Mixture models & EM

Sunday, September 29, 2024 9:38 PM

See Chapter 9, Pattern Recognition & Machine Learning

Supervised vs Unsupervised Learning:

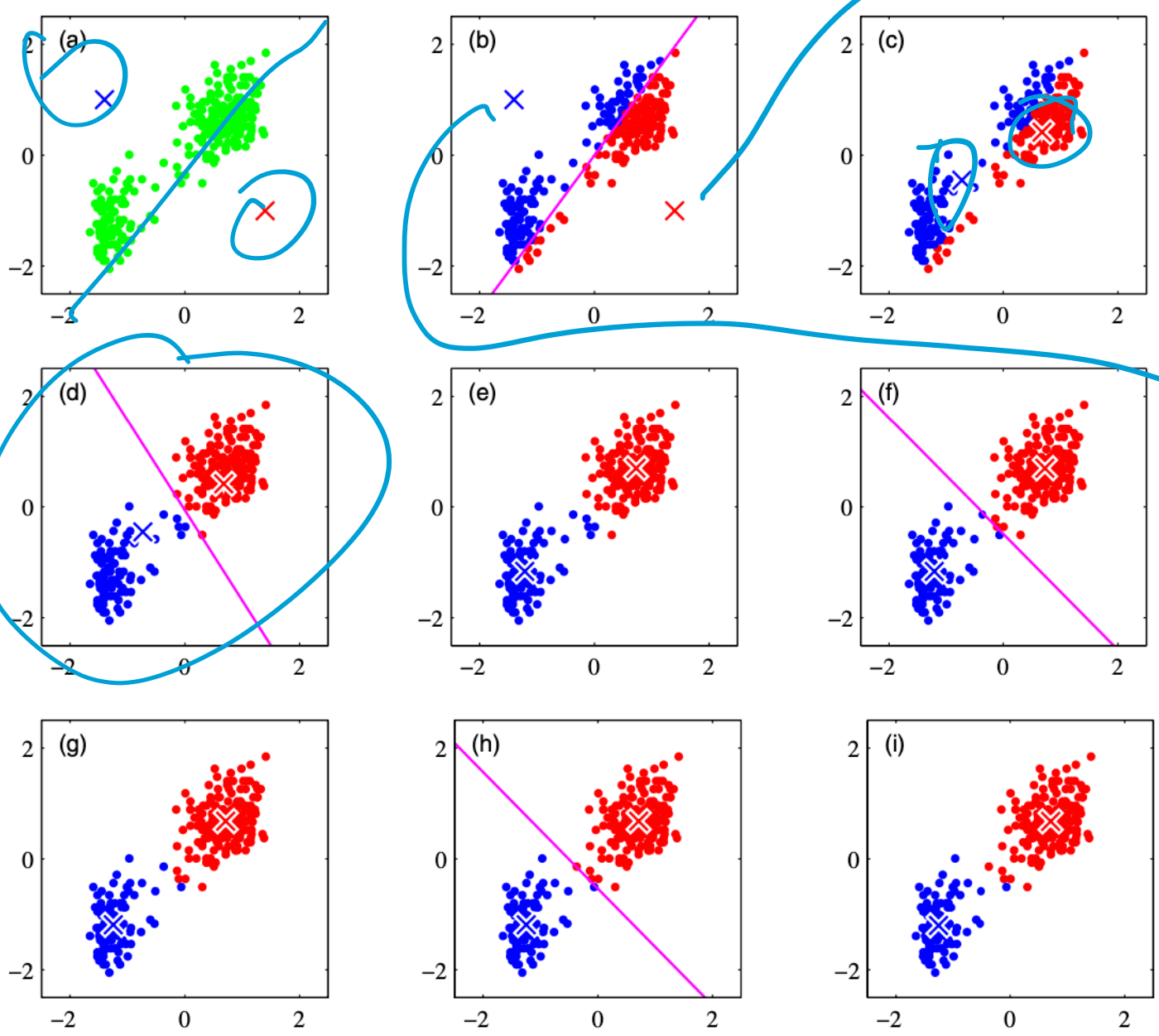
Supervised: Input x. Predict output y.

Unsupervised: Learn about x.

- Which values of x are "good".
- One version of "good"
  = Likely to be seen.
- $p(x)$ is high for training cases, $\therefore$ $p(x)$ is low elsewhere.

K means clustering

Sunday, September 29, 2024 9:41 PM



$\underline{\mu}_1 = \begin{pmatrix} \mu_{11} \\ \mu_{12} \end{pmatrix}$

2D data

$\underline{\mu}_2 = \begin{pmatrix} \mu_{21} \\ \mu_{22} \end{pmatrix}$

Figure 9.1 Illustration of the K-means algorithm using the re-scaled Old Faithful data set. (a) Green points denote the data set in a two-dimensional Euclidean space. The initial choices for centres μ_1 and μ_2 are shown by the red and blue crosses, respectively. (b) In the initial E step, each data point is assigned either to the red cluster or to the blue cluster, according to which cluster centre is nearer. This is equivalent to classifying the points according to which side of the perpendicular bisector of the two cluster centres, shown by the magenta line, they lie on. (c) In the subsequent M step, each cluster centre is re-computed to be the mean of the points assigned to the corresponding cluster. (d)–(i) show successive E and M steps through to final convergence of the algorithm.

k-means error function

Sunday, September 29, 2024 9:44 PM

$$E = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

K clusters / means

kth mean (vector)

vector datapoint

$r_{n1} \cdots r_{nK}$ is a "one hot encoding" of the cluster assignment of $\underline{x}_n$

$\underline{x}_n$ is in cluster 4

Eg, $r_{n1} \cdots r_{nK} = (0, 0, 0, 1, 0)$

Algorithm

Sunday, September 29, 2024 9:50 PM

Initialization

Method 1: Randomly initialize r_{nk}'s.
Proceed to update means.

Method 2: Randomly pick K $\underline{x}$'s, set $\underline{\mu}$'s to those

Iterate to convergence:

For $n=1 \ldots N$ set $r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\underline{x}_n - \underline{\mu}_j\|^2 \\ 0 & \text{otherwise} \end{cases}$

For $k=1 \sim K$ set

$$\underline{\mu}_k = \frac{\sum_{n=1}^{N} r_{nk} \underline{x}_n}{\sum_{n=1}^{N} r_{nk}}$$

Proof Part 1

Sunday, September 29, 2024 9:51 PM

$$E = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

$$\min E = \sum_{k=1}^{K} r_{nk} \|x_n - \mu_k\|^2$$

r_{nj}

Don't use r_{nk}

$$= \begin{cases} 1 \text{ if } j = \arg\min_l \|x_n - \mu_l\|^2 \\ 0 \text{ otherwise} \end{cases}$$

Proof Part 2

Sunday, September 29, 2024 9:52 PM

$$E \not{X} = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

min E: Set $\frac{\partial E}{\partial \mu_j} = 0$ → $\begin{pmatrix} \partial E/\partial \mu_{j1} \\ \partial E/\partial \mu_{j2} \\ \vdots \\ \partial E/\partial \mu_{jD} \end{pmatrix}$

μ_j

$$\frac{\partial E}{\partial \mu_j} = \frac{\partial}{\partial \mu_j} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} (\underline{x}_n - \underline{\mu}_k)^T (\underline{x}_n - \underline{\mu}_k)$$

$\frac{\partial r_{nk}(\|x_n - \mu_k\|^2)}{\partial \mu_j} = 0$ if $k \neq j$

$\underline{x}_n^T \underline{x}_n - 2\underline{x}_n^T \underline{\mu}_k + \underline{\mu}_k^T \underline{\mu}_k$

$$= \sum_{n=1}^{N} r_{nj} \frac{\partial}{\partial \mu_j} \left(\underline{x}_n^T \underline{x}_n - 2\underline{x}_n^T \underline{\mu}_j + \underline{\mu}_j^T \underline{\mu}_j \right)$$

$$= \sum_{n=1}^{N} r_{nj} \left(-2\underline{x}_n + 2\underline{\mu}_j \right) = 0$$

$$\sum_{n=1}^{N} r_{nj} \underline{x}_n = \sum_{n=1}^{N} r_{nj} \underline{\mu}_j \Rightarrow \mu_j = \frac{\sum_{n=1}^{N} r_{nj} \underline{x}_n}{\sum_{n=1}^{N} r_{nj}}$$

Error minimization example

Sunday, September 29, 2024 9:52 PM

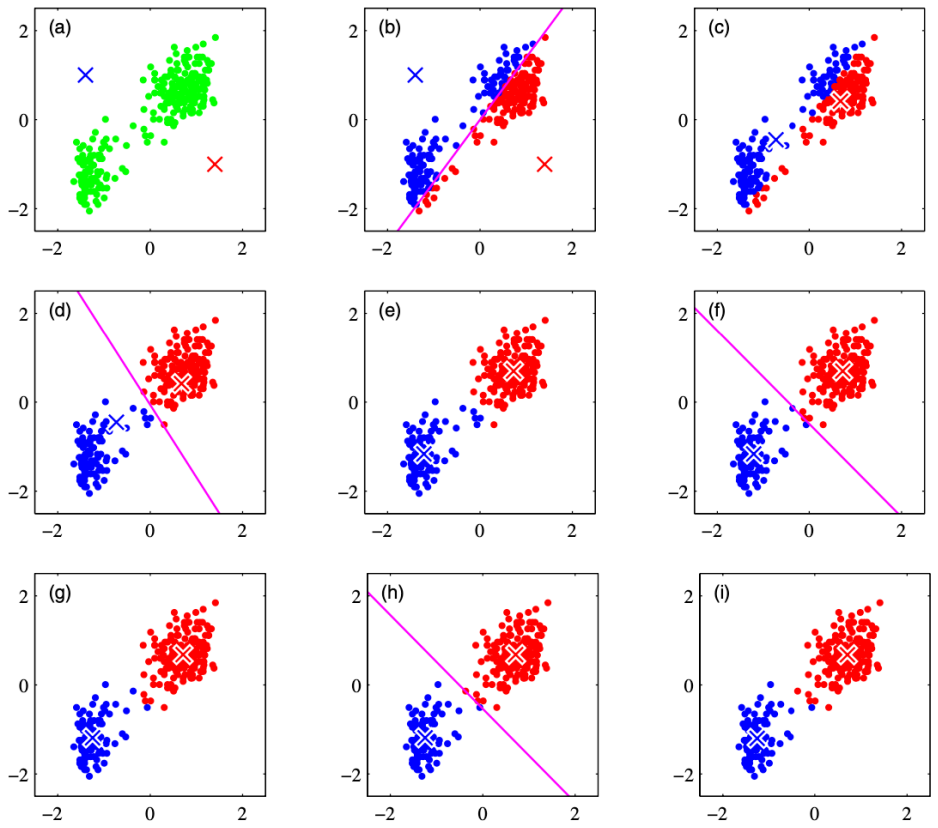
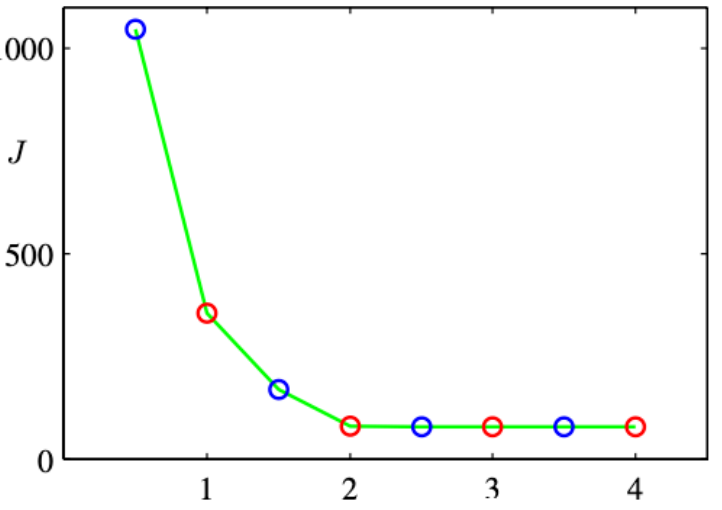


Figure 9.1 Illustration of the K-means algorithm using the re-scaled Old Faithful data set. (a) Green points denote the data set in a two-dimensional Euclidean space. The initial choices for centres $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ are shown by the red and blue crosses, respectively. (b) In the initial E step, each data point is assigned either to the red cluster or to the blue cluster, according to which cluster centre is nearer. This is equivalent to classifying the points according to which side of the perpendicular bisector of the two cluster centres, shown by the magenta line, they lie on. (c) In the subsequent M step, each cluster centre is re-computed to be the mean of the points assigned to the corresponding cluster. (d)–(i) show successive E and M steps through to final convergence of the algorithm.

Plot of the cost function J given by (9.1) after each E step (blue points) and M step (red points) of the K-means algorithm for the example shown in Figure 9.1. The algorithm has converged after the third M step, and the final EM cycle produces no changes in either the assignments or the prototype vectors.



