

$$\begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ 0 & & & \sigma_n \end{bmatrix}$$

$$X = \sum_{i=1}^{\text{rank}(X)} \sigma_i u_i v_i^T = U \Sigma V^T$$

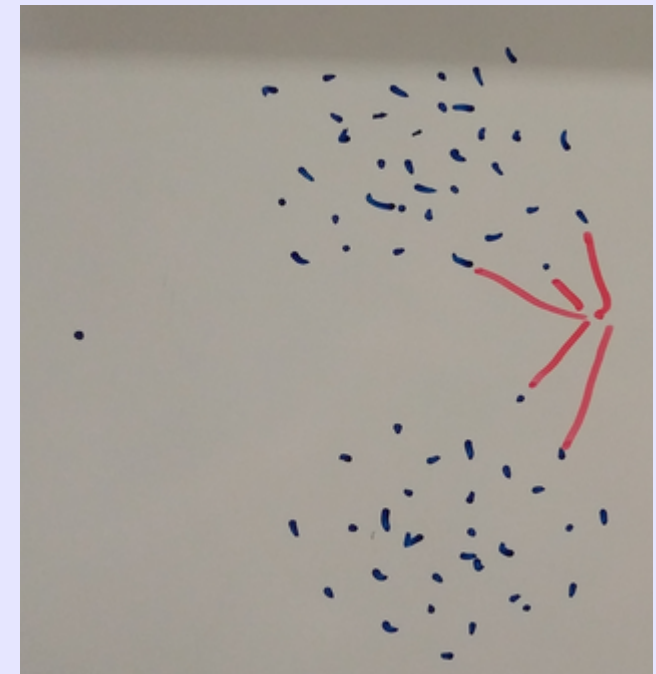
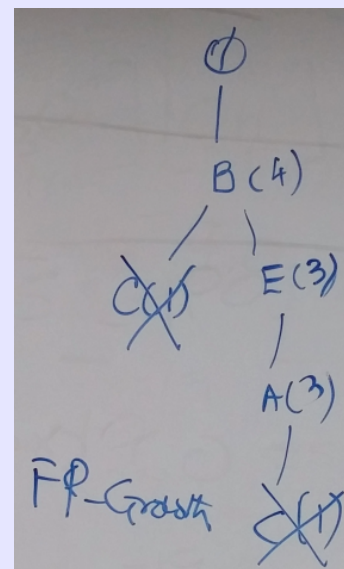
i^{th} singular value of X i^{th} left singular value of X (i^{th} column of U) i^{th} right singular vector of X (i^{th} column of V^T)

Captures the patterns among attributes
 Captures the patterns among the objects

$$\chi^2 = \sum \frac{(o - e)^2}{e}$$

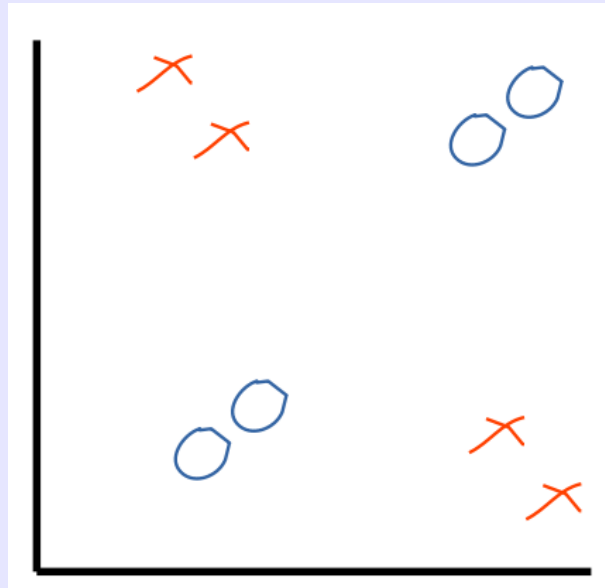
CS 422: Data Mining
 Vijay K. Gurbani, Ph.D.,
 Illinois Institute of Technology

