

## 1 Maximum Likelihood Estimator

Three properties of maximum likelihood estimator:

1.  $\theta_{ML}$  is consistent.  $\theta_{ML} \rightarrow \theta_0$  when  $n \rightarrow \infty$ .
2.  $\theta_{ML}$  is asymptotically normal.  $\sqrt{n}(\theta_{ML} - \theta_0) \sim \mathcal{N}(0, I_n(\theta_0))$  when  $n \rightarrow \infty$  and  $I_n(\theta_0)$  is the fisher information.
3.  $\theta_{ML}$  is asymptotically efficient.  $\theta_{ML}$  minimizes  $\mathbb{E}(\theta - \theta_0)^2$  when  $n \rightarrow \infty$  because the asymptotic variance equals the Rao-Cramer bound (MLE is asymptotically unbiased). Note: when  $n$  is finite,  $\theta_{ML}$  is not necessarily efficient, e.g., Stein estimator is universally more efficient for single sample.

Rao-Cramer bound: for any unbiased estimator  $\hat{\theta}$  of  $\theta_0$ ,  $\mathbb{E}(\hat{\theta} - \theta_0)^2 \geq 1/I_n(\theta_0)$ , where  $I_n(\theta) = -\mathbb{E}(\frac{\partial^2}{\partial \theta^2} \log f(X; \theta) | \theta) = \mathbb{E}(\frac{\partial}{\partial \theta} \log f(X; \theta) | \theta)^2$  is the fisher information.

Sketch of Proof: define  $\Lambda = \frac{\partial \log P(X; \theta)}{\partial \theta}$ . Cauchy-Schwarz says  $\text{Cov}^2(\Lambda, \hat{\theta}) \leq \text{Var}(\Lambda) \text{Var}(\hat{\theta}) = \mathbb{E}(\Lambda^2) \text{Var}(\hat{\theta})$  because  $\mathbb{E}\Lambda = 0$ . Note that  $\text{Cov}(\Lambda, \hat{\theta}) = \mathbb{E}(\Lambda \hat{\theta}) = \int_X \hat{\theta}(x) \frac{\partial}{\partial \theta} f(x; \theta) dx = \frac{\partial}{\partial \theta} \int_X \hat{\theta}(x) f(x; \theta) dx = \frac{\partial}{\partial \theta} \mathbb{E}\hat{\theta} = 1$ . Therefore,  $\text{Var}(\hat{\theta}) \geq 1/\mathbb{E}(\Lambda^2)$ .

However, when the dimension of problem goes to infinity while keeping the data-dim ratio fixed, MLE is biased and the  $p$ -values are unreliable.

## 2 Regression

### Bias-Variance trade-off

Let  $D$  be the training dataset and  $\hat{f}$  be the predictive function.  $\mathbb{E}_D \mathbb{E}_{Y|X}(\hat{f}(X) - Y)^2 = \mathbb{E}_D \mathbb{E}_{Y|X}[(\hat{f}(X) - \mathbb{E}_{Y|X}Y)^2 + (\mathbb{E}_{Y|X}Y - Y)^2] = \mathbb{E}_D(\hat{f}(X) - \mathbb{E}(Y | X))^2 + \mathbb{E}_D(\mathbb{E}(Y | X) - Y)^2 = \mathbb{E}_D(\hat{f}(x) - \mathbb{E}_D \hat{f}(x))^2 + (\mathbb{E}_D \hat{f}(x) - \mathbb{E}(Y | X))^2 + \mathbb{E}_D(\mathbb{E}(Y | X) - Y)^2$ . It means that expected square error (training) = variance of prediction + squared bias + variance of noise. The optimal trade-off is achieved by avoiding under-fitting (large bias) and over-fitting (large variance). Note that here the variance of output is computed by refitting the regressor on a new dataset.

### Regularization

Ridge and Lasso can be viewed as MAP (maximum a posterior) estimation. A Gaussian prior on  $\beta$  is equivalent to Ridge and a Laplacian prior is equivalent to Lasso. Using

SVD, we get Ridge has built-in model selection:  $X\beta^{\text{Ridge}} = \sum_{j=1}^d [d_j^2/(d_j^2 + \lambda)] u_j u_j^T Y$  (each  $u_j u_j^T Y$  can be viewed as a model). Lasso has more sparse estimations because the gradient of regularization does not shrink as in the case of Ridge.

