Probabilistic Machine Learning Notes

Tony Ruikang OuYang

Department of Engineering, University of Cambridge

ro352@cam.ac.uk

This note contains part of classical models in probabilistic machine learning, where most of the contents are from Pattern Recognition and Machine Learning and lectures in *Cambridge MLMI 23-24*.

It might not be that friendly for ML beginners. A preliminary of basic machine learning knowledge, *e.g.* linear regression, logistic regression, Bayesian inference, MLE and MAP, etc., is recommended. And I apologize that possible typos might occur in this note, and I will fix them smoothly as long as I have time to do so.

Contents

Chapter 1

Preliminary

1.1 Beta Distribution

Let's begin with the simple binary distribution,

$$P(x=1)=\mu, P(x=0)=1-\mu \tag{1.1.1}$$

then,

$$P(x)=\mu^{1-x}(1-\mu)^{1-x} \tag{1.1.2}$$

In Bayesian settings, given a binary likelihood, we're going to compute the posterior as follows:

$$P(\mu|x)=\frac{P(x|m)P(\mu)}{p(x)} \propto P(x|m)P(\mu) \tag{1.1.3}$$

For convenience, we construct a conjugate prior $P(\mu)$ with form:

$$P(\mu) \propto \mu^a(1-\mu)^b \tag{1.1.4}$$

Then, we construct the following proir, which is a conjugacy of binary dist., called Beta Dist.:

$$Beta(\mu|a,b)=\frac{\tau(a+b)}{\tau(a)\tau(b)}\mu^{a-1}(1-\mu)^{b-1} \tag{1.1.5}$$

$$with \quad \gamma(x)=\int_0^\infty u^{x-1}e^{-u}du \tag{1.1.6}$$

It's easy to check that:

$$E[Beta(\mu|a,b)]=\frac{a}{a+b} \tag{1.1.7}$$

$$Var(Beta(\mu|a,b))=\frac{ab}{(a+b)^2(a+b+1)} \tag{1.1.8}$$

1.2 Dirichlet Distribution

It's natural to extend the binary dist. to the multinomial case, e.g. there are K clusters in a dataset ($K \geq 2$). Let's say $x \in 0, 1^K$ and $\sum_k x_k = 1$, then,

$$P(x|\mu) = \Pi_{k=1}^K \mu_k^{x_k} \tag{1.2.1}$$

In N observations, we observe M_k datapoints lie in the k^{th} cluster, then a multinomial distribution is described as follow:

$$Mult(m_1, ..., m_k|N, \mu) = \frac{N!}{m_1!...m_k!\Pi_{k=1}^K \mu_k^{x_k}} \tag{1.2.2}$$

The same as before, to construct a conjugate prior for multinomial dist., we assume the prior has the following form:

$$P(\mu) \propto \Pi_{k=1}^K \mu_k^{\alpha_k - 1} \tag{1.2.3}$$

$$s.t. 0 \leq \mu_k \leq 1, \sum_k \mu_k = 1 \tag{1.2.4}$$

Then we obtain the prior by normalization, which is called Dirichelet distribution:

$$Dir(\mu|\alpha_1, ..., \alpha_K) = \frac{\tau(\alpha_0)}{\tau(\alpha_1)...\tau(\alpha_K)} \Pi_{k=1}^K \mu_k^{\alpha_k - 1} \tag{1.2.5}$$

$$with \quad \alpha_0 = \sum_k \alpha_k \tag{1.2.6}$$

Thus, posterior:

$$P(\mu|x, \alpha) \propto P(x|\mu)P(\mu|\alpha) \tag{1.2.7}$$

$$\propto \Pi_{k=1}^K \mu_k^{\alpha_k + m_k - 1} \tag{1.2.8}$$

$$\longrightarrow P(\mu|x, \alpha) = Dir(\mu|\alpha + m) \tag{1.2.9}$$

1.3 Gaussian Distribution

Definition 1.3.1 (Gaussian)**.**

$$N(x|\mu, \sigma) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|} exp(-\frac{1}{2})(x - \mu)^T \Sigma^{-1}(x - \mu)$$

Theorem 1.3.1. *Given p(Y—X) and p(X), then*

$$\mathbf{E}[Y] = \mathbf{E}_X \mathbf{E}_Y[Y|X] \tag{1.3.1}$$

$$Var(Y) = \mathbf{E}_X[Var(Y|X)] + Var_X(\mathbf{E}[Y|X]) \tag{1.3.2}$$

Proof.

1. $\mathbf{E}[Y]$ is easily derived by towel theorom.

2.

$$\begin{aligned}
Var(Y) &= \mathbf{E}[(Y - \mathbf{E}[\mathbf{Y}])(Y - \mathbf{E}[\mathbf{Y}])^T] \\
&= \mathbf{E}[(Y - \mathbf{E}[\mathbf{Y}] + \mathbf{E}_X[Y|X] - \mathbf{E}_X E[Y|X]) \\
&\quad (Y - \mathbf{E}[\mathbf{Y}] + \mathbf{E}_X[Y|X] - \mathbf{E}_X E[Y|X])^T] \\
&= \mathbf{E}_X[\mathbf{E}_{Y|X}[(Y - \mathbf{E}[Y|X])(Y - \mathbf{E}[Y|X])^T \\
&\quad + (\mathbf{E}[Y|X] - \mathbf{E}[Y])(\mathbf{E}[Y|X] - \mathbf{E}[Y])^T \\
&\quad + 2(Y - \mathbf{E}[Y|X])(\mathbf{E}[Y|X] - \mathbf{E}[Y])^T]] \\
&= \mathbf{E}_X[Var(Y|X)] + \\
&\quad \mathbf{E}_X[(\mathbf{E}[Y|X] - \mathbf{E}[\mathbf{E}[Y|X]])(\mathbf{E}[Y|X] - \mathbf{E}[\mathbf{E}[Y|X]])^T] \\
&= \mathbf{E}_X[Var(Y|X)] + Var_X(\mathbf{E}_\mathbf{Y}[\mathbf{Y}|\mathbf{X}])
\end{aligned}$$

□

Theorem 1.3.2 (Gaussian marginal and conditions)**.** *Given,* $p(x) = N(x|\mu, \Lambda^{-1})$ *and* $p(y|x) = N(y|Ax + b, L^{-1})$*, then,*

$$\begin{aligned}
p(y) &= N(x|A\mu + b, L^{-1} + A\Lambda^{-1}A^T) \\
p(x|y) &= N(y|\Sigma(A^T L(y - b) + \Lambda\mu), \Sigma)
\end{aligned}$$

where $\Sigma = (\Lambda + A^T LA)^{-1}$

Proof.

1. It is clear that y is Gaussian, then by theorem1.3.1:

$$\begin{aligned}
\mathbf{E}[y] &= \mathbf{E}_Y \mathbf{E}_X[y|x] = A\mu + b \\
Var[y] &= \mathbf{E}_X[L^{-1}] + Var_X(Ax + b) \\
&= L^{-1} + \mathbf{E}[A(x - \mu)(x - \mu)^T A^T] \\
&= L^{-1} + A\Lambda^{-1}A^T
\end{aligned}$$

2.

$$\begin{aligned}
p(x|y) &\propto N(y|Ax + b, L^{-1})N(x|\mu, \Lambda^{-1}) \\
&\propto exp((y - Ax - b)^T L(y - Ax - b) + (x - \mu)^T \Lambda(x - \mu)) \\
&\propto exp(x^T(ALA + \Lambda)x + 2(b^T LA - y^T LA - \mu^T \Lambda)x) \\
&\propto exp((x - m)^T \Sigma^{-1}(x - m))
\end{aligned}$$

with m $= \Sigma(A^T L(y - b) + \Lambda\mu)$ and $\Sigma = (\Lambda + A^T LA)^{-1}$

□

1.4 Periodic Variables

Consider an univariate θ, s.t.

$$P(\theta) \geq 0 \tag{1.4.1}$$

$$\int_0^{2\pi} P(\theta)d\theta = 1 \tag{1.4.2}$$

$$P(\theta + 2\pi) = P(\theta) \tag{1.4.3}$$

Suppose θ_0 is an origin, with radius r_0. We're going to construct the prob. P by transforming the probability distribution from Cartesian coordinates to Polar coordinates.

Consider a 2-D gaussian with mean (μ_1, μ_2), 0 covariance and same variance σ^2, i.e.

$$P(x_1, x_2) = \frac{1}{2\pi\sigma} exp(-\frac{(x_1-\mu_1)^2 - (x_2-\mu_2)^2}{2\sigma^2}) \tag{1.4.4}$$

Then, with $(x_1, x_2) = (r\cos\theta, r\sin\theta)$ and $(\mu_1, \mu_2) = (r_0\cos\theta_0, r_0\sin\theta_0)$, it's easy to compute that,

$$P(\theta) \propto P(x_1, x_2) = \frac{1}{2\pi\sigma} exp(\frac{r_0}{r}\cos(\theta-\theta_0) + const) \tag{1.4.5}$$

Then, we obtain the von-Mises distribution as follow,

$$P(\theta|\theta_0, m) = \frac{1}{2\pi I_0(m)} exp\{mcos(\theta-\theta_0)\} \tag{1.4.6}$$

$$where \quad I_0(m) = \frac{1}{2m}\int_0^{2\pi} exp(m\cos\theta)d\theta, \text{is the Bessel Function} \tag{1.4.7}$$

1.5 Laplacian Approximation

Given a density $p(z)$, our goal is to find a gaussian $q(z)$, such that $q(z)$ is an approximation of $p(z)$. In other words, we want to find:

$$p(z) \simeq q(z) = c\exp(-\frac{A(z-z_0)}{2}) \tag{1.5.1}$$

$$\iff lnp(z) \simeq const - \frac{A(z-z_0)}{2} \tag{1.5.2}$$

It's easy to notice that, finding the const and A is equivalent to figure out the 2nd order Taylor expansion of $p(z)$ at z_0, where $p'(z_0) = 0$. And thus,

$$lnp(z) \simeq lnp(z_0) + \frac{lnp''(z_0)}{2}(z-z_0)^2 \tag{1.5.3}$$

$$\implies A = -\frac{d^2}{dz^2}p(z)|_{z=z_0} \tag{1.5.4}$$

Thus,

$$q(z) = N(z; z_0, A^{-1}) \simeq p(z) \tag{1.5.5}$$

$$\text{with } \frac{d}{dz}lnp(z)|_{z=z_0} = 0 \text{ and } A = -\frac{d^2}{dz^2}lnp(z)|_{z=z_0}$$

Chapter 2

Linear models for Regression

2.1 Intro

Given $D = (x_n, y_n)_{n=1}^N$, we expect$y = \omega^T \phi(x)$, where ω is learnable param and ϕ is called basis function.

$$\begin{cases} \textbf{1. Decision-making perspective - Least Square:} \\ \quad \min_\omega \frac{1}{N} \sum_{n=1}^{N} (\omega^T \phi(x_n) - y_n)^2 \\ \textbf{2. Statistical perspective - MLE:} \\ \quad \text{suppose } y = \omega^T \phi(x) + \epsilon, \epsilon \sim N(0, \sigma^2)] \\ \quad \longrightarrow \max_\omega P(D|\omega) = N(\omega^T \phi(x), \sigma^2) \end{cases}$$

Then,

$$\omega = (\Phi^T \Phi)^{-1} \Phi^T y$$
$$\text{with } \Phi = [\phi(x_1), ..., \phi(x_N)]^T$$

2.2 regularization

$$\begin{cases} \textbf{1. Optim - lagrange: } \min_\omega L(\omega) + \lambda \|\omega\| \\ \textbf{2. Stats - bayesian reg: with a prior, maximize a posterior} \end{cases}$$

$$\longrightarrow \begin{cases} \textbf{1. L1: sparse solution, laplacian prior} \\ \textbf{2. L2: gaussian proir} \end{cases}$$

$$\longrightarrow \omega = (\lambda I + \Phi^T \Phi)^{-1} \Phi^T y \tag{2.2.1}$$

2.3 Bayesian view for Ridge Regression

In Bayesian view, we suppose the responses are given by $f(x) = \omega^T\phi(x) + \epsilon$, where $\epsilon \sim N(0, \sigma^2)$ and prior $\omega \sim N(0, \alpha^{-1}I)$. Then, we have $p(y|x, \omega, \beta) = N(y|\omega^T\phi(x), \beta^{-1}$. Thus, the posterior is given by:

$$\begin{aligned} p(\omega|D, \alpha, \beta) &= \Pi_{n=1}^{N} p(y_n|x_n, \omega, \beta)p(\omega|\alpha) \\ &= N(y|\Phi\omega, \beta^{-1}I)N(\omega|, 0, \alpha^{-1}I)) \\ &\propto exp\{\frac{1}{2}(\beta(y - \Phi\omega)^T(y - \Phi\omega) - 2\beta\omega^T\Phi y + \alpha\omega^T\omega)\} \\ &= exp\{\frac{1}{2}(\omega^T(\beta\Phi^T\Phi + \alpha I) - 2\beta\omega^T\Phi y)\} \\ &\propto exp\{\frac{1}{2}(\omega - m)^T\Sigma^{-1}(\omega - m)\} \end{aligned} \tag{2.3.1}$$

Thus,

$$\begin{aligned} p(\omega|D, \alpha, \beta) &= N(\omega|m, \Sigma) \\ where\ \Sigma &= (\beta\Phi^T\Phi + \alpha I)^{-1}\ and\ m = \Sigma\Phi y \end{aligned} \tag{2.3.2}$$

.

