Overview of Topics

MAT1841 – Fall 2021

Yun William Yu

List of topics

- Random hashing
- Probabilistic streaming and sketching algorithms
- Matrix decompositions (including NMF and probabilistic)
- Random graph theory / percolation theory
- Wavelet bases
- Complexity and entropy
- Nonlinear dimensionality reduction
- Computational topology

Possibly out-of-scope topics (because ML)

These are all super fun topics, but also covered in many other classes. Take e.g. Prof. Papyan's MAT1510 instead: https://sites.google.com/view/mat1510

- Clustering
- Linear classifiers
- Kernel methods
- Deep learning
- Graphical models

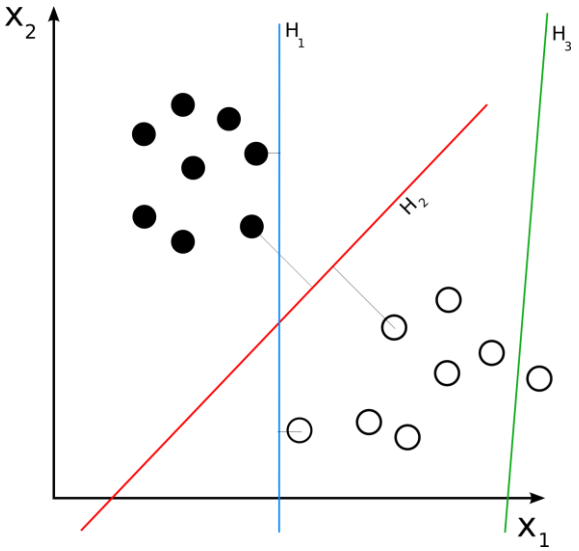
Machine Learning

- Intuition: try to learn the underlying probability distribution generating the data we care about.

- An algorithm builds a mathematical model based on training data, which it uses to make predictions or decisions on new data.
- We say that model parameters are “learned” from the data.
- We focus here on the supervised classification task, though many of the other topics in data science are sometimes “considered” ML.

ML: linear classifier

- Given an input vector $\boldsymbol{x}$, the output $y = f(\boldsymbol{w} \cdot \boldsymbol{x})$, where the weights $\boldsymbol{w}$ are learned from the data should match label l.
- Simple example: $f(a) = 1$ if $a > t$, for some threshold t, and 0 otherwise.
  - Dividing hyperplane, separating classes 0 and 1.



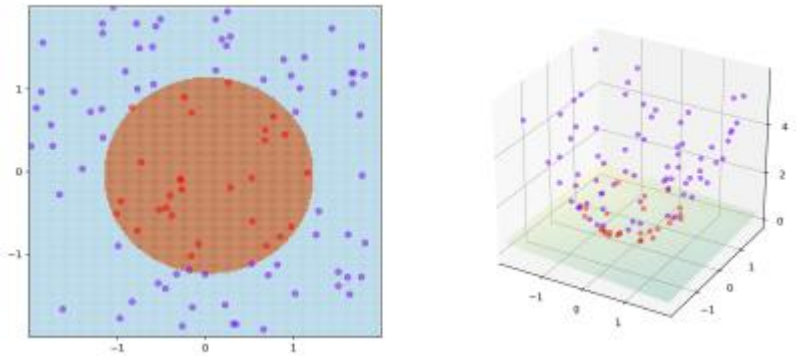
https://en.wikipedia.org/wiki/Linear_classifier

ML: linear classifiers

- Perceptron algorithm
  - Technical modification, $\hat{\boldsymbol{x}} = (\boldsymbol{x}, 1)$, $\hat{\boldsymbol{w}} = (\boldsymbol{w}, -t)$, making separating hyperplanes go through the origin.
  - Initialize with $\boldsymbol{w} \leftarrow 0$.
  - While there exists $\boldsymbol{x_i}$ with $\boldsymbol{x_i} l_i \cdot \boldsymbol{w} \leq 0$, update $\boldsymbol{w} \leftarrow \boldsymbol{w} + \boldsymbol{x_i} l_i$, where $l_i = \{-1, 1\}$ is class label.
- SVM (Support Vector Machine)
  - Tries to find the maximum-margin hyperplane, not just any hyperplane (like perceptron).

