1 General

P-Norm: $||x||_p = (\Sigma_{i=1}^n |x_i|^p)^{\frac{1}{p}}$
Frobenious Norm: $||A||_F = \sqrt{\Sigma_{i,j} a_{ij}^2}$
Chain rule: $D(f(g(x))) = Df(g(x)) * Dg(x)$
Positive (semi-)definiteness: $A \in \mathbb{R}^{n\times n}$ is p.(s.)d., then A is a real, symmetric matrix and $\forall x \in \mathbb{R}^n . x^T A x > (\geq) 0$.
Joint distribution: X, Y are RVs $F_{X,Y}(x,y) = P(X\le x, Y \le y)$
Joint density: $f_{X,Y}(x,y) = \frac{\delta^2 F}{\delta x \delta y}(x,y)$
Conditional Probability: $P(A|B) = \frac{P(A\cap B)}{P(B)}$
Total probability: $P(B) = \Sigma_{i=1}^n P(B|A_i)P(A_i)$
Bayes rule: $P(A|B) = P(B|A)\frac{P(A)}{P(B)}$
Variance: $\text{Var}(X) = \mathbb{E}[(X-\mathbb{E}[X])^2] = \mathbb{E}[X^2] - \mathbb{E}[X]^2 \geq 0$
Gaussian and Multivariate (2D) Gaussian:
$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(x-\mu)^2}{2\sigma^2})$,
$f(x) = \frac{1}{2\pi\sqrt{|\Sigma|}} e^{-\frac12 (x-\mu)^T\Sigma^{-1}(x-\mu)}$.
Convex function: $f:\mathbb{R}^d\to\mathbb{R}$ is convex $\Leftrightarrow$ $x_1,x_2\in\mathbb{R}^d, \lambda\in[0,1]: f(\lambda x_1 + (1-\lambda)x_2) \le \lambda f(x_1) + (1-\lambda) f(x_2)$.
A twice differentiable function $f:\mathbb{R}^d\to\mathbb{R}$ is convex iff $\forall x\in\mathbb{R}^d$ its Hessian is p.s.d.
If $w_i \geq 0$ and f_i all conv., then $\sum_i w_i f_i$ conv.

2 Regression

Linear Regression: Goal: Measure distance between predicted and target values
$f(x) = w_1x_1 + \cdots + w_d x_d + w_0 = \tilde{w}^T\tilde{x}$ with $\tilde{w} = [w_1 \cdots w_d,\ w_0]$ and $\tilde{x} = [x_1\cdots x_d,\ 1]$.
Residual: $r_i = y_i - w^T x_i,\ x_i\in\mathbb{R}^d,\ y_i\in\mathbb{R}$.
Objective function (convex): $\hat{R}(w) = \sum_{i=1}^n r_i^2$.
$w^* = \arg\min_w \sum_{i=1}^n (y_i - w^T x_i)^2$.
Closed form solution: $w^* = (X^TX)^{-1}X^Ty$.
Gradient: $\nabla_w \hat{R}(w) = -2\sum_{i=1}^n r_i x_i^T$.
Non-linear functions: $f(x) = \sum_{i=1}^D w_i\phi_i(x)$.
Gradient Descent: 1. Start $w_0 \in \mathbb{R}^d$.
2. For $t = 1,2,...$ do $w_{t+1} = w_t - \eta_t\nabla\hat{R}(w_t)$.
Empirical risk minimization: Assumption: Data set generated iid from unknown distribution P: $(x_i,y_i)\sim P(X,Y)$.
True risk: $R(w) = \int P(x,y)(y-w^Tx)^2 dxdy = \mathbb{E}_{x,y}[(y-w^Tx)^2]$
Emp. risk: $\hat{R}_D(w) = \frac{1}{|D|}\sum_{(x,y)\in D}(y-w^Tx)^2$
Generalization error: $|R(w) - \hat{R}_D(w)|$
Uniform convergence: $\sup_w |R(w) - \hat{R}_D(w)| \to 0$ as $|D|\to 0$
In general, it holds that: $\mathbb{E}_D[\hat{R}_D(\hat{w}_D)] \le \mathbb{E}_D[R(\hat{w}_D)]$, where $\hat{w}_D = \text{argmin}_w \hat{R}_D(w)$.
Cross-validation: For each model m and for $i=1:k$:
1. Split data: $D = D_{train}^{(i)} \uplus D_{val}^{(i)}$.
2. Train model: $\hat{w}_{i,m} = \arg\min_w \hat{R}_{train}^{(i)}(w)$.
3. Estimate error: $\hat{R}_m^{(i)} = \hat{R}_{val}^{(i)}(\hat{w}_{i,m})$.
Select model: $\hat{m} = \text{argmin}_m \frac1k\sum_{i=1}^k \hat{R}_m^{(i)}$.
Ridge regression: Regularization (corresponds to MAP estimation):
$\min_w \frac1n\sum_{i=1}^n (y_i - w^Tx_i)^2 + \lambda||w||_2^2$.
Closed form: $\hat{w} = (X^TX+\lambda I)^{-1}X^Ty$
Gradient: $-2\sum_{i=1}^n r_i x_i^T + 2\lambda w$
Standardization: Goal: each feature: $\mu=0$, $\sigma^2=1$: $\tilde{x}_{i,j} = \frac{(x_{i,j}-\hat{\mu}_j)}{\hat{\sigma}_j}$ with $\hat\mu_j = \frac1n\sum_{i=1}^n x_{i,j}$, $\hat\sigma_j^2 = \frac1n\sum_{i=1}^n (x_{i,j}-\hat\mu_j)^2$.

