0 Linear regression

Linear Model:
$f(x) = \tilde{w}^T\tilde{x} + w_0 = w^T x, x \in \mathbb{R}^d, y \in \mathbb{R}^n$
Error Measurement: $\hat{w} = \underset{f \in w}{\text{argmin}} \; ||y - Xw||^2$

$\nabla_w ||y - Xw||^2 = 2X^T(Xw - y)$, $O(nd)$, **Closed Form Sol**: $\hat{w} = (X^TX)^\dagger X^T y$, $O(nd^2)$, $n > d$: unique, $n < d$: exist

Huber loss : ignore outliers by giving less penalties

$$L_\delta(y, f(x)) = \begin{cases} 0.5 * (y - f(x))^2, & |y - f(x)| \le \delta \\ \delta * (|y - f(x)| - 0.5 * \delta), & otherwise \end{cases}$$

1 Optimization

Gradient descent update (steepest descent direction): $w^{t+1} = w^t - \eta \nabla_w L(w^t)$ Maximal Stepsize: $\eta \le \frac{2}{\lambda_{max}}$ λ: Eigenvalues of X^TX
Fastest Stepsize: $\eta = \frac{2}{\lambda_{\min} + \lambda_{\max}}$

Speeding up gradient descent: Momentum (prevent oscillation) $w^{t+1} - w^t = \alpha(w^t - w^{t-1}) - \eta \nabla L(w^t)$
Adaptive Methods (AdaGrad, RMSProp, Adam etc.) $w_i^{t+1} = w_i^t - \frac{\eta}{\sqrt{previouschange_i + \gamma}} \frac{\partial L}{\partial w_i}(w^t)$

Convexity:Guarantees local = global minimum
1. iff $L(\lambda w + (1-\lambda)v) \le \lambda L(w) + (1-\lambda)L(v)$; 2. or iff $L(v) \ge L(w) + \nabla L(w)^T(v - w)$ 3. or iff $\nabla^2 L(w)$ positive semi-definite
Operation for **preserving** Convexity: if f, g are convex
- $\alpha f + \beta g \quad \forall \alpha, \beta \ge 0$ - $h(x) = max\{f(x), g(x)\}$
g affine f convex **or** f non-decreasing and g conv: $f \circ g$
Strong convex: if $\exists m > 0, L(w) - \frac{m}{2}||w||^2$ is convex **or** $\nabla^2 L(w) > m\mathbf{I}^2$
For linear regression: Only one unique global minimum if $\nabla_X^2 L(w) = X^TX$ p.d.; many minima if p.s.d.
Effects of increasing sample size: Let d = sample dimension, n = sample number:
For noiseless case: square loss decreases when fixing d and increasing n; For noisy case, square increase and then decreases after $n \ge d$ (forced to fit the noise)

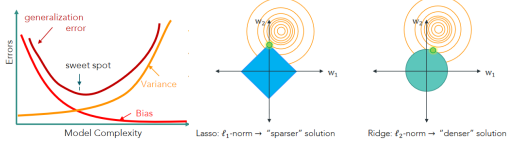
2 Model selection

Prediction Error vs. Estimation Error:
$R(f) = \mathbb{E}_{(x,y)}[(y - \hat{f}_D(x))^2] =$
$\mathbb{E}_x[(f^*(x) - \bar{f}(x))^2] + \mathbb{E}_x[(Var[\hat{f}_D(x)]] + \epsilon^2$ average prediction / generalization error = $bias^2$ + irreducible noise **cross validation**:
- (Typically choose K = 5 or 10 in practice) - Fit the model and compute the validation error on each fold k - Average the cross-validation error over K folds - Select the model with lowest CV error - Model training and evaluation(training set & test set)

LOOCV: if K very large, e.g. $K = |D_{rest}|$, we can get best approximation of $M_\Phi\{D_{rest}\}$.
Problem: Computationally intensive.

3 Regularization

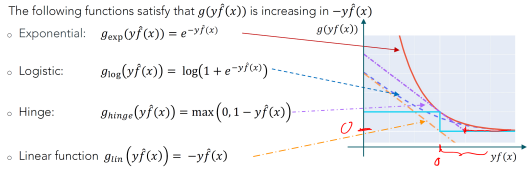
Bias of method M:
distance of average model $\bar{f} = \frac{1}{K}\sum_{k=1}^K \hat{f}_k$ to the ground truth $f^*(x)$: $\mathbb{E}_x(f^*(x) - \hat{f}(x))^2$
Variance of method M: average distance of individual models to average model $\mathbb{E}_x[\sum_{k=1}^K (f^*(x) - \hat{f}(x))^2]$
Bias variance decomposition and trade-off:



Geometric insight of lasso and ridge regression
Model complexity reflected in norms:
- The larger the norm, the larger the space in $\mathbb{R}$ you use → higher model complexity
- Fitting noise often causes norm/model complexity to increase by using more unnecessary features
Lasso regression: (convex) only use a few of these feature, encourage sparsity via limiting l_1-norm. (manually limiting the polynomial degree), **NO** close form $(\hat{w}_i^\lambda) = sign(\hat{w}_i)max(0, |(\hat{w}_i)| - \lambda))$
The problem formulation: $\arg\min_w \; ||y - \phi w||^2$ s.t. $||w||_1 \le R \rightarrow \arg\min_w \; ||y - \phi w||^2 + \lambda||w||_1$
Ridge regression The problem formulation: same as above except $||w||_2^2$ strict convex
Closed-form solution (find stationary point):
$\hat{w_\lambda} = (X^TX + \lambda I)^{-1}X^Ty$ (or use the gradient methods)
Cross-validation for λ selection
- Given a choice of features λ; - Find the best fit model for each fold:
$\hat{f}_k^\lambda = M_\lambda(D_k) = \arg\min_f L_0(f; D_k) + \lambda||f||$; - Others procedure same as normal CV

4 Classification

Average classification (generalization) error
$R(\hat{f}) = \mathbb{E}_{x,y}\mathbb{I}_{y \ne sign\hat{f}(x)}$ (the surrogate loss function is neither convex nor continuous, so it could not be used for minimizing training error!)
Convexify the surrogate losses



Logistic loss: binary classification
$\ell_{log}(\hat{f}(x), y) = log(1 + e^{-y\hat{f}(x)}) = -log(Prob(Y = y|x)$
(condi. log-likeli. param. by $\hat{f}(x) = (\hat{f}_1(x), \cdots, \hat{f}_k(x))$
via $p_i = softmax(\bar{f}_i) = \frac{exp(\alpha \bar{f}_i)}{\sum_{j=1}^K exp(\alpha \bar{f}_j)}, \alpha > 0)$

Cross-entropy loss: multi-class classification see 10.3
Maximum-Margin and SVM:
Motivation: for linearly separable data, **no unique solution** for the training loss by using the logistic loss.
General formulation of the optimization problem:
$w_{MM} = \arg\max_{||w||_2=1} margin(w)$ where
$\text{margin}(w) = \min_i y_i \langle w, x_i \rangle$
Soft-Margin SVM
If data is not linearly separable (constraints that allow some "slack" in the constraint):
$\min_{w,\xi} \frac{1}{2}||w||^2 + \lambda \sum_i \xi_i$ s.t. $y_i w^T x_i \ge 1 - \xi_i, \xi_i \ge 0$ for all i = 1, ..., n
converted to (l_2 penalized hinge loss):
$\min_w \frac{1}{2}||w||^2 + \lambda \sum_i \max(0, 1 - y_i w^T x_i)$
Area under the ROC (AUROC):
Ideal curve: higher TPR (y-axis) & lower FPR (x-axis)
F1-score:$F_1 score = \frac{2}{\frac{1}{recall} + \frac{1}{precision}}$ (Why not average? For both recall and precision to be large)
Accuracy: $\text{Acc}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_i 1(\hat{y}_i = y_i)$

Recall & Precision

Precision	$\frac{TP}{\#[\hat{y} = +1]} \sim P_n(y = 1 \mid \hat{y} = 1)$	FDR (= 1 - precision)	$\frac{FP}{\#[\hat{y} = +1]} \sim P_n(y = -1 \mid \hat{y} = 1)$
Recall (= TPR, ~ Power)	$\frac{TP}{\#[y = +1]} \sim P_n(\hat{y} = 1 \mid y = 1)$	FPR (~ Type I error)	$\frac{FP}{\#[y = -1]} \sim P_n(\hat{y} = 1 \mid y = -1)$

Other metrics in practice:
1. Worst group error (related to group fairness); 2. Advesarial pertubations (robustness againts data transformations); 3. Distribution shifts on the inputs (same label but data looks different)

5 Kernel Methods

Motivation: solve issue of feature explosion (extreme dimensionality of data): #training data= n + m-th degree polynomial features + $\phi(x) \in R^d + x \in R^p$

Dim of feature map $\phi(x)$: $p = (d+m, m) = \frac{(d+m)!}{m!d!}$ $O(d^m) = f(d)$ and $O(m^d) = f(m)$, size of total training data = np

Kernel trick Step I: global minimizer
$\hat{w} = \arg\min_{w \in R^p} \frac{1}{n}\sum_{i=1}^n l(y_i, f_w(x_i))$ has the form $\hat{w} = \phi^T\hat{\alpha}$ with $\hat{\alpha} \in R^n$ so that
$\hat{f}(x) = \sum_{i=1}^n \hat{\alpha}_i \langle \phi(x_i), \phi(x) \rangle$ and where $\hat{\alpha}$ only depends on x_i via inner products $\langle \phi(x_i), \phi(x_j) \rangle$ for $i, j = 1, ..., n$ (save memory $O(nd^m) \rightarrow O(n^2)$ - kernel matrix)
Step II: $\langle \phi(x), \phi(z) \rangle = (1 + \langle x, z \rangle)^m$ Efficient computation for the inner product of kernel function (from $O(p) \sim O(d^m)$ to $O(d+m)$) (kern.matr. K has n^2 kernels comput. complex. $O((d+m)n^2)$)
Kernelized regression:
$\hat{w} = \arg\min_w ||y - \phi w||^2 = \phi^T\hat{\alpha}$ where
$\hat{\alpha} = \arg\min_\alpha ||y - \phi\phi^T\alpha||^2 = \arg\min_\alpha ||y - K\alpha||^2$
Kernelized loss with ridge regularization:
$\arg\min_{\alpha \in \mathbb{R}^n} ||y - K\alpha||^2 + \lambda\alpha^T K\alpha$
Proof of kernel trick:
Kernel trick can limit search from R^d to S ($S = span\{\phi(x_1), ..., \phi(x_n)\}$)
Marcel's Theorem: For kernels $k : X \times X \rightarrow R$ on a compact domain $X \in R^d$, $\exists$ sequence $\{\mu_j\}_{j=1}^\infty$ and basis $\{\phi_j\}_{j=1}^\infty$ of $L_2(X)$ (a sequence of functions) such that $k(x, y) = \sum_{j=1}^\infty \mu_j\phi_j(x)\phi_j(y) = \langle \tilde{\phi}(x_j), \tilde{\phi}(y_j) \rangle$