## 3 BLR and GP

### Bayesian Linear Regression

$Y = X\beta + \epsilon$ ,  $\epsilon \sim \mathcal{N}(0, \sigma^2)$  and a prior  $\beta \sim \mathcal{N}(0, \Lambda^{-1})$ . By Bayesian,  $\beta | X, Y \sim \mathcal{N}(\mu_\beta, \Sigma_\beta)$ , where  $\mu_\beta = (X^T X + \sigma^2 \Lambda)^{-1} X^T Y$  and  $\Sigma_\beta = \sigma^2 (X^T X + \sigma^2 \Lambda)^{-1}$ . MAP estimation of this prior (i.e.,  $\mu_\beta$ ) is equivalent to Ridge regression given  $\Lambda = \lambda I$  and  $\sigma = 1$ .

When  $\Lambda = \lambda I$ , under the prior,  $Y \sim \mathcal{N}(0, \frac{1}{\lambda} X^T X + \sigma^2 I)$ . Therefore,  $\text{Cov}(y_i, y_j) = \frac{1}{\lambda} x_i^T x_j$ . It means a prior that closer samples is more similar, i.e.,  $\text{Cov}(y_i, y_j)$  is large when  $x_i^T x_j$  is large. The kernel  $X^T \Lambda^{-1} X$  is thus called linear kernel. When a general kernel is used, Gaussian Process appears.

### Gaussian Process

#### Kernel Function

A function is a kernel iff  $k(x, x') = k(x', x)$  (symmetry) and  $\int_{\Omega} k(x, x') f(x) f(x') dx dx' \geq 0$  for any  $f \in L_2$  and  $\Omega \in \mathcal{R}^d$  (semi-positiveness in continuous case) or  $K(X)$  is a valid covariance matrix for any  $X$  (semi-positiveness in discrete case). The latter is equivalent to either (1)  $\sum_{i,j} a_i a_j K(x_i, x_j) \geq 0$  for any  $a_i, j$  and  $k_{i,j}$ , or (2)  $k(x, x') = \phi(x)^T \phi(x')$  for some  $\phi$ .

Assume  $k_{1,2}$  are valid kernels, then the following are valid kernels:

1.  $k(x, x') = k_1(x, x') + k_2(x, x')$ .
2.  $k(x, x') = k_1(x, x') \cdot k_2(x, x')$ . Proof: let  $V \sim \mathcal{N}(0, K_1)$ ,  $W \sim \mathcal{N}(0, K_2)$  and is independent to  $V$ , then  $\text{Cov}(V_i W_i, V_j W_j) = \text{Cov}(V_i, V_j) \text{Cov}(W_i, W_j) = k_1 \cdot k_2(x_i, x_j)$ .
3.  $k(x, x') = c k_1(x, x')$  for constant  $c > 0$ .
4.  $k(x, x') = f(k_1(x, x'))$  if  $f$  is a polynomial with positive coefficients or the exp. Proof: polynomial can be proved by applying the product, positive scaling and addition. Exp can be proved by taking limit on the polynomial.
5.  $k(x, x') = f(x) k_1(x, x') f(x')$ .
6.  $k(x, x') = k_1(\phi(x), \phi(x'))$  for any function  $\phi$ .

Example: RBF kernel  $k(x, y) = \exp(-\frac{1}{2\sigma^2} \|x - y\|^2) = \exp(-\frac{1}{2\sigma^2} \|x\|^2) \exp(\frac{1}{\sigma^2} x^T y) \exp(-\frac{1}{2\sigma^2} \|y\|^2)$

is valid. Since  $x^T y$  is the linear kernel and thus  $\exp(\frac{1}{\sigma^2} x^T y)$  is a valid kernel, let  $f(x) = \exp(-\frac{1}{2\sigma^2} \|x\|^2)$ , we get the RBF function equals  $f(x)k(x, y)f(y)$ , which is a valid kernel.