Application

Sunday, September 29, 2024 9:55 PM

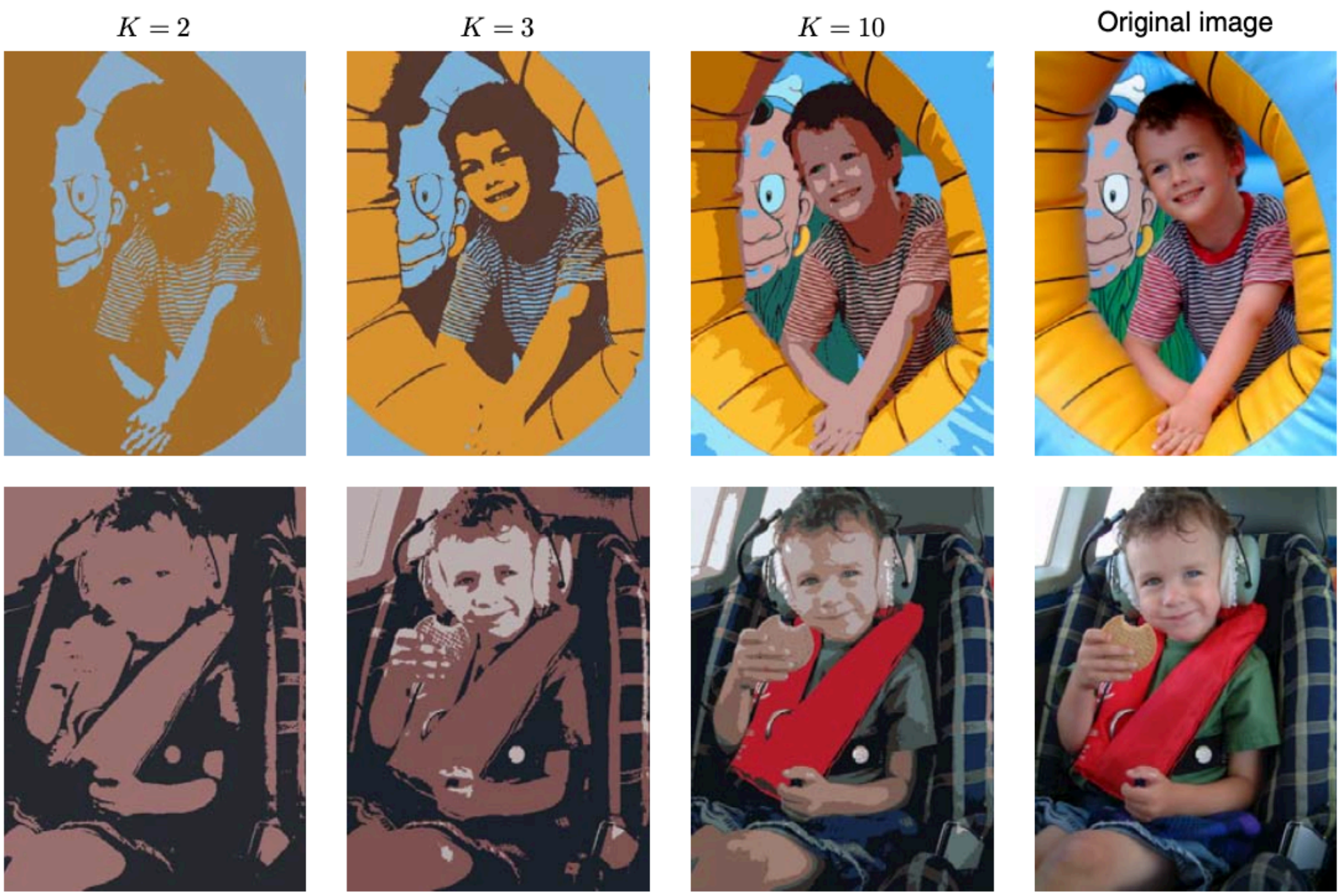
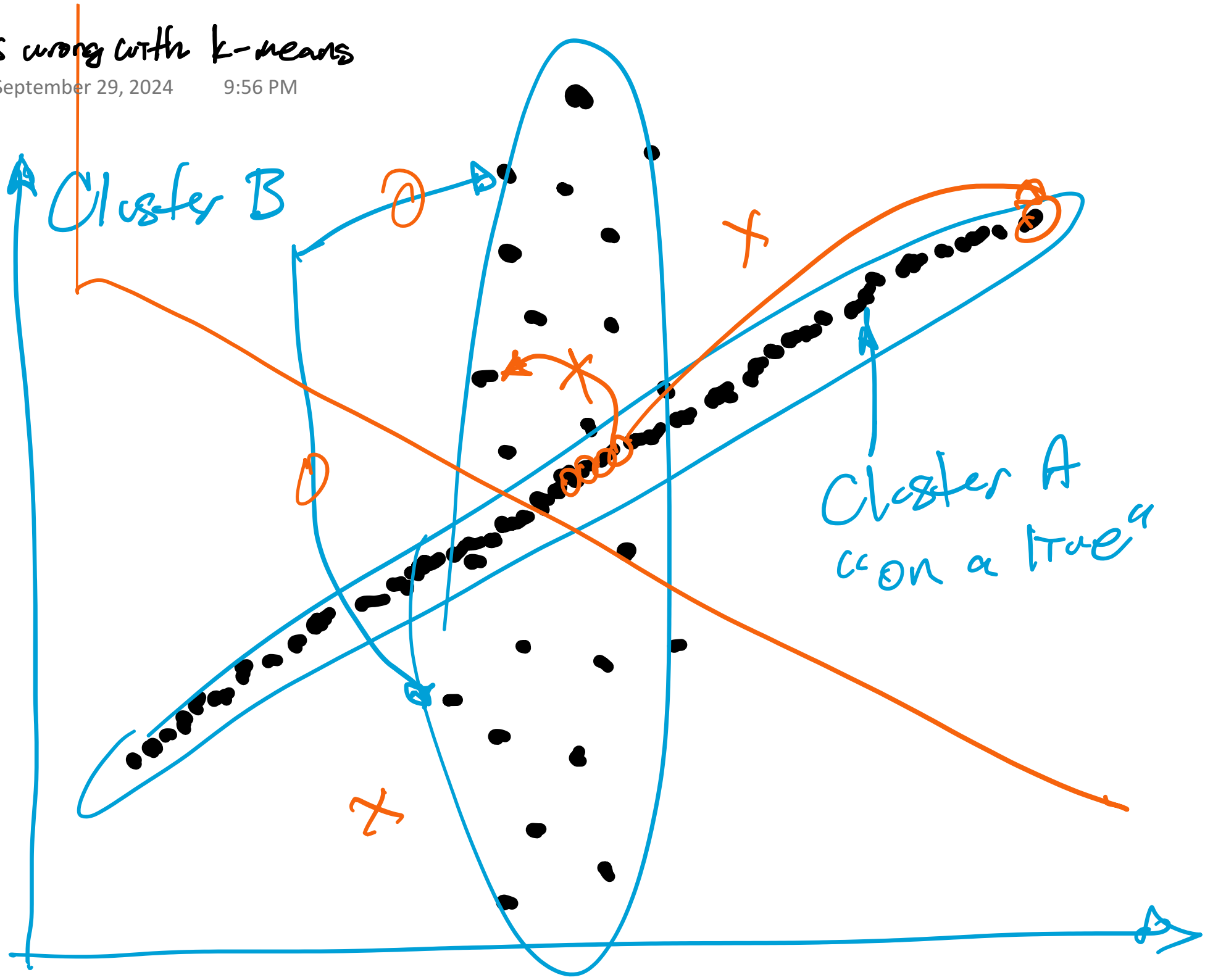


Figure 9.3 Two examples of the application of the K-means clustering algorithm to image segmentation showing the initial images together with their K-means segmentations obtained using various values of K. This also illustrates of the use of vector quantization for data compression, in which smaller values of K give higher compression at the expense of poorer image quality.

What's wrong with k-means

Sunday, September 29, 2024 9:56 PM



Mixture of Gaussians

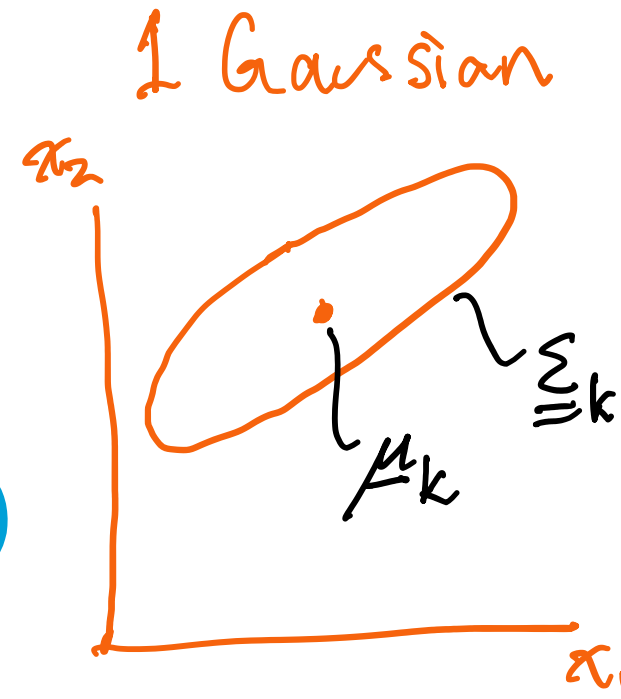
Sunday, September 29, 2024 9:57 PM

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

parameter not 3.14159...

$\mathcal{N}(\mu, \Sigma)$

Gaussian

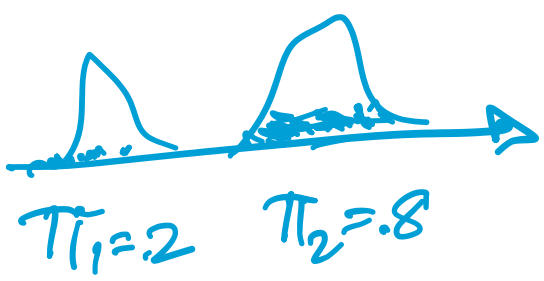


μ_k = mean of cluster k

Σ_k = covariance matrix of Gaussian k

π_k = mixing proportion of Gaussian k

Σ indicates a matrix



2D $\Sigma = \begin{pmatrix} \sigma_{11}^2 & \sigma_{12}^2 \\ \sigma_{12}^2 & \sigma_{22}^2 \end{pmatrix}$

Example: 2D Gaussian

Tuesday, October 1, 2024 10:45 PM

$$\underline{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \quad \underline{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \quad \underline{\underline{\Sigma}} = \begin{pmatrix} \sigma_{11}^2 & \sigma_{12}^2 \\ \sigma_{12}^2 & \sigma_{22}^2 \end{pmatrix}$$

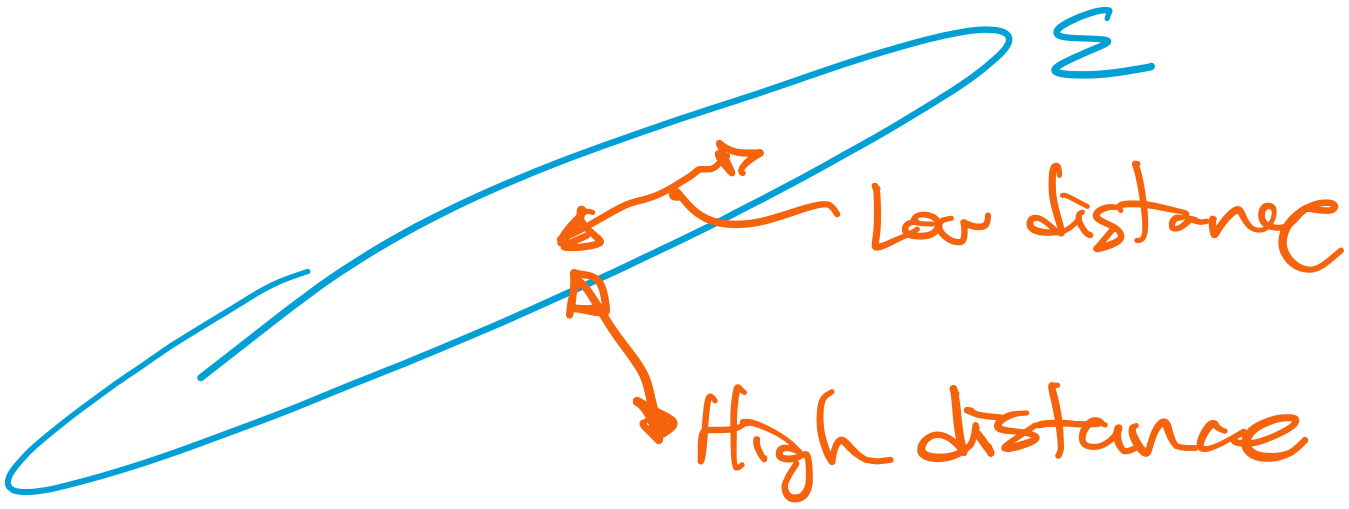
Transpose

Matrix inverse

$$N(\underline{x} \mid \underline{\mu}, \underline{\underline{\Sigma}}) = \frac{1}{|2\pi \underline{\underline{\Sigma}}|^{1/2}} \exp\left(-\frac{1}{2} (\underline{x} - \underline{\mu})^T \underline{\underline{\Sigma}}^{-1} (\underline{x} - \underline{\mu}) \right)$$

Determinant

$$\text{Scalar} = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$



Monday, September 30, 2024 10:36 AM

$D = 5 \Rightarrow$ Dimensionality of data

$$\Sigma = \begin{pmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \sigma_{13}^2 & \sigma_{14}^2 & \sigma_{15}^2 \\ \sigma_{12}^2 & \sigma_{22}^2 & \sigma_{23}^2 & \sigma_{24}^2 & \sigma_{25}^2 \\ \vdots & & & & \\ \sigma_{15}^2 & \sigma_{25}^2 & - & - & \sigma_{55}^2 \end{pmatrix}$$

Σ is separate, like 2D

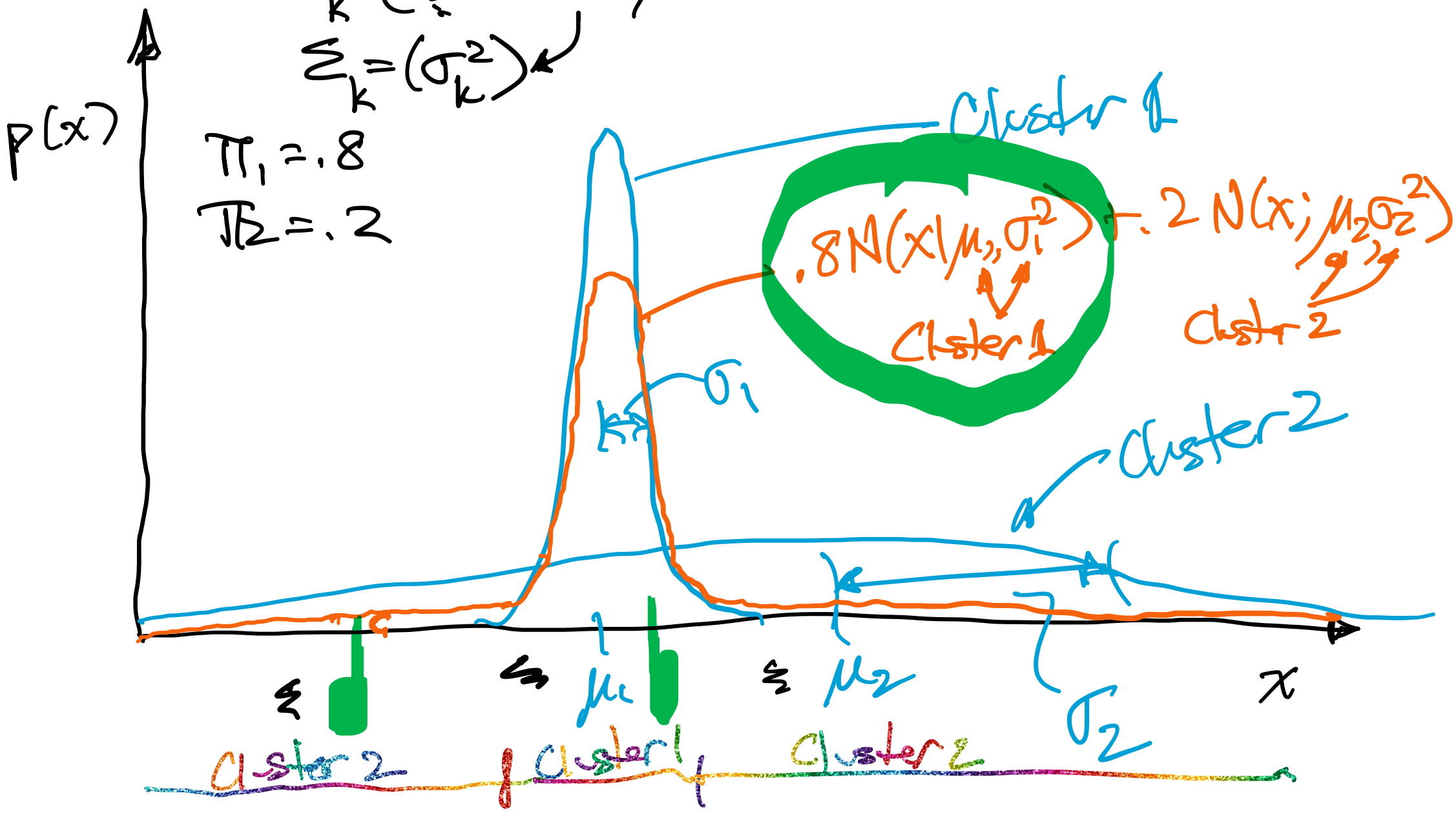
Scalar example (1-D)