ML: kernel trick

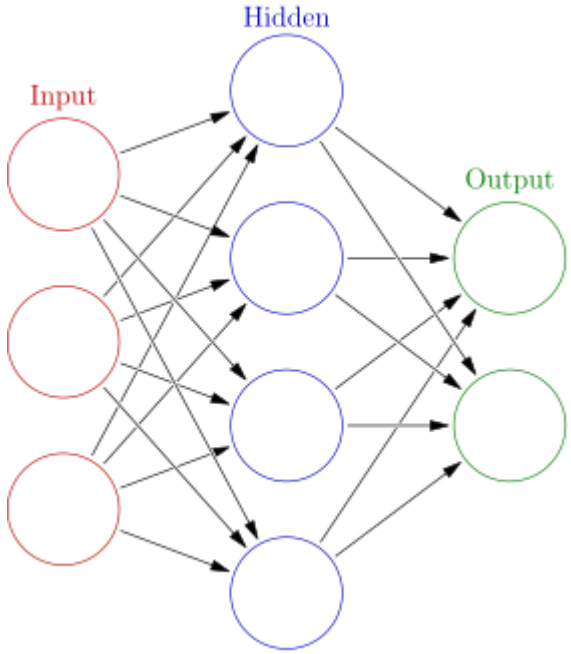
- Data may not be linearly separable
- But we can often map the data to another space where it is linearly separable.
- E.g. $\varphi\big((x_1, x_2)\big) = (x_1, x_2, x_1^2 + x_2^2)$
- Kernel: $K(\boldsymbol{x}, \boldsymbol{y}) = \varphi(\boldsymbol{x}) \cdot \varphi(\boldsymbol{y}) = \boldsymbol{x} \cdot \boldsymbol{y} + |\boldsymbol{x}|^2 |\boldsymbol{y}|^2$
- Careful choice of map allows using kernel function instead of explicit mapping.



https://en.wikipedia.org/wiki/Kernel_method

ML: deep learning

- Chaining together a bunch of simple nonlinear classifiers empirically improves classification.
- Each node represents a linear combination of parent node values, modified by a nonlinearity (often a ReLU).
- Empirically, using a deep network allows us to use a much simpler nonlinearity than more complicated kernel functions.



ML: back-propagation

- The network can be thought of as a function

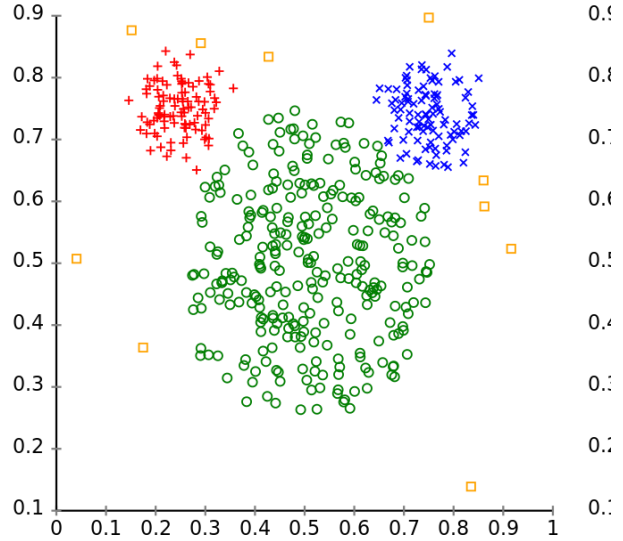
$$g(x) = f^L\left(W^L f^{L-1}(W^{L-1} \cdots f^1(W^1 x) \cdots)\right)$$

where, f^l is the nonlinearity, and W^l is a weights matrix at layer at layer l.