Valid kernels symmetric + psd: $min\{x, z\}$, const. > 0, x^Tx', $|A \cap B|$, RBF kernels: $k(x, z) = e^{-\frac{||x-z||^\alpha}{\tau}}$, $\tau \downarrow$ overfit (Gaussian: $\alpha = 2$ Laplacian: $\alpha = 1$)
k-Nearest Neighbor: can be kernelized
1. Sensitive to initialization (using cross-validation for choosing k); 2. becomes erratic in high dimensions (all points become far); 3. needs large n to perform well but computation $O(nd)$, can reduce to $O(n^\rho), \rho < 1$ if allowing some error probability
Decision trees: Leaf nodes of a binary tree. **But** - can easily overfit to noise; - inaccurate as the greedy method

6 Neural Networks

Motivation: train feature maps ϕ and weights w (generally non-convex, initialization matter)
$w^* = \arg\min_{w,\theta_j} \sum_{i=1}^n l(y_i; \sum_{j=1}^m w_j\phi(x_i; \theta_j))$

Activation functions 1. Identity; 2. Sigmoid: $\frac{1}{1+exp(-z)}$; 3. Tanh: $\frac{exp(z)-exp(-z)}{exp(z)+exp(-z)}$; 4. ReLU: $max(0, z)$

Derivative of activation functions
1. Sigmoid $\phi'(z) = (1 - \phi(z))\phi(z)$; 2. ReLU $\phi'(z) = 1$ if $z > 0$; 0 if $z < 0$ (not differential at z=0, manually define derivative at 0 is 0)
Backpropagation

$$\left(\nabla_{W^{(i)}}\ell\right)^T = \frac{\partial \ell}{\partial W^{(i)}} = \frac{\partial \ell}{\partial f}\frac{\partial f}{\partial z^{(L-1)}}\frac{\partial z^{(L-1)}}{\partial z^{(L-2)}} \cdots \frac{\partial z^{(i+1)}}{\partial z^{(i)}}\frac{\partial z^{(i)}}{\partial W^{(i)}}$$

or it can rewrite into Matrix form:
$\nabla_{W^{(i)}} l = \delta^{(i)}\nu^{(i-1)T}$ where error signal:

$$\delta^{(i)} = \begin{cases} \nabla_f l & \text{, output layer} \\ \varphi'(z^{(i)}) \odot (W^{(i+1)T}\delta^{(i+1)}) & \text{, hidden layer} \end{cases}$$

and $\nu^{(i)} = \varphi(W^{(i)}\nu^{(i-1)})$

Potential Issue Exploding or vanishing gradient (solve by using certain activation function, e.g. ReLU or keeping the magnitude of $\nu^{(i)}$)
Initializing weights Goal: Keep variance of weights approximately constant across layers to avoid vani. and explod. grad. **Random initialization** Glorot ($tanh$), He (ReLU) **Avoid overfitting** Regularization (weight decay); Early stopping; Dropout p (=Pr(remain), test: $\sigma(W \times p)$)

7 Convolutional Networks

CNN vs ANN:
- Invariance to the augumentation of the training set
- Fewer parameters (parameters sharing in CNN)
- The weights can still be optimized via backprop.
Batch normalization:
1. Use on the mini-batch; 2. Reduces internal covariate shift; 3. Enables larger learning rates; 4. Has regularizing effect 5. Solve vani./expl. grad.
1. Normalize each point with mini-batch mean and variance $\hat{x}_t = \frac{x_t - \mu_s}{\sqrt{\sigma_s^2 + \epsilon}}$; 2. Scale and shift with 2 learnable parameters $\tilde{x_t} = \gamma\hat{x_t} + \beta$ **Convolutions in 1D**: Given vector $w \in \mathbb{R}^k$ and $x \in \mathbb{R}^d$, convolution result: $z_i = \sum_{j=max(1,i-d+1)}^{min(i,k)} w_j x_{i-j+1}$
Output: m $f \times f$ filters, a $n \times n$ image as input, padding p and stride s: output size = $\frac{n+2p-f}{s} + 1$
Pooling layers: aggregate several units (max/average) to decrease the width of the width of the network
Residual Connections 1. Skip connections for effectively training deeper networks; 2. Allows identity as optimal solution (avoid vanishing gradients)

