10-701 Cheat Sheet

Distributions

Gaussian: $\ln \frac{1}{(2\pi)^{D/2}} \frac{1}{|\mathbf{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \mathbf{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\}$; **multinomial**: $p(\boldsymbol{x}|\boldsymbol{\mu}) = \prod \mu_k^{x_k}$, where only one of x_i is 1, and others are 0; **binary**: $\text{Bern}(x|\mu) = \mu^x(1-\mu)^{1-x}$; **binomial**: $\text{Bin}(m|N,\mu) = \binom{N}{m}\mu^m(1-\mu)^{N-m}$, with expectation $N\mu$ and variance $\mu(1-\mu)$. (reduced to binary for $N=1$)

Non-Parametric

MaxLikelihood learning window will give you delta functions, which is a kind of over fitting. Use Leave-one-out cross validation for model selection. Idea: Use some of the data to estimate density; Use other part to evaluate how well it works. Pick the parameter that works best.
$\log p(x_i|X \setminus \{x_i\}) = \log \frac{1}{n-1}\sum_{j\neq i} k(x_i,x_j)$, the sum over all points is $\frac{1}{n}\sum_{i=1}^n \log\left[\frac{n}{n-1}p(x_i) - \frac{1}{n-1}k(x_i,x_i)\right]$ where $p(x) = \frac{1}{n}\sum_{i=1}^n k(x_i,x)$.
why must we not check too many parameters? that you can overfit more; for a given dataset, a few particular parameter values might happen to do well in k-fold CV by sheer chance, where if you had a new dataset they might not do so well. Checking a reasonable number of parameter values makes you less likely to hit those "lucky" spots helps mitigate this risk.
Silverman's Rule for kernel size Use average distance from k nearest neighbors $r_i = \frac{r}{k}\sum_{x\in \text{NN}(x_i,k)} \|x_i - x\|$.
Watson Nadaraya 1. estimate $p(x|y=1)$ and $p(x|y=-1)$; 2. compute by Bayes rule
$p(y|x) = \frac{p(x|y)p(y)}{p(x)} = \frac{\frac{1}{m_y}\sum_{y_i=y} k(x_i,x)\cdot \frac{m_y}{m}}{\frac{1}{m}\sum_i k(x_i,x)}$. 3. Decision boundary
$p(y=1|x) - p(y=-1|x) = \frac{\sum_j y_j k(x_j,x)}{\sum_i k(x_i,x)} = \sum_j y_j \frac{k(x_j,x)}{\sum_i k(x_i,x)}$
Actually, we assume that p(x—y) is equal to $1/m_y * \sum_y k(x_i,x)$. Using this definition, we can see $p(x,-1)+p(x,1) = p(x|-1)p(-1) + p(x|1)p(1) = p(x)$.
This can be incorporated into the regression framework in chap 6 of PRML. Where we define $f(x-x_n, t\neq t_n) = 0$, and $f(x-x_n, t=t_n) = f(x-x_n)$. Using this definition, we can derive all the probabilities on this slide. (see my handwritten notes on chap 6 of PRML).
Regression case is the same equation.
kNN Let optimal error rate be p. Given unlimited **iid** data, 1NN's error rate is $\leq 2p(1-p)$.