Lecture: Introduction to Artificial Neural Networks



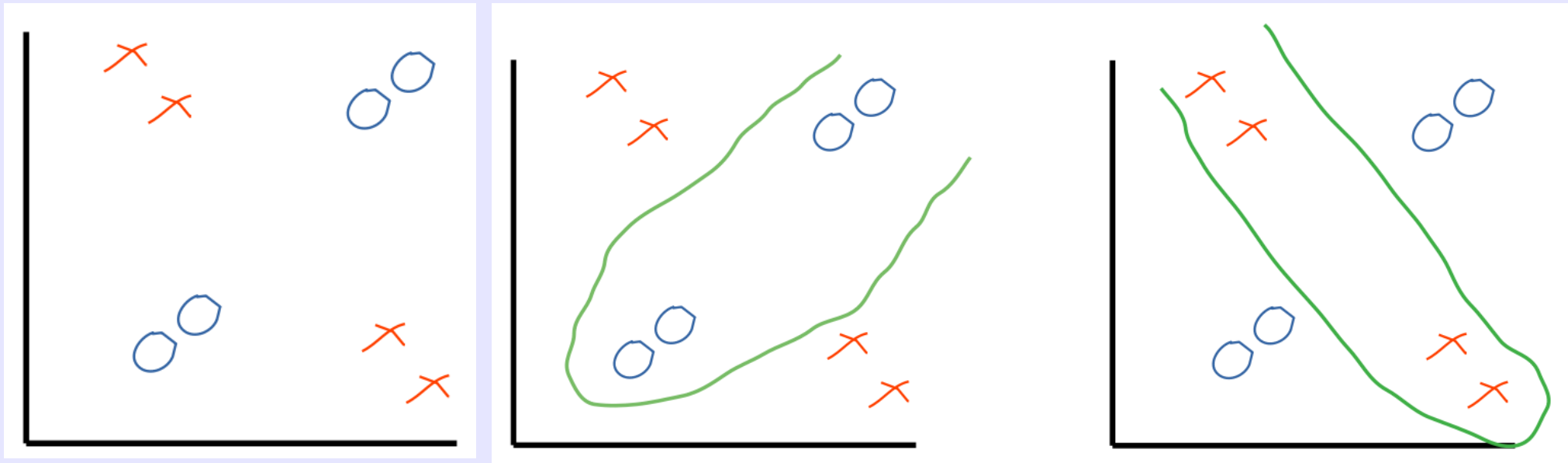
Introduction

- Hidden layers are needed if the data must be separated using a non-linear boundary.



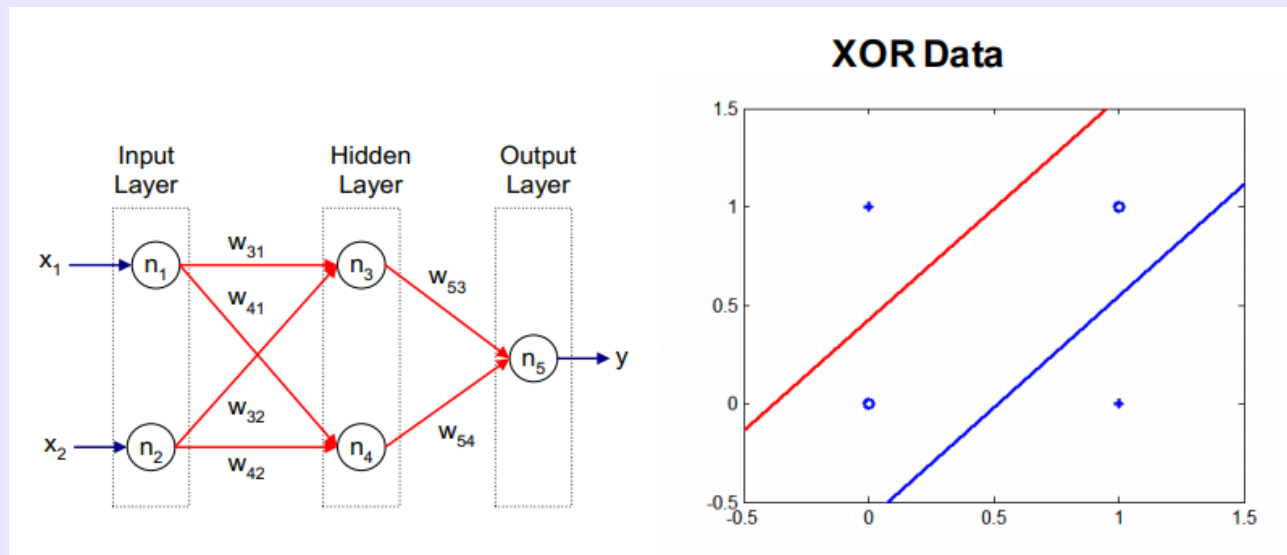
Introduction

- Hidden layers are needed if the data must be separated using a non-linear boundary.



Introduction

- Major difference between ANN and Perceptron is inclusion of hidden layers.
 - Consider the solution to the XOR problem in a FFNN:

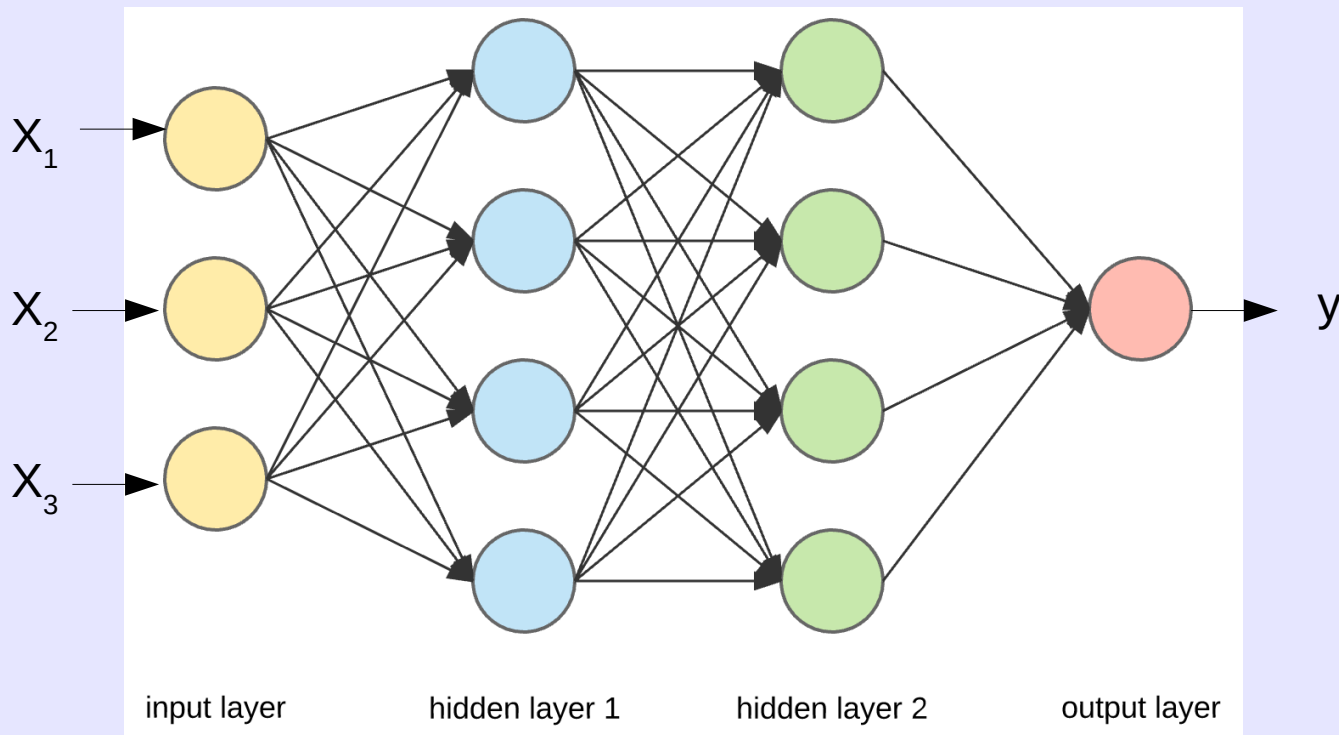


Introduction

- Major difference between ANN and Perceptron is inclusion of *hidden layers*.
 - Universal Approximation Theorem for Neural Networks: A FFNN with a single hidden layer containing an arbitrary number of neurons can **approximate** any **continuous** function on compact subsets of \mathbb{R}^n (Cybenko 1989).
 - The theorem was also proved for arbitrary number of *hidden layers*, each containing a limited number of neurons (Lu et al. 2017).
 - Hidden layers can represent arbitrary complex decision boundaries.

Introduction

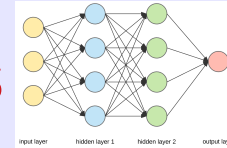
- Generalizes the concept of a single-neuron perceptron to the more complex architecture of nodes capable of learning non-linear decision boundaries.



Introduction

- Three kinds of neural networks:

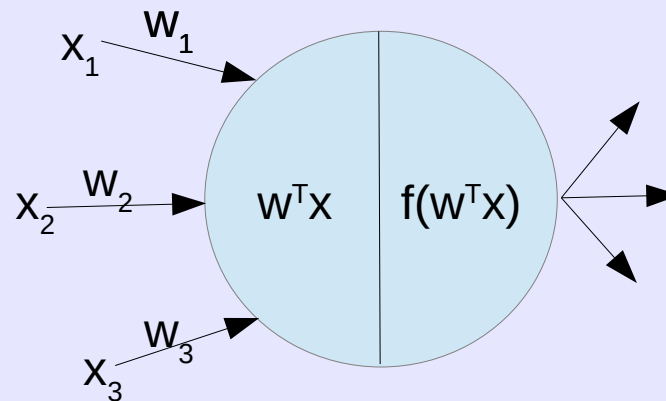
- **Feed forward neural networks**



- General neural networks for classification, regression, etc.
 - Convolutional neural networks
 - Excel at image recognition.
 - Recurrent neural networks
 - Excel at language tasks.

Introduction

- Hidden nodes learn *latent* representation (features useful for class boundaries).
- First hidden layer captures simpler features (since it receives the predictors as input).
- Subsequent hidden layers hone into specific patterns of the data to extract features.
- So, what does a neuron do?



$f(.)$ is an activation function.
You have seen this before ...
...