Therefore, ω is optimized by maximizing a posterior(MAP), which is equivalent to ridge regression.

Besides, the hyperparams α and β could be optimized by maximizing the evidence $p(D|\alpha, \beta) = \int p(D|\omega, \beta)p(\omega, \alpha)d\omega$, which is easy as $p(D|\alpha, \beta) = N(y|0, \beta^{-1}I + \alpha^{-1}\Phi\Phi^T)$(by Gaussian margins).

Choosing the optimal hyperparams α^* and β^*, then the prediction of a new input is:

$$\begin{aligned} p(y'|x', D,^*, \beta^*) &= \int p(y'|x', \omega, \beta^*)p(\omega|D, \alpha^*, \beta^*)d\omega \\ &= \int N(y'|\omega^T\phi(x'), \beta^{*-1}I)N(\omega|m, \Sigma)d\omega \\ &= N(y'|m^T\phi(x'), \beta^{*-1} + \phi(x')^T\Sigma\phi(x')) \end{aligned} \tag{2.3.3}$$

Chapter 3

Linear models for Classification

3.1 Intro

In classification settings, generally, there are two types of models: discriminant model and generative model.

3.1.1 Discriminant models

Given a dataset, we're going to find a hyperplane to classify the data. In statistics view, we're finding distributions $P(x \in C_k|x, \omega)$. Generally, we would like to use the following form to model the probs:

$$P(x \in C_k|x, \omega) = f(\omega_k^T x) \tag{3.1.1}$$

where f is called link function or activation function.

Then, the likelihood is

$$P(D|\omega) = \Pi_{n=1}^{N} \Pi_{k=1}^{K} P_{nk}^{I_{nk}} \tag{3.1.2}$$

$$\text{where, } P_{nk} = f(\omega_k^T x_n) \text{ and } I_{nk} = 1 \text{ if } x_n \in C_k \text{ ; else } 0 \tag{3.1.3}$$

We fit ω by MLE, which is equivalent to

$$\min_{\omega_1,\ldots,\omega_K} E(\omega) := -\sum_n \sum_k I_{nk} log P_{nk} \tag{3.1.4}$$

$$\nabla_{\omega_j} E(\omega) = -\sum_n \frac{I_{nj}}{P_{nj}} f'(\omega_j^T x_n) x_n \tag{3.1.5}$$

And we update the params by:

$$\omega_j^{t+1} = \omega_j^t - \alpha \nabla_{\omega_j} E(\omega) \tag{3.1.6}$$

The former objective function is also called Cross Entropy, with a more popular 2-D case (binary classification) as following:

$$E(\omega) = -\sum_n \{(1 - y_n) log p_n + y_n log(1 - p_n)\} \tag{3.1.7}$$

3.1.2 Generative models

Different from discriminant ones, generative models are modeling the data by fitting $P(x|C_k, \omega)$.

As,

$$P(x \in C_k|x, \omega) = \frac{P(x|C_k, \omega)P(C_k)}{P(x|\omega)} \tag{3.1.8}$$

we're going to model $P(x|C_k, \omega)$ and assign a prior for C_k, instead of modeling $P(x \in C_k|x, \omega)$ itself. For example, we could use Gaussians to model the generative process, i.e. $P(x|C_k, \omega) = N(x; \mu_k, \sum_k)$, and params are fitted by maximizing the likelihood as well.

3.2 Fisher's Discriminant model

Main idea: min within-class deviations and max between-class deviations.

For 2 classes case(C_1, C_2), given $D = (x_n, y_n)$, define,

$$\tilde{m_k} := \frac{1}{N_k} \sum_{n \in C_k} x_n,\ k = 1, 2 \tag{3.2.1}$$

$$m_k := \omega^T \tilde{m_k} \in \mathcal{R} \tag{3.2.2}$$

$$S_k := \frac{1}{N_k} \sum_{n \in C_k} (y_n - m_n)^2 \tag{3.2.3}$$

Then,

$$\max_{\omega} J(\omega) : = \frac{(m_1 - m_2)^2}{S_1 + S_2} \tag{3.2.4}$$

$$= \frac{\omega^T(\tilde{m_1} - \tilde{m_2})(\tilde{m_1} - \tilde{m_2})^T\omega}{\omega^T[\sum_k \frac{1}{N_k} \sum_{n \in C_k}(x_n - \tilde{m_k})(x_n - \tilde{m_k})^T]\omega} \tag{3.2.5}$$

$$: = \frac{\omega^T S_B \omega}{\omega^T S_W \omega} \tag{3.2.6}$$

Let $J(\omega) = 0$, then:

$$\omega^T S_B \omega S_W \omega = \omega^T S_W \omega S_B \omega \tag{3.2.7}$$

and notice that $\omega^T S_B \omega$ and $\omega^T S_W \omega$ are scalars, then we have

$$\omega \propto S_W^{-1} S_B \omega \propto S_W^{-1}(\tilde{m_1} - \tilde{m_2}) \tag{3.2.8}$$

And finally, we could assign $x^{new} \in C_1$ if $\omega^T x^{new} \geq y_0$, where y_0 could be modeled by a gaussian.

3.3 Logistic and it's variants

3.3.1 logistic

For 2-classes case, following the settings in 3.1.1, we select the sigmoid function as activation. By 3.1.5 and $\sigma'(x) = \sigma(x)(1 - \sigma(x))$,

$$\nabla E(\omega) = -\sum_n (y_n - \sigma(\omega^T x_n))x_n \tag{3.3.1}$$

Then, ω is estimated by gradient descent.

3.3.2 IRLS

To accelerate the training process for any 3.1.1 like model, we would like to use a 2nd order optimization algorithm, i.e. Newton-Raphson, which solves the fixed point problem $f(x) = x$ by an iterative algorithm $x^{new} = x^{old} - \frac{f'}{f''}|_{x=x^{old}}$.

Thus, we first compute the Hessian matrix for each param ω_j:

$$H(\omega_j) = \nabla\nabla_{\omega_j} E(\omega) \tag{3.3.2}$$

$$= -\nabla \sum_n \frac{I_{nj}}{P_{nj}} f'(\omega_j^T x_n)x_n \tag{3.3.3}$$

$$= -\sum_n \frac{I_{nj}}{P_{nj}^2}\{f''(\omega_j^T x_n)P_{nj} - (f'(\omega_j^T x_n))^2\}x_n x_n^T \tag{3.3.4}$$

Then, each param is updated by:

$$\omega_j^{t+1} = \omega_j^t - H(\omega_j)^{-1}\nabla_{\omega_j} E(\omega)|_{\omega=\omega^t} \tag{3.3.5}$$

3.3.3 Multiclass logistic

Instead of simple sigmoid activation, we use softmax as the activation function when dealing with multiclasses, which is:

$$P(x \in C_k|x, \omega) = \frac{exp(a_k)}{\sum_j exp(a_j)} \tag{3.3.6}$$

$$\text{where, } a_k = \sigma(\omega_k^T x) \tag{3.3.7}$$

According to 3.1.1, it's easy to derive that:

$$\nabla_{\omega_j} E(\omega) = -\sum_n (t_{nj} - P_{nj})x_n \tag{3.3.8}$$

$$\nabla_{\omega_k}\nabla_{\omega_j} E(\omega) = -\sum_n P_{nk}(I_{kj} - P_{nj})x_n x_n^T \tag{3.3.9}$$

3.3.4 Probit Reg

In logistic settings, we use sigmoid or softmax as activations, which might encounter some issues when outliers arising. In details, when the input x lies on the tails, i.e. $x \lim +\infty$, the prob would be close to 1 with exponential speed,

which might effect the training process a lot if the label is incorrect. Thus, we use an alternative activation as follows.

In a 2-classes case, We would like to find a threshold, such that $y = 0$ if $a \geq \theta$. Thus, given the density of θ, say, $p(\theta)$, $P(y = 1) = \int_{-\infty}^{a} p(\theta)d\theta$.

Indeed, let $p(\theta)$ be a standard gaussian, then

$$P(y = 1|x, \omega) = \int_{-\infty}^{\sigma(\omega^T x)} N(\theta; 0, 1)d\theta \tag{3.3.10}$$

$$= \int_{-\infty}^{\sigma(\omega^T x)} \frac{1}{\sqrt{2\pi}} exp(-\frac{\theta^2}{2})d\theta \tag{3.3.11}$$

$$= \Phi(\sigma(\omega^T x)) \tag{3.3.12}$$

Solving this by GD...

3.3.5 Bayesian logistic Reg

In addition to simple logistic, we adopt the Bayesian framework. By 3.1.2 and given prior $P(\omega)$, we compute the posterior as follow:

$$P(\omega|D) = \frac{1}{Z} P(D|\omega)P(\omega) \tag{3.3.13}$$

Then, we estimate ω by MAP:

$$\omega^{MAP} = \arg\min_{\omega} P(\omega|D) \tag{3.3.14}$$

$$= \arg\min_{\omega} logP(D|\omega) + logP(\omega) \tag{3.3.15}$$

Then, ω^{MAP} could be easily computed by any optimization algorithm, e.g. GD. However, the posterior is intractable, as computing the factor $Z = \int_{\omega} P(D|\omega)P(\omega)d\omega$ is hard, or even impossible.

To make the posterior tractable, we approximate it using Gaussian, i.e. Laplacian Approximation. We first choice the prior as $P(\omega) = N(\omega; m_0, S_0)$. Then, apply Laplacian Approximation using 1.5.5. Since ω^{MAP} is a maximum of posterior, then $\nabla P(\omega|D)|_{\omega=\omega^{MAP}} = 0$. Then,

$$A = -\nabla\nabla logP(\omega|D) = S_0^{-1} - \nabla\nabla P(D|\omega) \tag{3.3.16}$$

And we approximate the posterior as $P(\omega|D) \simeq N(\omega; \omega^{MAP}, A^{-1})$
e.g. for 2-classes case, $A = S_0^{-1} + \sum_n P_n(1 - P_n)x_n x_n^T$

Chapter 4

Kernel methods

4.1 Intro

To include more nonlinearity into our models, either regression or classification, we're going to introduce the concept of "kernel" in this section.