Matrix Cookbook

$\frac{\partial \boldsymbol{x}^T\boldsymbol{a}}{\partial \boldsymbol{x}} = \frac{\partial \boldsymbol{a}^T\boldsymbol{x}}{\partial \boldsymbol{x}} = \boldsymbol{a}$
$\frac{\partial \boldsymbol{a}^T\boldsymbol{X}\boldsymbol{b}}{\partial \boldsymbol{X}} = \boldsymbol{a}\boldsymbol{b}^T$, $\frac{\partial \boldsymbol{a}^T\boldsymbol{X}^T\boldsymbol{b}}{\partial \boldsymbol{X}} = \boldsymbol{b}\boldsymbol{a}^T$, $\frac{\partial \boldsymbol{a}^T(\boldsymbol{X}^T|\boldsymbol{X})\boldsymbol{a}}{\partial \boldsymbol{X}} = \boldsymbol{a}\boldsymbol{a}^T$
$W \in \boldsymbol{S}$, $\frac{\partial}{\partial \boldsymbol{s}}(\boldsymbol{x}-\boldsymbol{A}\boldsymbol{s})^T\boldsymbol{W}(\boldsymbol{x}-\boldsymbol{A}\boldsymbol{s}) = -2\boldsymbol{A}^T\boldsymbol{W}(\boldsymbol{x}-\boldsymbol{A}\boldsymbol{s})$,
$\frac{\partial}{\partial \boldsymbol{x}}(\boldsymbol{x}-\boldsymbol{s})^T\boldsymbol{W}(\boldsymbol{x}-\boldsymbol{s}) = 2\boldsymbol{W}(\boldsymbol{x}-\boldsymbol{s})$,
$\frac{\partial}{\partial \boldsymbol{s}}(\boldsymbol{x}-\boldsymbol{s})^T\boldsymbol{W}(\boldsymbol{x}-\boldsymbol{s}) = -2\boldsymbol{W}(\boldsymbol{x}-\boldsymbol{s})$,
$\frac{\partial}{\partial \boldsymbol{x}}(\boldsymbol{x}-\boldsymbol{A}\boldsymbol{s})^T\boldsymbol{W}(\boldsymbol{x}-\boldsymbol{A}\boldsymbol{s}) = 2\boldsymbol{W}(\boldsymbol{x}-\boldsymbol{A}\boldsymbol{s})$,
$\frac{\partial}{\partial \boldsymbol{A}}(\boldsymbol{x}-\boldsymbol{A}\boldsymbol{s})^T\boldsymbol{W}(\boldsymbol{x}-\boldsymbol{A}\boldsymbol{s}) = -2\boldsymbol{W}(\boldsymbol{x}-\boldsymbol{A}\boldsymbol{s})\boldsymbol{s}^T$.
$\text{Tr}(\boldsymbol{A}) = \sum_i \boldsymbol{A}_{ii}$. For two equal sized matrices, $\text{Tr}(\boldsymbol{A}^T\boldsymbol{B}) = \text{Tr}(\boldsymbol{B}^T\boldsymbol{A}) = \text{Tr}(\boldsymbol{A}\boldsymbol{B}^T) = \text{Tr}(\boldsymbol{B}\boldsymbol{A}^T) = \sum_{i,j}\boldsymbol{A}_{ij}\boldsymbol{B}_{ij}$.
$\text{Tr}(\boldsymbol{A}) = \text{Tr}(\boldsymbol{A}^T), \text{Tr}(\boldsymbol{A}+\boldsymbol{B}) = \text{Tr}(\boldsymbol{A}) + \text{Tr}(\boldsymbol{B})$. For square matricis,
$\text{Tr}(\boldsymbol{A}\boldsymbol{B}) = \text{Tr}(\boldsymbol{B}\boldsymbol{A}), \text{Tr}(\boldsymbol{A}\boldsymbol{B}\boldsymbol{C}) = \text{Tr}(\boldsymbol{C}\boldsymbol{A}\boldsymbol{B}) = \text{Tr}(\boldsymbol{B}\boldsymbol{C}\boldsymbol{A})$ (trace rotation).