### Mercer's Theorem

Assume  $k(x, x')$  is a valid kernel. Then there exists an orthogonal basis  $e_i$  and  $\lambda_i \geq 0$ , s.t.  $k(x, x') = \sum_i \lambda_i e_i(x) e_i(x')$ .

### Conditional Gaussian

$\mathbb{E}(y_2 | y_1) = \mu_2 + \Sigma_{21} \Sigma_{11}^{-1} (y_1 - \mu_1)$ ,  $\text{Var}(y_2 | y_1) = \Sigma_{22} - \Sigma_{21} \Sigma_{11}^{-1} \Sigma_{12}$ .

## 4 Linear Methods for Classification

### Concept Comparison

1. Probabilistic Generative, modeling  $p(x, y)$ : (1) can create new samples, (2) outlier detection, (3) probability for prediction, (4) high computational cost and (5) high bias.
2. Probabilistic Discriminative, modeling  $p(y | x)$ : (1) probability for prediction, (2) medium computational cost and (3) medium bias.
3. Discriminative, modeling  $y = f(x)$ : (1) no probability for prediction, (2) low computational cost and (3) low bias.

### Infer $p(x, y)$ for classification problems

Use  $p(x, y) = p(y)p(x | y)$ . Since  $y$  has finite states, model  $p(y)$  and  $p(x | y)$  for different  $y$ . The modeling requires to (1) guess a distribution family and (2) infer parameters by MLE.

### Compute $p(y | x)$ by discriminant analysis (DA)

#### Linear DA

Goal: classify a sample into two Gaussian distribution with  $\Sigma_0 = \Sigma_1$ . After calculation,  $p(y = 1 | x) = 1/(1 + \exp(-\log \frac{p(x|y=1)p(y=1)}{p(x|y=0)p(y=0)})) = 1/(1 + \exp(w_1^T x + w_0))$  since the quadratic term is eliminated due to  $\Sigma_0 = \Sigma_1$ .

#### Quadratic DA

Goal: classify a sample into two Gaussian distribution with  $\Sigma_0 \neq \Sigma_1$ . After calculation,  $p(y = 1 | x) = 1/(1 + \exp(x^T W x + w_1^T x + w_0))$ .

### Optimization Methods

#### Optimal Learning Rate for Gradient Descent

Goal: find  $\eta^* = \text{argmin}_{\eta} L(w^k - \eta \cdot \nabla L(w^k))$ .

By Taylor expansion of  $L(w^{k+1})$  at  $w^k$  and solve for the optimal  $\eta$ , we get  $\eta^* = \frac{\|\nabla L(w^k)\|^2}{\nabla L(w^k)^T H_L(w^k) \nabla L(w^k)}$ .

However, naive gradient descent has two

weaknesses: (1) it often has a zig-zag behavior, especially in a very narrow, long and slightly downward valley; (2) the gradient update is small near the stationary point. This can be mitigated by adding a momentum term in the update:  $w^{k+1} = w^k - \eta \nabla L(w^k) + \mu^k (w^k - w^{k-1})$  which speeds the update towards the “common” direction.

### Newton's Method

Taylor-expand  $L(w)$  at  $w_k$  to derive the optimal  $w^{k+1}$ :  $L(w) \approx L(w) + (w - w^k)^T \nabla L(w^k) + \frac{1}{2} (w - w^k)^T H_L(w^k) (w - w^k) \Rightarrow w^{k+1} = w^k - H_L^{-1}(w^k) \nabla L(w^k)$ .

Pros: (1) better updates compared to GD since it uses the second Taylor term and (2) does not require learning rate.

Cons: requires  $H_L^{-1}$  which is expensive.

### Bayesian Method

In most cases, the posterior is intractable. Use approximation of posterior instead.

### Laplacian Method

Idea: approximate posterior near the MAP estimation with a Gaussian distribution.  $p(w | X, Y) \propto p(w, X, Y) \propto \exp(-R(w))$ , where  $R(w) = -\log p(w, X, Y)$ . Let  $w^* = \text{argmin} R(w)$  be the MAP estimation and Taylor-expand  $R(w)$  at  $w^*$ :  $R(w) \approx R(w^*) + \frac{1}{2} (w - w^*)^T H_R(w^*) (w - w^*)$ . Therefore,  $p(w | X, Y) \propto \exp(-R(w^*) - \frac{1}{2} (w - w^*)^T H_R(w^*) (w - w^*))$  and thus  $(w | X, Y) \sim \mathcal{N}(w^*, H_R^{-1}(w^*))$ .