Beforing giving the definition of kernel, we first review the following concepts:

Definition 4.1.1 (Metric)**.** *a function D is called a metric on X, if $\forall x, y, z \in X$, it satisfies that:*

- $D(x, x) = 0$
- *positivity:* $D(x, y) \geq 0$
- *symmetry:* $D(x, y) = D = (y, x)$
- *triangle inequality:* $D(x, z) < D(x, y) + D(y, z)$

Definition 4.1.2 (Metric Space)**.** *A space, e.g. of X, equipped with a metric/norm, e.g. a distance function d, i.e.* (X, d)

Definition 4.1.3 (Inner Product)**.** $< x, y >= \sqrt{D(x, y)}, \forall x, y \in X$

Definition 4.1.4 (Inner Product Space)**.** *A metric space, whose metric/norm is defined by an inner product* $< \cdot, \cdot >$*, i.e.* $(X, \sqrt{< \cdot, \cdot >})$

Definition 4.1.5 (Complete Space)**.** *A metric space X, where for each $x \in X$, exists a sequence* $\{x_n\}_{n=1}^{\infty}$*, such that* $\lim_{n \longrightarrow \infty} x_n = x$.

Definition 4.1.6 (Hilbert Space)**.** *A complete Inner Product Space corresponding to the norm given by* $\sqrt{< \cdot, \cdot >_{\mathcal{H}}}$*, noted as* $\mathcal{H}(\mathcal{X}, \| \cdot \|)$

Follow the previous reviews, a kernel is defined as follows:

Definition 4.1.7 (Kernel)**.** *A function $k : X \times X \to \mathcal{R}$ is called a kernel, if there exits a Hilbert Space $\mathcal{H}$ and a map $\phi : X \to \mathcal{H}$ such that,*

$$\forall x, x' \in X, k(x, x') =< \phi(x), \phi(x') >_{\mathcal{H}}$$

Thus, kernel is the inner product on a Hilbert Space, which represents the inner-product of feature map ϕ.

Then, we're going to provide that ***k is a kernel if it is a symmetric and semi-definite positive function***, i.e.

$$\forall x, x' \in X, \begin{cases} k(x, x' = k(x', x)) \\ k(x, x) \geq 0 \end{cases}$$

Definition 4.1.8 (Positive semi-definite function)**.** *A function f is called positive semi-definite, if*

$$\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j f(x_i, x_j) \geq 0, \forall n \geq 1, \forall x_i, x_j \in X, \forall \alpha_i \in \mathcal{R} \tag{4.1.1}$$

Lemma 4.1.1. *All kernels are positive semidefinite functions.*

Proof.

$$\begin{aligned} \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) &= \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j < \phi(x_i), \phi(x_j) >_{\mathcal{H}_k} \\ &=< \sum_{i=1}^{n} \alpha_i \phi(x_i), \sum_{j=1}^{n} \alpha_j \phi(x_j) >_{\mathcal{H}} \\ &= \| \sum_{i=1}^{n} \alpha_i \phi(x_i) \|^2_{\mathcal{H}_k} \geq 0 \end{aligned}$$

□

Definition 4.1.9 (Reproducing Kernel and RKHS)**.** *let $\mathcal{H}$ be a Hilbert Space of functions: $f : X \to \mathcal{R}$, $k : X \times X \to \mathcal{R}$ is called a reproducing kernel if:*

- $\forall x \in X, k(\cdot, x) \in \mathcal{H}$
- $\forall x \in X, \forall f \in \mathcal{H}, < f, k(\cdot, x) >_{\mathcal{H}} = f(x)$

If $\mathcal{H}$ has a reproducing kernel, it is called a Reproducing Kernel Hiltert Space, RKHS.

Thus, any reproducing kernel is also a valid kernel with the feature map $\phi : X \to k(\cdot, x)$.

Proof.

$$k(x, x') =< k(\cdot, x'), k(\cdot, x) >_{\mathcal{H}} =< \phi(x), \phi(x') >_{\mathcal{H}}$$

□

Theorem 4.1.2 (Moore-Avonszajn)**.** *Every positive semidefinite function $k : X \times X \to \mathcal{R}$ is also a reproducing kernel with a unique corresponding RKHS.*

Theorem 4.1.3 (Representor Theorem)**.** *There is always a solution to*

$$f^* = \arg\min_{f\in\mathcal{H}_k} \hat{R}(f) + \Omega(\|f\|_{\mathcal{H}_k}) \tag{4.1.2}$$

that takes the form:

$$f^* = \sum_{n=1}^{N} a_n k(\cdot, x_n),\ a_n \in \mathcal{R} \tag{4.1.3}$$

Proof. Suppose f is one of a minimum and f_s is the projection of f onto subspace $span k(\cdot, x_n), n = 1, ..., N$, such that: $f = f_s + f_\perp$

$$\begin{aligned}
&\implies \|f\|^2_{\mathcal{H}_k} = \|f_s\|^2_{\mathcal{H}_k} + \|f_\perp\|^2_{\mathcal{H}_k} \geq \|f_s\|^2_{\mathcal{H}_k} \\
&\implies \Omega(\|f_s\|^2_{\mathcal{H}_k}) \leq \Omega(\|f\|^2_{\mathcal{H}_k})
\end{aligned}$$

Besides,

$$\begin{aligned}
f(x_n) =< f, k(\cdot, x_n) >_{\mathcal{H}_k} &=< f_s + f_\perp, k(\cdot, x_n) >_{\mathcal{H}_k} \\
&=< f_s, k(\cdot, x_n) >_{\mathcal{H}_k} = f_s(x_n)
\end{aligned}$$

$$\begin{aligned}
&\implies L(y_n, f(x_n), x_n) = L(y_n, f_s(x_n), x_n) \\
&\implies \hat{R}(f) = \hat{R}(f_s)
\end{aligned}$$

Thus, f_s is also a minimum. □

4.2 Example of kernels

1. Gaussian Kernel:

$$k(x, x') = exp\{\frac{1}{2\sigma^2}(\tilde{K}(x, x) + \tilde{K}(x', x')) - 2\tilde{K}(x, x')\}$$

   where $\tilde{K}$ is any other valid kernel and thus the Gaussian Kernel is not restricted to Euclidean Space.

2. Kernel for sets:

$$k(A_1, A_2) = 2^{|A_1 \cap A_2|}$$

3. Kernel for probs:

$$\begin{aligned}
k(x, x') &= \sum_i p(x|i)p(x'^{|i})p(i) \\
or &= \int p(x|z)p(x'^{|z})p(z)dz
\end{aligned}$$

4. Fisher Kernel:
   Given a generative model $p(x|\theta)$, the Fisher score is $g(\theta, x) := \nabla_\theta log p(x|\theta)$. Then the kernel is defined as:

$$k(x, x') = g^T(\theta, x)F^{-1}g(\theta, x)$$

   where F is the Fisher Information, $F = \mathbf{E}_X[g(\theta, x)g^T(\theta, x)]$

5. Sigmoid Kernel: $k(x, x') = tanh(a^T x^T x' + b)$
6. Matérn Kernels(also called Matérn covariance):

$$k(x, x') = \frac{1}{\Gamma(\nu)2^{\nu-1}}(\frac{\sqrt{2\nu}}{l}\|x - x'\|_2)^\nu K_\nu(\frac{\sqrt{2\nu}}{l}\|x - x'\|_2)$$

where $l > 0$, K_ν is a modified Bessel function and Γ is the gamma function. And when $\nu = s + \frac{1}{2}$, the corresponding Hilbert Space is a function space of all s-times differentiable functions. And especially,

(a) $\nu = \frac{1}{2}, k(x, x') = exp(-\frac{1}{l}\|x - x'\|_2)$

(b) $\nu = \frac{3}{2}, k(x, x') = exp(1 + \frac{\sqrt{3}}{l}\|x - x'\|_2)exp(-\frac{\sqrt{3}}{l}\|x - x'\|_2)$

4.3 Kernel constructions

1. Mapping between Space: $k(A(x), A(x'))$, with $A : X \to \tilde{X}$
2. Sums of kernels: $\sum_i k_i(x, x')$
3. Products of kernels: $k((x, y), (x'), y') = k_X(x, x'))k_Y(y, y')$ and moreover $k(x, x') = k_1(x, x')k_2(x, x')$

4.4 Kernel Regression: Nadaraya-Watson Model

Let's model the joint distribution as

$$p(x, y) = \frac{1}{N}\sum_{n=1}^{N} f_\theta(x - x_n, y - y_n) \tag{4.4.1}$$

where f is a function(some prob density) with param θ, then

$$\mathbf{E}[y|x] = \int yp(y|x)dy = \frac{\int yp(x, y)dy}{\int p(x, y)dy} \tag{4.4.2}$$

$$= \frac{\sum_{n=1}^{N}\int yf_\theta(x - x_n, y - y_n)dy}{\sum_{m=1}^{N}\int f_\theta(x - x_m, y - y_m)dy} \tag{4.4.3}$$

Suppose $\int yf(x, y)dy = 0$ and let $g(x) := \int_{-\infty}^{+\infty} f(x, y)dy$, then

$$\mathbf{E}[y|x] = \sum_n \frac{g(x - x_n)}{\sum_m g(x - x_m)}y_n \tag{4.4.4}$$

Therefore, the prediction has the form:

$$\mathbf{E}[y|x] = \sum_n k(x, x_n)y_n \tag{4.4.5}$$

where the kernel is $k(x, x_n) = \frac{g(x-x_n)}{\sum_m g(x-x_m)}$

e.g. let $f_\theta = N(o, \sigma^2 I)$

4.5 Representing Probability Distributions with Features

Chapter 5

Gaussian Process

In the perspective of Bayesian nonparametric approach, we treat the prediction function f as the random variable taking values in an infinite-dimension space of functions, i.e. given prior over functions $f \in \mathcal{F}$ and compute $P(f|D)$

By Bayesian,

$$P(f|D) \propto P(f)P(D|f) = P(f)\Pi_{i=1}^{n} P(y_i|f(x_i)) \tag{5.0.1}$$

which means that only require f to be evaluated at data points! And we could just define a stochastic process on the joint distribution of $(f(x_1), ...f(x_n))$ for all possible inputs.

5.1 GP Regression

Definition 5.1.1 (Gaussian Process)**.** *A GP is a stochastic process whose elements are jointly Gaussian:*

$$f = \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix} \sim N(\begin{bmatrix} m(x_1) \\ \vdots \\ m(x_n) \end{bmatrix}, \begin{bmatrix} k(x_1,x_1) & \dots & k(x_1,x_n) \\ \vdots & \ddots & \vdots \\ k(x_n,x_1) & \dots & k(x_n,x_n) \end{bmatrix}) \tag{5.1.1}$$

where m is the mean function, k is the covariance function, and they are all deterministic. And since k must be semidefinite positive, it's a kernel in $\mathcal{H}_k$!

To simplify, consider a zero mean function: $f \sim N(0, k)$.