Classifers and Regressors

Naive Bayes Conditionally independent:
$P(x_1,x_2,\ldots|C) = \prod_i P(x_i|C)$. One way to avoid divide by zero: add $(1,1,\ldots,1)$ and $(0,0,\ldots,0)$ to both classes.
Learns $P(x_i|y)$ for *Discrete* x_i – $P(x_i|y) = \frac{\#D(X_i=x_i,Y=y)}{\#D(Y=y)}$
For smoothing, use $P(x_i|y) = \frac{\#D(X_i=x_i,Y=y)+k}{\#D(Y=y)+n_i k}$, where n_i is the number of different possible values for X_i (In practice problem set, Jing Xiang used $k=1$?) *Continuous* x_i – Can use any PDF, but usually use Gaussian
$P(x_i|y) = \mathcal{N}(\mu_{X_i|y}, \sigma^2_{X_i|y})$, where $\mu_{X_i|y}$ and $\sigma_{X_i|y}$ are, respectively, the average and variance of X_i for all data points where $Y=y$. The Gaussian distribution already provides smoothing.
Perceptron Produces linear decision boundaries. *Classifies* using $\hat{y} = X_{test}\, w + b$ *Learns* w and b by updating w whenever $y_i(w^T x_i + b) \leq 0$ (i.e. incorrectly classified). Updates as $w \leftarrow w + x_i y_i, b \leftarrow b + y_i$ Repeat until all examples are correctly classified. w is some linear combination $\sum_i \alpha_i x_i (y_i * x_i)$ of data points, and decision boundary is the linear hyperplane $f(x) = w^T x + b$. **Note** that the perceptron is the same as stochastic gradient descent with a hinge loss function of $max(0, 1 - y_i[< w, x_i > + b])$ (**we can't remove 1 in the loss function; otherwise we can set** $w, b = 0$).
Convergence of perceptron proof 1 Here we use a perceptron without b. Assume we have $\boldsymbol{w}^*$ that has margin γ $(\min(\boldsymbol{w}^*)^T y_i x_i = \gamma\ )$, and $\|\boldsymbol{w}^*\| = 1, \|x_i\| = 1$. We start from $\boldsymbol{w}_0 = 0$. Assume that we have made M mistakes. We have 1) $\boldsymbol{w}_M \cdot \boldsymbol{w}^* = (\boldsymbol{w}_{M-1} + y_i x_i)\cdot \boldsymbol{w}^* \geq \boldsymbol{w}_{M-1}\cdot \boldsymbol{w}^* + \gamma$. So we have $\boldsymbol{w}_M \cdot \boldsymbol{w}^* \geq M\gamma$.
2) $\boldsymbol{w}_M \cdot \boldsymbol{w}_M = (\boldsymbol{w}_{M-1} + y_i x_i)\cdot(\boldsymbol{w}_{M-1} + y_i x_i) = \boldsymbol{w}_{M-1}\cdot \boldsymbol{w}_{M-1} + 2y_i x_i \cdot \boldsymbol{w}_{M-1} + (y_i x_i)\cdot(y_i x_i) \leq \boldsymbol{w}_{M-1}\cdot\boldsymbol{w}_{M-1} + 1$. So we have $\boldsymbol{w}_M \cdot \boldsymbol{w}_M < M$.
Combining them, using Cauchy-Schwarz, we have $M\gamma \leq \boldsymbol{w}_M \cdot \boldsymbol{w}^* \leq \|\boldsymbol{w}_M\|\|\boldsymbol{w}^*\| \leq \sqrt{M}$. So $M \leq 1/\gamma^2$. **proof 2** Let potential function $Q_i = \|\boldsymbol{w}_i\| - \boldsymbol{w}_i \cdot \boldsymbol{w}^*$, where i is the number of iterations. Assuming up to iteration i, we have M mistakes, so we have $Q_i \leq \sqrt{M} - M\gamma$. Clearly $Q_i \geq 0$ by Cauchy-Schwarz. So we have $\sqrt{M} - M\gamma \geq 0$.
Linear Regression For $y = \boldsymbol{\beta}^T\boldsymbol{x}$, $\boldsymbol{\beta}^* = (X^TX)^{-1}X^T\boldsymbol{y}$, where $X \in \mathbb{R}^{n\times d}$. If we add a regularizing term $\lambda\|\boldsymbol{\beta}\|^2$, $\boldsymbol{\beta}^* = (X^TX + \lambda I)^{-1}X^T\boldsymbol{y}$. Kernalized version of ridge regression: $\boldsymbol{\alpha}^* = (XX^T + \lambda I)^{-1}y, \boldsymbol{\beta}^* = X^T\boldsymbol{\alpha}^*$.