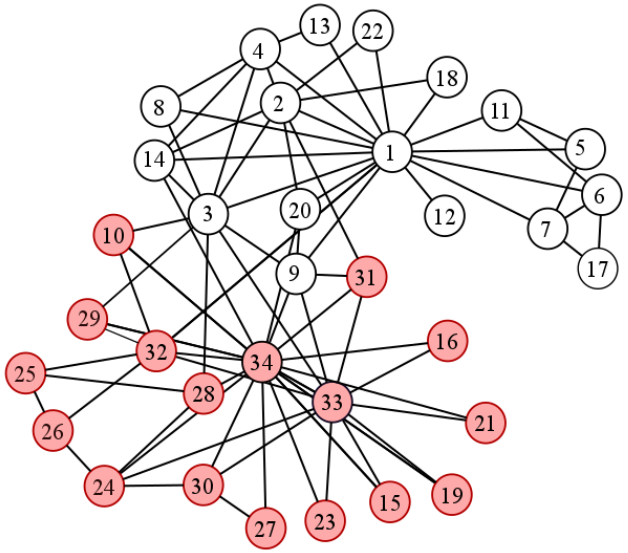
- We also have a loss/cost function $C\left(y_i, g(x_i)\right)$, where y_i is the true label of a data point x_i.
- We want to use gradient descent to optimize the weights based on the training data.
- Each individual component of the gradient $\partial C / \partial w_{jk}^l$ can be computed via the chain rule.
- The back-propagation algorithm avoids duplicate calculations by computing the gradient of each layer from back to front. (i.e. starting from the output layer)

Clustering

- Grouping together data points into “meaningful” groups.
- Also known variously as partitioning, community detection, finding spin glass states, etc.
- Two major versions
  - –High-dimensional space (not just vector spaces)
  - –On a graph
- Hard and soft (depending on group assignment)



https://en.wikipedia.org/wiki/K-means_clustering



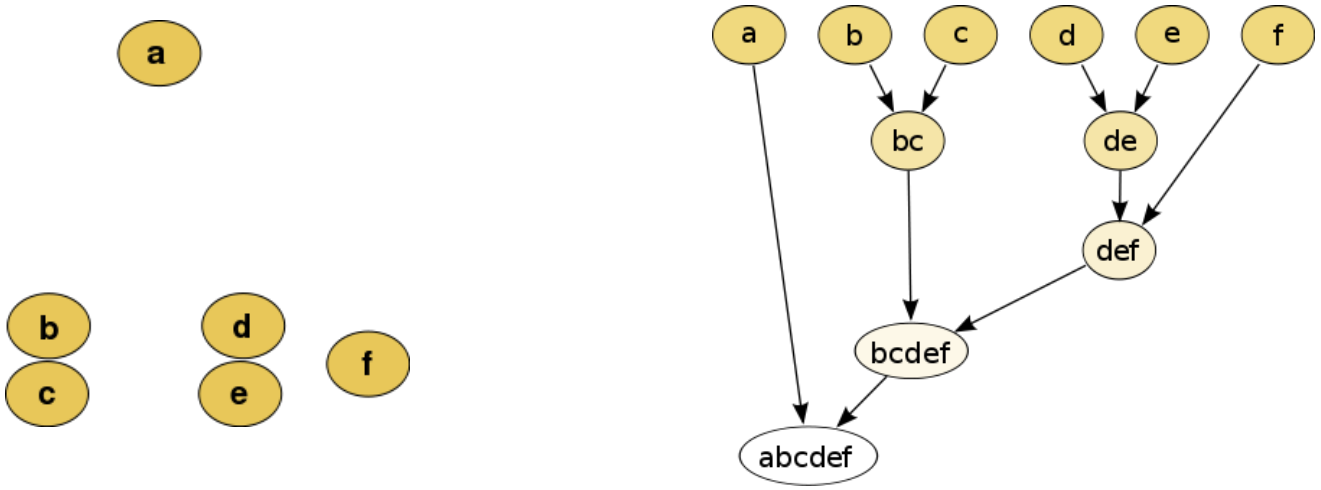
https://commons.wikimedia.org/wiki/File:Zachary%27s_karate_club.png