- Gaussian marginals and conditions:
  Let $Z \sim N(\mu, \Sigma)$, by blocking:

$$Z = \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix}, \mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}$$

$$\begin{aligned} \Longrightarrow & p(Z_1) = N(Z_1; \mu_1, \Sigma_1) \\ & p(Z_2) = N(Z_2; \mu_2, \Sigma_2) \\ & p(Z_2|Z_1) = N(Z_2; \mu_2 + \Sigma_{21}\Sigma_{11}^{-1}(Z_1 - \mu_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}) \end{aligned}$$

Suppose y_i are independent, let $y|f \sim N(f, \sigma^2 I)$, where $y = \begin{bmatrix} y_1 \dots y_n \end{bmatrix}^T$. Let $(K_{xx})_{ij} = k(x_i, x_j)$, $(K_{x'x'})_{ij} = k(x'_i, x'_j)$ and $(K_{x'x})_{ij} = k(x'_i, x_j)$. Then

$$f = \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix} \sim N(0, K_{xx}),\ f' = \begin{bmatrix} f(x'_1) \\ \vdots \\ f(x'_n) \end{bmatrix} \sim N(0, K_{x'x'})$$

$$\implies \begin{bmatrix} f' \\ y \end{bmatrix} \sim N(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K_{x'x'} & K_{x'x} \\ K_{xx'} & K_{xx} + \sigma^2 I \end{bmatrix})$$

Proof.

$$\begin{aligned} Cov(f', y) &= Cov(f', f + \sigma\epsilon) \text{ with } \epsilon \sim N(0, I) && (5.1.2)\\ &= \mathbf{E}[f'(f + \sigma\epsilon)^T] && (5.1.3)\\ &= \mathbf{E}[f' f^T] = K_{x'x} && (5.1.4) \end{aligned}$$

□

By Gaussian conditions,

$$f'|y \sim N(K_{x'x}(K_{xx} + \sigma^2 I)^{-1} y, K_{x'x'} - K_{x'x}(K_{xx} + \sigma^2 I)^{-1} K_{xx'}) \quad (5.1.5)$$

And it is noticable that the mean term $K_{x'x}(K_{xx} + \sigma^2 I)^{-1} y$ is exactly the result of Kernel Ridge Regression with $\lambda = \sigma^2$!

In Bayesian perspective:

1. prior: $f \sim GP(0, k)$
2. likelihood: $y_i|f \sim N(f(x_i), \sigma^2)$
3. posterior: $f|y \sim GP(m_{post}, k_{post})$, where

$$\begin{aligned} m_{post}(x) &= K_x^T(x)(K_{xx} + \sigma^2 I)^{-1} y \\ k_{post}(x, x') &= k(x, x') - K_x^T(x')(K_{xx} + \sigma^2 I)^{-1} K_x(x') \\ K_x(x) &= [k(x, x_1), ..., k(x, x_n)]^T \end{aligned}$$

For hyperparameter selection, we perform MLE on evidence. e.g. given $\theta = (\nu, \sigma^2)$, where ν is a param of kernel k_ν, then

$$\begin{aligned} p(D|\theta) &= \int p(D, f|\theta) df \\ &= \int p(D|f, \theta) p(f|\theta) df \\ &= \int N(y; f(x), \sigma^2 I) N(f; 0, K_\nu) df = N(y; 0, K_\nu + \sigma^2 I) && (5.1.6) \end{aligned}$$

Thus, we could optimize the params by

$$\max_{\theta=(\nu,\sigma^2)} logP(D|\theta) = -\frac{1}{2}|K_\nu + \sigma^2 I| - \frac{1}{2} y^T (K_\nu + \sigma^2 I)^{-1} y + const \quad (5.1.7)$$

5.1.1 GP vs linear regression

To discover the relationship between Gaussian Process and linear regression, we have to notice that(from last Chapter), the optimized linear regression model $f(x) = w^T\phi(x)$ is equivalent to:

$$f(x) = \sum_{n=1}^{N} a_n \phi(x_n)^T \phi(x)$$

It is easy to extend it to a Kernel linear regression, where a kernel function is defined as an inner-product of feature map in a Hilbert Space, $k(x, x') =< \phi(x), \phi(x') >_{\mathcal{H}}$. And there is no restriction about the feature map, which means we are going to consider an infinite feature map function, and what we need to do is just defining a valid kernel.
Thus, a GP is equivalent to a linear regression model with infinite basis functions!
Moreover, GP(0, k) is equivalent to the Kernel Bayesian Regression model with 0 mean prior.
Thus, the performance of a GP(0, k) model is well guaranteed. To improve its performance, specifying the mean function would be helpful, which might reduce the variance of the predictions:

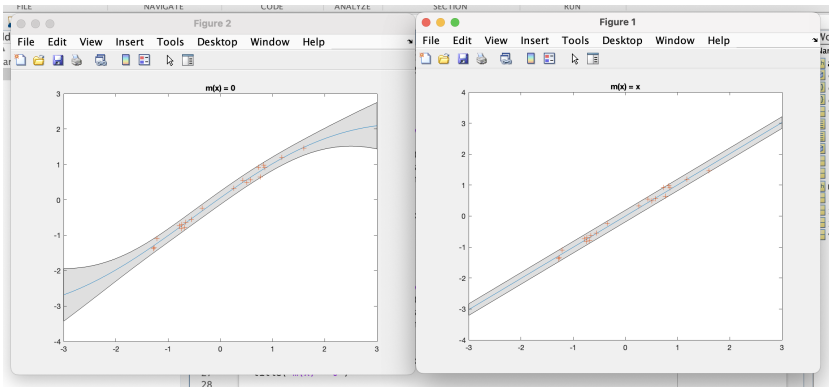


Figure 5.1: left: GP(0, k); right: GP(x, k)

Besides, we could specify a parameterised mean function, say m_γ, and we could jointly train the parameters by MLE:

$$\max_{\theta=(\gamma,\nu,\sigma^2)} logP(D|\theta) = -\frac{1}{2}|K_\nu+\sigma^2 I| - \frac{1}{2}(y-m_\gamma(x))^T(K_\nu+\sigma^2 I)^{-1}(y-m_\gamma(x)) + const \tag{5.1.8}$$

5.2 GP Classification

5.2.1 General case

To obtain categorical responses. we use link function $\psi = [\psi_1, ..., \psi_K]^T$ to transform GP outputs, i.e. $p(y = k|f(x)) = \phi_k(f(x))$. Recep that $f \sim N(f; 0, k)$

and $f'|f \sim N(f'; K_{x'x}K_{xx}^{-1}f, K_{x'x'} - K_{x'x}K_{xx}^{-1}K_{xx'})$, we aim to compute

$$\begin{aligned} p(f'|D) &= \int p(f', f|D)df \\ &= \int p(f'|f)p(f|D)df \end{aligned} \tag{5.2.1}$$

And

$$p(f|D) \propto p(D|f)p(f) = p(f)\Pi_i\Pi_k\psi_k(f(x_i))^{\mathbb{1}(y_i=k)} \tag{5.2.2}$$

To compute $p(f|D)$, we use Laplacian approximation:

$$\ln p(f|D) = \ln p(f) + \sum_i \sum_k \mathbb{1}(y_i = k) \ln psi_k(f(x_i)) + const \tag{5.2.3}$$

$$\nabla \ln p(f|D) = -K_{xx}^{-1}f + \sum_i \sum_k \mathbb{1}(y_i = k)\nabla \ln \psi_k(f(x_i)) \tag{5.2.4}$$

$$\nabla\nabla \ln p(f|D) = -K_{xx} + \sum_i \sum_k \mathbb{1}(y_i = k)\nabla\nabla \ln \psi_k(f(x_i)) \tag{5.2.5}$$

$$A = -\nabla\nabla \ln p(f|D)|_{f=f^{MAP}} \tag{5.2.6}$$

Then, we use $\nabla \ln p(f|D)$ to optimize and find f^{MAP}. And thus

$$p(f|D) \simeq N(f; f^{MAP}, A^{-1}) \tag{5.2.7}$$

Therefore,

$$\begin{aligned} p(f'|D) &= \int p(f'|f)p(f|D)df \\ &= \int N(f'; K_{x'x}K_{xx}^{-1}f, K_{x'x'} - K_{x'x}K_{xx}^{-1}K_{xx'})N(f; f^{MAP}, A^{-1}) \\ &= N(f'; K_{x'x}K_{xx}^{-1}f^{MAP}, K_{x'x'} - K_{x'x}A^{-1}K_{xx'}) \end{aligned} \tag{5.2.8}$$

5.2.2 Example - Binary case

For binary responses(-1/+1), we select sigmoid σ as link function, which is $p(y = +1|f(x)) = \sigma(f(x))$. Then following the previous subsection:

$$\ln p(f|D) = \ln p(f) + \sum_i \ln \sigma(y_i f(x_i)) + const \tag{5.2.9}$$

$$\nabla \ln p(f|D) = -K_{xx}^{-1}f + g_f \tag{5.2.10}$$

$$(g_f)_i = \sigma(-y_i f(x_i))y_i \tag{5.2.11}$$

$$\nabla\nabla \ln p(f|D) = -K_{xx} - D_f \tag{5.2.12}$$

$$D_f = diag(\sigma(f) \circ \sigma(f)) \tag{5.2.13}$$

$$\implies p(f'|y) \simeq N(f'; K_{x'x}K_{xx}^{-1}f^{MAP}, K_{x'x'} - K_{x'x}(K_{xx} + D_{f^{MAP}})^{-1}K_{xx'}) \tag{5.2.14}$$

5.3 Large-Scale Kernel Approximation

In GP regression or classification, it is usually expensive to compute the inverse of a $n \times n$ matrix, when the sample goes large. To avoid the $O(n^3)$ computation, scalable methods are necessary.

5.3.1 Low Rank Matrix Approximation

In GP regression5.1.5, we need to compute $(K_{xx}+\sigma^2 I)^{-1}$ which scales to $O(n^3)$. To avoid this, we approximate K_{xx} with $Q \in \mathcal{R}^{n\times m}, m << n$,

$$K_{xx} \simeq QQ^T \tag{5.3.1}$$

Apply the matrix inversion lemma:

$$(QQ^T + \sigma^2 I)^{-1} = \sigma^2 I - \sigma^2 Q(\sigma^2 I + Q^T Q)^{-1} Q^T \tag{5.3.2}$$

And thus, the computation of **??** is $O(m^3) << O(n^3)$

5.3.2 Random Fourier Features

Recap that in Ridge Linear Regression, $\omega = (\lambda I + \Phi^T\Phi)\Phi y$, where $\Phi = [\phi(x_1), ..., \phi(x_n) \in \mathcal{R}^{n\times m}]$. Therefore, once we figure out the feature mapϕ, the computation of parameters goes to $O(m^3)$. And thus the computation of Kernel Linear Regression is $\min\{O(n^3), O(m^3)\}$. And RFF performs a Fourier Transform on the kernel to approximate the feature map.
For $\forall$ stationary kernel, i.e. $k(x, x') = \kappa(x - x')$.

Theorem 5.3.1 (Bochner's theorem)**.** *A continuous shift-invariant kernel $k(x, y) = \kappa(xy)$ on $\mathcal{R}^P$ is positive definite if and only if $\kappa(\delta)$ is the Fourier transform of a non-negative measure.*

Given a non-negative measure $p(\omega)$(a prob density), we define the kernel as:

$$\kappa(\delta) = \int p(\omega) exp(i\omega^T\delta) d\omega \tag{5.3.3}$$

$$= \mathbf{E}_\omega[exp(i\omega^T\delta)] \tag{5.3.4}$$

To approximate the kernel with dimension m:

$$k(x, x') = \kappa(\delta) = \frac{2\kappa(0)}{m} \sum_{j=1}^{m} \cos(\hat{\omega}_j^T x + \hat{b}_j) cos(\hat{\omega}_j^T x' + \hat{b}_j) \tag{5.3.5}$$

with $\hat{b}_j \sim U(0, 2\pi)$ and $\hat{\omega}_j \sim p(\omega)$

Thus, we approximate the feature map ϕ as:

$$\phi(x) \simeq \phi_m(x) = \sqrt{\frac{2\kappa(0)}{m}}[\cos(\hat{\omega}_1^T x + \hat{b}_1), ..., \cos(\hat{\omega}_n^T x + \hat{b}_n)]^T \tag{5.3.6}$$

$$k(x, x') \simeq k_m(x, x') = \phi_m(x)^T \phi_m(x') \tag{5.3.7}$$

For more details about RFF: https://gregorygundersen.com/blog/2019/12/23/random-fourier-features/#a1-gaussian-kernel-derivation

Chapter 6

Sparse Kernel Machines

6.1 Support Vector Machine

6.1.1 Duality in Convex Optimization

Given a optimization problem:

$$\begin{aligned} &\min f_0(x) \\ &s.t. f_i(x) \leq 0, i = 1, ..., n \\ &\quad h_j(x) = 0, j = 1, ..., m \end{aligned} \tag{6.1.1}$$

The Lagrangian is defined as:

Definition 6.1.1 (Lagrangian)**.** $L(x, \lambda, \nu) = f_0(x) + \sum_i \lambda_i f_i(x) + \sum_j \nu_j h_j(x), \lambda_i \geq 0$

To optimize the primal problem, we set:

$$I_-(u) \begin{cases} 0, u \leq 0 \\ \infty, u > 0 \geq 0 \end{cases}, I_0(u) \begin{cases} 0, u = 0 \\ \infty, u \neq 0 \end{cases}$$

Let $\tilde{f}(x) := f_0(x) + \sum_i I_-(f_i(x)) + \sum_j I_0(h_j(x))$ Then, it is clear that the optimal value p^* is achieved by:

$$p^* = \inf_x \tilde{f}(x) \tag{6.1.2}$$

Besides,

$$\begin{aligned} \tilde{f}(x) &= f_0(x) + \sum_i I_-(f_i(x)) + \sum_j I_0(h_j(x)) \\ &= f_0(x) + \sum_i \sup_{\lambda_i} \lambda_i f_i(x) + \sum_j \sup_{\nu_j} \nu_j f_j(x) \\ &= \sup_{\lambda \geq 0, \nu} \{f_0(x) + \sum_i \lambda_i f_i(x) + \sum_j \nu_j h_j(x)\} \\ &= \sup_{\lambda \geq 0, \nu} L(x, \lambda, \nu) \end{aligned} \tag{6.1.3}$$

Thus,

$$p^* = \inf_x \sup_{\lambda \geq 0, \nu} L(x, \lambda, \nu) \tag{6.1.4}$$

which is a minmax problem. But it is unsolvable!