Sunday, September 29, 2024 10:06 PM

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \sum_{k=1}^{K} \pi_k \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu_k)^2}{2\sigma^2}}$$

$\Sigma_k = \begin{pmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \cdots & \sigma_{1D}^2 \\ \sigma_{21} & \cdots & & \end{pmatrix}$

$\Sigma_k = (\sigma_k^2)$



Example: Mixture of 3 Gaussians

Monday, September 30, 2024 10:43 AM

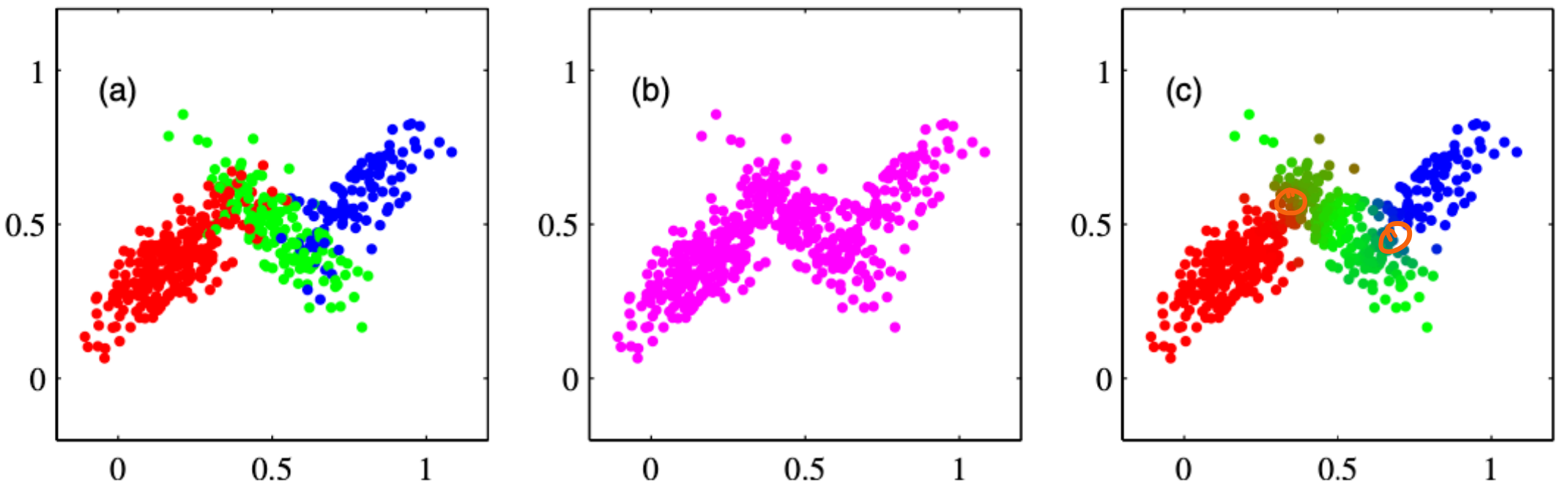


Figure 9.5 Example of 500 points drawn from the mixture of 3 Gaussians shown in Figure 2.23. (a) Samples from the joint distribution $p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$ in which the three states of $\mathbf{z}$, corresponding to the three components of the mixture, are depicted in red, green, and blue, and (b) the corresponding samples from the marginal distribution $p(\mathbf{x})$, which is obtained by simply ignoring the values of $\mathbf{z}$ and just plotting the $\mathbf{x}$ values. The data set in (a) is said to be *complete*, whereas that in (b) is *incomplete*. (c) The same samples in which the colours represent the value of the responsibilities $\gamma(z_{nk})$ associated with data point $\mathbf{x}_n$, obtained by plotting the corresponding point using proportions of red, blue, and green ink given by $\gamma(z_{nk})$ for $k = 1, 2, 3$, respectively

Likelihood of data - MoG

Sunday, September 29, 2024 10:14 PM

variable params

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Data likelihood

$X = \underline{x}_1, \underline{x}_2, \ldots, \underline{x}_N$, $p(X) = \prod_{n=1}^{N} p(\underline{x}_n)$ (IID assumption)

Data log-liketihood

$$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$

Notation: $f(x|\mu)$, we say "density of x given μ"

Relating k-means & MoG

Sunday, September 29, 2024 10:12 PM

k-means error: $E = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$

MoG log-likelihood: ($\log P(X)$) $\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$

Assume that for $\underline{x_n}$ almost $\underline{\text{all}}$ probability is accounted for by one component, denoted by r_{nk}

One-hot encoding

$$\log P(X) \simeq \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \log(\pi_k N(\underline{x}_n | \underline{\mu}_k, \underline{\underline{\Sigma}}_k))$$

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \left[\log \pi_k + \log N(\underline{x}_n | \underline{\mu}_k, \underline{\underline{\Sigma}}_k) \right]$$

$$= -\sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \left[\log \pi_k - \frac{1}{2} \log(2\pi\sigma_k^2) - \frac{1}{2} (x_n - \mu_k)^2 / \sigma_k^2 \right]$$

Scalar case.

Wrap Up

Sunday, September 29, 2024 10:22 PM

- k-means clustering
- Mixture of Gaussians (MoG)
- k-means as narrow case of MoG

Next class

- EM Algorithm for learning the MoG

Announcements

- Midterm, Oct 21, 9am - Noon
- Free days during midterm week
- Assignment 2 coming soon

Learning a MoG

Tuesday, October 1, 2024 9:56 PM

Recall: Prob of $\underline{x}$ under cluster k: $N(\underline{x}|\underline{\mu}_k, \underline{\underline{\Sigma}}_k)$

Total prob of $\underline{x}$: $p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

log P(X): $\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$

MoG: Hard assignment Alg

Tuesday, October 1, 2024 10:15 PM

Hard assignment algorithm:

Like k-means, but assign r's using $N(\cdot)$ instead of $\|\cdot\|^2$

Initialize $\pi_k = \frac{1}{K}$, $\underline{\underline{\Sigma}}_k$ = COV DATA, $\underline{\mu}_k$ = random $\underline{x}$'s.

Iterate until convergence

For $n = 1 \ldots N$ set $r_{nk} = \begin{cases} 1 & \text{if } k = \arg\max_j \pi_j N(\underline{x} | \underline{\mu}_j, \underline{\underline{\Sigma}}_j) \\ 0 & \text{otherwise} \end{cases}$

For $k = 1 \ldots K$, set

$$\pi_k = \frac{1}{N} \sum_{n=1}^{N} r_{nk} \qquad \underline{\mu}_k = \frac{\sum_{n=1}^{N} r_{nk} \underline{x}_n}{\sum_{n=1}^{N} r_{nk}}$$

$$\underline{\underline{\Sigma}}_k = \frac{1}{\sum_{n=1}^{N} r_{nk}} \sum_{n=1}^{N} r_{nk} (\underline{x}_n - \underline{\mu}_k)(\underline{x}_n - \underline{\mu}_k)^T$$

Vector Transpose

Sample covariance

MoG: EM Alg

Tuesday, October 1, 2024 10:23 PM

"Soft assignment" algorithm

Initialize $\pi_k = \frac{1}{K}$, $\underline{\underline{\Sigma}}_k$ = COV DATA, $\underline{\mu}_k$ = random $\underline{x}$'s.

Iterate until convergence:

E-Step For $n=1..N$, for $k=1..K$ set $r_{nk} = \frac{\pi_k N(\underline{x}_n | \underline{\mu}_k, \underline{\underline{\Sigma}}_k)}{\sum_j \pi_j N(\underline{x}_n | \underline{\mu}_j, \underline{\underline{\Sigma}}_j)}$ ← Responsibilities

Note: $\sum_k r_{nk} = 1$.

M-Step

For $k=1..K$, set

$$\pi_k = \frac{1}{N} \sum_{n=1}^{N} r_{nk} \qquad \underline{\mu}_k = \frac{\sum_{n=1}^{N} r_{nk} \underline{x}_n}{\sum_{n=1}^{N} r_{nk}}$$

$$\underline{\underline{\Sigma}}_k = \frac{1}{\sum_{n=1}^{N} r_{nk}} \sum_{n=1}^{N} r_{nk} (\underline{x}_n - \underline{\mu}_k)(\underline{x}_n - \underline{\mu}_k)^T$$

Vector Transpose

Sample covariance

Transforming Data

Wednesday, October 2, 2024 9:55 AM

Example of EM

Tuesday, October 1, 2024 10:29 PM

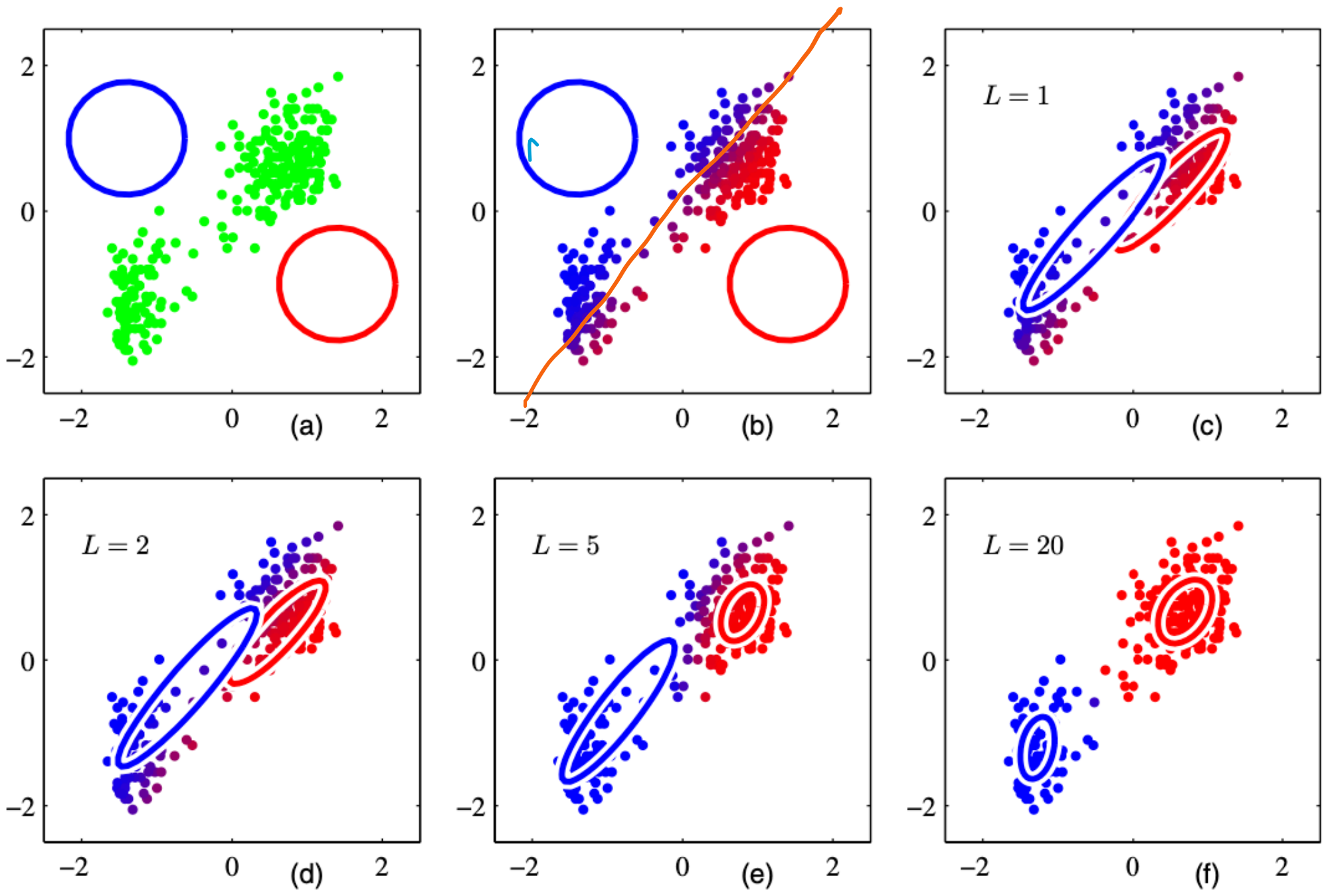
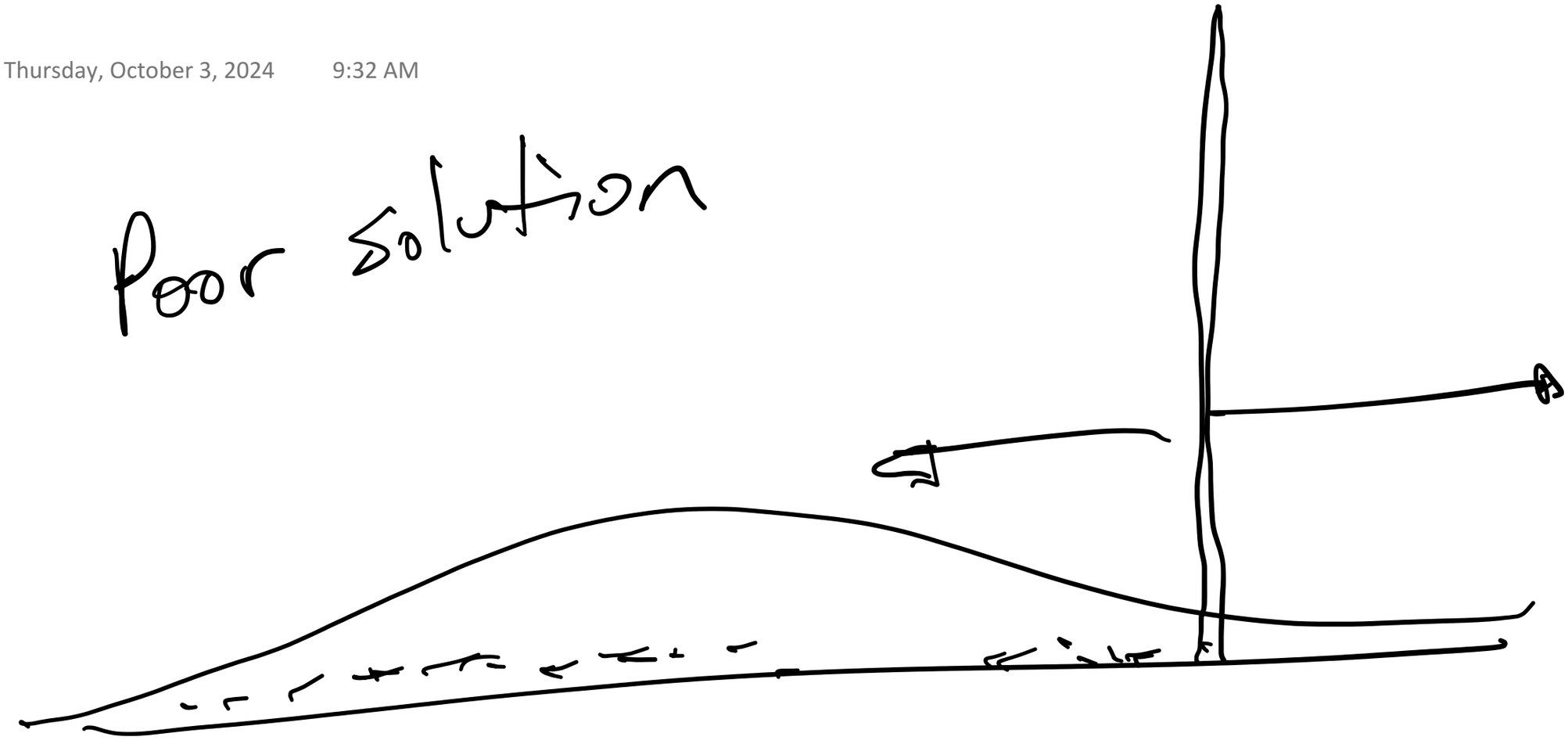


Figure 9.8 Illustration of the EM algorithm using the Old Faithful set as used for the illustration of the K-means algorithm in Figure 9.1. See the text for details.