K-means

- Given a set of observations $(x_1, \ldots, x_n)$, $x_i \in \mathbb{R}^d$, find a partition $\boldsymbol{S} = \{S_1, \ldots, S_k\}$ that minimizes squared distances to cluster centers.
- Naïve k-means algorithm
  - Initialize means (e.g. with random choice)
  - Iterate until convergence:
    - Assign each observation to nearest cluster center
    - Calculate new cluster means based on assignment.
- Converges if using Euclidean distance

Hierarchical clustering

- E.g. on a graph, repeatedly cut the graph in half to minimize the cut weight.
- Alternately, iteratively link together pairs of points that are closest together.



https://en.wikipedia.org/wiki/Hierarchical_clustering

Scoring functions

- E.g. Girvan-Newman modularity.
  - The fraction of edges within clusters minus the expected fraction if edges were distributed at random (under several different random graph models).
- Related to Hamiltonian of spin glass in physics. (i.e. energy of a system where adjacent nodes want the same spin).
- Cluster scoring function independent of number of clusters. Often paired with a hierarchical clustering algorithm to allow choosing the correct level.

(Gaussian) mixture models

- Recall we covered a simple Gaussian mixture model where we assumed our dataset was generated by a combination of different radially symmetric Gaussians.
- In general, let $p(\theta) = \sum_{i=1}^{K} \phi_i \mathcal{N}(\mu_i, \Sigma_i)$, where ϕ_i is a weight associated with each multivariate Gaussian distribution $\mathcal{N}(\mu_i, \Sigma_i)$.
- How can we estimate $p(\theta)$ from a bunch of samples drawn from it?

Expectation-maximization iterative algorithm

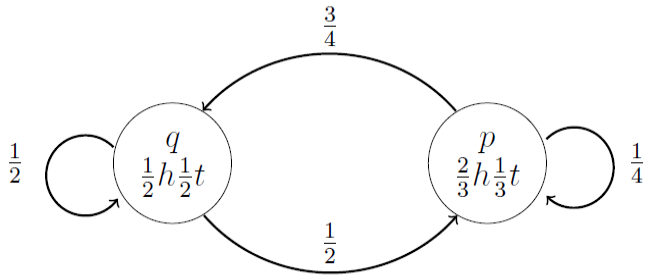
- One commonly used iterative technique to fit parameters θ and missing latent variables Z is the EM-algorithm.
- Algorithm:
  - Initialize parameters θ to random values
  - Compute the probability of each possible value of Z, given θ (E-step).
  - Then, use the just-computed values of Z to compute a better estimate for the parameters θ (M-step)
  - Iterate the last two steps until convergence.

Hidden Markov models (HMM)

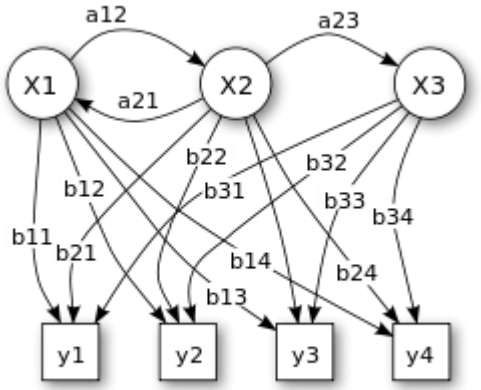
- Let X_n and Y_n be discrete-time stochastic processes and $n \geq 1$. The pair (X_n, Y_n) is a Hidden Markov Model if X_n is a Markov process and not directed observable and $P(Y_n \in A | X_1 = x_1, \ldots, X_n = x_n) = P(Y_n \in A | X_n = x_n)$.
- Generalization of a mixture model where the hidden (latent) variables controlling the mixture component are related through a Markov chain instead of independent.
- System being modelled is assumed to be a Markov process with unobservable (hidden) states.
- Can be learned using a variation of the EM algorithm.