To solve the primal problem, we consider constructing a dual problem and solve it instead. And it is natural to think about $\sup_{\lambda \geq 0, \nu} \inf_x L(x, \lambda, \nu)$, which is of the maxmin format. Let $g(\lambda, \nu) = \inf_x L(x, \lambda, \nu)$

$$d^* = \sup_{\lambda \geq 0, \nu} \inf_x L(x, \lambda, \nu) \tag{6.1.5}$$

$$= \sup_{\lambda \geq 0, \nu} g(\lambda, \nu) \tag{6.1.6}$$

But it is noticeable that $\inf_x \sup_{\lambda,\nu} L(x, \lambda, \nu) \geq \sup_{\lambda,\nu} \inf_x L(x, \lambda, \nu)$, which means that the result computed by optimizing the dual problem is a lower bound of the primal problem. Thus, solving the dual problem would not get a better solution. And $p^* - d^*$ is called *optimal duality gap*. If the duality gap is 0, i.e. $p^* - d^* = 0$, a strong duality holds.

Definition 6.1.2 (K.K.T conditions)**.** *The K.K.T conditions for a given convex optimization problem is that:*

$$\begin{cases} f_i(x) \leq 0, i = 1, ..., n \\ h_j(x) = 0, j = 1, ..., m \\ \lambda_i \geq 0, i = 1, ..., n \\ \lambda_i f_i(x) = 0, i = 1, ..., n \\ \nabla f_0(x^*) + \sum_i \lambda_i^* \nabla f_i(x^*) + \sum_j \nu_j^* \nabla h_j(x^*) = 0 \end{cases}$$

Theorem 6.1.1. *If objective function is differentiable, and constraint functions satisfy Slater's Condition(i.e.*$\exists x^* \in \inf \tilde{f}$ *such that* $f_i(x^*) < 0$ *and* $h_j(x^*) = 0$*), then*

K.K.T conditions hold $\iff$ *global optimality exists*

Proof. Here's an informal proof, but could give a hint why this theorem comes. If strong duality holds:

$$\begin{aligned} f_0(x^*) &= g(\lambda^*, \nu^*) \\ &= \inf_x f_0(x) + \sum_i \lambda_i^* f_i(x) + \sum_j \nu_j^* h_j(x) \\ &\leq f_0(x^*) + \sum_i \lambda_i^* f_i(x^*) + \sum_j \nu_j^* h_j(x)^* \\ &\leq f_0(x^*) \\ \implies &\sum_i \lambda_i^* f_i(x^*) = 0 \\ \iff &\lambda_i^* f_i(x^*) = 0 \ \textit{(Complementary Slackness)} \end{aligned}$$

$$\iff \begin{cases} \lambda_i^* > 0 \implies f_i(x^*) = 0 \\ f_i(x^*) < 0 \implies \lambda_i^* = 0 \end{cases} \textit{(Complementary Slackness)}$$

Besides, since $x*$ is an optimality, then

$$\begin{aligned}&\nabla\tilde{f}(x)|_{x=x^*}=0\\ \Longleftrightarrow&\nabla f_0(x^*)+\sum_i\lambda_i^*\nabla f_i(x^*)+\sum_j\nu_j^*\nabla h_j(x^*)=0\end{aligned}$$

Above all, that's why K.K.T conditions come. □

6.1.2 SVM for classification

Linear-seperable case

When the data is linear seperable, we're going to find a hyperplane such that $\hat{y}=1$ if $\omega^Tx+b\geq 1$. To find the best hyperplane, we're going to maximize the margin, i.e.

$$\begin{aligned}&\max_{\omega,b}\frac{2}{\|\omega\|}\\ &s.t.\ y_i(\omega^Tx_i+b)\geq 1, i=1,...,n\\ \Longleftrightarrow&\min_{\omega,b}\frac{1}{2}\|\omega\|^2\\ &s.t.\ y_i(\omega^Tx_i+b)\geq 1, i=1,...,n\end{aligned}\tag{6.1.7}$$

Non-linear-seperable case

If the data is not linear seperable, we have to allow a certain number of errors, by adding a $0-1$ loss term:

$$\min_{\omega,b}\frac{1}{2}\|\omega\|^2+C\sum_{i=1}^n\mathbb{1}(y_i(\omega^Tx_i+b)<0)\tag{6.1.8}$$

where C controls the trade-off between maximum margin and loss. However, $0-1$ loss is nondifferentiable at 0, which might cause inconvience in GD-based optimization methods. To solve that, we use Hinge loss instead:

Definition 6.1.3 (Hinge loss)**.**

$$h(\alpha)=(1-\alpha)_+\begin{cases}1-\alpha,\ \textit{if}\ 1-\alpha>0\\ 0,\ \textit{else}\end{cases}$$

Thus, the primal problem becomes:

$$\min_{\omega,b}\frac{1}{2}\|\omega\|^2+C\sum_{i=1}^n h(y_i(\omega^Tx_i+b))\tag{6.1.9}$$

$$\Longleftrightarrow\min_{\omega,b,\xi}\frac{1}{2}\|\omega\|^2+C\sum_{i=1}^n\xi_i\tag{6.1.10}$$

$$s.t.\ \begin{cases}1-y_i(\omega^Tx_i+b)\leq\xi_i\\ \xi_i\geq 0\end{cases}$$

which is called c-SVM. Then, the Lagrangian is:

$$L(\omega, b, \xi, \alpha, \lambda) = \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i(1 - y_i(\omega^T x_i + b) - \xi_i) + \sum_{i=1}^{n}\lambda_i(-\xi_i) \tag{6.1.11}$$

and,

$$\begin{cases} \dfrac{\partial L}{\partial \omega} = \omega - \sum\limits_i \alpha_i y_i x_i = 0 \Longrightarrow \omega^* = \sum\limits_{i=1}^{n}\alpha_i y_i x_i \\ \dfrac{\partial L}{\partial b} = -\sum\limits_i \alpha_i y_i = 0 \\ \dfrac{\partial L}{\partial \xi_i} = C - \alpha_i - \lambda_i = 0 \Longrightarrow \alpha_i = C - \lambda_i \end{cases} \tag{6.1.12}$$

Inputting ω^*, b^* and ξ^* from 6.1.12 to g, we get

$$\begin{aligned} g(\alpha, \lambda) &= \frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j y_i y_j x_i^T x_j + \sum_i \alpha_i - \sum_i \alpha_i y_i (\sum_j \alpha_j y_j x_j)^T x_i \\ &= \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j \end{aligned} \tag{6.1.13}$$

Combing the conditions from 6.1.12,

$$\max_{\alpha,\lambda} g(\alpha, \lambda) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j \tag{6.1.14}$$

$$s.t. \quad \begin{cases} \sum\limits_i \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C \end{cases}$$

This optimization problem could be easily solved, as it is a constrained quadratic optimization over α!

After that, the optimal params are obtained by:

$$\begin{cases} \omega^* = \sum\limits_{i=1}^{n}\alpha^* y_i x_i \\ b^* = \dfrac{1}{y_i} - \omega^{*T} x_i, \textit{ for any margin support vectors, i.e. those } 0 < \alpha_i < C \end{cases} \tag{6.1.15}$$

Definition 6.1.4 (Support vectors)**.**

1. *Non-margin support vectors:* $\alpha_i = C > 0$
2. *Margin support vectors:* $0 < \alpha_i < C$
3. *Non support vectors:* $\alpha_i = 0$

By this definition, it is clear that params of SVM are only determined by support vectors!

Kernel SVM

To include nonlinearity, we combine SVM with kernel method. Suppose the response function is given by $f(x) = \omega^T \phi(x) + b$, instead of $\omega^T x + b$. Following the notion in Chapter4, given a kernel function k, note $k(x, x') = \phi(x)^T \phi(x')$, $K \in \mathcal{R}^{n \times n}$ and $K_{ij} = k(x_i, x_j)$. Following the same procedure in the former subsection, we get:

$$\begin{aligned} g(\alpha, \lambda) &= \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \\ &= \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K_{ij} \end{aligned} \tag{6.1.16}$$

$$\omega^* = \sum_{i=1}^{n} \alpha^* y_i \phi(x_i) \tag{6.1.17}$$

$$f(x) = \sum_{i=1}^{n} \alpha^* y_i k(x, x_i) + b^* \tag{6.1.18}$$

6.1.3 SVM for regression

Similar with SVM for classification, in regression task, SVM use another error function $E_\epsilon(\alpha)$, which is called ϵ-insensitive error function.