Kernel

Kernel function $k(\boldsymbol{x},\boldsymbol{x}') = \phi(\boldsymbol{x})^T\phi(\boldsymbol{x}')$ for some $\phi(\cdot)$. For a set of data points $\{\boldsymbol{x}_i\}$, we have Gram matrix (kernel matrix) $K_{ij} = k(\boldsymbol{x}_i,\boldsymbol{x}_j)$. A **necessary and sufficient** condition for being a valid kernel function: K always positive semidefinite.
Proof: $\boldsymbol{\alpha}^T K\boldsymbol{\alpha} = \sum_{ij}\alpha_i\alpha_j K_{ij} = \sum_{ij}\alpha_i\alpha_j\langle\phi(\boldsymbol{x}_i),\phi(\boldsymbol{x}_j)\rangle = \langle\sum_i \alpha_i\phi(\boldsymbol{x}_i), \sum_j \alpha_j\phi(\boldsymbol{x}_j)\rangle \geq 0$.
Mercer's Theorem for any symmtric function $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ which is square integrable and satisfying $\int_{\mathcal{X}\times\mathcal{X}} k(x,x')f(x)f(x')\mathrm{d}x\mathrm{d}x' \geq 0$ for $f \in L_2(\mathcal{X})$, we have a feature space $\Phi(x)$ and $\boldsymbol{\lambda} \geq 0$ that $k(x,x') = \sum_i \lambda_i\phi_i(x)\phi_i(x')$.
new kernel from old ones Given kernels $k_1(x,x'), k_2(x,x')$, $ck_1(x,x'), f(x)k_1(x,x')f(x'), k_1(x,x') + k_2(x,x'), k_1(x,x')k_2(x,x')$ are new valid kernels. Proof: 1) write the kernel as the dot product of two vectors; 2) use Mercer's Theorem; 3) any Gram matrix derived from it is positive semidefinite. $k_1(x,x') - k_2(x,x')$ is **invalid**: let $k_1(x,x') = 1$ for $x = x'$, and 0 otherwise, and $k_2(x,x') = 2k_1(x,x')$. The Gram matrix of new kernel is not PSD.
PSD matrices Products of two PSD matrices are not always PSD. Let A be 2×2 PSD, and B be diag(1, 2). AB is not PSD (columns scaled differently). **PSD's eigen decomposition** $A = UDU^T$. $A^{m+1} = AA^m = UDU^T(UD^mU^T) = UD(U^TU)D^mU^T = UD^{m+1}U^T$. **Any PSD is a covariance matrix** Let $\boldsymbol{x}$ be a random vector with covariance I, PSD Q is the covariance matrix for $Q^{1/2}\boldsymbol{x}$: $\text{cov}(Q^{1/2}\boldsymbol{x}) = Q^{1/2}\text{cov}(\boldsymbol{x})Q^{1/2} = Q^{1/2}Q^{1/2} = Q$.
examples polynomial: $(\langle x,x'\rangle + c)^d, c \geq 0$. For $c = 0$, it's a polynomial having all terms of order d; for $c > 0$, it contains all terms of order up to d. **gaussian rbf** $\exp(-\lambda\|x - x'\|^2)$. **laplacian rbf** $\exp(-\lambda|x - x'|^2)$.

Convexity

Convex Sets

Definition: A set C is *convex* if the line segment between any two points in C lies in C, i.e. if for any $x_1, x_2 \in C$ and any θ with $0 \leq \theta \leq 1$, we have

$$\theta x_1 + (1-\theta)x_2 \in C$$

Examples:

- Empty set $\emptyset$, single point x_0, the whole space $\mathbb{R}^n$
- Hyperplane $\{x|a^Tx = b\}$, halfspaces $\{x|a^Tx \leq b\}$
- Euclidean balls $\{x|\|x - x_c\|_2 \leq r\}$
- Positive semidefinite marices $S_+^n = \{A \in S^n | A \succeq 0\}$ (S^n is the set of symmetric $n\times n$ matrices)

