Clustering
Input: $S_n = \left\{\overline{x}^{(i)}\right\}_{i=1}^n \overline{x} \in \mathbb{R}^d$
Output: a set of cluster assignments $c_1, \ldots, c_n$, where $c_i \in \{1, \ldots, k\}$
K-means
Datapoints $\overline{x}^{(1)}, \ldots, \overline{x}^{(n)}$ and fixed k. Initial means $\overline{\mu}^{(1)}, \ldots, \overline{\mu}^{(k)}$
Iteratively,
1. reassign $\overline{x}^{(i)}$ to $c_i = \arg\min_j \left\|\overline{x}^{(i)} - \overline{\mu}^{(j)}\right\|^2$
2. recompute $\overline{\mu}^{(j)} = \frac{\sum_i [[c_i=j]]\overline{x}^{(i)}}{\sum_i [[c_i=j]]}$
That is, define $J(\overline{c}, \overline{\mu}) = \sum_{i=1}^n \left\|\overline{x}^{(i)} - \overline{\mu}^{(c_i)}\right\|^2$
First, fix $\overline{\mu}$, choose $\overline{c}$ to minimize J. Second, fix $\overline{c}$, choose $\overline{\mu}$ to minimize J.
k-means performs coordinate descent on the objective function. k-means is guaranteed to converge (because objective function is monotonically decreasing), but not necessarily the global minimum. Solution: vary initialization of k-means, and pick clustering with lowest (eventual) objective function value (for a fixed k).
Spectral Clustering
Weight matrix: W, where w_{ij} represents the similarity between v_i and v_j.
Degree of a vertex: $d_i = \sum_{j=1}^n w_{ij}$
Degree matrix D: $D_{ii} = \sum_{j=1}^n w_{ij}$, $D_{ij} = 0$ for $i \neq j$
Cost of a cut between A and $\overline{A}$: $\text{cut}(A, \overline{A}) = \sum_{i \in A} \sum_{j \in \overline{A}} w_{ij}$
Graph Laplacian: $L = D - W$.
Build matrix with the first k eigenvectors (corresponding to the k smallest eigenvalues) as columns interpret rows as new data points
Apply k-means to new data representation
Hierarchical Clustering
1. Assign each pt. its own cluster, for each point i, $C_i = \{\overline{x}^{(i)}\}$
2. Find the closest clusters & merge, repeat until convergence

$$\arg\min_{i,j} d\left(C_i, C_j\right) \text{ where } i \neq j$$

Recommender Systems
Approach #1: **Nearest Neighbor Prediction**
User: a, b.Y_{ai} is missing, i.e., a has not rated movie i.
Define $R(a, b)$: set of movies rated by both users a and b
$\tilde{Y}_{a:b} = \frac{1}{|R(a,b)|} \sum_{j \in R(a,b)} Y_{a_j}$

$$\text{sim}(a,b) = \frac{\sum_{j \in R(a,b)} \left(Y_{a_j} - \tilde{Y}_{a:b}\right)\left(Y_{b_j} - \tilde{Y}_{b:a}\right)}{\sqrt{\sum_{j \in R(a,b)} \left(Y_{a_j} - \tilde{Y}_{a:b}\right)^2 \sum_{j \in R(a,b)} \left(Y_{b_j} - \tilde{Y}_{b:a}\right)^2}}$$

Define $KNN(a, i)$: k nearest neighbors, i.e., the k most similar users to a, who have rated movie.

$$\hat{Y}_{a_i} = \overline{Y}_a + \frac{1}{\sum_{b \in kNN(a,i)} |\text{sim}(a,b)|} \sum_{b \in kNN(a,i)} \text{sim}(a,b)\left(Y_{b_i} - \overline{Y}_b\right)$$

Approach #2: **Matrix Factorization**
Let $A \in \mathbb{R}^{n \times m}$, and $rank(A) = r$. Then there exists $X \in \mathbb{R}^{n \times r}$, $Y \in \mathbb{R}^{r \times m}$, such that $A = XY$
Given Y with empty cells, construct low rank-d $\widehat{Y}$ with no empty cells. We may think of Y as being approximated by $\widehat{Y} = UV^T$, where $U \in \mathbb{R}^{n \times d}$ contains the relevant features of the user, and $V \in \mathbb{R}^{m \times d}$ contains the relevant features of the movie.

$$J(U,V) = \frac{1}{2} \sum_{(a,i) \in D} (Y_{ai} - [UV^T]_{ai})^2 + \frac{\lambda}{2} \sum_{a=1}^n \sum_{k=1}^d U_{ak}^2 + \frac{\lambda}{2} \sum_{i=1}^m \sum_{k=1}^d V_{ik}^2$$