Definition 6.1.5 (ϵ-insensitive error function)**.**

$$E_\epsilon(\alpha) = \begin{cases} 0, \ if \ |\alpha| < \epsilon \\ |\alpha| - \epsilon, \ else \end{cases} \tag{6.1.19}$$

Thus, the optimization problem becomes,

$$\min_{\omega, b} \frac{1}{2} \|\omega\|^2 + C \sum_{i=1}^{n} E_\epsilon(\omega^T \phi(x_i) + b - y_i) \tag{6.1.20}$$

By introducing the slack variables ξ and $\hat{\xi}$, then the previou problem is equivalent to

$$\min_{\omega, b, \xi} \frac{1}{2} \|\omega\|^2 + C \sum_{i=1}^{n} (\xi_i + \hat{\xi}_i) \tag{6.1.21}$$

$$s.t. \begin{cases} \xi_i \geq 0 \\ \hat{\xi}_i \geq 0 \\ \xi_i \geq \omega^T \phi(x_i) + b - y_i - \epsilon \\ \hat{\xi}_i \geq y_i - \omega^T \phi(x_i) - b - \epsilon \end{cases} \tag{6.1.22}$$

$$\begin{aligned}\implies L(\omega, b, \xi, \hat{\xi}, \alpha, \hat{\alpha}, \lambda, \hat{\lambda}) = \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{n}(\xi_i + \hat{\xi}_i) - \sum_i \lambda_i\xi_i - \sum_i \hat{\lambda}_i\hat{\xi}_i \\ + \sum_i \alpha_i(\omega^T\phi(x_i) + b - y_i - \epsilon - \xi_i) \\ + \sum_i \hat{\alpha}_i(y_i - \omega^T\phi(x_i) - b - \epsilon - \hat{\xi}_i) \end{aligned} \tag{6.1.23}$$

$$\implies \begin{cases} \dfrac{\partial L}{\partial \omega} = \omega + \sum_i \alpha_i\phi(x_i) - \sum_i \hat{\alpha}_i\phi(x_i) = 0 \\ \dfrac{\partial L}{\partial b} = \sum_i (\alpha_i - \hat{\alpha}_i) = 0 \\ \dfrac{\partial L}{\partial \xi_i} = C - \alpha_i - \lambda_i = 0 \\ \dfrac{\partial L}{\partial \hat{\xi}_i} = C - \hat{\alpha}_i - \hat{\lambda}_i = 0 \end{cases} \tag{6.1.24}$$

By combing 6.1.21 and 6.1.24,

$$\begin{aligned} g(\alpha, \hat{\alpha}, \lambda, \hat{\lambda}) = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i - \hat{\alpha}_i)(\alpha_j - \hat{\alpha}_j)K_{ij} \\ - \sum_i (\alpha_i - \hat{\alpha}_i)y_i - \epsilon\sum_i (\alpha_i + \hat{\alpha}_i) \end{aligned} \tag{6.1.25}$$

$$\omega^* = \sum_{i=1}^{n}(\hat{\alpha}_i - \alpha_i)\phi(x_i) \tag{6.1.26}$$

$$\tag{6.1.27}$$

By K.K.T conditions,

$$\begin{cases} \alpha_i(f(x_i) - y_i - \epsilon - \xi_i) = 0 \\ \hat{\alpha}_i(y_i - f(x_i) - \epsilon - \hat{\xi}_i) = 0 \\ \lambda_i\xi_i = (C - \alpha_i)\xi_i = 0 \\ \hat{\lambda}_i\hat{\xi}_i = (C - \hat{\alpha}_i)\hat{\xi}_i = 0 \end{cases} \tag{6.1.28}$$

By K.K.T conditions,

1. if $\alpha_i \neq 0 \implies f(x_i) - y_i - \epsilon - \xi_i = 0 \implies f(x_i) - y_i - \epsilon \geq 0$, which means that the data point lies on or above the uppper boundary of ϵ-tube
2. if $\hat{\alpha}_i \neq 0$, similarly, the data point lies on or below the lower boundary of ϵ-tube
3. if $\alpha_i = \hat{\alpha}_i = 0$, the data point lies within the ϵ-tube

Thus, the support vectors are those $\alpha_i \neq 0$ or $\hat{\alpha}_i \neq 0$. And for $0 < \alpha_i < C$, by K.K.T, $\xi_i = 0$. Then $f(x_i) - y_i - \epsilon = 0$ and thus,

$$b^* = y_i + \epsilon - \omega^{*T}\phi(x_i) \tag{6.1.29}$$

And thus the predictive function is

$$f(x) = \sum_{i=1}^{n}(\hat{\alpha}_i - \alpha_i)k(x, x_i) + b^* \tag{6.1.30}$$

6.2 Relevance Vector Machines

$$cons\ of\ SVMs \begin{cases} 1.output\ are\ decisions,\ not\ posterior \\ 2.2\text{-}classes\ for\ classification \\ 3.hyperparam\ C \end{cases}$$

$\Longrightarrow$RVM: A Bayesian sparse kernel technique.

6.2.1 RVM for Reg

Intro

Suppose

$$p(y|x,w,\beta) = N(y|f(x),\beta^{-1}) \tag{6.2.1}$$

$$with \begin{cases} \beta = \sigma^2 \\ f(x) = \sum_{i=1}^{M} w_i\phi_i(w) = w^T\phi(w) \end{cases} \tag{6.2.2}$$

Similar with SVM, we form $f(x)$ as a linear combination of a kernel function evaluating at each pair (x, x_i):

$$f(x) = w_0 + \sum_{i=1}^{N} w_i k(x,x_i), w \in \mathcal{R}^{N+1} \tag{6.2.3}$$

We then obtain the likelihood:

$$p(D|w) = \Pi_{i=1}^{N} N(y_i|f(x_i),\beta^{-1}) \tag{6.2.4}$$

Besides, assign a prior for w:

$$p(w|\alpha) = N(w|0, diag(\alpha_i^{-1})) \tag{6.2.5}$$

By 1.3.2 and we have posterior:

$$p(w|D,\alpha,\beta) = N(w|m,\Sigma) \tag{6.2.6}$$

$$with \begin{cases} m = \beta\Sigma Ky \\ \Sigma = (diag(\alpha_i) + \beta K^T K)^{-1} \end{cases} \tag{6.2.7}$$

and evidence:

$$p(D|\alpha,\beta) = N(y|0, \beta^{-1}I + K^T diag(\alpha_i^{-1})K) \tag{6.2.8}$$

Hyperparam Optimization

To optimize α and β, we maximize the evidence:

$$\begin{aligned} \hat{\alpha,\beta} &= \arg\max_{\alpha,\beta} p(D|\alpha,\beta) \\ &= \arg\max_{\alpha,\beta} \log p(D|\alpha,\beta) \\ &= \arg\max_{\alpha,\beta} -\frac{1}{2}(\log|C| + y^T C^{-1} y) \end{aligned} \tag{6.2.9}$$

where C is the covariance matrix of evidence: $\beta^{-1}I + K^T diag(\alpha_i^{-1})K$.

Relevance vectors

Similar to SVMs, if $w_i^* \neq 0$, then x_i is called relevance vector.

Prediction

$$p(y'|x', D, \alpha^*, \beta^*) = \int p(y'|w, x', \beta^*)p(w|D, \alpha^*)dw \tag{6.2.10}$$

$$= \int N(y'|f(x'), \beta^{*-1})N(w|m, \Sigma)dw \tag{6.2.11}$$

$$= N(y'|m^T K_{x'x}, \beta^{*-1} + K_{x'x}^T \Sigma K_{x'x}) \tag{6.2.12}$$

6.2.2 RVM for clf

Let,

$$p(y = k|x, w) = f_k(x, w) = \frac{a_k}{\sum_j a_j} \tag{6.2.13}$$

$$with\ a_k = \sum_{i=1}^{N} w_i k(x, x_i) + w_0 \tag{6.2.14}$$

and we use the same prior for α as RVM Reg6.2.5, then the likelihood is:

$$p(D|w, \alpha, \beta) = \Pi_{n=1}^{N} \Pi_{k=1}^{K} f_k(x_n, w)^{y_{nk}} \tag{6.2.15}$$

and log-posterior:

$$\log p(w|D, \alpha, \beta) = \sum_n \sum_k y_{nk} \log f_k(x, w) + \sum_{j=1}^{M} \log N(w_j|0, \alpha_j^{-1}) \tag{6.2.16}$$

Chapter 7

Graphical Models

Definition 7.0.1 (Graph)**.**

1. *DAG: directed acycle graph, leads to Bayesian Net*
2. *UG: undirected graph, leads to Markov Random Field*

7.1 Bayesian Network

7.1.1 Intro

Given a graph (G, X) with N nodes, we could compute the prob of this graph by:

$$p(X) = \Pi_{n=1}^{N} p(x_n|pa_n) \tag{7.1.1}$$

$$with\ pa_n = \{a|a\ is\ a\ parent\ of\ x_n\} \tag{7.1.2}$$

e.g.

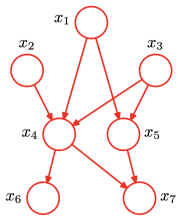


Figure 7.1: DAG

$$P(x_1, ..., x_7) = P(x_1)P(x_2)P(x_3)P(x_4|x_1, x_2, x_3)P(x_5|x_1, x_3)P(x_6|x_4)P(x_7|x_4, x_5)$$

7.1.2 Sample from graph

To sample nodes in a graph, we could sequencially sample the parant pa_n first, then $x_n|pa_n$

7.1.3 Num of params, example of discrete variables

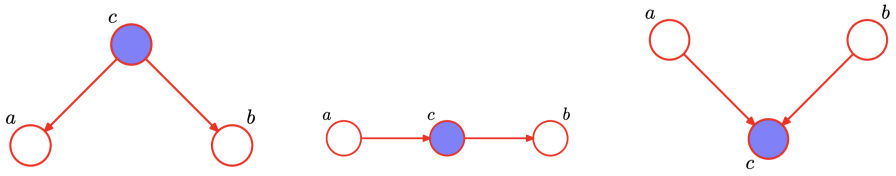
suppose there are K states for each node, then

$$For \begin{cases} fully\ connceted\ net:\ \#params = N^K - 1 \\ isolated\ net:\ \#params = N(K-1) \\ chain:\ \#params = K - 1 + (N-1)K(K-1) \end{cases}$$

7.1.4 Linear Gaussian Models

$$p(x_i|pa_i) = N(x_i|\sum_{j \in pa_i} w_{ij}x_j + b_i, v_i) \tag{7.1.3}$$

7.2 Conditional Independence



(a) tail-to-tail (b) head-to-tail (c) head-to-head

Figure 7.2: Conditional independence

$$\begin{cases} c\ is\ tail\text{-}to\text{-}tail \Longrightarrow a \perp b|c \\ c\ is\ head\text{-}to\text{-}tail \Longrightarrow a \perp b|c \\ c\ is\ head\text{-}to\text{-}head \Longrightarrow a \not\perp b|c, (\ but\ a \perp b) \end{cases}$$

7.3 Markov Random Field

Definition 7.3.1 (Markov Random Field)**.** *Markov Random Field(MRF), also called Markov Net, is an undirected graph model.*

Definition 7.3.2 (Clique)**.** *Given a UG (G, X), a subset $\tilde{X} \subseteq X$ is called a clique, if $\forall \tilde{x}_i, \tilde{x}_j \in \tilde{X}$, $\tilde{x}_i \leftrightarrow \tilde{x}_j$*

Definition 7.3.3 (Maximum clique)**.** *A clique is called a maximum clique if adding any other node into this clique makes it no longer a clique.*

Suppose C is a set of maximum cliques of an undirected graph G, then,

$$p(X) = \frac{1}{Z}\Pi_c \psi_c(X_c) \tag{7.3.1}$$

$$Z = \int_{\mathcal{X}} \Pi_c \psi_c(X_c) dX \tag{7.3.2}$$

where $\psi(\cdot)$ is a non-negative potential and usually,

$$\psi(X_c) = exp(-E(X_c)) \tag{7.3.3}$$

where $E(\cdot)$ is called energy function. Thus,

$$p(X) = \frac{1}{Z} exp(-\sum_{c} E(X_c)) \tag{7.3.4}$$

7.4 Inference in Graphical Models

7.4.1 Inference on a chain

Given a chain with nodes $\{x_n\}_{n=1}^N$, by Bayesian Net,

$$P(X) = \frac{1}{Z}\psi_{12}(x1, x2)...\psi_{N-1,N}(x_{N-1}, x_N) \tag{7.4.1}$$

To compute the marginal $P(x_n)$,

$$P(x_n) = \frac{1}{Z}\sum_{x_1}...\sum_{x_{n-1}}\sum_{x_{n+1}}...\sum_{x_N}\psi_{12}(x_1, x_2)...\psi_{n-1,n}(x_{n-1}, x_n) \tag{7.4.2}$$

$$= \frac{1}{Z}\{\sum_{x_1}...\sum_{x_{n-1}}\psi_{12}(x_1, x_2)...\psi_{n-1,n}(x_{n-1}, x_n)\} \tag{7.4.3}$$

$$\{\sum_{x_{n+1}}...\sum_{x_N}\psi_{12}(x_{n+1}, x_{n+2})...\psi_{N-1,N}(x_{N-1}, x_N)\} \tag{7.4.4}$$

$$= \frac{1}{Z}\{\sum_{x_{n-1}}\psi_{n-1,n}(x_{n-1}, x_n)...[\sum_{x_2}\psi_{23}(x_2, x_3)[\sum_{x_1}\psi_{12}(x_1, x_2)]]\} \tag{7.4.5}$$

$$\{\sum_{x_{n+1}}\psi_{n+1,n+2}(x_{n+1}, x_{n+2})...[\sum_{x_{N-1}}\psi_{N-1,N-2}(x_{N-1}, x_{N-2})[\sum_{x_N}\psi_{N-1,N}(x_{N-1}, x_N)]]\} \tag{7.4.6}$$

let,

$$\mu_\alpha(x_n) = \sum_{x_{n-1}}\psi_{n-1,n}(x_{n-1}, x_n)...[\sum_{x_2}\psi_{23}(x_2, x_3)[\sum_{x_1}\psi_{12}(x_1, x_2)]] \tag{7.4.7}$$

$$\mu_\beta(x_n) = \sum_{x_{n+1}}\psi_{n+1,n+2}(x_{n+1}, x_{n+2})...[\sum_{x_N}\psi_{N-1,N}(x_{N-1}, x_N)] \tag{7.4.8}$$

then, the calculation of marginal is given by the forward-backward algorithm:

$$P(x_n) = \frac{1}{Z}\mu_\alpha(x_n)\mu_\beta(x_n) \tag{7.4.9}$$

$$\mu_\alpha = \sum_{x_{n-1}}\psi_{n-1,n}(x_{n-1}, x_n)\mu_\alpha(x_{n-1}) \tag{7.4.10}$$

$$\mu_\beta = \sum_{x_{n+1}}\psi_{n,n+1}(x_n, x_{n+1})\mu_\alpha(x_{n+1}) \tag{7.4.11}$$

$$P(x_{n-1}, x_n) = \mu_\alpha(x_{n-1})\psi_{n-1,n}(x_{n-1}, x_n)\mu_\beta(x_{n+1}) \tag{7.4.12}$$