Thursday, October 3, 2024 9:32 AM

Poor solution



within the Afs:

If $\sum_{n=1}^{N} r_{nk} < 5$ (must have at least 5 dp)

Re-initialize Gaussian k.

Wrap Up

Tuesday, October 1, 2024 10:42 PM

- k-means clustering
- Mixture of Gaussians (MoG)
- k-means as narrow case of MoG
- EM Algorithm for learning the MoG

Announcements

- Midterm, Oct 21, 9am - 11am
- Free days during midterm week
- Assignment 2 coming soon

In todays news !!!

Wednesday, October 9, 2024 2:53 PM

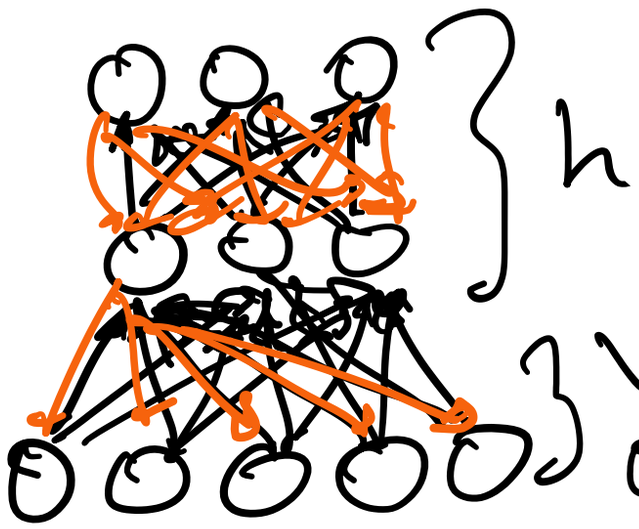
- Geoff Hinton & John Hopfield win Nobel Prize in Physics for their work on neural networks
  - Very relevant to this week's lectures!
  - Big deal for U of T!

- David Baker, Demis Hassabis & John Jumper win Nobel Prize in Chemistry for AlphaFold, the deep learning system for protein structure prediction

My work with Hinton

Wednesday, October 9, 2024 9:18 AM

The Helmholtz Machine



h

v

observed

Recognition net:
Distribution over h: Q(h)

Generative net
Distribution over h,v: P(h,v)

Physics, free energy, $F = \langle \text{Energy} \rangle - \text{Entropy}$

$\text{Energy} = -\log P(h,v)$, $\text{Entropy} = -\sum_h Q(h) \log Q(h)$

$$F = -\sum_h Q(h) \log P(h,v) + \sum_h Q(h) \log Q(h)$$

$$= \sum_h Q(h) \log \frac{Q(h)}{P(h,v)}$$

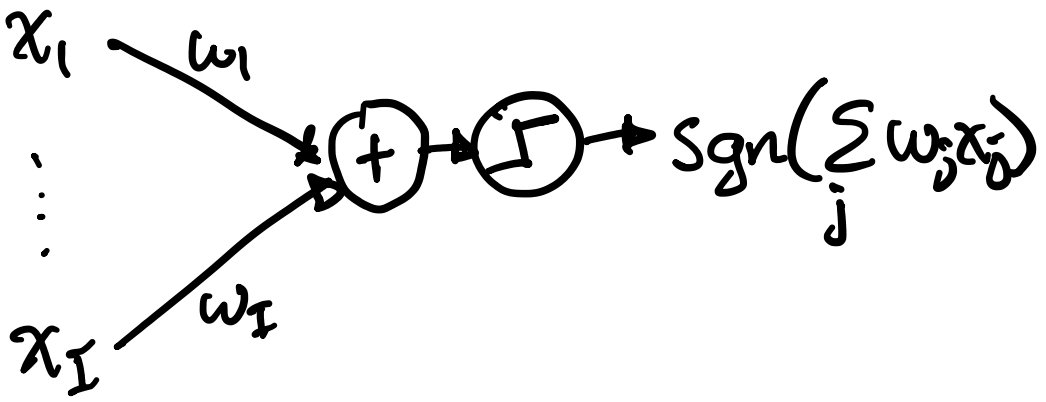
... $F = KL\{Q,P\} - \log P(v)$

$F \geq -\log P(v)$

Deep Learning

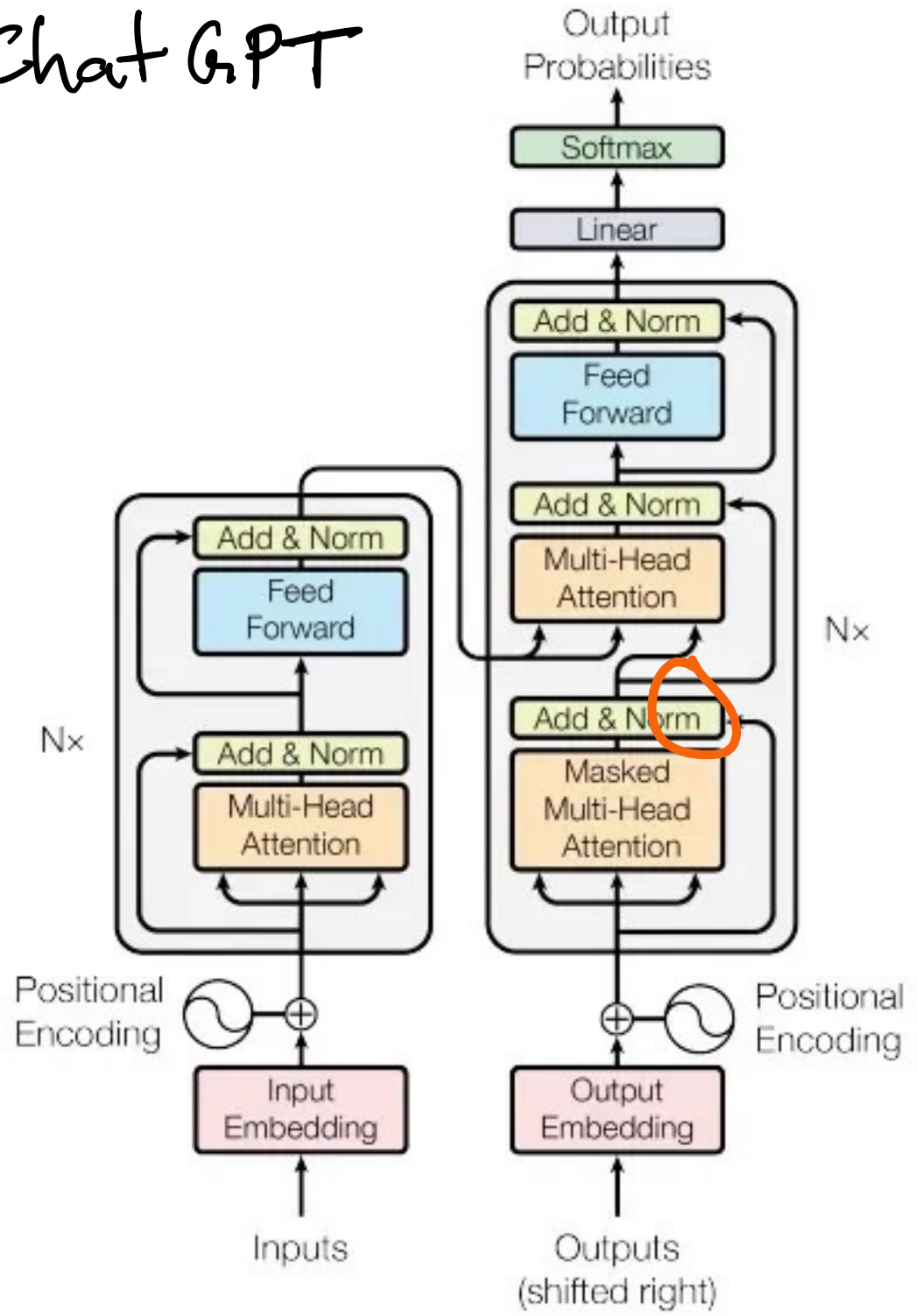
Saturday, October 5, 2024 1:52 PM

A) Perceptron



How do we go from A to B?

B) Chat GPT



Recall Perceptron

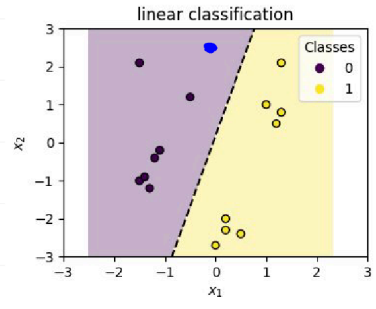
Saturday, October 5, 2024 1:59 PM

- Determine a classification rule

$$y = \text{sign}\left(\sum_{j=0}^{d} w_j x_j\right)$$

to minimize classification error

We will see Perceptron Learning algorithm, logistic regression and Gradient descent



Perceptron Learning Algorithm (PLA)

Input: training set $\mathcal{D}$ that is linearly separable

Output: $\underline{w} \in \mathbb{R}^{d+1}$ that achieves $E_{in}(\underline{w}) = 0$

Initialization: choose arbitrary $\underline{w}$, e.g., $\underline{w} = \underline{0}$

Step 1: check if $E_{in}(\underline{w}) = 0$. If yes, stop and return $\underline{w}$.

Step 2: Let $(\underline{x}_n, y_n)$ be a miss-classified point,
i.e., $y_n \neq \hat{y}_n$ (including the points on the boundary)

If $y_n = +1$, $\underline{w} \leftarrow \underline{w} + \underline{x}_n$

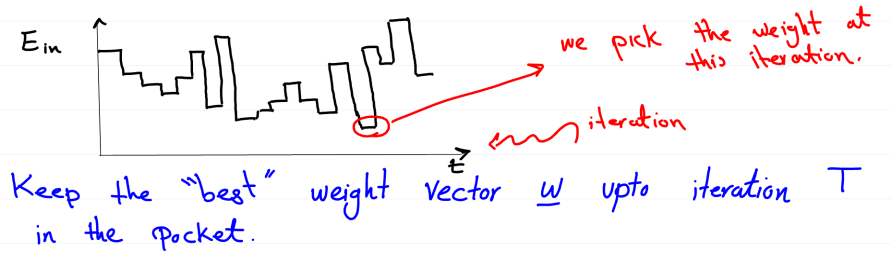
If $y_n = -1$, $\underline{w} \leftarrow \underline{w} - \underline{x}_n$

} $\underline{w} \leftarrow \underline{w} + y_n \underline{x}_n$

Go to Step 1.

Pocket Algorithm

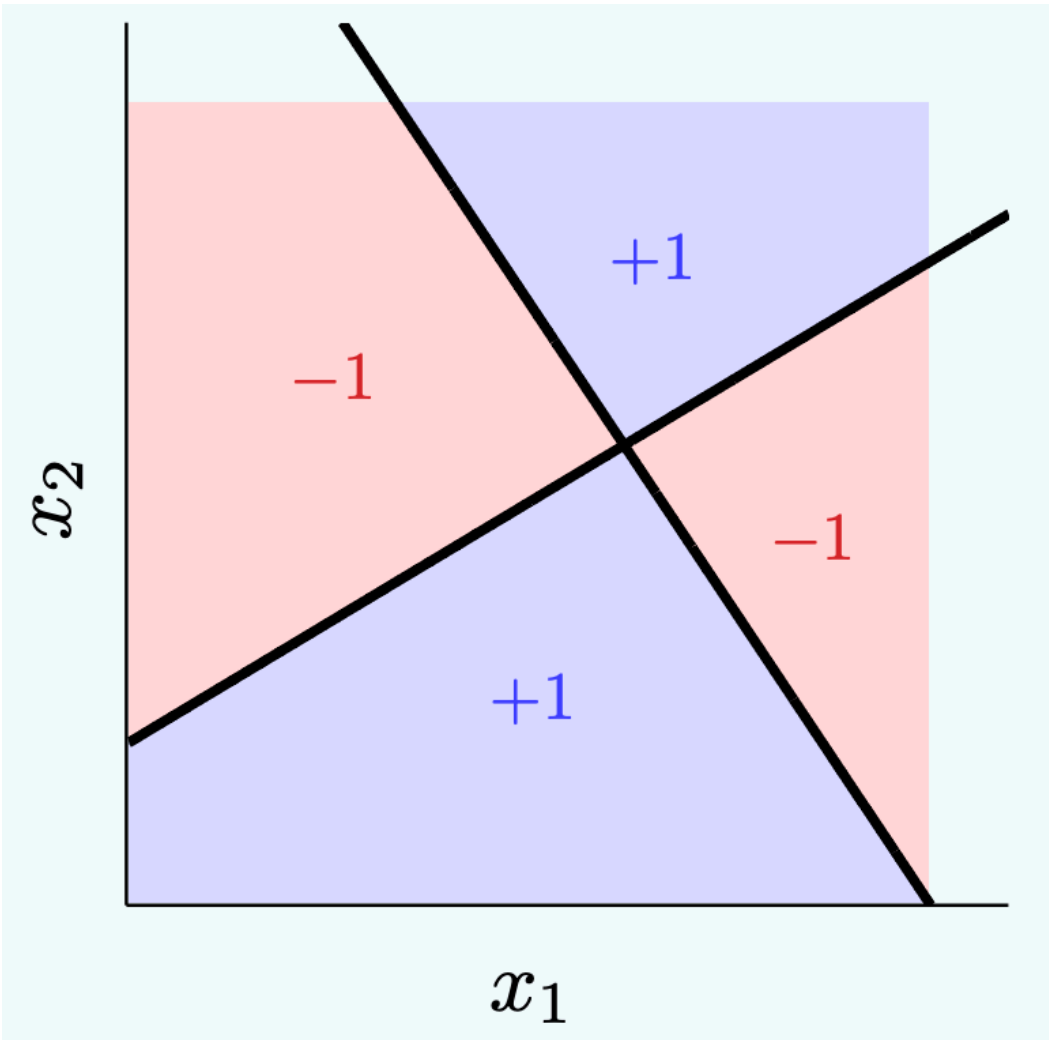
- Pocket Algorithm extends PLA for dataset that are not linearly separable.