3 Classification

0/1 loss: $\ell_{0/1}(w;x,y) = [y \neq \text{sign}(w^Tx)]$
Perc. loss: $\ell_p(w;x,y) = \max(0, -yw^Tx)$
Hinge loss: $\ell_H(w;x,y) = \max(0, 1-yw^Tx)$
$\nabla_w \ell_p = \begin{cases} 0, & \text{if } w^Tx_iy_i \geq 0 \text{ (1 if } \ell_H) \\ -y_ix_i & \text{else}\end{cases}$
Fisher consistency: A surrogate loss $\psi: Y\times S\to\mathbb{R}$, is said to be consistent to the loss $L: Y\times S\to\mathbb{R}$, if every minimizer f of surrogate risk function $R_\psi(f)$ is also a minimizer of risk function $R_L(f)$. E.g. hinge and logistic losses are consistent with 0/1 loss.
SGD: GD requires sum over all data → slow.
1. Choose random initial $w_0\in\mathbb{R}^d$
2. For $t=1,2,\ldots$ do:
(a) Choose $(x,y)\in D$ u.a.r (w/ replacement)
(b) Set $w_{t+1} = w_t - \eta_t\nabla\ell(w_t;x,y)$
SGD converges if $\sum_t\eta_t=\infty$ and $\sum_t\eta_t^2<\infty$.
Mini-batch: Choose multiple datapoints at random; may converge faster.
Perceptron: SGD with ℓ_p and $\eta=1$.
$\hat w = \arg\min_w \frac1n\sum_{i=1}^n \ell_p(w;x_i,y_i)$
If data linearly separable finds separator.
SVM: SGD with ℓ_H and regularization.
$\hat w = \arg\min_w \frac1n\sum_{i=1}^n \ell_H(w;x_i,y_i) + \lambda||w||_2^2$
$w_{t+1} = w_t(1-2\eta_t\lambda) + \eta_t x_i y_i [y_i w_T x_i < 1]$
Often $\eta_t = \frac{1}{\lambda t}$. Works on non-linearly separable data, finds best separator w.r.t. ℓ_H.

4 Feature selection

Greedy: Greedily add (or remove) features to maximize cross-validated prediction accuracy w.r.t. cost $c: V\Rightarrow\mathbb{R}$ of using features in subset of V. Forward (start with empty set) is faster, but backward is more resilient to "dependent" features. Applies to any method, but slow b/c trains many models and can be suboptimal.
L1-Regularization: regularize loss with $\|w\|_1$, automatic feature selection. Can be used for regression (Lasso), classification (L1-SVM) by replacing $||w||_2^2$ with $||w||_1$. Only works for linear models, but is fast.