8 Clustering (unsup. classification)

K-Means problem: Non-convex Opt. + NP hard
Pick centers to minimize sum of squared distances:
$\hat{R}(\mu) = \hat{R}(\mu_1, \cdots, \mu_k) = \sum_{i=1}^n \min_{j \in \{1 \cdots k\}} ||x_i - \mu_j||_2^2$
Lloyd's heuristic: 1.Initialize Cluster center 1...k
2.While not converged: Assign points to closest center and update center with mean of its points
Properties: Guaranteed to converge (to a local optimum); Sensitive to initialization; Number of iterations required can be exponential; Determining k is difficult; Cannot well model clusters of arbitrary shape
K-Means++ 1. Choose the first centroid μ uniformly rand. from X 2. For each $x \in X$ compute $D(x) := \min_j ||x - \mu_j||_2^2$ 3. Sample the next μ_j from X with probability $P(\mu_j = x) \propto D(x)^2$ 4. Repeat the last two steps until k centroids are chosen
(Expected cost is $\mathcal{O}(logk)$ times that of optimal k-Means solution $\hat{R}(\mu_{++}) \le \mathcal{O}(logk)\min_\mu \hat{R}(\mu)$)
Kernelized k-means k-means algorithm is kernelizable (objective only depends on XX^T); can use appropriate features $\phi(x)$ to cluster non-spherical and non-linearly separable clusters using k-means.

9 Dimension reduction (unsup. reg)

PCA: Compress data with low dim. representation
$(\boldsymbol{w}^*, \boldsymbol{z}^*) = \underset{||\boldsymbol{w}||_2^2=1, \boldsymbol{z}}{\text{argmin}} \sum_{i=1}^n ||z_i\boldsymbol{w} - \boldsymbol{x}_i||_2^2$, $z_i^* = \boldsymbol{w}^T\boldsymbol{x}_i$,

$\boldsymbol{w}^* = \underset{||\boldsymbol{w}||_2^2=1}{\text{argmin}} \sum_{i=1}^n ||\boldsymbol{w}\boldsymbol{w}^T\boldsymbol{x}_i - \boldsymbol{x}_i||^2$

$= \underset{||\boldsymbol{w}||_2^2=1}{\text{argmax}} \sum_i^n (\boldsymbol{w}^T\boldsymbol{x}_i)^2$ if $\boldsymbol{w}$ is a 1-D vector.(k=1)

$(\boldsymbol{W}, \boldsymbol{z}_1, ..., \boldsymbol{z}_n) = \underset{\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}_k, \boldsymbol{z}}{\text{argmin}} \sum_i^n ||\boldsymbol{W}\boldsymbol{z}_i - \boldsymbol{x}_i||_2^2$

with orthogonal $\boldsymbol{W} = [\boldsymbol{v}_1|...|\boldsymbol{v}_k] \in \mathbb{R}^{d \times k}$ and $\boldsymbol{z}_i = \boldsymbol{W}^T\boldsymbol{x}_i$ where $\boldsymbol{v}_{1...k}$ are the first k columns of $\boldsymbol{V}$ ($SVD : \boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$)

Empirical Covariance: $\mathbf{\Sigma} = \frac{1}{n}\sum_i \boldsymbol{x}_i\boldsymbol{x}_i^T$ with Eigenvalues $\lambda_1 \geq \lambda_2 \geq ... \Rightarrow \mathbf{\Sigma} = \sum_i \lambda_i \boldsymbol{v}_i\boldsymbol{v}_i^T$
$\frac{1}{n}\ \underset{\boldsymbol{W}^T\boldsymbol{W}=\boldsymbol{I}_k}{\text{argmin}}\ \sum ||\boldsymbol{W}\boldsymbol{z}_i - \boldsymbol{x}_i||_2^2 = \sum_{i=k+1}^d \lambda_i$

Kernel PCA: $w = \sum_{j=1}^n \alpha^{(j)}\phi(x_j)$

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}^T\boldsymbol{K}\boldsymbol{\alpha}=1}{\text{argmax}}\ \boldsymbol{\alpha}^T\boldsymbol{K}^T\boldsymbol{K}\boldsymbol{\alpha} = \underset{\boldsymbol{\alpha}}{\text{argmax}}\frac{\boldsymbol{\alpha}^T\boldsymbol{K}^T\boldsymbol{K}\boldsymbol{\alpha}}{\boldsymbol{\alpha}^T\boldsymbol{K}\boldsymbol{\alpha}}$$

with solution $\alpha^{(i)} = \frac{1}{\sqrt{\lambda_i}}v_i$ from $K = \sum \lambda_i v_i v_i^T$

A new point x is projected by: $z_i = \sum_{j=1:n}\alpha_j^{(i)}k(x_j, x)$

Autoencoders initialization matters
$f(x,\theta) = f_{dec}(f_{enc}(x;\theta_{enc});\theta_{dec})\quad \mathbb{R}^d \to \mathbb{R}^{k<d} \to \mathbb{R}^d$ If linear activ. func., AE (non-conv.) equivalent to PCA.