Keep the "best" weight vector $\underline{w}$ upto iteration T in the pocket.

Limitation of Perceptron

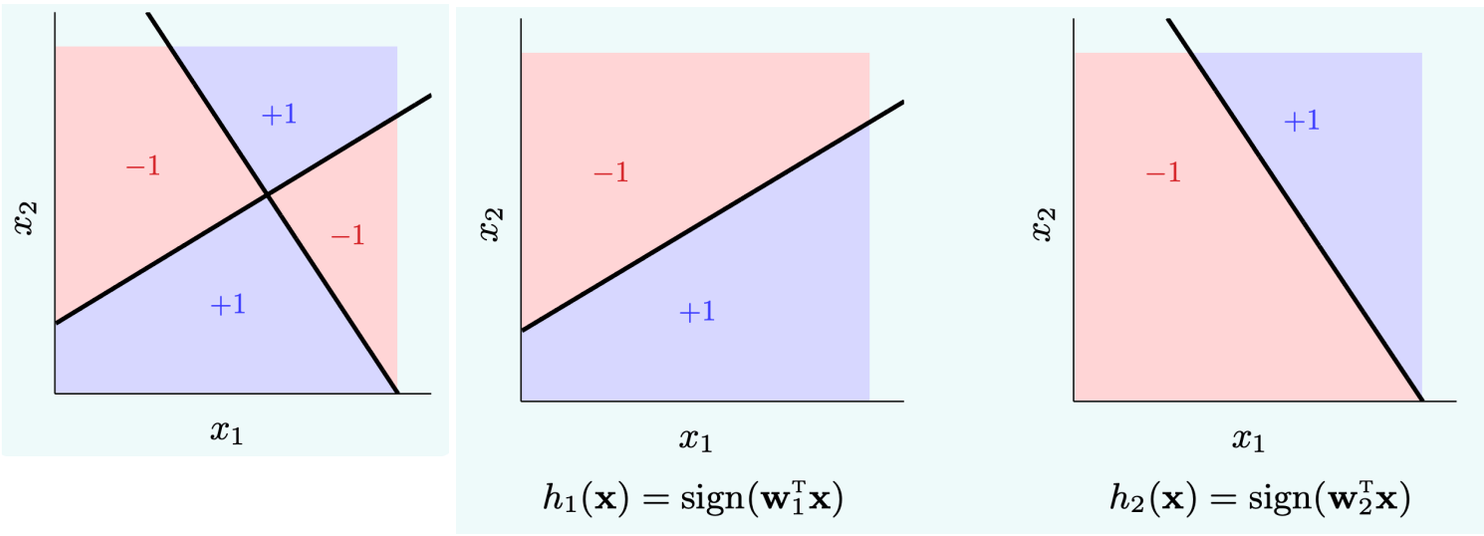
Saturday, October 5, 2024 2:07 PM



Problem!

Decomposing a complex problem

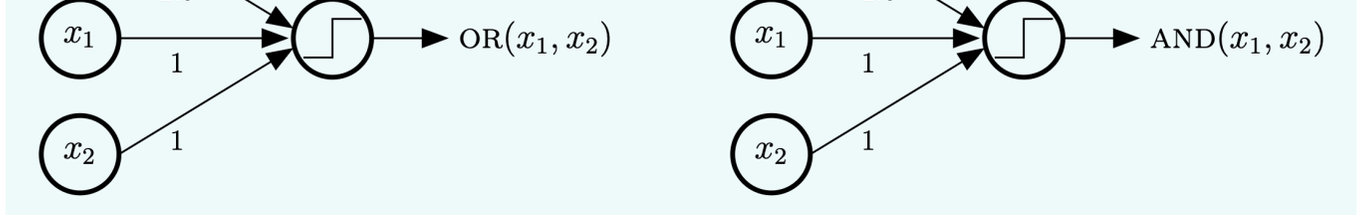
Saturday, October 5, 2024 2:11 PM



AND NOT OR

$$f = h_1\overline{h_2} + \overline{h_1}h_2$$

Implementing OR, AND

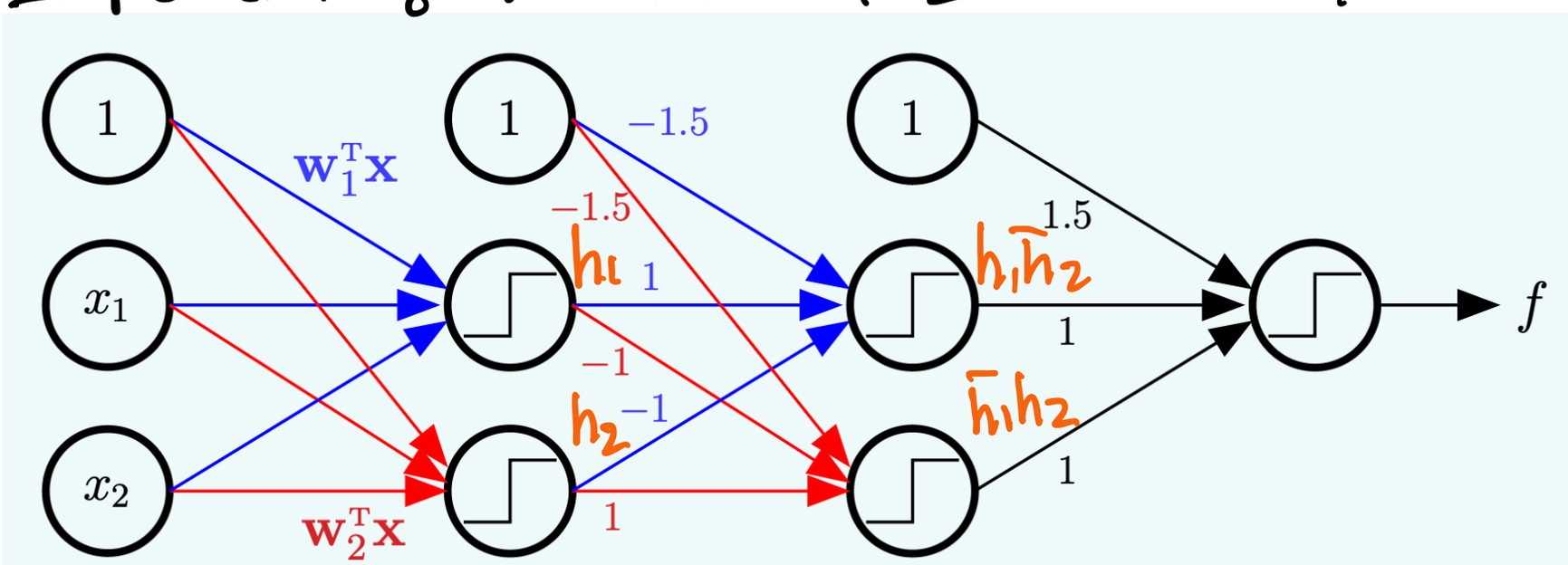


False True

$x_1 \in \{-1, 1\}$

$x_2 \in \{-1, 1\}$

Implementing $f = h_1\bar{h}_2 + \bar{h}_1 h_2$ "Multilayer Perceptron"



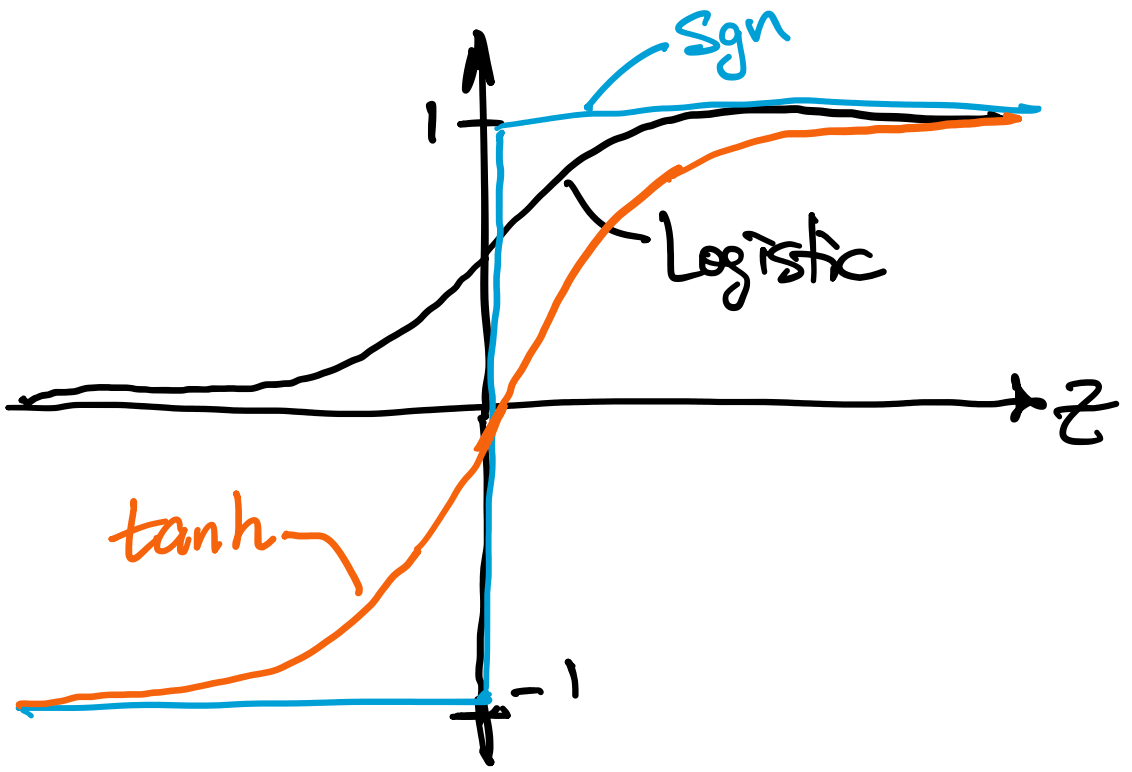
Differentiable activation

Saturday, October 5, 2024 3:22 PM

- Recall that for linear/logistic regression, k-means clustering, MoG, etc, we <u>differentiated</u> the objective function to derive a <u>learning algorithm</u>.
- $\text{sgn}(\underline{w}^T \underline{x})$ is <u>not</u> differentiable
- Introduce a continuous activation function

Logistic: $g(z) = \frac{1}{1+e^{-z}}$

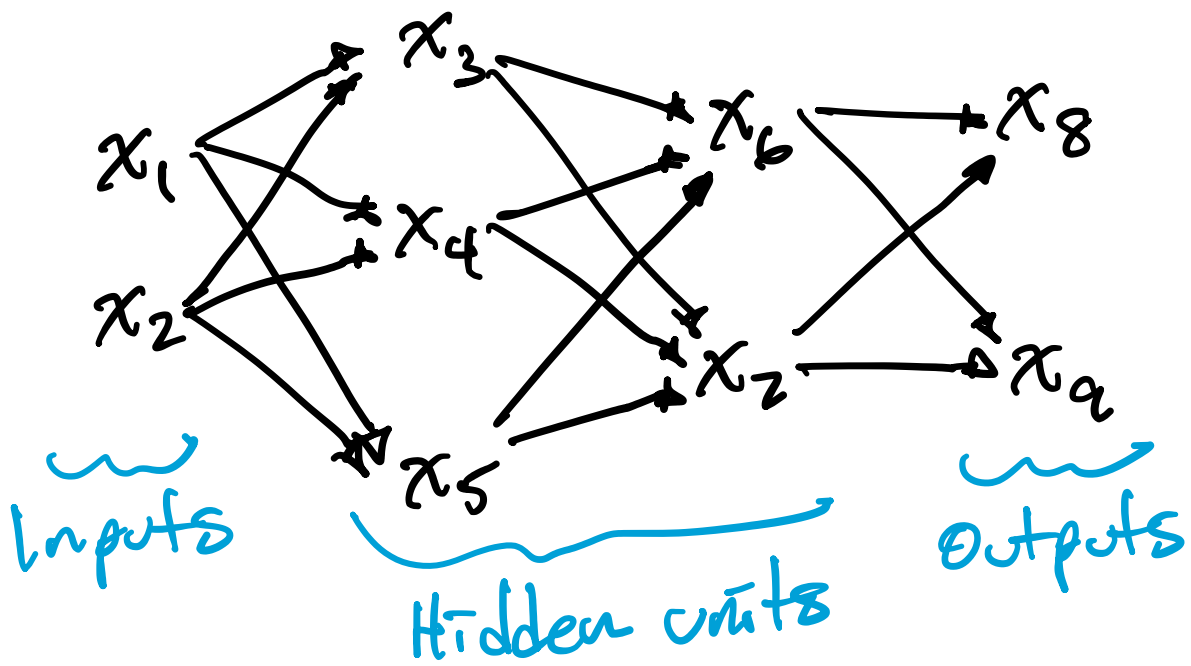
Tanh: $g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$