5 Class imbalance

Downsample loses data but fast, upsample random pertubation maybe unsafe. Use cost-sensitive metrics controlling tradeoff:
$\ell_{CS} = c_y\ell(w;x,y)$.

		True label		
		Positive	Negative	Σ
Predict.	Positive	TP	FP	p_+
	Negative	FN	TN	p_-
	Σ	n_+	n_-	

Accuracy bad metric for imbalanced data.
Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$,
Precision: $\frac{TP}{TP+FP}$, TPR, Recall: $\frac{TP}{TP+FN}$,
FPR: $\frac{FP}{TN+FP}$, F1 score: $\frac{2TP}{2TP+FP+FN}$
Precision Recall Curve: Precision (y-axis) vs. Recall (x-axis). Precision = 1 and Recall = 1 is optimal. Area under curve (AUC) can be used for comparison of algos.
Receiver Operator Characteristic (ROC) Curve: TPR (y-axis) vs. FPR (x-axis). Random guessing achieves TPR = FPR line. TPR > FPR is better than random guessing. TPR = 1 and FPR = 0 is optimal. Area under curve (AUC) can be used for comparison of algos.
Theorem: Alg 1 dominates Alg 2 in terms of ROC curve ⇔ Alg 1 dominates Alg 2 in terms of Precision Recall curve.
One-vs-all: c classifiers, one for each class, pick highest confidence. Class may not be lin. sep. from all others. Note Scaling + Imbalance.
One-vs-one $c(c-1)/2$ classifiers, voting scheme with highest number of positive prediction wins: no confidence needed.
Ideally $\mathcal{O}(\log c)$ classifiers, theor. optimum.
Multi-class SVM: c weight vectors, $w^{(y)T}x \geq \max\{w^{(i)T}x\}+1$ for correct label y.
$\ell_{MC-H}(w^{(1)},...,w^{(c)};x,y) = \max(0, 1+\max_{j\neq y} w^{(j)T}x - w^{(y)T}x)$

6 Kernels

Kernel trick: 1. Express problem s.t. it only depends on inner products $x_j^Tx_i$.
2. Replace inner products by kernels.
Reformulate problem: Fundamental insight: Optimal separating hyperplane lives in the span of data: $\hat w = \sum_{i=1}^n \alpha_i y_i x_i$, e.g. Perceptron:
$\hat w = \arg\min_w \frac1n\sum_{i=1}^n \max(0,-y_iw^Tx_i)$.
$\hat a = \arg\min_\alpha \frac1n\sum_{i=1}^n \max(0, -\sum_{j=1}^n \alpha_j y_i y_j x_j^T x_i)$.
Replace inner products: For some feature transform $\phi: x\mapsto\phi(x)$, kernels solve $\phi(x)^T\phi(x')$ efficiently as $k(x,x')$.
Perceptron example (Training):
1. Initialize $\alpha_1=\alpha_2=\cdots=\alpha_n=0$.
2. For $t=1,2,\ldots$:
(a) Pick $(x_i,y_i)\in D$ u.a.r.
(b) Predict $\hat y = \text{sign}\left(\sum_{j=1}^n\alpha_jy_jk(x_j,x_i)\right)$.
(c) If $\hat y\neq y_i$ set $\alpha_i \Leftarrow \alpha_i+\eta_t$
Kernel properties: $k: X\times X\Rightarrow\mathbb{R}$ must be *symmetric*: $k(x,x') = k(x',x)$
Gram matrix K must be p.s.d. ($\forall x. x^TKx\geq 0$) for any n, any set $\{x_1,\ldots,x_n\}\subseteq X$. All p.s.d. matrices are some kernel.

$$K = \begin{bmatrix} k(x_1,x_1) & \ldots & k(x_1,x_n)\\ \vdots & \ddots & \vdots \\ k(x_n,x_1) & \ldots & k(x_n,x_n)\end{bmatrix}$$

For k_1,k_2 kernel, $c>0$ and f polyn. with pos. coef. or exp. k_1+k_2, $k_1\cdot k_2$, $c\cdot k_1$ and $f(k_1(x,x'))$ are kernels.