### Bayesian Information Criterion (BIC)

We use prior  $w \sim \mathcal{N}(\mu_0, \alpha_0 I_d)$  for  $\alpha_0$  sufficiently large (little prior). Since  $p(w | X, Y) = p(w, X, Y)/p(X, Y) \approx \exp(-R(w^*) - \frac{1}{2} (w - w^*)^T H_R(w^*) (w - w^*)) / p(X, Y)$  is a Gaussian distribution, we have  $p(X, Y) \approx e^{-R(w^*)} (2\pi)^{-d/2} |H_R(w^*)|^{-1/2}$ . Therefore,  $\log p(X, Y) \approx -R(w^*) - \frac{d}{2} \log(2\pi) - \frac{1}{2} \log |H_R(w^*)| = \log p(w^*) + \log p(X, Y | w^*) - \frac{d}{2} \log(2\pi) - \frac{1}{2} \log |H_R(w^*)|$ . Further, notice that  $H_R(w^*) = \frac{\partial^2}{\partial w w^T} (-\log p(w^*) - \log p(X, Y | w^*)) = -(\alpha_0 I_d)^{-1} - N \mathbb{E}_{x,y} (\frac{\partial^2}{\partial w w^T} \log p(x, y | w^*)) \approx N \mathbf{I}$ , where  $\mathbf{I}$  is the fisher information. Therefore, we define  $\text{BIC} = -\log p(X, Y | w^*) + \frac{d}{2} \log N$  and thus  $\log p(X, Y) \approx \text{const} - \text{BIC}$ . A lower BIC means (approximately) a larger evidence (log-likelihood of samples) and thus a better model.

## LDA by loss minimization

### Perceptron

Goal: for  $y_i \in \{0, 1\}$ , find  $w$ , s.t.  $y_i w^T x_i > 0$  for any  $i$ . The classification function is  $c(x) = \text{sgn}(w^T x)$ .

$L(y, c(x)) = 0$  if  $y w^T x > 0$  and  $L(y, c(x)) = -y w^T x$  o.w. By gradient descent, the Perceptron is guaranteed to converge if (1) the data is linearly separable, (2) learning rate  $\eta(k) > 0$ , (3)  $\sum_k \eta(k) \rightarrow +\infty$  and (4)  $(\sum_k \eta(k)^2) / (\sum_k \eta(k))^2 \rightarrow 0$ . However, there exists multiple solutions if the data is linearly separable.

### Fisher's LDA

Idea: project the two distribution into one dimension and maximize the ratio of the variance between the classes and the variance within the classes, i.e.,  $\max(w^T u_1 - w^T u_0)^2 / (w^T S w)$ , where  $S = \Sigma_0 + \Sigma_1$ . Let gradient be zero and solve for  $w^*$ , we get  $w^* \propto S^{-1}(u_1 - u_0)$ .

We first compute  $w^*$  and fit distributions of the two-class projection. Then apply Bayesian decision theory to make classification.

## 5 Convex Optimization

Definition: the objective is convex and the feasible set is convex. The standard form is to minimize a convex function ( $f(w)$  for convex  $f$ ) under affine equality constraint ( $g_i(w) = 0$  for affine  $g_i$ ) and convex non-positive constraint ( $h_i(w) \leq 0$  for convex  $h_i$ ).