Neural network

Saturday, October 5, 2024 4:32 PM

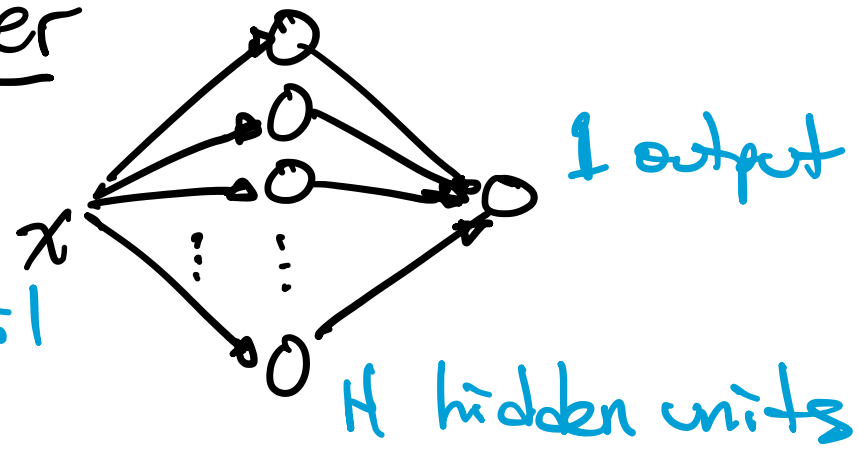
Example



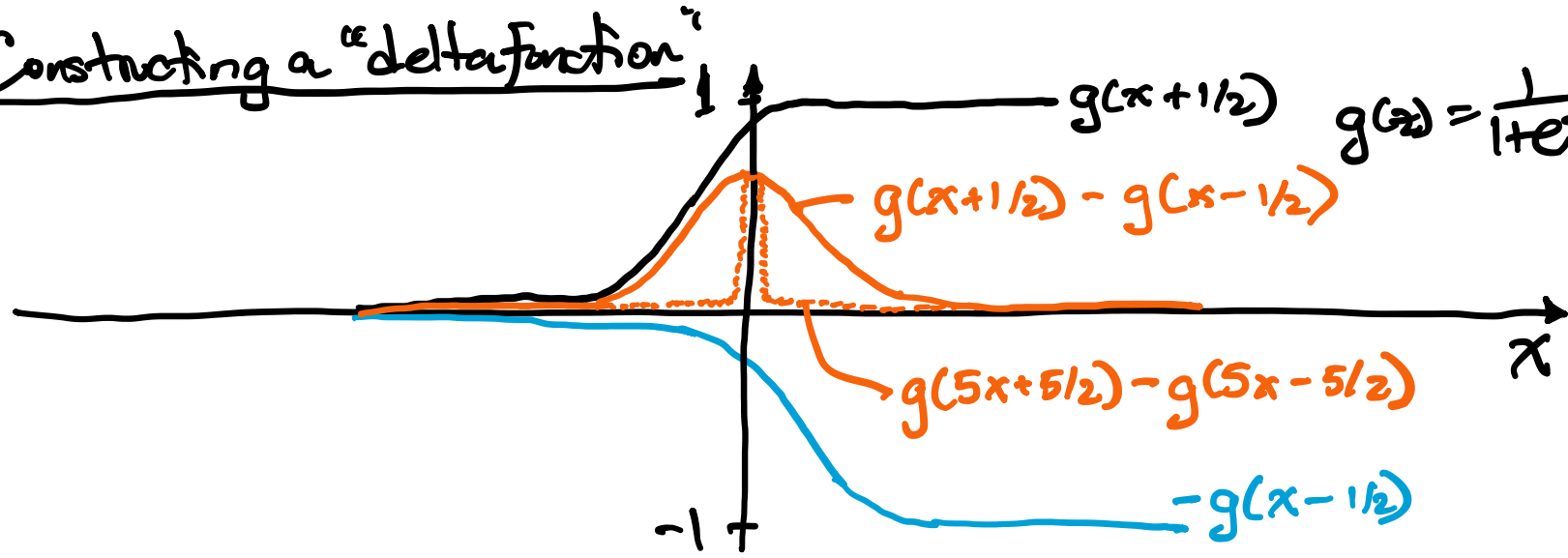
Arbitrary 1-D fn approximation

Saturday, October 5, 2024 3:40 PM

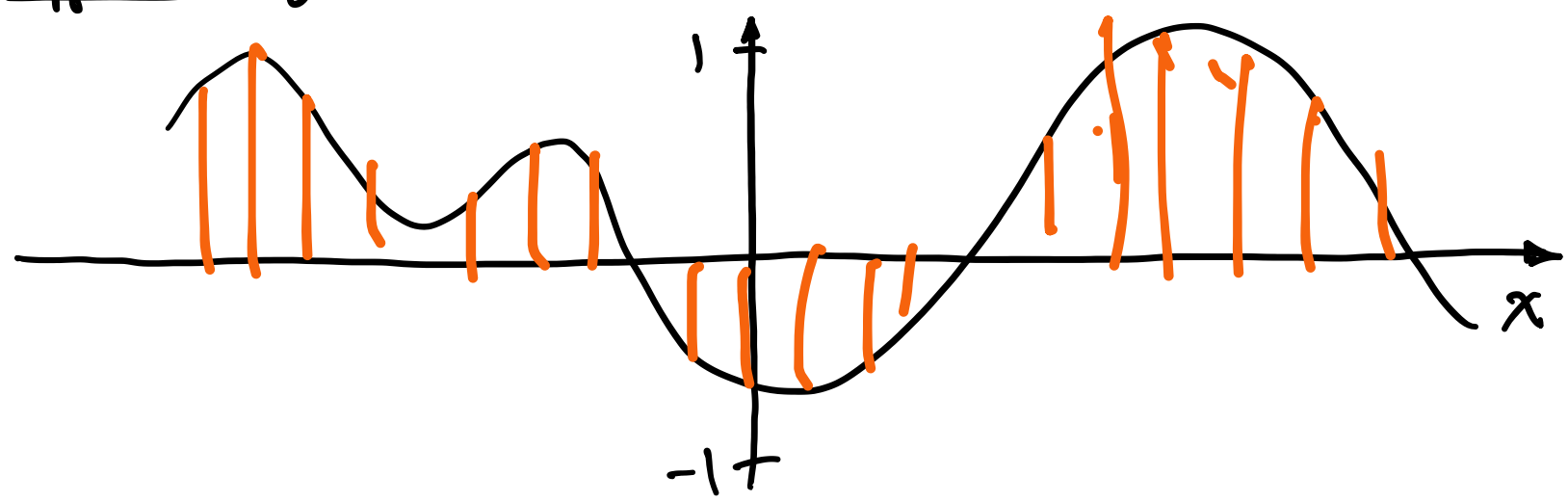
One hidden layer



Constructing a "delta function"