Poly. degree $= d$	$(x^Tx')^d$
Poly. degree $\le d$	$(x^Tx'+1)^d$
Gaussian(RBF)	$\exp(-\|x-x'\|_2^2/(2h^2))$
Lapacian	$\exp(-\|x-x'\|_1/h)$

Note: $h>0$ is bandwidth, $h\to 0$ overfits.
k-NN: $y = \text{sign}(\sum_{i=1}^n y_i[x_i \text{ is k-NN of } x])$
No training, but depends on all data.
Can use kernel as *similarity function*: $y = \text{sign}\left(\sum_{i=1}^n y_i\alpha_ik(x_i,x)\right)$, Improved performance, depends only on wrongly classified data, can capture global trends, but requires training.
Parametric vs. nonparametric learning: *Parametric* have finite set of parameters (regression, perceptron), while *nonparametric* increase complexity with size of data (kernelized perceptron, k-NN).
Can kernelize other tasks, such as SVM: $\arg\min_\alpha \frac1n\sum_{i=1}^n\max\{0, 1-y_i\alpha^Tk_i\} + \lambda\alpha^TD_yKD_y\alpha$ with $k_i = [y_1k(x_1,x_i),...,y_nk(x_n,x_i)]$.
Linear regression: Training: $\hat\alpha = \arg\min_\alpha\frac1n\|\alpha^TK-y\|_2^2 + \lambda\alpha^TK\alpha$, Closed form: $\hat\alpha = (K+n\lambda I)^{-1}y$, Prediction: $\hat y = \sum_{i=1}^n\hat\alpha_ik(x_i,x)$
Est. kernel parameters via CV. Choosing kernels requires domain knowledge. Deal w/ overfit by regularization.

7 Neural Networks

Parametrized feature maps. Can learn nonlinear features. *Forward propagation* in layer ℓ: $(v^{(0)}=x)$: $v^{(\ell)} = \varphi(z^{(\ell)}) = \varphi(W^{(\ell)}v^{(\ell-1)})$.
Output layer: $y = f = W^{(L-1)}v^{(L-1)}$.
Weight optimization: Apply loss $\ell(y - f(x,W))$, optimize weights to minimize loss.
For multi-outputs sum losses.
Sigmoid: $\frac{1}{1+\exp(-z)}$, Tanh: $\frac{\exp(z)-\exp(-z)}{exp(z)+\exp(-z)}$,
ReLu: $\max(0,z)$. Sigmoid and Tanh: diff. everywhere, but gradient ≈ 0 if $x\neq 0$. ReLu: not diff. at 0, $\varphi'=1$ if $z>0$.
Back prop.: $\delta^{(L)} = l'(f) = [l'(f_1),...,l'(f_p)]$, and $\nabla_{W^{(L)}}\ell(W;y,x) = \delta^{(L)}v^{(L-1)T}$. For $\ell<L$: $\delta^{(\ell)} = \varphi'(z^{(\ell)})\odot(W^{(\ell+1)T}\delta^{(\ell+1)})$ and $\nabla_{W^{\ell}}\ell(W;y,x) = \delta^{(\ell)}v^{(\ell-1)T}$
Nonconvex optimization, initialization matters.

Glorot (tanh): $w_{i,j} \sim \mathcal{N}(0, 1/n_{in})$

$w_{i,j} \sim \mathcal{N}(0, 2/(n_{in}+n_{out}))$

He (ReLU): $w_{i,j} \sim \mathcal{N}(0, 2/(n_{in}))$

Learning rate η_t *decreasing* (e.g. $min(0.1, 100/t)$) to prevent oscillation.

Momentum ($m < 1, a \leftarrow m \cdot a + \eta_t \nabla_w \ell(W; y, x), W \leftarrow W - a$), avoid local minima.