Convexit preserving set operations:

- Translation $\{x + b|x \in C\}$
- Scaling $\{\lambda x|x \in C\}$
- Affine function $\{Ax + b|x \in C\}$
- Intersection $C \cap D$
- Set sum $C + D = \{x + y|x \in C, y \in D\}$

Convex Functions

Definition: A function $f : \mathbb{R}^n \to \mathbb{R}$ is *convex* if $\mathbf{dom} f$ is a convex set and if for all x, $y \in \mathbf{dom} f$, and θ with $0 \le \theta \le 1$, we have

$$f(\theta x + (1-\theta)y) \le \theta f(x) + (1-\theta) f(y)$$

First-order conditions: Suppose f is differentiable. Then f is convex if and only if $\mathbf{dom} f$ is convex and

$$f(y) \ge f(x) + \triangledown f(x)^T (y - x)$$

Second-order conditions: Assume that f is twice differentiable. Then f is convex if and only if $\mathbf{dom} f$ is convex and its Hessian is positive semidefinite: for all $x \in \mathbf{dom} f$,

$$\triangledown^2 f(x) \succeq 0$$

Strict convexity: Whenever $x \neq y$ and $0 < \theta < 1$, we have

$$f(\theta x + (1-\theta)y) < \theta f(x) + (1-\theta) f(y)$$

Or

$$f(y) > f(x) + \triangledown f(x)^T (y - x)$$

Or **sufficient but not necessary condition:**

$$\triangledown^2 f(x) \succ 0$$

Strong convexity: There exists an $m > 0$ such that

$$f(y) \ge f(x) + \triangledown f(x)^T (y - x) + \frac{m}{2} \|y - x\|_2^2$$

Or

$$\triangledown^2 f(x) \succeq mI$$

Convex function examples:

- Exponential. e^{ax} is convex on $\mathbb{R}$, for any $a \in \mathbb{R}$
- Powers. x^a is convex on $\mathbb{R}_{++}$ when $a \ge 1$ or $a \le 0$, and concave for $0 \le a \le 1$
- Powers of absolute value. $|x|^p$ for $p \ge 1$ is convex on $\mathbb{R}$
- Logatithm. $\log x$ is concave on $\mathbb{R}_{+}+$
- Norms. Every norm on $\mathbb{R}^n$ is convex
- $f(x) = \max x_1, ..., x_n$ is convex on $\mathbb{R}^n$
- Log-sum-exp. $f(x) = \log(e^{x_1} + ... + e^{x_n})$ is convex on $\mathbb{R}^n$

Convexity preserving function operations Convex functions $f(x), g(x)$

- Nonnegative weighted sum: $af(x) + bg(x)$
- Pointwise maximum: $f(x) = \max f_1(x), ..., f_m(x)$
- Composition with affine function: $f(Ax + b)$
- Composition with nondecreasin convex function g: $g(f(x))$

Duality

primal problem (standard form): $\min f_0(x), \text{s.t.} f_i(x) \le 0, i = 1, \ldots, m, h_i(x) = 0, i = 1, \ldots, p$. **Lagrangian**: $\mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to R$, $L(x, \lambda, \nu) = f_0(x) + \sum_i \lambda_i f_i(x) + \sum_i \nu_i h_i(x)$. **Lagrange dual function** $\mathbb{R}^m \times \mathbb{R}^p \to R$, $g(\lambda, \nu) = \inf_{x \in \mathcal{D}} L(x, \lambda, \nu)$. g is concave (can be $-\infty$ for some λ, ν). **lower bound property** if $\lambda \succeq 0$, $g(\lambda, \nu) \le p^*$. **Lagrange dual problem** $\max g(\lambda, \nu), \text{s.t.} \lambda \succeq 0$. **KKT conditions** For (x^*, λ^*, ν^*), we have 1) $f_i(x^*) \le 0$, 2) $h_i(x^*) = 0$, 3) $\lambda_i^* \ge 0$, 4) $\lambda_i^* f_i(x^*) = 0$, 5) $\nabla_x L(x, \lambda, \nu) = 0$. Or 1) primal constraints; 2) dual constraints; 3) complementary slackness; 4) gradient of Lagrangian with respect to x is zero. If strong duality holds, then optimal (x^*, λ^*, ν^*) must satisfy KKT. For a convex problem, finding (x^*, λ^*, ν^*) satifsying KKT means they're optimal (and proves the problem has strong duality). If Slater's condition is satisfied, x is optimal iff there exists (λ, ν) that satisfy KKT.
dual for LP for standard LP $\min c^T x, \text{s.t.} Ax = b, x \succeq 0$, we have dual $\max -b^T \nu, \text{s.t.} A^T \nu - \lambda + c = 0, \lambda \succeq 0$. for inequality LP $\min c^T x, \text{s.t.} Ax \preceq b$, we have the dual problem $\max -b^T \lambda, \text{s.t.} A^T \lambda + c = 0, \lambda \succeq 0$.