10 Statistical perspective

10.1 Estimate data distribution

Generalization:
Assume data is generated iid. Goal: identify a hypothesis $f : X \to Y$ that minimized **expected loss** (**prediction error**, **population risk**) :
$R(f) = \int p(x,y)\ell(y;f(x))dxdy = \mathbb{E}_{x,y}[\ell(y,f(x))]$
Empirical risk: $\hat{R}_D(f) = \frac{1}{|D|}\sum_{(x,y)\in D}\ell(y;f(x))$

Generalization error: $|\hat{R}_D(f) - R(f)| \to 0, |D| \to \infty$
Traing data D, test data D' from the same distribution:
Solution: $\hat{f}_D = \underset{f\in F}{\arg\min}\,\hat{R}_D(f)$
Evaluation: $\hat{R}_{D'}(\hat{f}_D) = \frac{1}{|D'|}\sum_{(x,y)\in D'}\ell(y,\hat{f}_D(x))$
Obtain an overly optimistic estimate:
$\mathbb{E}_D[\hat{R}_D(\hat{f}_D)] <= \mathbb{E}_D[R_D(\hat{f}_D)]$ (biased if no test)
$\mathbb{E}_{D'}[\hat{R}_{D'}(\hat{f}_D)] = R(\hat{f}_D)$ (unbiased with test)

10.2 Regression

Optimal predictor for the squared loss:
$f^*(x) = \underset{f:X\to\mathbb{R}}{\arg\min}\,R(f) = \underset{f:X\to\mathbb{R}}{\arg\min}\,\mathbb{E}_{x,y}[\ell(y,f(x))]$
Bayes' optimal predictor for the squared loss:
$f^*(x) = \mathbb{E}[Y|X=x]$
Least-squares regression = Gaussian MLE:
Assume $y = f(x) + \epsilon, \epsilon \sim \mathcal{N}(x;0,\sigma^2)$, $f(x) = w^Tx$,
$p(y|x) = \mathcal{N}(y;w^Tx,\sigma^2)$,
$\hat{w}_{MLE} = \text{argmax}_w\ p(y_{1:n}|x_{1:n},w^Tx,\sigma^2) =$
$\text{argmin}_w\ -\sum_{i=1}^n \log\left[P(y_i|x_i;w^Tx,\sigma^2)\right] =$
$\text{argmin}_w\ n/2log(2\pi\sigma^2) + \sum_{i=1}^n(y_i - w^Tx_i)^2/(2\sigma^2)$
Ridge regression = Gaussian MAP estimation:
Assume noise $p(y|x,w)$ is iid Gaussian, $w \sim \mathcal{N}(0,\beta^2 I)$
$\text{argmax}_w\ p(w|D) =$
$\text{argmax}_w\ p(w)\prod_{i=1}^n \log P(y_i|x_i;w) =$
$\text{argmin}_w\ \frac{1}{2\beta^2}\|w\|_2^2 + \frac{1}{2\sigma^2}\sum_{i=1}^n(y_i - w^Tx_i)^2$
L1-regul.: Laplace prior: $p(x;\mu,b) = \frac{1}{2b}exp(-\frac{|x-\mu|}{b})$
Student-t likelihood:

$$P(y|x,w,\nu,\sigma^2) = \frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi\sigma^2}\,\Gamma(\frac{\nu}{2})}\left(1 + \frac{(y-w^Tx)^2}{\nu\sigma^2}\right)^{-\frac{(\nu+1)}{2}}$$

10.3 Classifier

Population risk with 0-1 loss:
$R(f) = P(y \neq f(x)) = \arg\min \mathbb{E}_{x,y}[(y \neq f(x)]$
Bayes' optimal classifier:
$f^*(x) = \arg\max_y p[Y=y|X=x]$ most probable class
Logistic regression = Bernoulli MLE:
$p(y|x) = Ber(y;\sigma(w^Tx))$
$\hat{w}_{MLE} = \text{argmax}_w\ p(y_{1:n}|x_{1:n},w^Tx,\sigma^2) =$
$\text{argmin}_w\ -\sum_{i=1}^n \log\left[P(y_i|x_i;w^Tx,\sigma^2)\right] =$
$\text{argmin}_w\ \sum_{i=1}^n log(1 + exp(-y_iw^Tx_i))$
Logistic loss is **convex**

Regularized logistic regression = Bernoulli MAP
L2 (Gaussian prior):
$\text{argmin}_w\ \sum_{i=1}^n log(1 + exp(-y_iw^Tx_i)) + \lambda\|w\|_2^2$
L1 (Laplace prior):
$\text{argmin}_w\ \sum_{i=1}^n log(1 + exp(-y_iw^Tx_i)) + \lambda\|w\|_1^2$
Classification: $P(y|x,\hat{w}) = \frac{1}{1+exp(-y\hat{w}^Tx)}$
Multi-class logistic regression:

$$P(y=i|x,\hat{w}_1,\cdots,\hat{w}_c) = \frac{exp(\hat{w}_i^Tx)}{\sum_{j=1}^c exp(\hat{w}_j^Tx)}$$

