CS189 Midterm 1 Study Guide

Equations

Probability:

Normal distribution	$\frac{1}{\sigma\sqrt{2\pi}}\exp(-(x-\mu)^2/(2\sigma^2))$
Multivariate normal	$\frac{1}{\sqrt{(2\pi)^k\|\Sigma\|}}\exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$
Poisson distribution	$p(x,\lambda)=(e^{-\lambda}\lambda^x)/x!$
Bayes' rule	$P(A\|B)=P(A)P(B\|A)/P(B)$
Covariance	$\mathbb{E}[(x_i-\mu_i)(x_j-\mu_j)]x$
Expectation	$\mathbb{E}[cX]=c\mathbb{E}[X]$
Variance	$Var(cX)=c^2Var(X)$

Matrices:

$(AB)^T$	B^TA^T
$(A^T)^{-1}$	$(A^{-1})^T$
$\partial(x^Ta)/\partial x$	a
$\partial(x^TAx)/\partial x$	$(A+A^T)x$
$\partial Trace(XA)/\partial X$	A^T
$\partial(a^TXb)/\partial X$	ab^T

Support Vector Machines

Linear SVM: classifier that draws decision boundary. hyperparameter C (tradeoff between training error vs. model complexity, allows points to be misclassified)

Kernel Trick:

parametric	$f(x)=w\cdot\Phi(x)$
	$w=\sum_k\alpha_k\Phi(x^k)$
non-parametric	$f(x)=\sum_k\alpha_k k(x^k,x)$
	$k(x^k,x)=\Phi(x^k)\cdot\Phi(x)$

Learning Rules

Gradient Descent

Stochastic: Differentiate regularized loss w/ respect to w, find Δw proportional to the negative gradient. Obtain dual form $\Delta\alpha$ by using kernel trick, NOT by differentiating loss. Does not exploit convexity of risk function; best approach for big data but NOT when N or d is small.

Hebb's rule:

$w=\sum_k y^kx^k$

$w\leftarrow w+y_k\Phi(x^k)$

$\alpha_k\leftarrow\alpha_k+y_k$

Perceptron uses same rule, but only updates on misclassification $y_kf(x^k)<0$.

Update rules:

perceptron	$\Delta w_i=\eta y\Phi_i(x_i)$ if $z\le 0$
	$\Delta\alpha_k=\eta y^k$
large margin perceptron:	$\Delta w_i=\eta y\Phi_i(x_i)$ if $z\le 1$
optimum margin perceptron:	$\Delta w_i=\eta y\Phi_i(x_i)$ if $min(z)$
least mean squares:	$\Delta w_i=\eta(y-f(x))\Phi_i(x)$
	$\Delta\alpha_k=\eta(y^k-f(x^k))$

Risk Minimization

Risk/Loss Functionals

risk	$(1/N)\sum_{k=1:N}L(f(x^k,w),y^k)$
true gradient	$w\leftarrow w-\eta\nabla_wR$
stochastic gradient	$w\leftarrow w-\eta\nabla_wL$
SRM/regularization	$w_i\leftarrow(1-\gamma)w_i-\eta\partial R_{train}/\partial w_j$
linear discriminant	$f(x)=\sum_i w_ix_i=wx$
functional margin	$z=yf(x),y=\pm1$

Risk Types

guaranteed risk: $R_{gua}[f]=R_{train}[f]+\epsilon(\delta,C/N)$ with high probability $(1-\delta)$

regularized risk: $R_{reg}[f]=R_{train}[f]+\lambda\|w\|^2$

Regularization: penalizes model complexity at expense of more training error, often explicitly part of loss function.

Norms: $\|x\|_p=(|x_1|^p+|x_2|^p+\cdots)^{1/p}$

L0 norm	penalizes number of features considered
L1 norm	makes weight vector more sparse
L2 norm	shrinks weight vector to reduce variance

Hessian: $H=[\partial^2R/\partial w_i\partial w_j]$

Logistic Regression

Like Hebb's rule but weighted; misclassifications are more heavily weighted (multiplied by $S(-z)$.