7.5 Inference on Factor Graph

Given a graph with two type of nodes: original nodes(called nodes) and factor nodes(called factors, is a function), where each nodes are associated via factors.

Thus,

$$P(X) = \Pi_s f_s(X_s) \tag{7.5.1}$$

where f_s is a factor and $X_s \subseteq X$ represents nodes associated with f_s.

e.g.

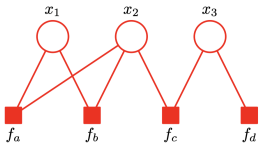


Figure 7.3: Factor graph

$$P(x_1, x_2, x_3) = f_a(x_1, x_2) f_b(x_1, x_2) f_c(x_2, x_3) f_d(x_3)$$

Given a tree structure factor graph, a joint distribution could be expressed by the product of factors which are its neighbors, that is

$$P(X) = \Pi_{s \in ne(x)} F_s(x, X_s) \tag{7.5.2}$$

where $ne(x)$ indicates factor nodes that are neighbor to x. Then the marginal is:

$$P(x) = \sum_{X \setminus x} \Pi_{s \in ne(x)} F_s(x, X_s) \tag{7.5.3}$$

$$= \Pi_{s \in ne(x)} \sum_{X_s} F_s(x, X_s) \tag{7.5.4}$$

Define,

$$\mu_{f_s \to x}(x) = \sum_{X_s} F_s(x, X_s) \tag{7.5.5}$$

Let $X_s = (x_1, ..., x_M)$, and notice that F_s could be decomposed by

$$F_s(x, X_s) = f_s(x, X_s) G_1(x_1, X_{s1}) ... G_M(x_M, X_{sM}) \tag{7.5.6}$$

Then,

$$\mu_{f_s \to x}(x) = \sum_{X_s} f_s(x, X_s) \Pi_{m \in ne(f_s) \setminus x} G_m(x_m, X_{sm}) \tag{7.5.7}$$

$$= \sum_{X_s} f_s(x, X_s) \Pi_{m \in ne(f_s) \setminus x} \sum_{X_{sm}} G_m(x_m, X_{sm}) \tag{7.5.8}$$

Define,

$$\mu_{x_m \to f_s}(x_m) = \sum_{X_{sm}} G_m(x_m, X_{sm}) \tag{7.5.9}$$

Also,

$$G_m(x_m, X_{sm}) = \Pi_{l \in ne(x_m) \setminus s} F_l(x_m, X_{ml}) \tag{7.5.10}$$

Thus,

$$P(x) = \Pi_{s \in ne(x)} \mu_{f_s \to x}(x) \tag{7.5.11}$$

$$\mu_{f_s \to x}(x) = \sum_{X_s} f_s(x, X_s) \Pi_{m \in ne(f_s) \setminus x} \mu_{x_m \to f_s}(x_m) \tag{7.5.12}$$

$$\begin{aligned} \mu_{x_m \to f_s}(x_m) &= \sum_{X_{sm}} \Pi_{l \in ne(x_m) \setminus s} F_l(x_m, X_{ml}) \\ &= \Pi_{l \in ne(x_m) \setminus s} \mu_{f_l \to x_m}(x_m) \end{aligned} \tag{7.5.13}$$

$$\textit{with leaf nodes: } \mu_{f \to x}(x) = f(x) \textit{ and } \mu_{x \to f}(x) = 1 \tag{7.5.14}$$

The previous procedure is called Sum-Product Algorithm.

e.g. calculating marginal of x_2

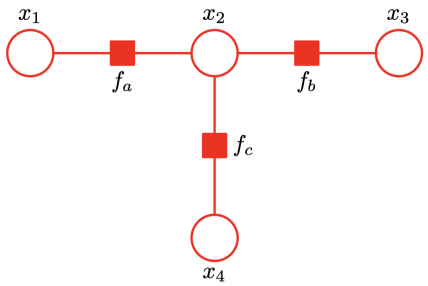


Figure 7.4: Sum-product algorithm example

$$\mu_{x_1 \to f_a}(x_1) = 1, \mu_{x_3 \to f_b}(x_2) = 1, \mu_{x_4 \to f_c}(x_3) = 1$$

$$\mu_{f_a \to x_2}(x_2) = \sum_{x_1} f_a(x_1, x_2) \mu_{x_1 \to f_a}(x_1) = \sum_{x_1} f_a(x_1, x_2)$$

$$\mu_{f_b \to x_2}(x_2) = \sum_{x_3} f_a(x_2, x_3) \mu_{x_1 \to f_b}(x_3) = \sum_{x_3} f_b(x_2, x_3)$$

$$\mu_{f_c \to x_2}(x_2) = \sum_{x_4} f_a(x_2, x_4) \mu_{x_1 \to f_c}(x_4) = \sum_{x_4} f_c(x_2, x_4)$$

$$\begin{aligned} p(x_2) &= \Pi_{s \in ne(x_2)} \mu_{f_s \to x_2}(x_2) \\ &= \mu_{f_a \to x_2}(x_2) \mu_{f_b \to x_2}(x_2) \mu_{f_c \to x_2}(x_2) \\ &= \sum_{x_1} f_a(x_1, x_2) \sum_{x_3} f_b(x_2, x_3) \sum_{x_4} f_c(x_2, x_4) \end{aligned}$$

Chapter 8

EM

8.1 EM algorithm

8.1.1 EM in an optimization viewpoint

In some probabilistic modeling cases, we might encounter hidden variables, like mixture gaussians, resulting that we have to consider a joint distribution about the observation and hidden states. For simplification, we assume Z is discrete, and for continous variable, it's easy to just change the summation into an integral. For example, in a maximum likelihood settings,

$$\max_{\theta} \log P(X|\theta) = \max_{\theta} \log \sum_{Z} P(X, Z|\theta) \tag{8.1.1}$$

However, even if X and Z could be assumed as variables from exponential family, the summation inside the logarithm makes the joint distribution intractable. To solve this problem, an intuition is to switch the log and $\sum$, with the form: $\max_\theta \sum_z \log P(X, Z|\theta)$ which results in a type of algorithm called EM algorithm.

By Cauchy-Schwarz Inequality:

$$\log \sum_{Z} P(X, Z|\theta) = \log \sum_{Z} q(Z) \frac{P(X, Z|\theta)}{q(Z)} \tag{8.1.2}$$

$$\geq \sum_{Z} q(Z) \log \frac{P(X, Z|\theta)}{q(Z)} \tag{8.1.3}$$

to make the inequality tight, we must have: $\frac{P(X,Z|\theta)}{q(Z)}$ is constant with the variable Z. While $P(X, Z|\theta) \propto P(Z|X, \theta)P(X|\theta)$, then we must have:

$$q(Z) = P(Z|X, \theta) \tag{8.1.4}$$

then we have,

$$\log P(X|\theta) \geq \sum_{Z} P(Z|X, \theta) \log \frac{P(X, Z|\theta)}{P(Z|X, \theta)} \tag{8.1.5}$$

$$= \mathbf{E}_{Z\sim P(Z|X,\theta)}[\log \frac{P(X, Z|\theta)}{P(Z|X, \theta)}] \tag{8.1.6}$$

Then, instead of optimizing the likelihood $\log P(X|\theta)$, we optimize its lower bound:

$$\max_{\theta} \mathbf{E}_{Z\sim P(Z|X,\theta)}[\log \frac{P(X,Z|\theta)}{P(Z|X,\theta)}] \tag{8.1.7}$$

resulting to the EM algorithm:

1. initialize θ randomly, i.e. $\theta^{(0)}$
2. sample $Z = (Z_1, ..., Z_M)$ from its posterior $P(Z|X, \theta^{(t)})$
3. E step: compute $Q(\theta, \theta^{(t)}) = \frac{1}{M}\sum_{m=1}^{M} \log \frac{P(X,Z_m|\theta)}{P(Z_m|X,\theta^{(t-1)})}$
4. M step: $\theta^{(t+1)} = \arg\max_{\theta} Q(\theta, \theta^{(t)})$

8.1.2 EM in KL-divergence viewpoint

In other perspective, we could derive EM from KL-divergence. First recap the definition of KL-divergence, which measures the distance between two probability distributions.

Definition 8.1.1 (KL-divergence)**.**

$$KL(q||p) = -\sum_{z} q(z) \log \frac{p(z)}{q(z)} \tag{8.1.8}$$

, $KL(q||p) \geq 0$ and it is not symmetric, i.e. $KL(q||p) \neq KL(p||q)$.

And we define $Q(q, \theta) = \sum_Z q(Z) \log \frac{P(X,Z|\theta)}{q(Z)}$, then,

$$\begin{aligned}
\log P(X|\theta) &= \log \frac{P(X,Z|\theta)}{P(Z|X,\theta)} \\
&= \sum_Z q(Z) \log \frac{P(X,Z|\theta)}{P(Z|X,\theta)} \\
&= \sum_Z q(Z) \log \frac{P(X,Z|\theta)q(Z)}{q(Z)P(Z|X,\theta)} \\
&= \sum_Z q(Z) \log \frac{P(X,Z|\theta)}{q(Z)} - \sum_Z q(Z) \log \frac{P(Z|X,\theta)}{q(Z)} \\
&= Q(q,\theta) + KL(q||P(Z|X,\theta))
\end{aligned} \tag{8.1.9}$$

Notice that KL-divergence is non-negative, then

$$\log P(X|\theta) \geq Q(q, \theta) \tag{8.1.10}$$

the inequality is tight if and only if $KL(q||P(Z|X,\theta)) = 0$, which indicates $q(Z) = P(Z|X,\theta)$. And thus,

$$\log P(X|\theta) = Q(P(Z|X,\theta),\theta) = \mathbf{E}_{Z\sim P(Z|X,\theta)}[\log \frac{P(X,Z|\theta)}{P(Z|X,\theta)}] \tag{8.1.11}$$

Thus, the EM algorithm could be revisited in a coordinate-ascent perspective:

$$\text{E step} : q^{(t)} = \arg\max_{q} Q(q, \theta^{(t)}) \tag{8.1.12}$$

$$\text{M step} : \theta^{(t+1)} = \arg\max_{\theta} Q(q^{(t)}, \theta) \tag{8.1.13}$$

8.1.3 Convergence of EM

From the coordinate ascent viewpoint:

$$\log P(X|\theta^{(t)}) = Q(\theta^{(t)}, q) + KL(q||P(Z|X, \theta^{(t)})) \tag{8.1.14}$$
$$= Q(\theta^{(t)}, q^{(t)}) \tag{8.1.15}$$
$$\leq Q(\theta^{(t+1)}, q^{(t)}) \tag{8.1.16}$$
$$\leq Q(\theta^{(t+1)}, q^{(t+1)}) \tag{8.1.17}$$
$$= \log P(X|\theta^{(t+1)}) \tag{8.1.18}$$

thus, the EM algorithm would converge.