Many parameters → overfitting! *Early stop* or *regularization* ($\lambda \|W\|_F^2$) to prevent. Also *dropout*, randomly set weights to 0 with prob. p, set $W = W \odot p$ after training.

Batch normalization, standardize some batch $\{x_{1...m}\}$ and set $y_i = \gamma \hat{x}_i + \beta = BN_{\gamma,\beta}(x_i)$. Then $\varphi(Wx) = \varphi(W(BN_{\gamma,\beta}(x)))$.

Convolutional NN: Apply m $f \times f$ filters to $n \times n$ image, padding p and stride s: Leaves with $\alpha \times \alpha \times m$ output, where $\alpha = \frac{n+2p-f}{s}$.

8 Unsupervised Learning

k-Means clust.: Assign point $x_i \in \mathbb{R}^d$ to nearest center $\mu_j \in \mathbb{R}^d$. $\hat{R}(\mu) = \sum_{i=1}^n \min_{j \in [k]} \|x_i - \mu_j\|_2^2$, nonconvex.

Lloyd: Initialize $\mu^{(0)}$. Then assign x_i to closest center $z_i^{(t)} = \arg\min_j \|x_i - \mu_j^{(t-1)}\|_2^2$. Then update mean: $\mu_j^{(t)} = \frac{1}{|i:z_i^{(t)}=j|} \sum_{i:z_i^{(t)}=j} x_j$.

$\mathcal{O}(nkd)$ per it., converg. pot. slowly but monotonically to local optimum → Mult. iter.

k-Means++: Let $\mu^{(0)} = x$ u.a.r. from X. Assign centers $2 \dots k$ randomly, prop. to sq. dist. to closest sel. cent. $\mathbb{E}[\text{cost}]$ is $\mathcal{O}(\log k)$ of opt. $\Pr[\mu_j = x_i] = \frac{1}{z} \min_{k \in [j-1]} \|\mu_k - x_i\|_2^2$

Choosing k difficult. Heuristic: When $k+1$ yields *diminishing returns*, or *regularization* with λk. Only models circ. clust.: use kernels.

Dimension Reduction: Embed $\{x_1, \dots, x_n\}$, $x_i \in \mathbb{R}^d$ in $\mathbb{R}^k$ where $k < d$.

PCA: *Center data* $\mu = \frac{1}{n}\sum_{i=1}^n x_i = 0$ and construct *emp. cov. matrix*: $\Sigma = \frac{1}{n}\sum_{i=1}^n x_i x_i^T$.

PCA problem: $(W, z_1, \dots, z_n) = \arg\min_{W,z} \sum_{i=1}^n \|W z_i - x_i\|_2^2$ where W orthogonal, $z_i \in \mathbb{R}^k$ has sol. $z_i = W^T x_i$.

Sol. for W is k principal eigenvectors of Σ. $W = (v_1|...|v_k)$ where $\Sigma = \sum_{i=1}^d \lambda_i v_i v_i^T$ with $\lambda_1 \geq ... \geq \lambda_d \geq 0$. Can apply to any matrix $X = USV^T$, first k columns of V are first k principal components. Choose k by CV for feature ind. or s.t. var is explained (see k-Means).

Kernel PCA: $w = \sum_{j=1}^n \alpha_j \phi(x_j)$, $K = \sum_{i=1}^n \lambda_i v_i v_i^T$, $\arg\max_{\|w\|_2=1} \sum_{i=1}^n (w^T \phi(x_i))^2 = \arg\max_{\alpha^T K \alpha = 1} \alpha^T K^T K \alpha$, $\alpha^{(i)} = v_i/\sqrt{\lambda_i}$

New point: $z_i = w^{(i)T} x = \sum_{j=1}^n \alpha_j^{(i)} k(x, x_j)$

Centering a kernel: $K' = K - KE - EK + EKE$ where $E = \frac{1}{n}[1,...,1][1,...,1]^T$.

Complexity grows with the # of data points.