Cross-entropy loss:
$\ell(y,x;\hat{w}_1,\cdots,\hat{w}_c) = -log(p(Y=y|,x,\hat{w}_1,\cdots,\hat{w}_c))$
Kernelized logistic regression:
Learning:
$\hat{\alpha} = \text{argmin}_\alpha\ \sum_{i=1}^n log(1 + exp(-y_i\alpha^TK_i)) + \lambda\alpha^TK\alpha$
Classification: $P(y|x,\hat{\alpha}) = \frac{1}{1+exp(-y\sum_{j=1}^n\hat{\alpha}_j^Tk(x_j,x)}$

11 Bayesian decision theory

Given:
1. Conditional distribution over labels $p(y|x)$ for $y \in Y$
2. Set of actions A 3. Cost function $C : Y \times A \to \mathbb{R}$
BDT: minimize the expected cost
$a* = \text{argmin}_{a\in A}\,\mathbb{E}_y[C(y,a)|x]$

11.1 Regression:

Cond. dist: $p(y|x) = \mathcal{N}(x;f(x),\sigma^2)$, Act. set: $A = \mathbb{R}$, Cost func.:
1. $C(y,a) = (y-a)^2$: $a^* = \mathbb{E}[y|x] = f(x)$
2. Asymmetric cost:
$C(y,a) = c_1\max(y-a,0) + c_2\max(a-y,0)$:
$a^* = f(x) + \Phi^{-1}(\frac{c_1}{c_1+c_2})$, CDF for Gaussian: $\Phi(z)$

11.2 Classification:

Cond. dist: $p(y|x) = Ber(y;\sigma(f(x)))$, Act. set: $A = \{+1,-1\}$, Cost func.:
1. $C(y,a) = [y \neq a]$: $a^* = \text{argmax}_y\ p(y|x) = sign(f(x))$
$\mathbb{E}_y[C(y,a)|x] = 1 - p(y=a|x)$
2. Asymmetric cost:

$$C(y,a) = \begin{cases} c_{FP} & \text{y=-1, a=+1} \\ c_{FN} & \text{y=+1, a=-1} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$c_+ = \mathbb{E}_y[C(y,+1)|x] = (1-p)\cdot c_{FP}$, $p = p(y=+1|x)$
$c_- = \mathbb{E}_y[C(y,-1)|x] = p\cdot c_{FN}$
$a^* = 1 : c+ <= c_- \Rightarrow p >= \frac{c_{FP}}{c_{FP}+c_{FN}}$
c_{FN}/c_{FP} increases, more points classified as pos, TPR FPR increase
3. With abstention (doubtful l.r.): $A = \{+1,-1,D\}$, $c < 0.5$, Less c: more likely to choose D (doubtful)

$$C(y,a) = \begin{cases} c_m[y \neq a] & a \in \{+1,-1\} \\ c & a = D \end{cases} \quad (2)$$

$$a^* = \begin{cases} y & P(y|x) \geq 1 - c/c_m \\ D & p > c/c_m \text{ or } p < 1 - c/c_m \end{cases} \quad (3)$$

$c_+ = (1-p)c_m, c_- = pc_m, c_D = c$
Active sampling: minimize the number of labels; *violates i.i.d. assumption*; get stuck with bad model
Uncertainty sampling: Given: unlabeled examples $D_U = \{x_1,\cdots,x_n\}$, labled dataset $D_L = \{\}$. For $t = 1,2,3,\cdots$ Estimate $p(y|x)$ given current D_L
Pick unlabled example that we are most uncertain about (highest entropy): $i_t \in \text{argmin}_{x\in D_U}H(p(y|x))$
Query label y_{i_t} and set $D_L \leftarrow D_L \cup \{(x_{i_t},y_{i_t})\}$

12 Generative Model

Discriminative: $p(y|x)$ **Generative:** $p(x,y)$
Can derive condi. from joint distr., but not vice versa!
Generative models for classification:
1. $P(X,Y) = P(X)P(Y|X)$
density over inputs, probabilistic classifier
2. $P(X,Y) = P(Y)P(X|Y)$
prior over classes, appearence for each class

