neuron AND other neurons in the neighborhood of the winner



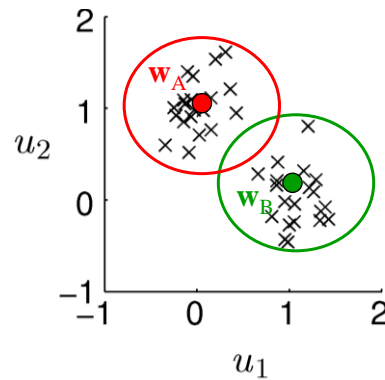
8

Image Source: Wikimedia Commons

PCA/Hebb/Covariance rule



Competitive Learning



Is there a principled way of learning models of input data?

9

Image Source: Dayan & Abbott textbook

Enter... Unsupervised Learning

Causes v

Prior

$$p[v; G]$$

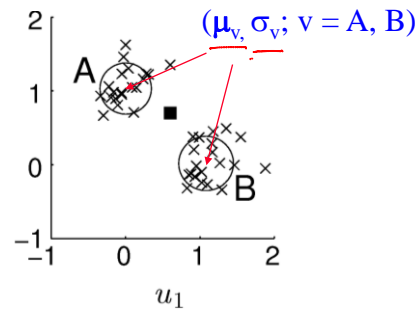
Generative model

Data

points \mathbf{u}

Likelihood

$$p[\mathbf{u} | v; G]$$



Mixture of Gaussians Model:

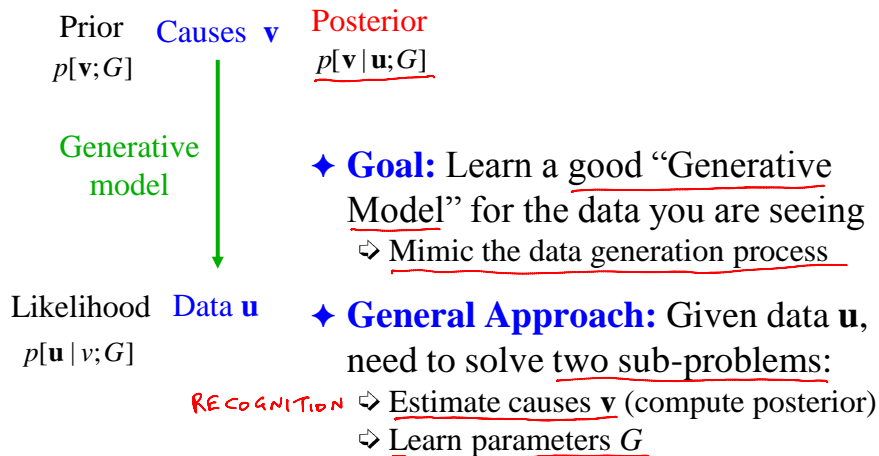
$$p[\mathbf{u}; G] = \sum_v p[\mathbf{u} | v; G] p[v; G]$$

Parameters $G = (\mu_v, \sigma_v, \gamma_v)$

Gaussian γ_v

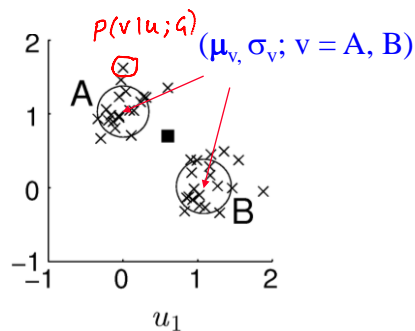
Prior γ_v

The Goal of Unsupervised Learning



11

Clustering as Unsupervised Learning



How do we learn the model parameters
 $G = (\underline{\mu_v}, \underline{\sigma_v}, \underline{\gamma_v})?$

Mixture of Gaussians Model:

$$p[\mathbf{u}; G] = \sum_v p[\mathbf{u} | \mathbf{v}; G] p[\mathbf{v}; G]$$

Gaussian Prior = γ_v

12

EM algorithm for Unsupervised Learning

- ♦ Stands for Expectation-Maximization algorithm E STEP
M STEP

- ♦ *Iterate* the following two steps until convergence:

⇒ E step: Compute posterior distribution of v ($= A, B$) for each \mathbf{u} :

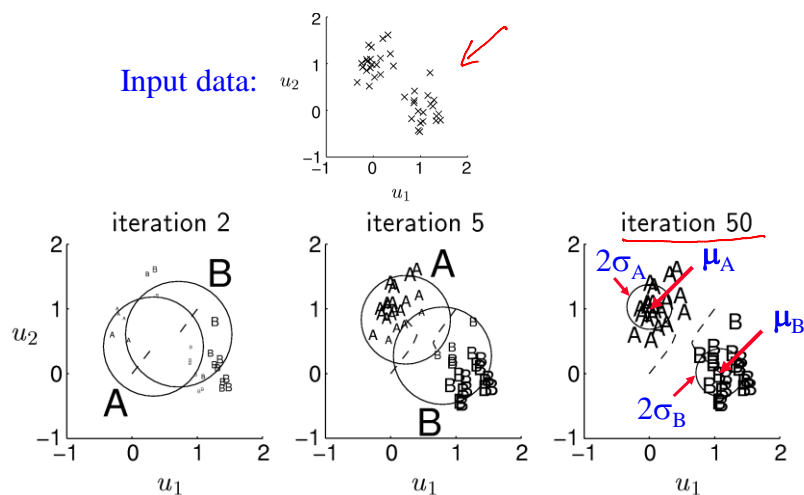
$$p[v | \mathbf{u}; G] = \frac{p[\mathbf{u} | v; G] p[v; G]}{p[\mathbf{u}; G]} = \frac{N(\mathbf{u}; \boldsymbol{\mu}_v, \sigma_v I) \cdot \gamma_v}{\sum_v N(\mathbf{u}; \boldsymbol{\mu}_v, \sigma_v I) \cdot \gamma_v} \quad \begin{array}{l} \text{(Bayes rule)} \\ \text{"Soft" competition} \\ \text{(not winner-takes-all)} \end{array}$$

⇒ M step: Change parameters G using results from E step

$$\begin{aligned} \boldsymbol{\mu}_v &= \frac{\sum_{\mathbf{u}} p[v | \mathbf{u}; G] \cdot \mathbf{u}}{\sum_{\mathbf{u}} p[v | \mathbf{u}; G]}, \quad \sigma_v^2 = \frac{\sum_{\mathbf{u}} p[v | \mathbf{u}; G] |\mathbf{u} - \boldsymbol{\mu}_v|^2}{\sum_{\mathbf{u}} p[v | \mathbf{u}; G]} \quad \text{(Update parameters)} \\ \gamma_v &= \sum_{\mathbf{u}} p[v | \mathbf{u}; G] / N_u \end{aligned}$$

13

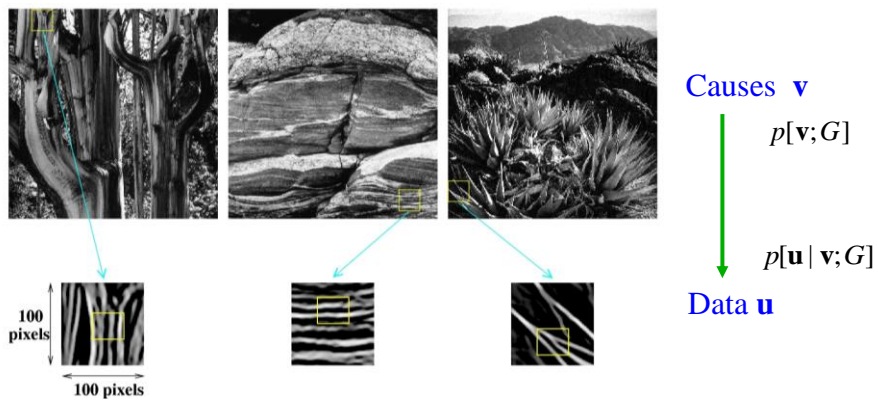
Results from the EM algorithm



14

Image Source: Dayan & Abbott textbook

Suppose the Data are Natural Images...



What kind of generative model would you use?
How do you learn the “causes” of such images?

Image Source: Rao & Ballard, 1999