$U = [\overline{u}^{(1)}, \ldots, \overline{u}^{(n)}]^T$, and $V^T = [\overline{v}^{(1)}, \ldots, \overline{v}^{(m)}]$
Algorithm Overview (coordinate descent)
1. Initialize V to small (random) values.
2. Iterate until convergence
Fix $\overline{v}^{(1)}, \ldots, \overline{v}^{(m)}$. Solve for $\overline{u}^{(1)}, \ldots, \overline{u}^{(n)}$

$$\min_{\overline{u}^{(a)}} \frac{1}{2} \sum_{i \in D_a} (Y_{ai} - \overline{u}^{(a)} \cdot \overline{v}^{(i)})^2 + \frac{\lambda}{2} \|\overline{u}^{(a)}\|^2$$

Fix $\overline{u}^{(1)}, \ldots, \overline{u}^{(n)}$. Solve for $\overline{v}^{(1)}, \ldots, \overline{v}^{(m)}$

$$\min_{\overline{v}^{(i)}} \frac{1}{2} \sum_{a \in D_i} (Y_{ai} - \overline{u}^{(a)} \cdot \overline{v}^{(i)})^2 + \frac{\lambda}{2} \|\overline{v}^{(i)}\|^2$$

i.i.d assumption
Identically drawn from $\mathcal{D}$, each trial independent from each other.
Gaussian Distribution

$$\mathcal{N}\left(x|\mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$$

Spherical Gaussian

$$\mathcal{N}\left(\overline{x}|\overline{\mu}, \sigma^2\right) = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(-\frac{1}{2\sigma^2}\|\overline{x} - \overline{\mu}\|^2\right)$$

MLE results

$$\overline{\mu} = \frac{1}{n} \sum_{i=1}^n \overline{x}^{(i)}, \ \sigma^2 = \frac{1}{nd} \sum_{i=1}^n \|\overline{x}^{(i)} - \overline{\mu}\|^2$$

GMM, EM
Iterate until convergence:
E step: use current estimate of mixture model to assign examples to clusters
M step: re-estimate each cluster model separately based on the points assigned to it (similar to the "known cluster" case)
Bayes' Theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \frac{P(B|A)P(A)}{\sum_A P(B|A)P(A)}$$

E-step: fix parameters $\overline{\theta} = [\gamma_1, \ldots, \gamma_k, \overline{\mu}^{(1)}, \ldots, \overline{\mu}^{(k)}, \sigma_1^2, \ldots, \sigma_k^2]$, compute posterior distribution

$$P(j|i) = \frac{\gamma_i \cdot \mathcal{N}(\overline{x}^{(i)}|\overline{\mu}^{(j)}, \sigma_j^2)}{\sum_{t=1}^k \gamma_t \cdot \mathcal{N}(\overline{x}^{(i)}|\overline{\mu}^{(t)}, \sigma_t^2)}$$

M-step: fix posterior distribution $P(j|i)$, compute MLE parameters $\overline{\theta}$.

$$n_j = \sum_{j=1}^n P(j|i), \quad \gamma_j = \frac{n_j}{n}$$

$$\overline{\mu}^{(j)} = \frac{1}{n_j} \sum_{i=1}^n P(j|i)\overline{x}^{(i)}, \quad \sigma_j^2 = \frac{1}{n_j d} \sum_{i=1}^n P(j|i)\|\overline{x}^{(i)} - \overline{\mu}\|^2$$

EM in general
Observed data x, latent variable z (e.g., cluster labels)
$l(\overline{\theta}; x) = \sum_{i=1}^n \log \sum_{z^{(i)}} \mathrm{P}(\overline{x}^{(i)}, z^{(i)}; \overline{\theta})$
E-step: compute expectations to "fill in" missing values according to current estimate of $\overline{\theta}$, compute $\mathrm{P}(z^{(i)} = k|\overline{x}^{(i)}; \overline{\theta})$
M-step: re-estimate parameters with "weighted" values.

$$\overline{\theta}^{t+1} = \arg\max_{\overline{\theta}} \sum_i \sum_k \mathrm{P}(z^{(i)} = k|\overline{x}^{(i)}; \overline{\theta}^{(t)}) \log \mathrm{P}(z^{(i)} = k, \overline{x}^{(i)}; \overline{\theta}^{(t)})$$

Bayesian Networks
For a given graph, the joint distribution can be written as a product of the conditional probability of each variable given its parents

$$P\left(X_1, \ldots, X_d\right) = \prod_{i=1}^d P\left(X_i|X_{pa_i}\right)$$

Three Rules

1. $P(X,Y) = P(X|Y)P(Y)$
2. $P(X) = \sum_Y P(X,Y)$
3. $\sum_X P(X|Y) = 1$

d-separation

Step 1: keep only ancestral graph of the variables of interest

Step 2: ("moralize") add undirected edge between any two variables in ancestral graph that have a common child & change to undirected.

Step 3: (i) if there is **no** path between variables of interest, they are marginally independent.

(ii) if **all** paths go through a particular node, then the variables are independent given that node.