12.1 Bayes Classifier (supervised)

Naive Bayes Model: class prior:
$P(Y=y) = p_y\ y \in \mathcal{Y} = \{1,\cdots,c\}$
features as conditionally independent given Y:
$P(X_1,\cdots,X_d|Y) = \prod_{i=1}^d P(X_i|Y)$
Gaussian NBC: continuous RV
Learning: 1. MLE: $P(Y=y) = \frac{Count(Y=y)}{n}$
2. MLE: $P(x_i|y) = \mathcal{N}(x_i|\mu_{y,i},\sigma_{y,i}^2)$
$\hat{\mu}_{y,i} = \frac{1}{Count(Y=y)}\sum_{j:y_j=y}x_{j,i}$
$\sigma_{y,i}^2 = \frac{1}{Count(Y=y)}\sum_{j:y_j=y}(x_{j,i} - \hat{\mu}_{y,i})^2$
Prediction:
$y = \text{argmax}_{y'}\,P(y'|x) = \text{argmax}_{y'}\,P(y')\prod_{i=1}^d P(X_i|Y)$
Categorical NBC: discrete RV
Learning: 1. MLE: $P(Y=y) = \frac{Count(Y=y)}{n}$
2. MLE: $P(X_i=x|Y=y) = \theta_{x|y}^{(i)} = \frac{Count(X_i=x,Y=y)}{Count(Y=y)}$
Prediction:
$y = \text{argmax}_{y'}\,P(y'|x) = \text{argmax}_{y'}\,P(y')\prod_{i=1}^d P(X_i|Y)$
Decision rules for binary classification: Goal:
$y = \text{argmax}_{y'}P(y'|x) \Rightarrow y = sign(\log\frac{P(Y=1|x)}{P(Y=-1|x)})$
GNB, c=2, shared variance (= logistic regression = linear classifier): *discriminant function:*
$f(x) = \log\frac{P(Y=1|x)}{P(Y=-1|x)} = w^Tx + w_0$, $w_i = \frac{\mu_{+,i}-\mu_{-,i}}{\sigma_i^2}$

$$w_0 = \log\frac{\hat{p}_+}{1-\hat{p}_+} + \sum_{i=1}^d\frac{\hat{\mu}_{-,i}^2 - \hat{\mu}_{+,i}^2}{2\hat{\sigma}_i^2}$$

Class distri.: $P(Y=1|x) = \frac{1}{exp(-f(x))} = \sigma(w^Tx + w_0)$
GBC: class prior: $y \in \mathcal{Y} = \{1,\cdots,c\}$,
$P(Y=y) = p_y = \frac{Count(Y=y)}{n}$
features as generated by multivariate Gaussian:
$P(x|y) = \mathcal{N}(x;\mu_y,\Sigma_y)$, $\hat{\mu}_y = \frac{1}{Count(Y=y)}\sum_{i:y_i=y}x_i$
$\Sigma_y = \frac{1}{Count(Y=y)}\sum_{i:y_i=y}(x_i - \hat{\mu}_y)(x_i - \hat{\mu}_y)^T$
Fisher's LDA: $y = sign(f(x)) = sign(\boldsymbol{w}^T\boldsymbol{x} + w_0)$,
$\boldsymbol{w} = \hat{\Sigma}^{-1}(\hat{\mu}_+ - \hat{\mu}_-)$, $w_0 = \frac{1}{2}(\hat{\mu}_-^T\Sigma^{-1}\hat{\mu}_- - \hat{\mu}_+^T\Sigma^{-1}\hat{\mu}_+)$
Conjugate priors:

Prior / Posterior	Likelihood function
Beta	Bernoulli/Binomial
Dirichlet	Categorical/Multinomial
Gaussian (fixed covariance)	Gaussian
Gaussian-inverse Wishart	Gaussian
Gaussian process	Gaussian

12.2 Gaussian Mixture Model (unsup.)

Gaussian Mixture:
$P(x|\theta) = \sum_{y\in\{1,\cdots,i\ \cdots\}}\underbrace{w_i}_{p(y=i|\theta)}\ \underbrace{\cdot\mathcal{N}(x;\mu_i,\Sigma_i)}_{p(x|y=i,\theta)}$
Hard-EM Algorithm:
E-Step: Predict most likely class for each data point:
$y_i^{(t)} = \text{argmax}_y P(y|x_i,\theta^{(t-1)}) =$
$\text{argmax}_y P(y|\theta^{(t-1)})P(x_i|y,\theta^{(t-1)})$
M-step: Compute MLE for GBC (closed form):
$\theta^{(t)} = \text{argmax}_\theta P(D^{(t)}|\theta)$
Problem: Hard EM assigns a fixed label for uncertain x, even though the model is uncertain.
Work poorly if clusters are **overlapping**
Soft EM: *(or just EM)*
E-Step: Get cluster Membership with $\mu^{(t-1)},\Sigma^{(t-1)},w^{(t-1)}$.

$$\gamma_j(x) = P(y=j|x,\Sigma,\mu,w) = \frac{w_jP(x|\Sigma_j,\mu_j)}{\sum_l w_lP(x|\Sigma_l,\mu_l)}$$

M-Step: Fit Clusters to weighted data points.

$$w_j^{(t)} = \frac{1}{n}\sum_i\gamma_j^{(t)}(x_i);\ \mu_j^{(t)} = \frac{\sum_i\gamma_j^{(t)}(x_i)x_i}{\sum_i\gamma_j^{(t)}(x_i)}$$

$$\Sigma_j^{(t)} = \frac{\sum_i\gamma_j^{(t)}(x_i)(x_i-\mu_j^{(t)})(x_i-\mu_j^{(t)})^T}{\sum_i\gamma_j^{(t)}(x_i)}$$

Initialization: *Weights:* typically uniform distr.; *Means:* randomly initi./k-means++; *Variances:* Spherical: $\Sigma_i = \sigma_i^2I_d$, Diagonal: $\Sigma_i = diag(\sigma_1 = \cdots = \sigma_d)$, Tied: $\Sigma_1 = \cdots = \Sigma_k$
Degeneracy of GMM: For #Clusters = #points the GMM overfits to $\mu = x$, $\sigma = 0$. $\Rightarrow$ add $+\nu^2\mathbb{I}$ to the $\Sigma_j^{(t)}$ update.