### How to Solve

- (1) Define Lagrangian:  $L(w, \lambda, \alpha) = f(w) + \sum_i \lambda_i g_i(w) + \sum_j \alpha_j h_j(w)$  and  $\alpha_j \geq 0$ .
- (2) Check whether strong duality holds using Slater's condition (sufficient but not necessary): there is a strictly feasible point, i.e.,  $\exists w_0$ , s.t.  $g_i(w_0) = 0$  and  $h_i(w_0) < 0$ .
- (3) Solve for  $dL(w) = 0$ ,  $g_i(w) = 0$ ,  $\alpha \geq 0$ ,  $h_j(w) \leq 0$  and  $\alpha_j h_j(w) = 0$  (complementary slackness).
- (4) Weak duality always holds and when Slater's condition holds it equals  $\min_w f(w)$ : solve  $\max_{\lambda, \alpha} \Theta(\lambda, \alpha)$  s.t.  $\alpha \geq 0$ , where  $\Theta(\lambda, \alpha) = \min_w L(w, \lambda, \alpha)$ . The optimal in the weak duality is a lower bound for  $\min_w f(w)$  since  $\Theta(\lambda, \alpha) \leq \min_w f(w)$  for any  $\lambda$  and  $\alpha$ .

## 6 Support Vector Machine

### Hard-Margin SVM

The optimization form is to minimize  $\frac{1}{2} \|w\|^2$ , s.t.  $y_i(w^T x_i + w_0) \geq 1$  ( $y_i \in \{-1, 1\}$ ). It is convex and the Slater's condition holds under the

assumption of linear separability.

By solving  $\min_{w, w_0} L(w, w_0, \alpha)$ , the duality form is  $\max_{\alpha} -\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j + \sum_i \alpha_i$ , s.t.  $\alpha_i \geq 0$ ,  $\sum_i \alpha_i y_i = 0$ .  $w^* = \sum_{i \in \text{support}} \alpha_i^* y_i x_i$ .

### Soft-Margin SVM & Kernel SVM

When the data is not linearly separable, one solution is to add slack variables:  $\min_{\xi, w, w_0} \frac{1}{2} \|w\|^2 + C \sum_i \xi_i$ , s.t.  $y_i(w^T x_i + w_0) \geq 1 - \xi_i$  and  $\xi_i \geq 0$ . Note: when the data is linearly separable, soft-margin does not necessarily produce the same cut-plane as the hard-margin SVM, especially when the margin is small due to few support vectors. When  $C \rightarrow \infty$ , it becomes hard-margin SVM.

The only difference in the dual form is that we need an extra condition  $0 \leq \alpha \leq C$ .  $\xi_i^* = \max(0, 1 - y_i(w^{*T} x_i + w_0))$ .

Another solution, Kernel SVM, is to use  $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$  to replace  $x_i^T x_j$ .

### Multi-class SVM

Idea: use  $M := \# \text{class}$  hyperplanes and maximize the generalized margin:  $\min \sum_{i=1}^M w_i^T w_i$  s.t.  $w_{y_i}^T x_i + w_{y_i,0} - \max_{y \neq y_i} w_y^T x_i + w_{y,0} \geq 1$ .

### Structural SVM

Goal: predict a structured output label, e.g., parsing trees.

Difficulty of predicting structures:

1. Compact representation of output space (cannot use one model for each class). Sol: use a score function and feature map  $\hat{y} = \text{argmax}_y w^T \Psi(x, y)$ .
2. Efficient prediction (cannot use brute-force search to find argmax). Sol: assume structures such as decomposable output spaces.
3. Define prediction error (cannot use 0/1 errors). Sol: use a loss function  $\Delta(y, \hat{y})$ .
4. Efficient training (cannot evaluate all constraints). Sol: iteratively add new constraints to the training.

## 7 Ensemble: Adaboost

Adaboost has following properties:

1. It minimizes exponential loss forwardly.
2. It trains max-margin classifiers.
3. It, as well as Random Forest, is spiky self-averaging interpolators, which localize the effect of noise.
4. It falls into the double descent regime: over-parameterized models can have better generalization.

## 8 Generative Models

Variational Autoencoder: use approximated probability to model  $p(z|x)$  and the  $p(x|z)$ .

- Pro: can be used as feature representation.
- Con: generated images are blurrier and relatively low compared to GAN.

Generative Adversarial Network: use 2-player game to sample from  $p(x)$ .

- Pro: SOTA.
- Con: (1) unstable, (2) potential mode collapse and (3) cannot solve  $p(x)$ .

## 9 Non-parametric Bayesian Inference (BI)

### Exact Conjugate Prior of Multivariate Gaussian

Data:  $x_i \sim \mathcal{N}(\mu, \Sigma)$  i.i.d.. Inverse Wishart:  $\Sigma \sim \mathcal{W}^{-1}(S, \nu) \propto |\Sigma|^{(\nu+p+1)/2} \exp(-\text{Tr}(\Sigma^{-1}S)/2)$ .