Autoencoder: NN where hidden layers usually smaller (k) than in- and output (d). Try to learn identity function. Compr. from input to smallest HL, Decompr. from smallest HL to output. $f(x;\theta) = f_{dec}(f_{enc}(x;\theta_1);\theta_2)$. If $\varphi(z) = z$ then autoencoder is equivalent to PCA.

9 Probabilistic modeling

Bayes optimal predictor: $h^*(x) = \mathbb{E}[Y \mid X = x]$, unattainable in pr. Can try to estimate conditional distr. $\hat{P}(Y \mid X)$.

Parametric estimation: Have $\hat{P}(Y \mid X, \theta)$, MLE: $\theta^* = \arg\max_\theta \hat{P}(Y \mid X, \theta) = \arg\min_\theta -\sum_{i=1}^n \log \hat{P}(y_i \mid x_i, \theta)$

E.g. Gauss. noise, lin. reg.: $y_i \sim \mathcal{N}(w^T x, \sigma^2)$ $\arg\min_\theta = \frac{n}{2}\log(2\pi\sigma^2) \sum_{i=1}^n \frac{(y_i - w^T x_i)^2}{2\sigma^2}$, MLE equiv. to LSQ est.. In general, MLE with gauss. noise with const. var. equiv. to LSQ sol.

Bias Variance Tradeoff: Prediction error = Bias² + Variance + Noise. Bias: Risk of best model compared to minimal risk given $P(X,Y)$. Variance: Risk due to estimating model from limited data. Noise: Risk by optimal model. Trade bias and variance via model selection / regularization

MAP est.: $\arg\max_w P(w|x_{1:n}, y_{1:n}) = \arg\max_w \frac{P(w)P(y_{1:n}|x_{1:n},w)}{P(y_{1:n}|x_{1:n})} = \arg\min_w -\log P(w) - \log P(y_{1:n}|x_{1:n}, w)$

For Gaussian noise and Gaussian prior: MAP = Ridge regression. Regularized estimation can often be understood as MAP inference: $\arg\min_w \sum_{i=1}^n \ell(w^T x_i; x_i, y_i) + C(w) = \arg\max_w \prod_i P(y_i|x_i, w) P(w) = \arg\max_w P(w|D)$ where $C(w) = -\log P(w)$ and $\ell(w^T x_i; x_i, y_i) = -\log P(y_i|x_i, w)$.

1. Choose likelihood function → loss function
2. Choose prior → regularizer
3. Optimize for MAP parameters, choose hyperparameters through cross-validation
4. Make predictions via B. Decision Theory

Bayes optimal classifier: $h^*(x) = \arg\max_y P(Y = y \mid X = x)$

Logistic regression: $P(Y = y|x) = \frac{1}{1+\exp(-yw^T x)}$. Replaces Gauss. noise assumption with Ber. noise: $P(y|x,w) = \text{Ber}(y; \sigma(w^T x))$. MLE: $\hat{R}(w) = \sum_{i=1}^n log(1 + exp(-y_i w^T x_i))$, is convex and $\nabla_w \ell_w = \frac{1}{1+exp(-yw^T x)} exp(-yw^T x)(-yx)$, and $exp(-yw^T x) = 1$ if misclassified. Thus $\nabla_w \ell_w = \frac{-yx}{1+exp(yw^T x)}$. With L2 regularizer, take step in direction $w(1 - 2\lambda\eta_t) - \eta_t \nabla_w \ell_w(y, x)$. Nonlinear classification via kernels: $\hat{\alpha} = \arg\min_\alpha \sum_{i=1}^n \log(1 + \exp(-y_i \alpha^T K_i)) + \lambda \alpha^T K \alpha$ and $\hat{P}(y|x, \hat{\alpha}) = \frac{1}{1+\exp(-y\sum_{j=1}^n \alpha_j k(x_j, x))}$ with $w = \sum_i \alpha_i x_i$.

M-class: $P(Y = i|x, w_{[c]}) = \frac{\exp(w_i^T x)}{\sum_{j=1}^c \exp(w_j^T x)}$.

Can obtain class probablities, but dense sols.