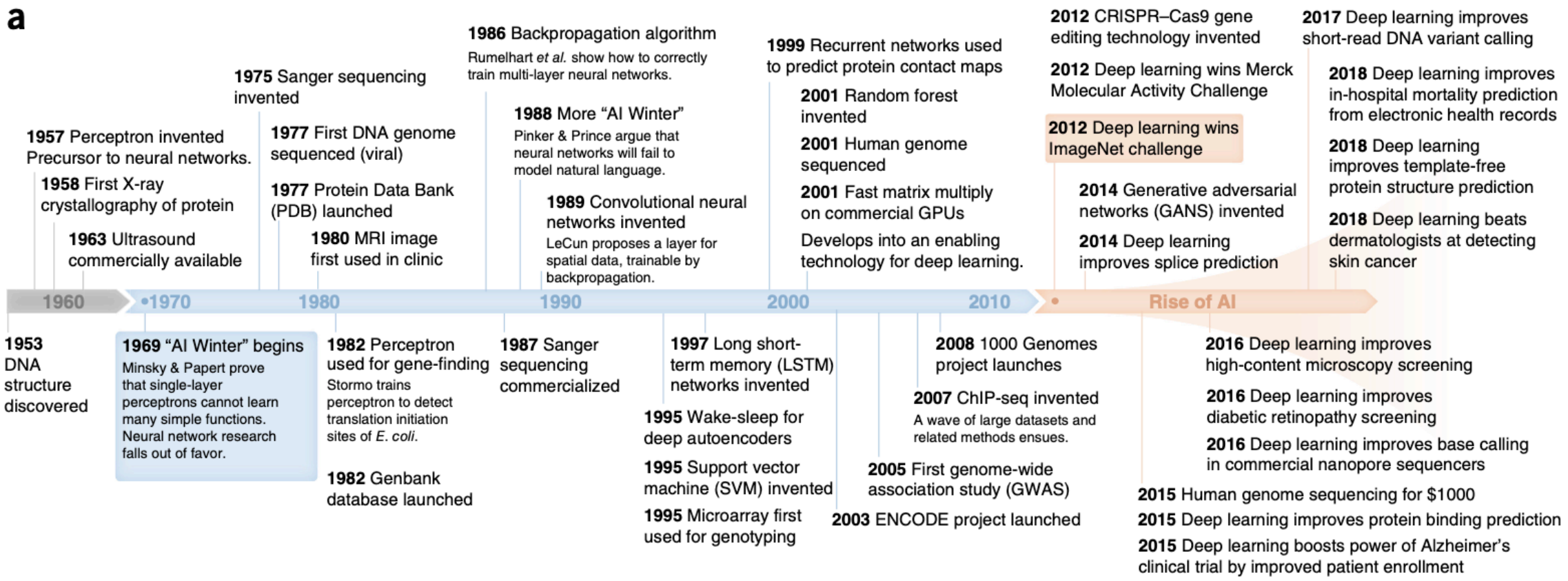


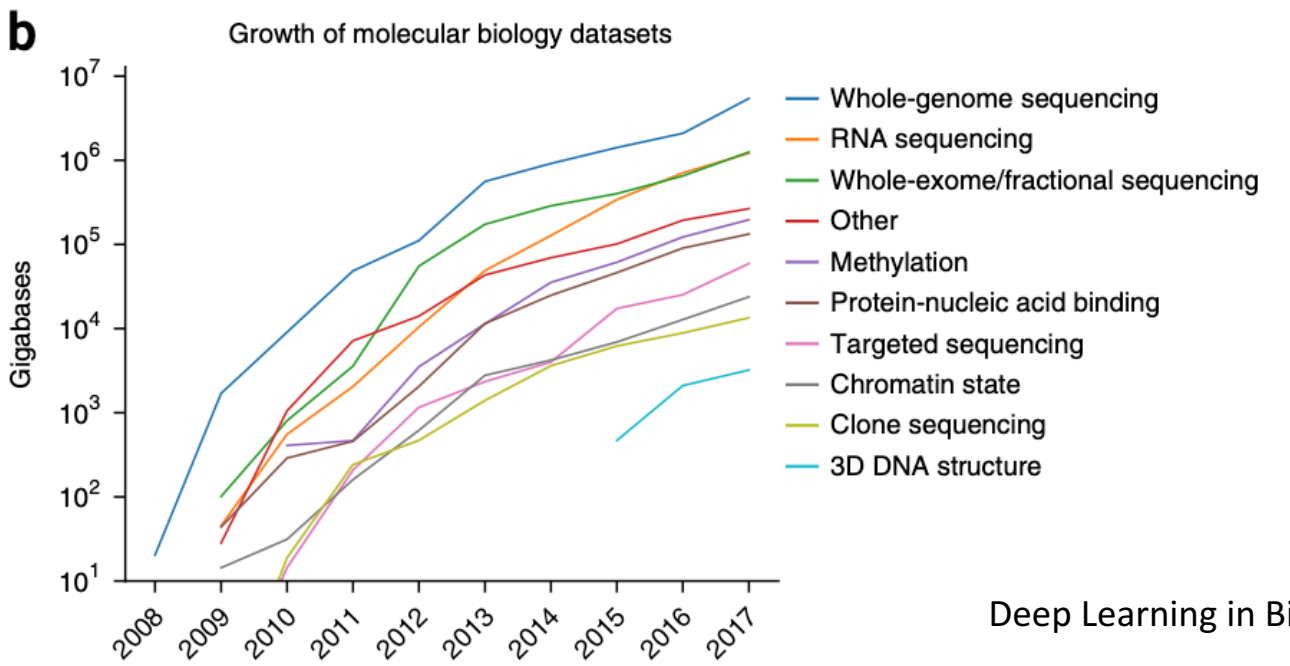
Approximating any f:n



Deep Learning in genomics for medicine

Sunday, October 6, 2024 9:59 PM

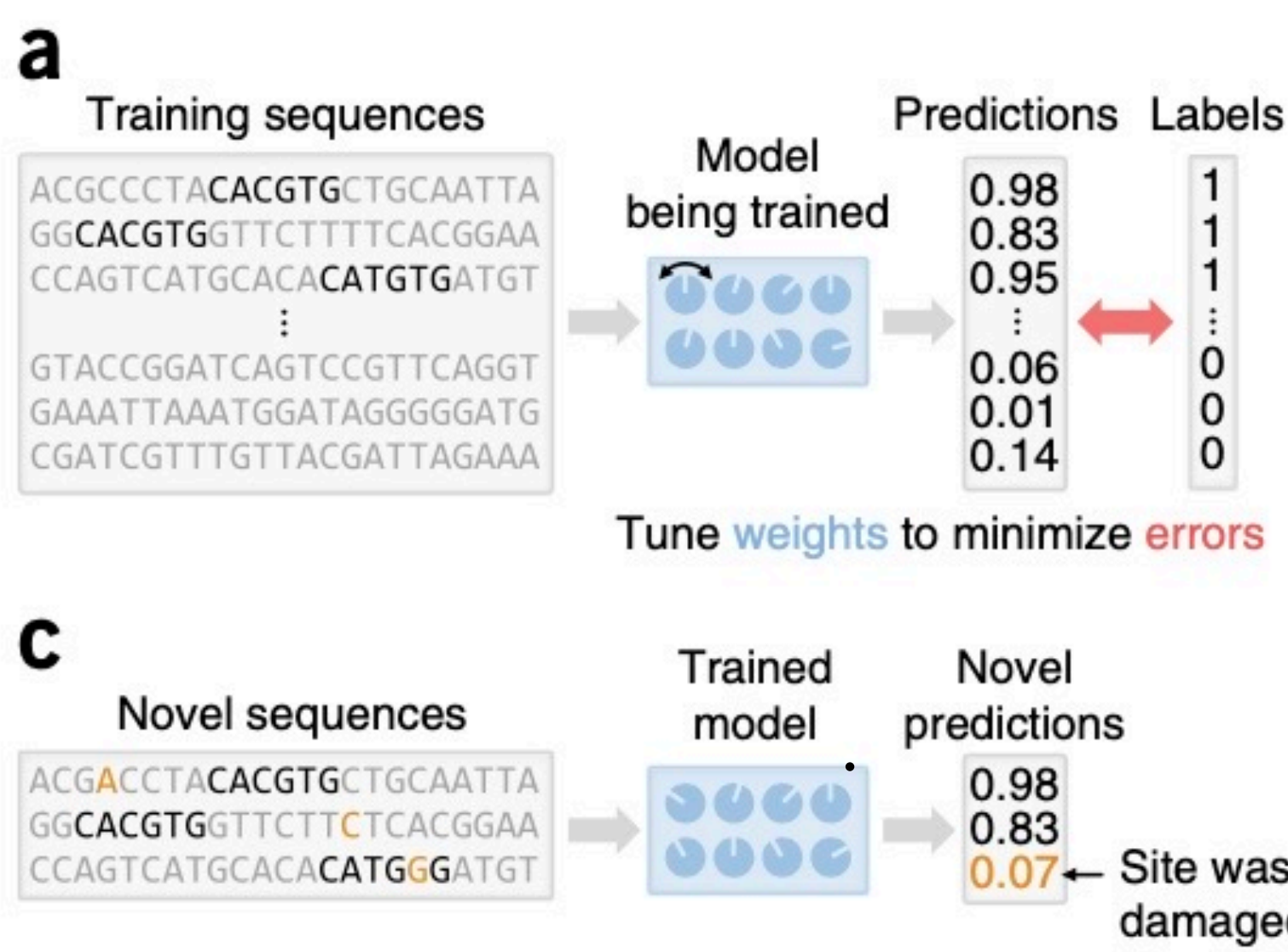




Deep Learning in Biomedicine, Nature Biotechnology, 2018

Detecting protein binding sites in DNA

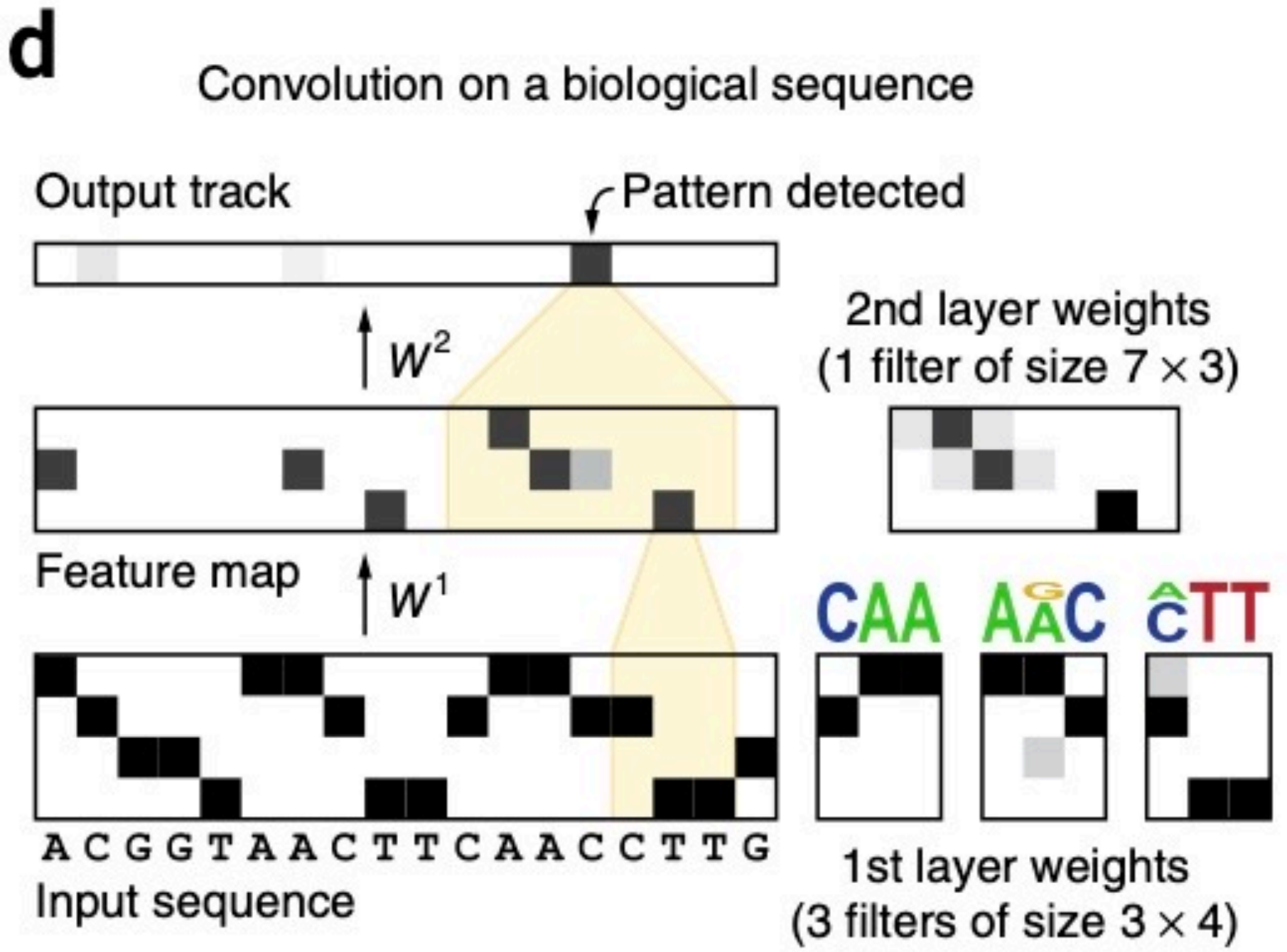
Sunday, October 6, 2024 10:13 PM



Deep Learning in Biomedicine, Nature Biotechnology, 2018

Seq2Seq DNA model

Sunday, October 6, 2024 10:11 PM



A mostly complete chart of

Neural Networks

Backfed Input Cell
Input Cell
Noisy Input Cell
Hidden Cell
Probablistic Hidden Cell
Spiking Hidden Cell
Output Cell
Match Input Output Cell
Recurrent Cell
Memory Cell
Different Memory Cell
Kernel
Convolution or Pool

Perceptron (P)
Feed Forward (FF)
Radial Basis Network (RBF)
Deep Feed Forward (DFF)
Recurrent Neural Network (RNN)
Long / Short Term Memory (LSTM)
Gated Recurrent Unit (GRU)
Auto Encoder (AE)
Variational AE (VAE)
Denoising AE (DAE)
Sparse AE (SAE)
Markov Chain (MC)
Hopfield Network (HN)
Boltzmann Machine (BM)
Restricted BM (RBM)
Deep Belief Network (DBN)
Deep Convolutional Network (DCN)
Deconvolutional Network (DN)
Deep Convolutional Inverse Graphics Network (DCIGN)
Generative Adversarial Network (GAN)
Liquid State Machine (LSM)
Extreme Learning Machine (ELM)
Echo State Network (ESN)
Deep Residual Network (DRN)
Kohonen Network (KN)
Support Vector Machine (SVM)
Neural Turing Machine (NTM)

Forward propagation

Saturday, October 5, 2024 4:21 PM

- I inputs, O outputs, H hidden units
- Total units $M = I + H + O$
- Index units so that connection $i \to j$ satisfies $i < j$
- Ordered activations: Standard form

$$x_1 \dots x_I, x_{I+1} \dots x_{I+H}, x_{M-O+1} \dots x_M$$

- It is convenient to let x_i be the pre-activation value for unit i
- Combining the above, we have

For $j > I$, $x_j = \sum_{i=1}^{j-1} w_{ij} g_i(x_i)$

← If i is not connected to j set $w_{ij} = 0$

↖ Activation function for x_i

Forward propagation: Notes

Sunday, October 6, 2024 10:20 PM

For $j > I$, $x_j = \sum_{i=1}^{j-1} w_{ij}\, g_i(x_i)$

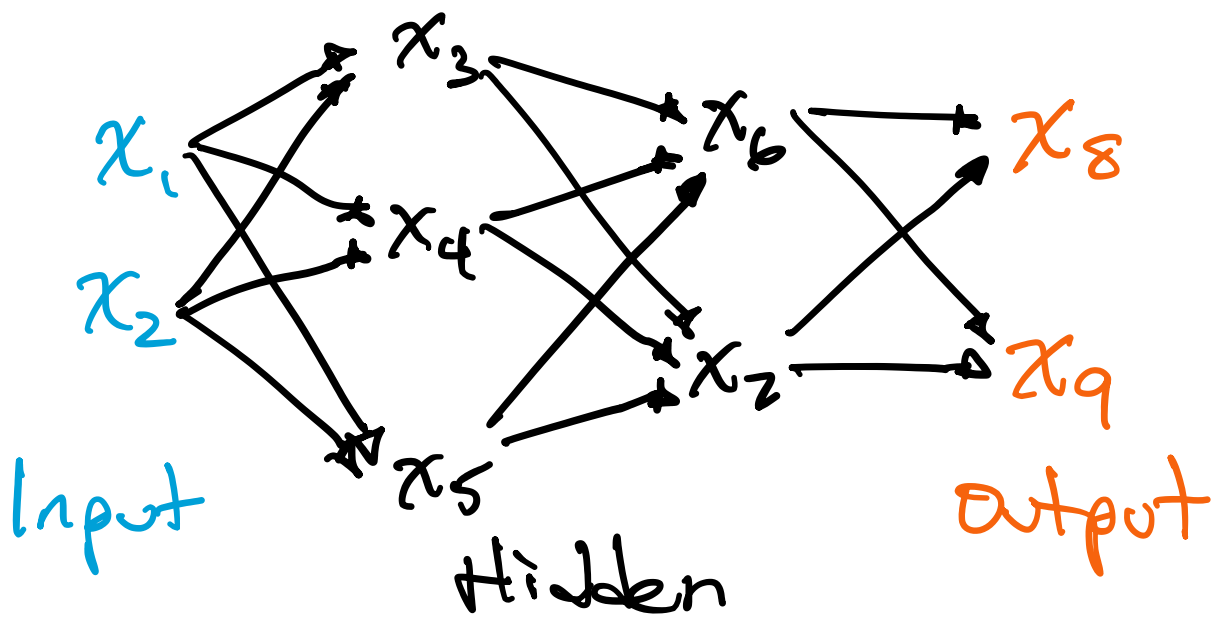
← If i is not connected to j, set $w_{ij} = 0$

↖ Activation function for x_i

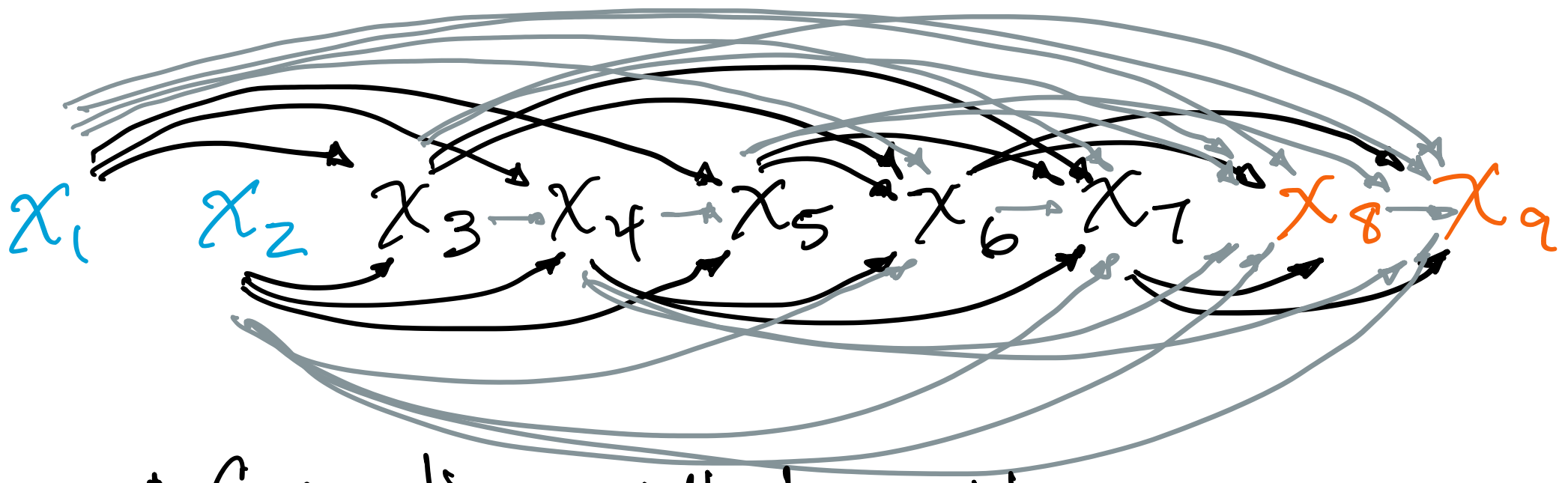
- If input unit x_i, $i \in \{1 \dots I\}$ is not transformed, set $g_i(x_i) = x_i$, ie, $g_i()$ is identity.
- The outputs of the neural network are $g_{M-o+1}(x_{M-o+1}) \dots g_M(x_M)$

Monday, October 7, 2024 10:49 AM

Layered net



Standard form (fully connected DAG)



→ Connections with learnable w

→ Connections for which w clamped to 0

Summary

Sunday, October 6, 2024 9:56 PM

- Limitations of the perceptron
- Decomposing a complex problem
- Differentiable activation functions (sgn → logistic, tanh)

- Neural Networks
- Applications to genomics

- Forward propagation

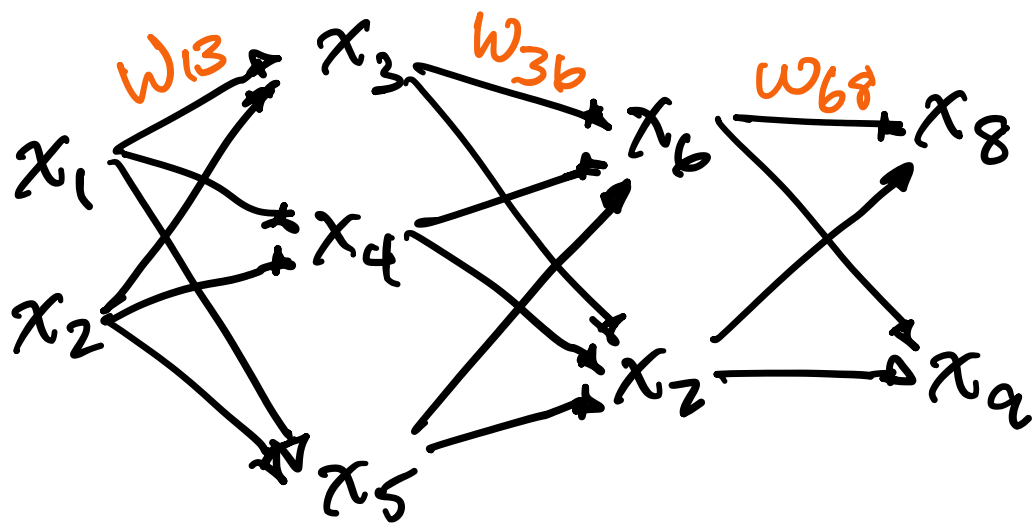
- Next: Backpropation learning

Learning Neural Networks

Saturday, October 5, 2024 4:48 PM

Recall for perceptron: $\underline{w} \leftarrow \underline{w} + y\,\underline{x}_n$ (y: $+1$ or -1)

How do we learn the weights $\{w_{ij}\}$?