SVM

primal form (hard margin problem, C is $+\infty$) $\min_{w,b} 1/2\|w\|^2, \text{s.t.} (w^T x_i + b) y_i \ge 1$. **Lagrangian** $L(w, b, \alpha) = 1/2\|w\|^2 - \sum_i \alpha_i [(w^T x_i + b) y_i - 1]$. Minimize w.r.t. w, b, we have $\partial_w L(w, b, \alpha) = w - \sum_i \alpha_i y_i x_i = 0, \partial_b L(w, b, \alpha) = \sum_i \alpha_i y_i = 0$.
dual form Plugging back, we have $\max -1/2 \sum_{ij} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle + \sum_i \alpha_i$, with constraints $\sum_i \alpha_i y_i = 0$ and $\alpha_i \ge 0$. After solving this, we can preserve the input vectors that have associated $\alpha_i > 0$. These are support vectors. b can be solved using support vectors, since they satisfy the inequality tightly. **why large margin** Maximum robustness relative to uncertainty;
soft margin problem with slack variables $\min_{w,b} 1/2\|w\|^2 + C \sum_i \xi_i, \text{s.t.} (w^T x_i + b) y_i \ge 1 - \xi_i$ and $\xi_i \ge 0$. A trivial solution (upper bound is number of samples) is $w = 0, b = 0, \xi_i = 1$. **Lagrangian** $L(w, b, \xi, \alpha, \eta) = 1/2\|w\|^2 - \sum_i \alpha_i [(w^T x_i + b) y_i + \xi - 1] - \sum_i \eta_i \xi_i$. Minimize w.r.t. w, b, ξ, we have the third one $C - \alpha_i - \eta_i = 0$.
dual form Plugging back, we have the same objective, with additional box constraints on $a_i \in [0, C]$. By limiting α_i, we set limit on each sample vector's impact on the decision boundary. Make the result more robust. When C is small, more errors made; C is large, converge to hard margin case. C regulates the size of $\|w\|^2$.
Basically, as C increases, the margin becomes narrower.
kernel trick by changing $\langle x_i, x_j \rangle$ by $k(x_i, x_j)$, we have kernelized nonlinear version. With increasing C, the boundaries becomes more and more wiggly. Increasing C allows for more nonlinearities. Decreases number of errors
risk and loss $C \sum_i \xi_i$ can be reformulated as $C \max[0, 1 - y_i[\langle w, x_i \rangle + b]$, without constraint. $\max[0, 1 - y_i[\langle w, x_i \rangle + b]$ is hinge loss function. Other choices are possible.
optimally, the loss function should be 1 for $1 - y_i[\langle w, x_i \rangle + b`0$, and 0 otherwise. Hinge loss function is a convex approximation of it.
logistic: $\log[1 + e^{-f(x)}]$, Huberized loss: 0 for $f(x) > 1$, $0.5(1 - f(x))^2$ for $f(x) \in [0, 1]$, and $0.5 - f(x)$ for $f(x) < 0$. Choosing a quadratic penality when ξ_i is small means we don't care small loss that much.