10 Decision Theory

Have $P(y \mid x)$, actions $\mathcal{A}$ and cost $C : \mathcal{Y} \times \mathcal{A} \to R$. Min. Exp. cost: $a^* = \arg\min_a \mathbb{E}_y[C(y,a) \mid x]$. Asymmetric costs: Assume $A = \{-1, +1\}, c_{FP}, c_{FN} > 0$. Let $\hat{P}(y \mid x) = p$, then $\mathbb{E}_y[C(y, +1)] = (1-p)c_{FP}$ and $\mathbb{E}_y[C(y,-1)] = pc_{FN}$. Predict $+1$ when $p > \frac{c_{FP}}{c_{FP}+c_{FN}}$. Uncertainty sampling: Ask user to label example we most uncertain about: $i_t \in \arg\min_i |0.5 - \hat{P}(Y_i|x_i)|$. Active learning violates i.i.d. assumption.

11 Generative Modeling

Est. *joint distribution* $P(X, Y)$ instead of $P(Y \mid X)$. Pred. $y = \text{sign}(f(x))$

Cond. distr. derived from joint: 1. Est. $P(Y)$, 2. Est. $P(x|y) \forall y$, 3. Use Bayes' rule: $P(y|x) = P(y)P(x|y)/P(x)$ where $P(x) = \sum_{y'} P(x, y')$. For $c = 2$ *discriminant function* $f(x) = \log \frac{P(Y=1|x)}{P(Y=-1|x)}$, which is +1 if $P(Y = 1 \mid x) > 0.5$.

Naive Bayes: Model y as categorical, and features conditionally independent given y, e.g. Gaussian NB, assumes $P(x_i|y) = \mathcal{N}(x_i|\mu_{y,i}, \sigma^2_{y,i})$. Is linear class., equiv. to log. reg. if assumpt. are met. $f(x) = w^T x + w_0$ where $w_0 = \log \frac{p_+}{1-p_+} + \sum_{i=1}^d \frac{\mu^2_{-,i} - \mu^2_{+,i}}{2\sigma_i^2}$ and $w_i = \frac{\mu_{+,i} - \mu_{-,i}}{\sigma_i^2}$. Cond. indep. assumpt. can make pred. overconfident. # parameters = $O(cd)$, Compl. (mem. + inference) lin. in d.

Categorical Naive Bayes: $P(X_i = c|Y = y) = \theta^{(i)}_{c|y}$. MLE prior: $\hat{p}_y = \frac{\#y}{n}$, MLE feat. distr.: $\theta^{(i)}_{c|y} = \frac{\text{Count}(X_i=c, Y=y)}{\#y}$. Exponentially (in d) parameters, Way to overfit.

Gaussian Bayes: $P(x|y) = \mathcal{N}(x; \mu_y, \Sigma_y)$. MLE prior: $\hat{p}_y = \frac{\#y}{n}$, MLE feat. distr.: $\hat{\mu}_k = \frac{1}{\#y}\sum_{i:y_i=y} x_i$ and $\hat{\Sigma}_y = \frac{1}{\#y}\sum_{i:y_i=y}(x_i - \hat{\mu}_y)(x_i - \hat{\mu}_y)^T$. $f(x) = \log\frac{p}{1-p} + \frac{1}{2}[\log\frac{|\hat{\Sigma}_-|}{|\hat{\Sigma}_+|} + ((x - \hat{\mu}_-)^T \hat{\Sigma}_-^{-1}(x - \hat{\mu}_-)) - ((x - \hat{\mu}_+)^T \hat{\Sigma}_+^{-1}(x - \hat{\mu}_+))]$. Capt. corr. of feat., no overconf., # param. = $O(cd^2)$, compl. quadr. in d.

Fischer's LDA: Assume $p = 0.5$ and $\hat{\Sigma}_- = \hat{\Sigma}_+ = \hat{\Sigma}$. Then $f(x) = x^T \hat{\Sigma}^{-1}(\hat{\mu}_+ - \hat{\mu}_-) + \frac{1}{2}\left(\hat{\mu}_-^T \hat{\Sigma}^{-1}\hat{\mu}_- - \hat{\mu}_+^T \hat{\Sigma}^{-1}\hat{\mu}_+\right)$.