Hidden Markov Models

- Inference tasks:
  - given parameters of a model, compute probability of a particular output sequence.
  - Figure out the distribution over hidden states of the last latent variable at the end of the sequence.



Blum, Hopcroft, Kannan, 2020



https://en.wikipedia.org/wiki/Hidden_Markov_model

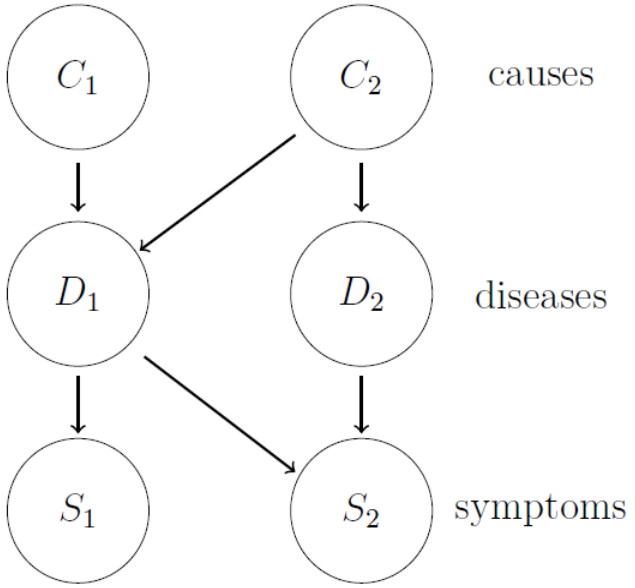
Graphical models

- “Graphical” in the sense of “graph theory”
- A graphical model is a compact representation of a probability distribution over n variables $x_1, \ldots, x_n$.
- When using a directed acyclic graph, is known as a Bayesian or belief network.
- When using an undirected graph, is known as a Markov random field

Bayesian or Belief networks

$$p(x_1, \ldots, x_n) = \Pi_{i=1}^{n} p(x_i | \text{parents of } x_i)$$

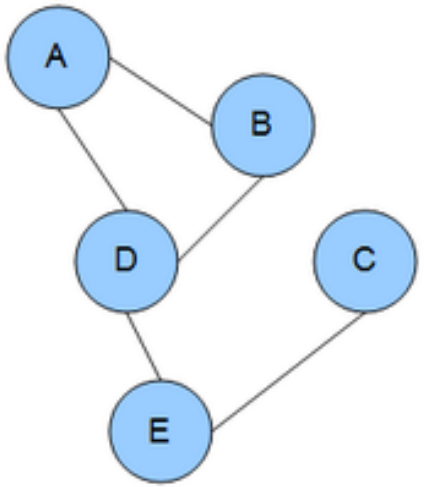
- Each directed edge from y to x represents a conditional probability $p(y|x)$.
- A variable without any in-edges has an unconditional probability distribution.
- We observe only certain variables, known as "evidence".
- E.g. A doctor observes an ill patient's symptoms
  - What disease does the patient have?
  - What is the probability of a specific disease?



Blum, Hopcroft, Kannan, 2020

Markov random field

- Given an undirected graph $G = (V, E)$, a set of random variables $X = (X_v)_{v \in V}$ indexed by V form a Markov random field with respect to G if every variable is conditionally independent of all other variables given its neighbors.



https://en.wikipedia.org/wiki/Markov_random_field

Markov random field examples

- Application: Ising model of spin glasses / community detection.
  - Each particle $x_1, \ldots, x_n$ can have a spin ± 1, and the energy of the system is $\exp\left(c \sum_{i \sim j} \left| x_i - x_j \right|\right)$.
  - Minimizing the energy, subject to specified constraints, is a Markov random field.
- Application: Image reconstruction
  - Each pixel is a graph vertex, and we may wish nearby pixels to be similar, with some penalty.