Linear logistic regression

$\log[P_f(Y=1|X=x)/P_f(Y=-1|X=x)]=w\cdot x+b$

logistic function: $S(t)=g^{-1}(t)=1/(1+e^{-t})$

$$R(f)=(1/N)\sum_{k=1:N}\ln(1+e^{-z}) \quad (1)$$

$$\begin{aligned}\Delta w_i&=-\eta\partial L/\partial w_i=-\eta\partial L/\partial z.\partial z/\partial w_i\\&=\eta S(-z)y\Phi_i(x)\end{aligned}$$

Update equations:

$$\begin{aligned}\Delta w_i&=(-\gamma w_i)+\eta S(-z)y\Phi_i(x)\\\Delta\alpha_k&=(-\gamma\alpha_k)+\eta S(-z)y^k \text{ for example k}\\\Delta\alpha_h&=(-\gamma\alpha_h)\text{ for other examples}\end{aligned}$$

Ridge Regression

$\sum_i w_ix_i^k=y^k$ for all $k=1..m$

$$\begin{aligned}Xw^T&=y\\X^TXw^T&=X^Ty\\w^T&=(X^TX)^{-1}X^Ty\end{aligned}$$

Optimal solution: $w^T=X^+y$

Pseudo-inverse

Case 1) $N>d$ overdetermined, no exact solution. Optimal RSS solution is

$X^+=\lim_{\lambda\to0}(X^TX+\lambda I)^{-1}X^T$

Case 2) $N<d$ underdetermined, optimize for $\min(\|w\|)$

$X^+=\lim_{\lambda\to0}X^T(XX^T+\lambda I)^{-1}$

Not limit when $\lambda\to0$, but find optimal value through cross-validation.

Residual: $y-\hat{y}=(I-XX^+)y$

Kernel trick: In case 2, dimensionality of features can approach ∞. Instead, replace XX^T by a (N,N) kernel matrix $K=k(x^k,x^h)$. $\alpha=(K+\lambda I)^{-1}y$ yields the nonlinear regression function $f(x)=\sum_k\alpha_kk(x,x_k)$.

Principal Component Analysis (PCA): decrease dimensionality of features by constructing linear combinations of the features such that the reconstructed patterns are as close as possible to the original features (minimize RSS). We do this by removing the dimensions with the smallest eigenvalues (smallest variance, affects the data the least).

Kernel Machines

Kernels are dot products in a potentially infinite Φ space. Good kernels are symmetric: $k(x,x')=k(x',x)$. Kernel matrix should be invertible, possibly after regularization $(K+\lambda I)$. Satisfied if matrix is PSD, all eigenvalues ¿ 0.

$f(x)=\sum_{k=1:N}y_kk(x,x_k)$

Radial kernels:

Parzen window: $k(x,x_k)=1(\|x-x_k\|^2<\sigma^2)$

Gaussian kernel: $\exp-(\|x-x_k\|^2/2\sigma^2)$

Non-radial kernels:

Linear kernel: $k(x,x_k)=x\cdot x_k$

Polynomial kernel: $k(x,x_k)=(1+x\cdot x_k)^q$

Bayesian Decision Theory

Datasets should be IID.

Bonferroni Correction: $p'=mp$, where we use m classifiers.

Kernel Methods

Performance Evaluation

Model Selection

Gaussian Classification

Assumptions: Generating model (draw y first, draw x given y), variance in dataset explained by Gaussian noise, independence of features in given class (no covariance), same variance for all classes. NOT optimum Bayes classifier because assumptions almost always violated. We can post-fit bias term by adjusting the threshold.

If two classes have same variance, Gaussian classifier is a linear discriminant (equivalent to centroid method, Hebb's rule with target values $1/N_1$ and $-1/N_0$).

Bayes rule: $P(X,Y)=P(X)P(Y|X)=P(Y)P(X|Y)$

Gaussian classifier:

$P(Y=y|X=x)\ P(Y=y)P(X=x|Y=y)$

Prior: $P(Y=y)$, relative class abundance (occurrences over total count)

Likelihood: $P(X=x|Y=y)$, probability x belongs to class y, proportional to $\exp(-\|x-\mu^{|y|}\|^2/2\sigma^2)$

LDA

Data transforms:

Centering: subtracting mean of the features

Standardizing (sphering): subtracting by mean, dividing by standard deviation (component-wise)

Whitening: multiply by square root of inverse covariance matrix, $\Phi=X(X^TX)^{-1/2}$

Linear Discriminant Analysis:

Generalization of Gaussian classifier for cases where the input variables aren't statistically independent, but all classes have same covariance matrix

PCA, ridge regression use covariance matrix of all data combined. LCA uses pooled, within-class covariance

Useful for multi-class classification and data visualization.