Deriving $\partial E / \partial w_{lm}$

Tuesday, October 8, 2024 4:55 PM

We know that for $j = I+1 \dots M$, $x_j = \sum_{i=1}^{j-1} w_{ij}\, g_i(x_i)$ Eq 1 (Forward recursion)

$\partial E / \partial w_{lm}$ for one training case: w_{lm} influences E only via x_m, so $\frac{\partial E}{\partial w_{lm}} = \frac{\partial E}{\partial x_m} \frac{\partial x_m}{\partial w_{lm}}$, $\frac{\partial E}{\partial w_{lm}} = \frac{\partial E}{\partial x_m} g_l(x_l)$ ← Apply Eq 1

If x_m is <u>not</u> an output unit, then x_m influences E via $x_{m+1} \dots x_M$, and we obtain a recursion:

$\frac{\partial E}{\partial x_m} = \sum_{k=m+1}^{M} \frac{\partial E}{\partial x_k} \frac{\partial x_k}{\partial x_m}$, $\frac{\partial E}{\partial x_m} = \sum_{k=m+1}^{M} \frac{\partial E}{\partial x_k} w_{mk}\, g'_m(x_m)$ ← Apply Eq 1

If x_m <u>is</u> an output unit, then

$\frac{\partial E}{\partial x_m} = \frac{\partial (y_m - g_m(x_m))^2}{\partial x_m}$, $\frac{\partial E}{\partial x_m} = -2(y_m - g_m(x_m))\, g'_m(x_m)$

Backpropagation

Tuesday, October 8, 2024 5:32 PM

Forward propagation: for $j = I+1 \dots M$, $x_j = \sum_{i=1}^{j-1} w_{ij}\, g_i(x_i)$

Output unit error derivatives:

For $m = M \dots M-O+1$: $\frac{\partial E}{\partial x_m} = -2(y_m - g_m(x_m))\, g_m'(x_m)$

Backpropagation:

For $m = M-O \dots M-O-H$: $\frac{\partial E}{\partial x_m} = \sum_{k=m+1}^{M} \frac{\partial E}{\partial x_k}\, w_{mk}\, g_m'(x_m)$

Weight derivatives:

For $l = 1 \dots M-O+1$, for $m = l+1 \dots M$

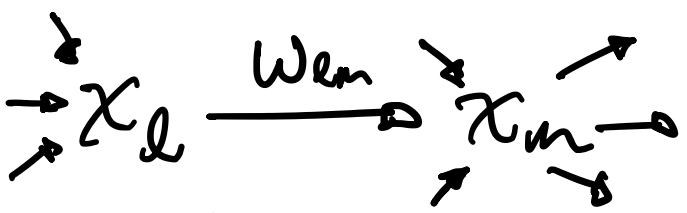
$$\frac{\partial E}{\partial w_{lm}} = \frac{\partial E}{\partial x_m}\, g_l(x_l), \quad w_{lm} \leftarrow w_{lm} - \eta \frac{\partial E}{\partial w_{lm}}$$

Learning rate

Relationship to perceptron learning

Wednesday, October 9, 2024 3:13 PM

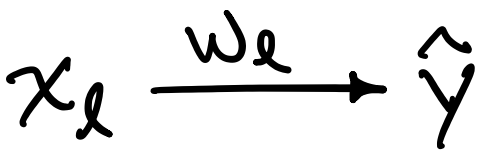
Neural net:



Say $g_\ell(x_\ell) = x_\ell$

$$w_{\ell m} \leftarrow w_{\ell m} - \eta \frac{\partial E}{\partial x_m} x_\ell$$

Perceptron:



$\eta = 1$

$$w_\ell \leftarrow w_\ell + y x_\ell$$

Homework exercise

Wednesday, October 9, 2024 3:07 PM

Forward propagation: for $j = I+1 \dots M$, $x_j = \sum_{i=1}^{j-1} w_{ij}\, g_i(x_i)$

Output unit error derivatives:

For $m = M \dots M-O+1$: $\frac{\partial E}{\partial x_m} = -2(y_m - g_m(x_m))\, g_m'(x_m)$

Backpropagation:

For $m = M-O \dots M-O-H$: $\frac{\partial E}{\partial x_m} = \sum_{k=m+1}^{M} \frac{\partial E}{\partial x_k} w_{mk}\, g_m'(x_m)$

Weight derivatives:

For $l = I \dots M-O-H$, for $m = l+1 \dots M$

$\frac{\partial E}{\partial w_{lm}} = \frac{\partial E}{\partial x_m} g_l(x_l)$, $w_{lm} \leftarrow w_{lm} - \eta \frac{\partial E}{\partial w_{lm}}$ (learning rate)

Rewrite these formulas in terms of the post activation values, $\hat{x}_i \triangleq g_i(x_i)$ $i = 1 \dots M$.

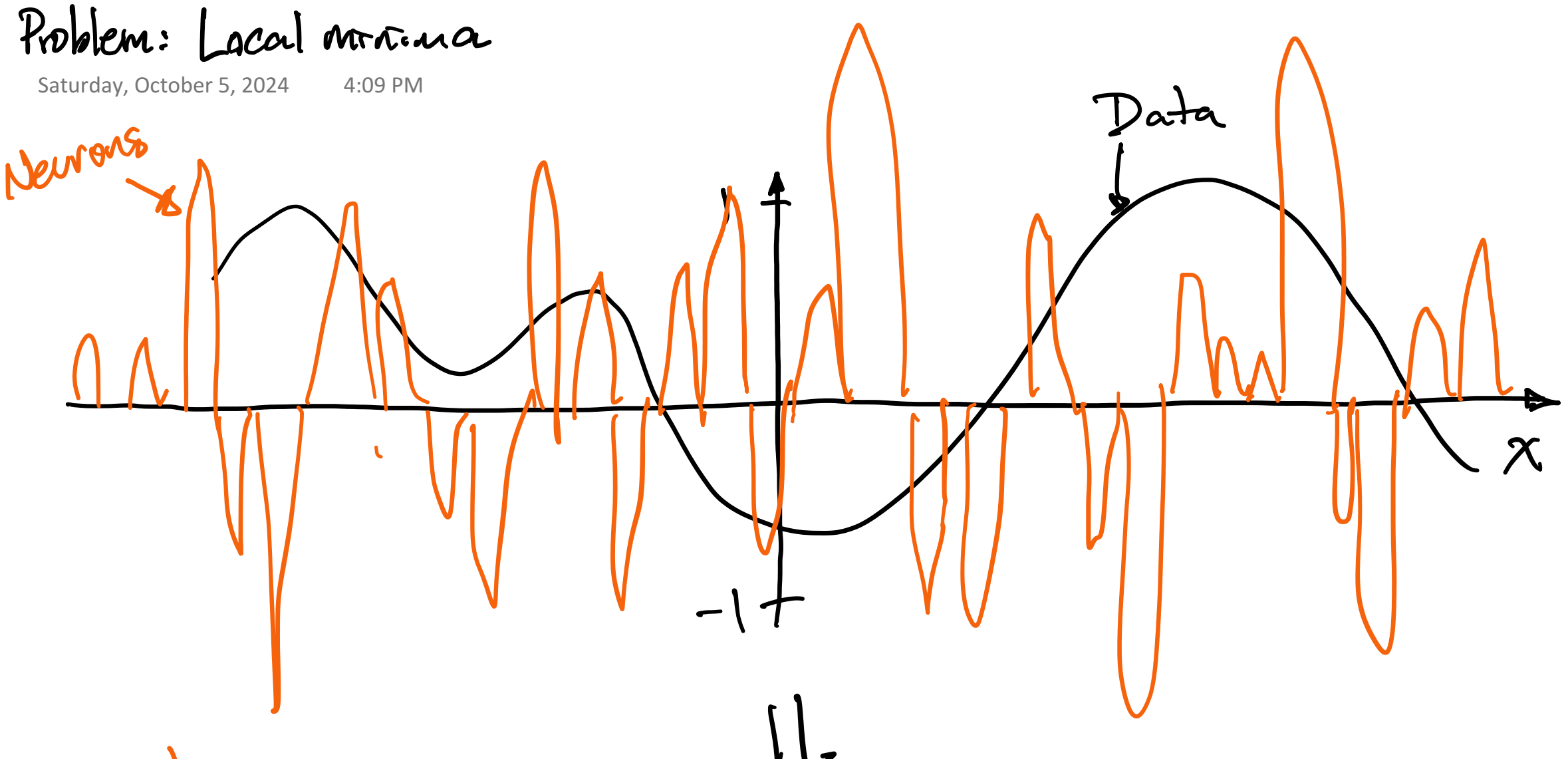
Stochastic gradient descent

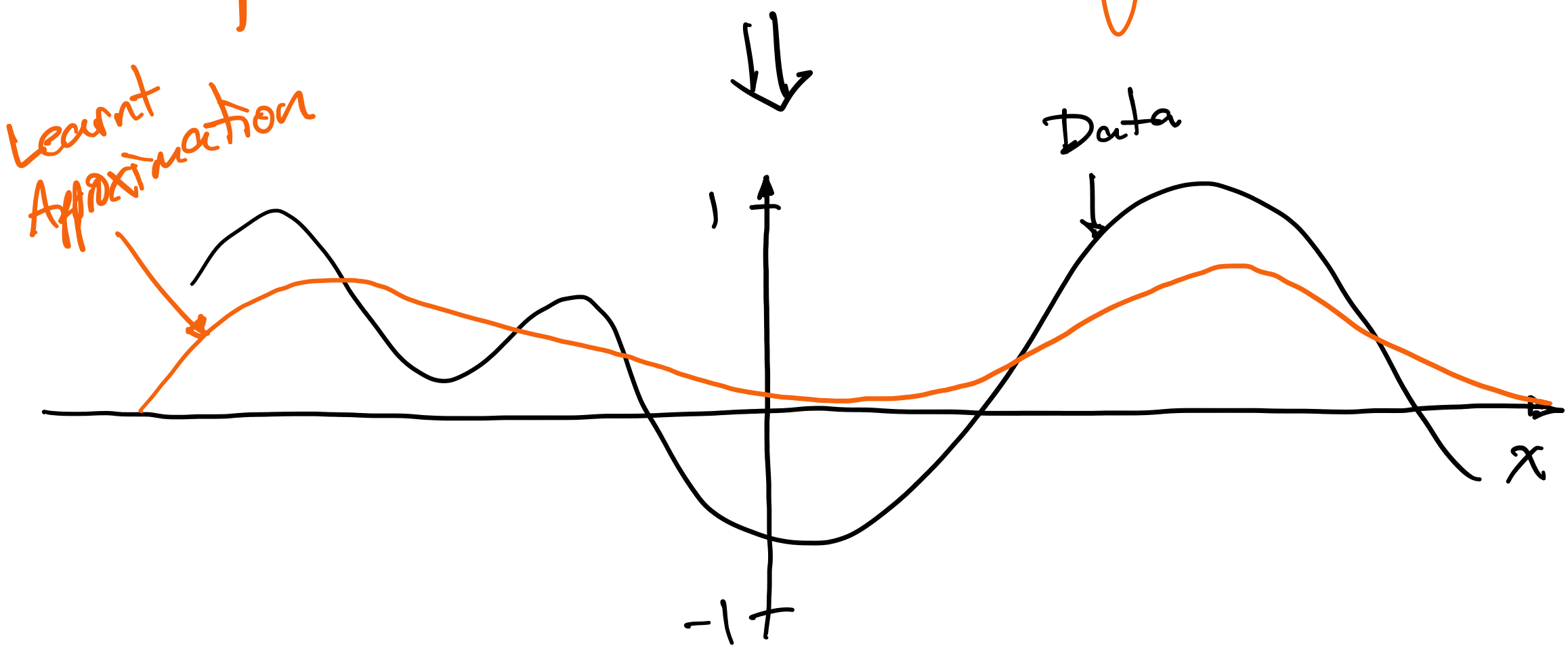
Saturday, October 5, 2024 4:48 PM

- Initialize weights, eg, to small random values
- Iterate until stopping criteria met:
  - Select a training case
  - Apply forward propagation to get x's
  - Compute output unit error derivatives
  - Apply backpropagation to get $\partial E/\partial x$'s
  - Compute weight derivatives $\partial E/\partial w_{lm}$ and update weights, $w_{lm} \leftarrow w_{lm} - \eta \partial E/\partial w_{lm}$

Problem: Local minima

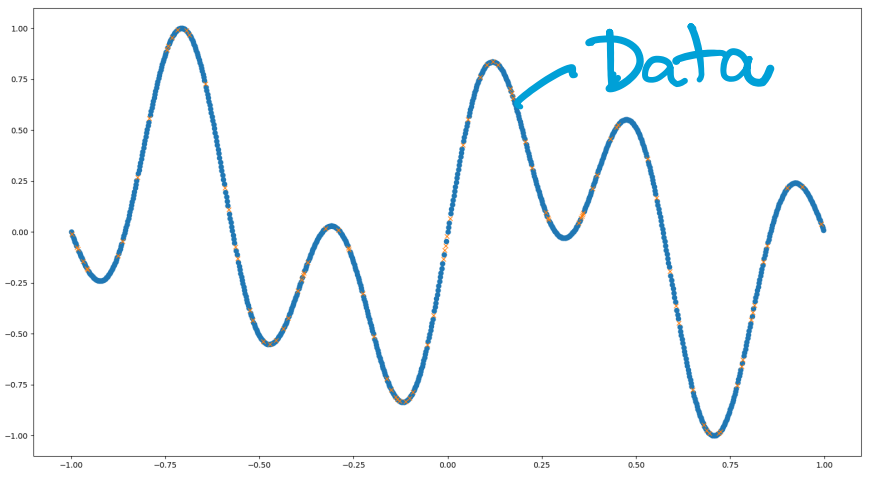
Saturday, October 5, 2024 4:09 PM



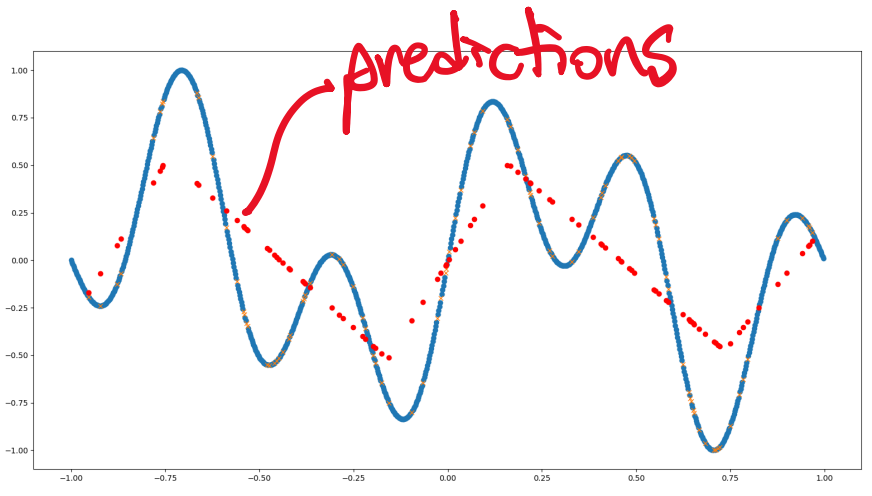


Example

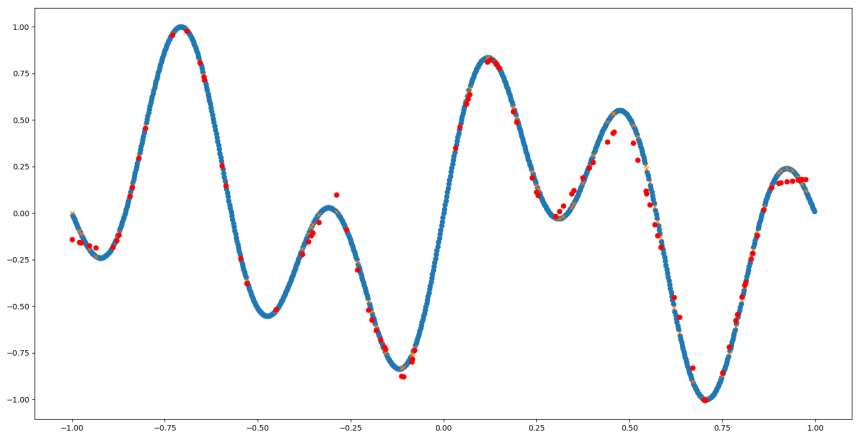
Saturday, October 5, 2024 4:01 PM



Training data



Neural network predictions
1 hidden layer of 30 units
after training to convergence



Neural network predictions
4 hidden layers of 20, 40, 50 and 30 units

After training to convergence