

basics

- sigmoid(z) = $\frac{1}{1+e^{-z}} = \frac{e^z}{e^z+1}$
- $p(x|\mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right]$

linear algebra

- SVD: $A = U\Sigma V^T$
- PSD matrices: $A \succeq 0 \Rightarrow A = UDU^T$, D are evals, U are evecs
- PCA: $A = U\Sigma V^T$, row of V^T are principal components, project onto them to reduce dimension of data
- Use PCA to find direction of maximum variance in data; $X^T X$ is covariance

info theory

- $H(X) = -\sum p(x) \log p(x)$
- $H(Y|X) = -\sum_x p(x) \sum_y p(y|x) \log p(y|x)$
- $D(p||q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$
- $I(X;Y) = \sum_X \sum_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$
- Jensen's: $E[f(X)] > f(E[X]) \sim f$ convex

optimization

- $0 \preceq \frac{mI}{\text{strong convexity}} \preceq \nabla^2 f(x) \preceq \frac{MI}{\text{smoothness}}$
- exact line search: min over stepsize
- backtracking line search: keep doubling while below some approx
- newton's: $\theta^{(t+1)} = \theta^{(t)} - \nabla^2 f(\theta^{(t)})^{-1} \nabla f(\theta^{(t)})$
- strongly convex:
 $f(x_2) \geq f(x_1) + \nabla f(x_1)^T (x_2 - x_1) + m/2 \|x_2 - x_1\|_2^2$

Uninformed Search/ A^*

- Uninformed Search
 - * Depth First Search (DFS): search of deepest/leftmost solution (not complete)
 - * Breadth First Search (BFS): search of shallowest/leftmost solution (complete)
 - * Uniform Cost Search (UCS): search expanding minimum cost (complete and optimal)
- Informed Search, use heuristics to estimate goal distance
 - * Greedy Search: expand node closest to goal
 - * A^* : search according to heuristic
 - * A^* tree search with admissibility $h(n) \leq h^*(n)$ will yield optimal solution
 - * A^* graph search with consistency $h(A) - h(C) \leq \text{cost}(A, C)$ will yield optimal solution

Constraint Satisfaction Problems

- Consist of variables (X_i), one domain per var (D_i), constraints (C).
- k -consistency: for any set of $k-1$ vars and for any consistent assignment to those variables, a consistent value can be assigned to the k th variable. Path-consistency is 3-consistent, arc-consistency is 2-consistent, node consistency 1-consistent.

- Backtracking search occurs when a path leads to an inconsistent assignment. Tricks can be employed here: Minimum-remaining values heuristic breaks assignment ties, conflict directed back-jumping can choose good points to backtrack to,
- Tree-structured CSPs can be topologically sorted: then, solved with a forward pass.

Game Trees

- $\alpha-\beta$ pruning: makes solving minimax trees more efficient
- For large trees, use evaluation functions to estimate minimax value at a state
- Rational agents follow principle of maximum utility

Decision theory / VPI

- $EU(\alpha, \mathbf{e}) = \max_{\mathbf{a}} \sum_{\mathbf{s}'} \mathbf{P}(\mathbf{RESULT}(\mathbf{a}) = \mathbf{s}' | \mathbf{a}, \mathbf{e}) \mathbf{U}(\mathbf{s}')$
- $EU(\alpha_{e_j}, \mathbf{e}, \mathbf{e}_j) = \max_{\mathbf{a}} \sum_{\mathbf{s}'} \mathbf{P}(\mathbf{RES}(\mathbf{a}) = \mathbf{s}' | \mathbf{a}, \mathbf{e}, \mathbf{e}_j) \mathbf{U}(\mathbf{s}')$
- $VPI_{\mathbf{e}}(E_j) = (\sum_k P(E_j = e_{jk} | \mathbf{e}) EU(\alpha_{e_{jk}} | \mathbf{e}, \mathbf{E}_j = \mathbf{e}_{jk}) - EU(\alpha | \mathbf{e}))$

MDPs

- $Q(s, a)$ = expected discounted rewards with s, a
- $U(s) = \max$ expected discounted rewards with s
- Bellman: $U(s) = \max_a \sum_{s'} \underbrace{P(s'|s, a)[R(s, a, s') + \gamma U(s')]}_{Q(s, a)}$
- val: iterate bellman, pol: iterate with no max
- $\pi(s) = \underset{a}{\operatorname{argmax}} U^\pi(s)$

Passive RL

- ADP: learn $P(s'|s, a)$, $U(s) \rightarrow$ plug into Bellman eqn
- TD: $U(s) = (1 - \alpha)U(s) + \alpha[R(s) + \gamma U(s')]$

Active RL

- (TD) Q-learning:
 $Q(s, a) = (1 - \alpha)Q(s, a) + \alpha[R(s) + \gamma \max_{a'} Q(s', a')]$
- SARSA: $Q(s, a) = (1 - \alpha)Q(s, a) + \alpha[R(s) + \gamma Q(s', a')]$
- evaluation functions + policy search
- approximate: $w_i = w_i + \alpha \cdot \text{diff} \cdot f_i(s, a)$ (derivative of linear $Q(s, a)$)
- diff = what used to be rhs

logic

- propositional logic: only facts
 - * horn clause (at most one positive), definite clause (exactly one positive)
 - * TT-ENTAILS: check everything (forward/backward chaining)
 - * DPLL - TT-ENTAILS w/ improvements
- first-order logic: facts, objects, relations
 - * inference: forward-chaining, backward chaining