Equiv. to log. reg. if assump. are met. LDA can be viewed as proj. to a 1-dim. subsp. that maxes ratio of between-class and within-class var.. PCA (k=1) maxes the var. of the res. 1-dim. proj.. Gen. model, used to detect outliers, not rob. against viol. of normality of X assumpt..

Conjugate distr.: Posterior is same family as prior. Ex.: Beta$(\theta, \alpha_+, \alpha_-)$ and Beta$(\theta, n_+ + \alpha_+, n_- + \alpha_-)$.

12 Generalized Mixture Models

Labels maybe unknown, want to cluster data. Model $P(x, \theta)$ as *conv. comb.* of Gauss. distr.: $\sum_{i=1}^c w_i \mathcal{N}(x; \mu_i, \Sigma_i)$ with $w_i > 0$ and $\sum_i w_i = 1$. $(\mu^*, \Sigma^*, w^*) = \arg\min -\sum_{i=1}^n \log \sum_{j=1}^k w_j \mathcal{N}(x_i; \mu_j, \Sigma_j)$.

Constr. (Σ p.s.d.) hard to maint. in SGD. Fitting GMM = Train a GBC without labels.

Hard-EM: E-step: Pred. most likely class for all x_i: $z_i^{(t)} = \arg\max_z P(z \mid x_i, \theta^{(t-1)}) = \arg\max_z P(z|\theta^{(t-1)})P(x_i|z, \theta^{(t-1)})$ with $\theta^{(t)} = [w^{(t)}_{1:c}, \mu^{(t)}_{1:c}, \Sigma^{(t)}_{1:c}]$. M-step: Comp. $\theta^{(t)}$ as MLE as for GBC: $\theta^{(t)} = \arg\max_\theta P(D^{(t)} \mid \theta)$.

Too much info extr. from each label, overlapping clusters not detected, fixed label if model uncertain. k-Means Algo is equiv. to Hard-EM if $w_j = 1/k$ and $\Sigma_j = I \cdot \sigma^2$.

Soft-EM: $\gamma_j(x)$ is prob. x in clust. j: $\gamma_j(x) = P(Z = j \mid x, \Sigma, \mu, w) = \frac{w_j P(x|\Sigma_j, \mu_j)}{\sum_\ell w_\ell P(x|\Sigma_\ell, \mu_\ell)}$

E-Step: Calculate prob $\gamma_j(x_i)$ for all i, j based on $\mu^{(t-1)}, \Sigma^{(t-1)}$ and $w^{(t-1)}$.

M-Step: Adjust parameters: $w_j^{(t)} = \frac{1}{n}\sum_{i=1}^n \gamma_j^{(t)}(x_i)$, $\mu_j^{(t)} = \frac{\sum_{i=1}^n \gamma_j^{(t)}(x_i) x_i}{\sum_{i=1}^n \gamma_j^{(t)}(x_i)}$ and $\Sigma_j^{(t)} = \frac{\sum_{i=1}^n \gamma_j^{(t)}(x_i)(x_i - \mu_j^{(t)})(x_i - \mu_j^{(t)})^T}{\sum_{i=1}^n \gamma_j^{(t)}(x_i)}$.

Init. sensitive, nonconvex objective. Init. w with unif. distr., μ by k-means++ and Σ as spherical (acc. to emp. var. in data). Choose k via CV. Avoid degeneracy by add. term $\nu^2 I$ to $\Sigma_j^{(t)}$ → Wishart-prior.

Semi-supervised learning: For points with label y_i: $\gamma_j^{(t)}(x_i) = [j = y_i]$

GANs: $\min_{w_G} \max_{w_D} \mathbb{E}_{x \sim \text{Data}} \log D(x; w_D) + \mathbb{E}_{z \sim \mathcal{N}} \log(1 - D(G(z; w_G); w_D))$, Mode Collapse: gen. prod. limit. variety of samples, Data memorization, Simult. training → oscillations, Can't comp. likelih. on holdout set.