8.1.4 EM for Bayesian

In a Bayesian settings, we are optimizing:

$$\max_{\theta} \log P(\theta|X) \Leftrightarrow \max_{\theta} \log P(X|\theta) + \log P(\theta) \tag{8.1.19}$$

Thus, to use EM to optimize the posterior, we just need to modify the M-step by:

$$\theta^{(t+1)} = \arg\max_{\theta} Q(q^{(t)}, \theta) + \log P(\theta) \tag{8.1.20}$$

8.2 EM examples

8.2.1 K-means algorithm

To cluster a set of data into K clusters, given K initial points $m_1, ...m_K$, we iteratively update the clusters by:

$$s_n = \arg\min_k \|x_n - m_k\|^2$$
$$m_k = \frac{1}{N_k} \sum_{n \in C_k} x_n$$

which sequentially minimizes the cost function:

$$c(\{s_{nk}\}, \{m_k\}) = \sum_k \sum_{n \in C_k} \|x - m_k\|^2$$

And the previous optimization could be viewed as an EM algorithm:

$$\textit{E-step} : m_k = \arg\min_{m_k} c(\{s_{nk}\}, \{m_k\}) = \frac{1}{N_k} \sum_{n \in C_k} x_n$$
$$\textit{M-step} : s_{nk} = \arg\min_{s_{nk}} c(\{s_{nk}\}, \{m_k\}) = \arg\min_k \|x_n - m_k\|^2$$

Cons of K-means:

1. hard-allocation: there is no prob distribution about the clustering results
2. Sensitive to initializations: K-means++, which first apply K-means to a small subset of training data, then use its clusters' mean as the original points and start K-means for whole dataset.

8.2.2 Mixture Gaussians

In mixture gaussians setting, we suppose an observation is formed as:

$$p(x|\pi,\mu,\Sigma) = \sum_{k=1}^{K} \pi_k N(x|\mu_k,\Sigma_k) \tag{8.2.1}$$

Introduce a hidden state $z \in \{0,1\}^K$ indicates the belonging of gaussians, with $\sum_k z_k = 1$, then

$$p(x|z=k,\mu_k,\Sigma_k) = N(x|\mu_k,\Sigma_k) \tag{8.2.2}$$

$$p(X,Z|\pi,\mu,\Sigma) = \Pi_{n=1}^{N}\Pi_{k=1}^{K}(\pi_k N(x_n|\mu_k,\Sigma_k))^{z_{nk}} \tag{8.2.3}$$

Let $\theta = (\pi,\mu,\Sigma)$, with $z \in \{0,1\}^K$:

$$q(z) = p(z|x,\theta) \tag{8.2.4}$$

$$\propto p(x|z,\theta)p(z|\theta) \tag{8.2.5}$$

$$= \Pi_{k=1}^{K}(\pi_k N(x|\mu_k,\Sigma_k))^{z_k} \tag{8.2.6}$$

$$\Rightarrow \mathbf{E}[z_{nk}] = p(z_{nk}=1) = \frac{\pi_k N(x_n|\mu_k,\Sigma_k)}{\sum_j \pi_j N(x_n|\mu_j,\Sigma_j)} \tag{8.2.7}$$

$$:= \gamma(z_{nk}) \tag{8.2.8}$$

Thus,

$$Q(\theta,q) = \mathbf{E}_{q(Z)}[\log \frac{P(X,Z|\theta)}{q(Z)}] \tag{8.2.9}$$

$$= \mathbf{E}_{q(Z)}[\log P(X,Z\theta)] - H(q(Z)) \tag{8.2.10}$$

$$\mathbf{E}_{q(Z)}[\log P(X,Z\theta)] = \mathbf{E}_{q(Z)}[\sum_n \sum_k z_{nk}\{\log \pi_k + \log N(x_n|\mu_k,\Sigma_k)\}] \tag{8.2.11}$$

$$= \sum_n \sum_k \gamma(z_{nk})\{\log \pi_k + \log N(x_n|\mu_k,\Sigma_k)\} \tag{8.2.12}$$

Thus,

$$\text{E step} : \gamma(z_{nk}^{(t)}) = \frac{\pi_k^{(t)} N(x_n|\mu_k^{(t)},\Sigma_k^{(t)})}{\sum_j \pi_j N(x_n|\mu_j^{(t)},\Sigma_j^{(t)})} \tag{8.2.13}$$

$$\text{M step} : \theta^{(t+1)} = \arg\max_{\theta=(\pi,\mu,\Sigma)} \sum_n \sum_k \gamma(z_{nk}^{(t)})\{\log \pi_k + \log N(x_n|\mu_k,\Sigma_k)\} \tag{8.2.14}$$

8.2.3 Mixture Bernoulli

Given data $\{x_n\}_{n=1}^N$, with $x_n \in \mathcal{R}^D$. Let,

$$p(x|\mu) = \Pi_{d=1}^{D} \mu_d^{x_d}(1-\mu_d)^{1-x_d} \tag{8.2.15}$$

consider:

$$p(x|\mu,\pi) = \sum_{k=1}^{K} \pi_k p(x|\mu_k) \tag{8.2.16}$$

$$p(x|\mu_k) = \Pi_{d=1}^{D} \mu_{kd}^{x_d}(1-\mu_{kd})^{1-x_d} \tag{8.2.17}$$

assign a hidden state $z \in \{0,1\}^K$:

$$p(x|z_k = 1, \mu) = \Pi_{d=1}^{D} \mu_{kd}^{x_d}(1-\mu_{kd})^{1-x_d} \tag{8.2.18}$$

$$p(X, Z|\pi, \mu) = \Pi_{n=1}^{N}\Pi_{k=1}^{K}(\pi_k p(x_n|z_{nk}=1,\mu))^{z_{nk}} \tag{8.2.19}$$

$$P(Z|X,\pi,\mu) \propto P(X,Z|\pi,\mu) \tag{8.2.20}$$

$$\Rightarrow \mathbf{E}[z_{nk}] = \frac{\pi_k \mu_{kd}^{x_{nd}}(1-\mu_{kd})}{\sum_j \pi_j \mu_{jd}^{x_{nd}}(1-\mu_{jd})} := \gamma(z_{nk}) \tag{8.2.21}$$

$$Q(\theta, q) = \mathbf{E}_{q(Z)}[\log P(X, Z|\pi,\mu)] - H(q(Z)) \tag{8.2.22}$$

$$\begin{aligned}\mathbf{E}_{q(Z)}[\log P(X,Z|\pi,\mu)] = \sum_n \sum_k \gamma(z_{nk})\{\log \pi_k + \sum_d x_{nd} \log \mu_{kd} \\ + \sum_d (1-x_{nd})\log(1-\mu_{kd})\}\end{aligned} \tag{8.2.23}$$

Thus, we optimize Q and obtain:

$$N_k = \sum_n \gamma(z_{nk}) \tag{8.2.24}$$

$$\overline{X_k} = \frac{1}{N_k}\sum_n \gamma(z_{nk}) x_n \tag{8.2.25}$$

$$\hat{\mu}_k = \overline{X_k} \tag{8.2.26}$$

$$\hat{\pi}_k = \frac{N_k}{N} \text{ (by Lagrange, constrainting that)} \sum_k \pi_k = 1 \tag{8.2.27}$$

then,

$$\text{E step} : \gamma(z_{nk}^{(t)}) = \frac{\pi_k^{(t)} \mu_{kd}^{x_{nd}(t)}(1-\mu_{kd}^{(t)})}{\sum_j \pi_j \mu_{jd}^{x_{nd}(t)}(1-\mu_{jd}^{(t)})} \tag{8.2.28}$$

$$\text{M step} : \theta^{(t+1)} = \left(\frac{\sum_n \gamma(z_{nk}^{(t)}) x_n}{\sum_n \gamma(z_{nk}^{(t)})}, \frac{\sum_n \gamma(z_{nk}^{(t)})}{N}\right) \tag{8.2.29}$$

8.2.4 Bayesian Linear Reg

In the bayesian linear reg setting, we have

$$\log p(y, w|x, \alpha, \beta) = \log p(y|w, x, \beta) + \log p(w|\alpha) \tag{8.2.30}$$

with $p(y|w,x,\beta) = N(y|w^T\phi(x), \beta^{-1})$ and $p(w|\alpha) = N(w|0, \alpha^{-1}I)$. Then the posterior of w is:

$$p(w|D, \alpha, \beta) = N(w|m, \Sigma) \tag{8.2.31}$$

$$m = \Sigma \Phi^T y \tag{8.2.32}$$

$$\Sigma = (\beta^{-1}\Phi^T\Phi + \alpha^{-1}I)^{-1} \tag{8.2.33}$$

We now consider using EM to optimize the hyperparams $\theta = (\alpha, \beta)$:

$$\max_\theta Q(\theta, q) = \max_\theta \mathbf{E}_{q(w)}[\log \frac{p(Y, w|X, \theta)}{q(w)}] \tag{8.2.34}$$

$$= \max_\theta \mathbf{E}_{q(w)}[\log p(Y|w, X, \theta) + \log p(w|\theta)] \tag{8.2.35}$$

$$= \max_\theta \mathbf{E}_{q(w)}[\sum_n \log N(y_n|w^T\phi(x_n), \beta^{-1})] + \log N(w|0, \alpha^{-1}I) \tag{8.2.36}$$

$$= \max_\theta \frac{N}{2}\log\frac{\beta}{2\pi} + \frac{M}{2}\log\frac{\alpha}{2\pi} - \mathbf{E}_{q(w)}[\frac{\beta}{2}\|Y - \Phi w\|^2 + \frac{\alpha}{2}w^T w] \tag{8.2.37}$$

$$= \max_\theta \frac{N}{2}\log\frac{\beta}{2\pi} + \frac{M}{2}\log\frac{\alpha}{2\pi} - \frac{\beta}{2}\{Y^T Y + m^T\Phi^T\Phi m - 2m^T\Phi^T y + \Sigma\} - \frac{\alpha}{2}(Tr(\Sigma) + \sum_i m_i^2) \tag{8.2.38}$$

thus, we're optimizing:

$$\max_\theta Q(\theta, q) = \max_\theta \frac{N}{2}\log\frac{\beta}{2\pi} + \frac{M}{2}\log\frac{\alpha}{2\pi} - \frac{\beta}{2}\{\|Y - \Phi m\|^2 + \Sigma\} - \frac{\alpha}{2}(Tr(\Sigma) + \sum_i m_i^2) \tag{8.2.39}$$

and finally we have:

$$m^{(t)} = \Sigma^{(t)}\Phi^T y \tag{8.2.40}$$

$$\Sigma^{(t)} = (\frac{1}{\beta^{(t)}}\Phi^T\Phi + \frac{1}{\alpha^{(t)}}I)^{-1} \tag{8.2.41}$$

$$q^{(t)}(w) = N(w|m^{(t)}, \Sigma^{(t)}) \tag{8.2.42}$$

$$\beta^{(t+1)} = \frac{N}{\|Y - \Phi m^{(t)}\|^2 + \Sigma^{(t)}} \tag{8.2.43}$$

$$\alpha^{(t+1)} = \frac{M}{Tr(\Sigma^{(t)}) + \sum_i (m_i^{(t)})^2} \tag{8.2.44}$$