For semi-supervised GMM: Set $\gamma_j^{(t)}(x_i) = \mathbb{1}_{\{j=y_i\}}$ for labeled data
To select k=#gaussians we can use cross-validation
Reasons mixture model useful: 1. Can encode assumptions about "shape" of clusters 2. Can be part of more complex statistical models E.g., classifiers 3. Probabilistic models can output likelihood $P(x)$ of a point x. (Useful for *anomaly/outlier detection*) 4. Can be naturally used for semi-supervised learning
EM vs k-means
K-Means equals to Hard-EM with equal weights $w = P(y|\theta) = \frac{1}{k}$ and spherical Variance $\Sigma_{1:k} = \sigma^2\mathbb{I}$
General Expectation Maximation:
E-Step:
$Q(\theta;\theta^{(t-1)}) = \mathbb{E}_{z_{1:n}}\left[\log P(x_{1:n},z_{1:n}|\theta)|x_{1:n},\theta^{(t-1)}\right]$
$= \sum_i\sum_{z_i}\gamma_{z_i}(x_i)\log P(z_i|\theta)P(x_i|z_i,\theta)$
which equals to computing $\gamma_z(x_i) = P(z|x,\theta^{(t-1)})$
M-Step: $\theta^{(t)} = \text{argmax}_\theta\ Q(\theta;\theta^{(t-1)})$

13 GANS

$$\min_{w_G}\max_{w_D}\underbrace{\mathbb{E}_x\log D(x;w_D)}_{\text{real images}} + \underbrace{\mathbb{E}_y\log\left[1 - D(G(y;w_G);w_D)\right]}_{\text{fake images}}$$

For a fixed generator G the optimal discriminator D is:

$$D_G^*(x) = \frac{P_{data}(x)}{P_G(x)+P_{data}(x)}$$

Simultanous gradient descent: Find saddle point

$$w_G^{(t+1)} = w_G^{(t)} - \eta_t\nabla_{w_G}M(w_G,w_D^{(t)})$$
$$w_D^{(t+1)} = w_D^{(t)} + \eta_t\nabla_{w_D}M(w_G^{(t)},w_D)$$

Duality Gap: $DG = \max_{w_D'}M(w_G,w_D') - \min_{w_G'}M(w_G',w_D)$
$DG = 0$ if w_G, w_D form a pure equilibrium, else $DG > 0$

14 Utilities

$\nabla_x x^TA = A \qquad \nabla_x a^Tx = \nabla_x x^Ta = a$
$\nabla_x b^TAx = A^Tb \qquad \nabla_x x^Tx = 2x \qquad \nabla_x x^TAx = 2Ax$
$Y = XW$: $\frac{\partial L}{\partial X} = \frac{\partial L}{\partial Y}W^T$, $\frac{\partial L}{\partial W} = X^T\frac{\partial L}{\partial W}$
Bayes: $P(y|x) = \frac{P(x|y)P(y)}{P(x)}$ $Var(x) = \mathbb{E}[X^2] - \mathbb{E}[X]^2$
Complexity ($X \in \mathbb{R}^{n\times d}$):
$X^TX \to \mathcal{O}(nd^2)$; $X^{-1} \to \mathcal{O}(d^3)$; $X^Ty \to \mathcal{O}(nd)$
Orthogonal matrix: $W^TW = I_k$; Trace: $tr(A) = tr(A^T)$
$X \sim \mathcal{N}(0,I_d), Y = AX + \mu \sim \mathcal{N}(\mu,AA^T)$
Maximum (conditional) likelihood estimation(**MLE**):
$\theta^* = \text{argmax}_\theta\hat{p}(y_1,\ldots,y_n|\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n,\theta)$
Maximum a posteriori estimate(**MAP**):
$p(w|\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n,y_1,\ldots,y_n) = \frac{p(w)p(y_1,\ldots,y_n|\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n,w)}{p(y_1,\ldots,y_n|\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n)}$

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)$$

$$f_{\mathbf{X}}(x_1,\ldots,x_k) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)}{\sqrt{(2\pi)^k|\boldsymbol{\Sigma}|}}$$

Poisson: $P(x) = \frac{e^{-\lambda}\lambda^x}{x!}$, $E(X) = V(X) = \lambda$
Exponential: $P(x;\lambda) = \lambda e^{-\lambda}$, $\mathrm{E}[X] = \frac{1}{\lambda}$, $\mathrm{Var}[X] = \frac{1}{\lambda^2}$,
Bernoulli: $f(k;p) = p^k(1-p)^{1-k}$ for $k \in \{0,1\}$,
$\mathrm{E}(X) = p$, $\mathrm{Var}[X] = pq = p(1-p)$
Binomial: $f(k;n,p) = \Pr(X=k) = \binom{n}{k}p^k(1-p)^{n-k}$
Marginal: $P(z) = \sum_{x_i\in X}P(z,x_i)$