Normal Inverse Wishart as conjugate prior:

$p(\mu, \Sigma | m_0, k_0, \nu_0, S_0) = \mathcal{N}(\mu | m, \Sigma / k_0) \mathcal{W}^{-1}(\Sigma | S_0, \nu_0)$ . Update rule:  $m_p = (k_0 m_0 + N \bar{x}) / (k_0 + N)$ ,  $k_p = k_0 + N$ ,  $\nu_p = \nu_0 + N$ ,  $S_p = S_0 + k_0 m_0 m_0^T - k_p m_p m_p^T + \sum (x_i - \bar{x})(x_i - \bar{x})^T$ .

### BI with Semi-Conjugate Prior

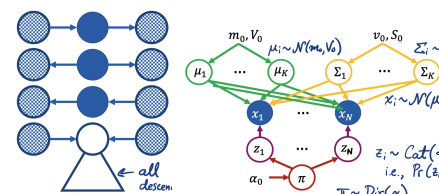
New prior:  $\mu \sim \mathcal{N}(m_0, V_0)$ ,  $\Sigma \sim \mathcal{W}^{-1}(S_0, \nu_0)$ , then posterior  $p(\mu, \Sigma | X)$  is intractable, but condition posterior is exact,  $p(\mu | \Sigma, X) = \mathcal{N}(m_p, V_p)$ ,  $V_p^{-1} = V_0^{-1} + N \Sigma^{-1}$ ,  $V_p^{-1} m_p = V_0^{-1} m_0 + N \Sigma^{-1} \bar{x}$ ;  $p(\Sigma | \mu, X) = \mathcal{W}^{-1}(S_p, \nu_p)$ ,  $\nu_p = \nu_0 + N$ ,  $S_p = S_0 + \sum x_i x_i^T + N \mu \mu^T - 2N \bar{x} \mu^T$ .

**Gibbs sampling:** random variable  $p(z_1, \dots, z_p)$  intractable, cyclically resample  $z_i$  according to tractable conditional distribution  $p(z_i | z_{-i})$   $n$  times, when  $n \rightarrow \infty$ ,  $(z_1, \dots, z_p) \sim p(z_1, \dots, z_p)$ . Finally, replace posterior with MC sampling:  $\mathbb{E}_{\theta | X} f(x | \theta) \approx \sum f(x | \theta_i) / N$ .

### BI for Gaussian Mixture Model

Data model: latent  $K$  class variable  $z_i \sim \text{Cat}(\pi)$ , observed  $x_i \sim \mathcal{N}(\mu_{z_i}, \Sigma_{z_i})$ . Prior:  $\mu_k \sim \mathcal{N}(m_0, V_0)$ ,  $\Sigma_k \sim \mathcal{W}^{-1}(S_0, \nu_0)$ ,  $\pi \sim \text{Dir}(\alpha) \propto \prod_k p_k^{\alpha_k - 1}$ . Prior also intractable.

**Goal** Gibbs sampling for BI, but to simplify conditional distribution.



**d-seperation:** for verifying conditional independence. Given with observed variable set

$C$ , if any path from variable  $A$  to  $B$  is blocked on probability graph, then  $A$  and  $B$  are independent condition on  $C$ . By this thm: (1)  $z_i, z_j$  (2)  $\mu, \pi$  (3)  $\Sigma, \pi$  all independent condition on other parameter. Sampling procedure: (1)  $z^{(t)} \leftarrow p(\cdot | x, \mu^{(t-1)}, \Sigma^{(t-1)})$ , (2)  $\mu^{(t)} \leftarrow p(\cdot | x, \Sigma^{(t-1)}, z^{(t)})$ , (3)  $\Sigma^{(t)} \leftarrow p(\cdot | x, \mu^{(t)}, z^{(t)})$ , (4)  $\pi^{(t)} \leftarrow p(\cdot | x, z^{(t)})$ .

### BI for Non-Parametric GMM

Goal: sample from infinite categorical distri.