Planning

- Goal: to achieve stated goals with a plan of action.
- Developed language: Planning Domain Definition Language, PDDL. Consists of *States* (collection of atomic fluents), *Available Actions (s)* (applicable if preconditions are satisfied by *s*), *Results(s,a)*: specified in terms of what changes, and *Goal Test*. Comparable to search of Ch.7 but broader, more expressive.
- Complexity of planning: planSAT (is there a plan that solves the goal?) and bounded planSAT (plan that solves in $\leq k$ steps?) are both PSPACE, solvable by deterministic Turing Machine in poly time.
- Backwards and Forward search exist, forward more popular because more heuristics exist (ignore preconditions heuristic, ignore delete lists heuristics, etc.)

Knowledge Representation

- Ontology: question of whether things exist, and how to organize facts about the world
- Upper ontologies break existence into hierarchies (exception-ridden without gradations to express non-absolutes.)
- General Ontology must unify all special purpose domains (may be impossible.)
- Categorization creates hierarchies, which allows qualities to be inherited down to subcategories

linear regression

- convergence criteria: $0 < \alpha < 0.5\lambda_{\max}(X^T X)$
- normal eq: $\hat{w} = (X^T X)^{-1} X^T y$
- schur comp. like 2d det but divide by inverse (e.g. M/A)

Independence and Factorization

- $P(x_1, \dots, x_n) = \prod_i P(X_i | \text{Parents}(X_i))$
- Node is conditionally independent of all other nodes given its parents, children, and children's parents (Markov Blanket)
- $X_A \perp X_C | X_B \Rightarrow p(x_A, x_C | x_B) = p(x_A | x_B) p(x_C | x_B)$ or $p(x_A | x_B, x_C) = p(x_A | x_B)$
- Edges do not imply dependence, but lack of edges imply independence
- d-separation: use bayes ball algorithm to find conditional independencies
- Undirected GM: $\prod_{C \in \text{cliques}} \phi(C)$ potential function

Clustering, Mixture of Gaussians, k-means

- ex. GMMs: $p(x|\theta) = \sum_i \pi_i \mathcal{N}(x|\mu_i, \Sigma_i)$
- ex. mixture of linear regressions:

$$p(y|x, \theta) = \sum_i \underbrace{\pi_i(x, \xi)}_{\text{mixing prop.}} \cdot \underbrace{\mathcal{N}(y|\beta_i^T x, \sigma_i^2)}_{\text{mixture comp.}}$$

EM Algorithm

- $L(q, \theta) = \sum_z q(z|x) \log \frac{p(x, z|\theta)}{q(z|x)}$
- Expectation: assign to observed *X*'s the unobserved (latent) variable classes. With *q* as averaging distribution, $q^{(t+1)} = \text{argmax}_q L(q, \theta^{(t)})$
- Maximization: holding assignments constant, compute MAP or MLE estimates of cluster parameters.
 $\theta^{(t+1)} = \text{argmax}_\theta L(q^{(t+1)}, \theta)$

logistic regression

- $P(Y = 1|x, w) = \text{sigmoid}(w^T x)$
- minimizes cross-entropy: $-\sum_z P(y = z) \log P(\hat{y} = z)$

decision trees

- info gain: $H(\text{parent}) - E[H(\text{children})]$

svms

- soft-margin:

$$\min_{w, b, \xi} \quad \frac{1}{2} \|w\|_2^2 + C \sum_i \xi_i \quad (1)$$

$$s.t. \quad y_i(w^T x_i - b) \geq 1 - \xi_i \quad \forall i \quad (2)$$

$$\xi_i \geq 0 \quad \forall i \quad (3)$$

- binary: $\min_{w, b} \quad \frac{1}{2} \|w\|^2 + C \sum_i \max(1 - y_i(w^T x_i - b), 0)$

duality

- primal: $p^* = \min f_0(x)$
 $s.t. \quad f_i(x) \leq 0$
 $h_i(x) = 0$
- dual: $d^* = \max_{\lambda, \nu} \inf_x \overbrace{f_0(x) + \sum \lambda_i f_i(x) + \sum \nu_i h_i(x)}^{\text{dual function } g(\lambda, \nu)}$
 $s.t. \quad \lambda \succeq 0$
 $\text{Lagrangian } L(x, \lambda, \nu)$

kernel methods

- matrix XX^T has each element be dot product

statistical concepts

- $\overbrace{p(\theta|x)}^{\text{posterior}} = \frac{\overbrace{p(x|\theta)}^{\text{likelihood}} \overbrace{p(\theta)}^{\text{prior}}}{p(x)}$
- $\hat{\theta}_{MLE} = \text{argmax}_\theta p(x|\theta)$
- $\hat{\theta}_{MAP} = \text{argmax}_\theta p(\theta|x)$
- $\hat{\theta}_{Bayes} = \int \theta p(\theta|x) d\theta$