MLE learning on Bayesian Networks

log-likelihood: $l(\theta; S_n, G) = \sum_{t=1}^{n} \sum_{i=1}^{d} \log \theta_i(x_i^{(t)}|x_{pai}^{(t)})$

Corresponding MLE:

$$\hat{\theta}_i(x_i|x_{pai}) = \frac{\#(X_i = x_i, X_{pai} = x_{pai})}{\sum_{x_i'} \#(X_i = x_i', X_{pai} = x_{pai})}$$

Bayesian Information Criterion, n is number of training data

$$\text{BIC}(D;\overline{\theta}) = l(D;\overline{\theta}) - \frac{\#\,\text{param}}{2}\log(n)$$

M^{th} order Markov Model

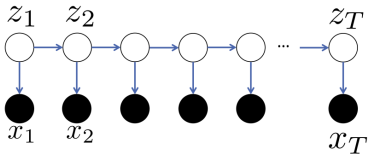
Assumption: the t^{th} observation is independent of previous observations given the $(t\text{–}1)^{\text{st}}$,...,$(t\text{–}M)^{\text{th}}$ observations

Joint pdf: $P(x_1, \ldots, x_T) = \ldots \Pi_{t=M+1}^{T} P(x_t|x_{t-1}, \ldots, x_{t-M})$

Number of parameters: $O(k^{M+1})$

Hidden Markov Model (HMM)

Assumption: the hidden variable (z's) are discrete random variables (x's are observed).



Transition Probabilities: $A(h_i, h_j) = P(z_{t+1} = h_j | z_t = h_i)$

Emission Probabilities: $B(h_i, o_l) = P(x_t = o_l | z_t = h_i)$

Starting State Probability: $\pi(h_i) = P(z_1 = h_i)$

Joint probability distribution: $P(x_1, \ldots, x_T, z_1, \ldots, z_T) = \pi(z_1) \prod_{t=1}^{T-1} A(z_t, z_{t+1}) \prod_{t=1}^{T} B(z_t, x_t)$

Viterbi Algorithm (Dynamic Programming)

Input: the observations $x_1, \ldots, x_T$, and model parameterss $\theta = \{\Pi, A, B\}$

Output: Infer the underlaying $z_1, \ldots, z_T$, namely,

$$\underset{z_1,\ldots,z_T}{\text{argmax}} P(x_1, \ldots, x_T, z_1, \ldots, z_T; \theta)$$

If we define $A(z_0, z_1) = \Pi(z_1)$, then one can define

$$P(x_1, \ldots, x_T, z_1, \ldots, z_T; \theta) = \prod_{t=1}^{T} (A(z_{t-1})B(z_t, x_t))$$

Define $r(z_1, \ldots, z_k) = \prod_{t=1}^{k} (A(z_{t-1})B(z_t, x_t))$, $1 \le k \le T$.

Define $s(k, v)$ to be set of all sequences of length k that end in $z_k = v$.

Define $\psi(k, v) = \max_{s(k,v)} r(z_1, \ldots, z_k)$.

Base case: $\psi(1, v) = \Pi(v)B(v, x_1)$.

Recursive steps:

$$\psi(k, v) = \max_{u \in H} \{\psi(k-1, u)A(u, v)B(v, x_k)\}$$

Time complexity: $O(M^2T)$.

Backtrack: $\hat{z_T} = \arg\max_{v \in H} \psi(v, T)$.

$\hat{z_{T-1}} = \arg\max_{u \in H} \{\psi(T-1, u)A(u, \hat{z_T})\}$

$\hat{z_{T-2}} = \arg\max_{u \in H} \{\psi(T-2, u)A(u, \hat{z_{T-1}})\}$ ...

Parameter Estimation

MLE: $\Pi(v) = \frac{\#(z=v)}{\sum_{v' \in H} \#(z=v')}$, $A(z, z') = \frac{\#(z \to z')}{\sum_{v \in H} \#(z \to v)}$,

$B(z, x) = \frac{\#(z \to x)}{\sum_{x' \in O} \#(z \to x')}$.

Stuffs before Midterm

Lagrange Multipliers:

$\min_{\overline{\theta}} f(\overline{x}; \overline{\theta})$ subject to $h_i(\overline{x}; \overline{\theta}) \le 0, \forall i = 1, \ldots, n$ is equivalent to

$$\min_{\overline{\theta}} \max_{\overline{\alpha}} \quad f(\overline{x}; \overline{\theta}) + \sum_{i=1}^{n} \alpha_i h_i(\overline{x}; \overline{\theta}) \text{ s.t. } \alpha_i \ge 0, \forall i = 1, \ldots, n$$

For $h_i < 0$, $\alpha_i = 0$, for $h_i = 0$, $\alpha_i > 0$, for $h_i > 0$, $\alpha_i = \infty$.

Shannon Entropy and Information Gain

$$H(X) = -\sum_{i=1}^{k} \Pr(X = x_i) \log_2 \Pr(X = x_i)$$

$$H(Y|X = x_i) = -\sum_{j=1}^{k} \Pr(Y = y_j | X = x_i) \log_2 \Pr(Y = y_j | X = x_i)$$

$$H(Y|X) = \sum_{i=1}^{m} \Pr(X = x_i) \cdot H(Y|X = x_i)$$

$$IG(X, Y) = H(Y) - H(Y|X)$$