**Dirichlet Process (DP):**  $\Theta$  parameter space,  $H$  prior distri on  $\Theta$ ,  $A_1, \dots, A_r$  arbitrary partition of  $\Theta$ .  $G$  a categorical distribution over  $\{A_i\}$  is  $G \sim \text{DP}(\alpha, H)$  if  $(G(A_1), \dots, G(A_r)) \sim \text{Dir}(\alpha H(A_1), \dots, \alpha H(A_r))$ .

**Posterior:**  $G | \{\theta_i\}_{i=1}^n \sim \text{DP}\left(\alpha + n, \frac{\alpha H + \sum_{i=1}^n \delta_{\theta_i}}{\alpha + n}\right)$

**Condition on  $\theta$ , Margin over  $G$ :**  $\theta_{n+1} | \theta_1, \dots, \theta_n \sim \frac{1}{\alpha + n} (\alpha H + \sum_{i=1}^n \delta_{\theta_i})$ . Leads to CRP

### Three Methods of Sampling from DP

In  $K \rightarrow \infty$  GMM,  $\theta$  in DP is  $z$ ,  $G$  is  $\pi$ .

**(1) Chinese Restaurant Process (CRP)**, sample  $z$ , marginalize over  $\pi$ :

$$p(z_n = k | \theta_{i < n}) = \begin{cases} n_k / (\alpha + n - 1), & \text{existing } k \\ \alpha / (\alpha + n - 1), & \text{new } k \end{cases}$$

**Expect # of Class**  $\sum_{i=1}^n \frac{\alpha}{\alpha + i - 1} \simeq \alpha \log\left(1 + \frac{n}{\alpha}\right)$

**(2) Stick-breaking Construction** samples  $\pi$ :

$$\beta_k \sim \text{Beta}(1, \alpha), \theta_k^* \sim H, \pi_k = \beta_k \prod_{l=1}^{k-1} (1 - \beta_l)$$

**(3) Marginalize over  $\mu, \Sigma$  when sampling  $z$  (if intractable), less variance (Rao-Blackwell).**

**Exchangeability:**  $p(\{\theta_i\}) = \prod_{n=1}^N p(\theta_n | \{\theta_{i < n}\})$  unchanged after permuting sampling order.

**DeFinetti's Thm** any exchangeable distri is a mixture model  $P(\{\theta_i\}) = \int \prod_{i=1}^n G(\theta_i) dP(G)$

## 10 PAC Learning

Definitions:

- A learning algorithm  $\mathcal{A}$  can learn  $c \in \mathcal{C}$  if there is a  $\text{poly}(\dots)$ , s.t. for (1) any distribution  $\mathcal{D}$  on  $\mathcal{X}$  and (2)  $\forall 0 < \epsilon < \frac{1}{2}, 0 < \delta < \frac{1}{2}$ ,  $\mathcal{A}$  outputs  $\hat{c} \in \mathcal{H}$  given a sample of size at least  $\text{poly}(\frac{1}{\epsilon}, \frac{1}{\delta}, \text{size}(c))$  such that  $P(\mathcal{R}(\hat{c}) - \inf_{c \in \mathcal{C}} \mathcal{R}(c) \leq \epsilon) \geq 1 - \delta$ .
- $\mathcal{A}$  is called an efficient PAC algorithm if it runs in polynomial of  $\frac{1}{\epsilon}$  and  $\frac{1}{\delta}$ .
- $\mathcal{C}$  is (efficiently) PAC-learnable from  $\mathcal{H}$  if there is an algorithm  $\mathcal{A}$  that (efficiently) learns  $\mathcal{C}$  from  $\mathcal{H}$ .

VC inequality:

- For an ERM  $\hat{c}_n^*$ ,  $\mathbf{P}(\mathcal{R}(\hat{c}_n^*) - \inf_{c \in \mathcal{C}} \mathcal{R}(c) > \epsilon) \leq 2|\mathcal{C}| \exp\left(-\frac{n\epsilon^2}{2}\right)$ .
- Finite VC-dimension means PAC-learnable.

**11    Appendix**

$$\frac{\partial}{\partial \Sigma} \log |\Sigma| = \Sigma^{-T}.$$