Fall 2020 Last Update: January 21, 2021

Lecture Note: Advanced Machine Learning

Prof. Joachim M. Buhmann and Carlos Cotrini

Tao SUN
Dept. of Computer Science, ETH Zürich
taosun47@student.ethz.ch

Contents

Acknowledgement

This summary was made during the 2020 Fall Semester of the course *Advanced Machine Learning* by Prof. Buhmann and Cotrini at ETH Zürich.

The main purpose of writing this is to familiarize me with the concepts and mathematical derivations in the course. Therefore, I do not guarantee the correctness and completeness of it.

This note is mainly based on the lecture slides, tutorial materials and exercises. Some of the contents are referenced from the related books or papers. Also, some of the notes are cited from previous students, inclining @michaelaerni[1]. The reference sources are stated in the footnotes. Many thanks to them!

[1] Introduction to ML Lecture Notes, https://github.com/michaelaerni/eth-introml-lecturenotes

Chapter 0

Math Preliminaries

0.1 Matrix Algebra

0.1.1 Derivatives

For $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$, $f'(\mathbf{x}) \in \mathbb{R}^n$	
$\frac{\partial}{\partial x}(\mathbf{b}^\top \mathbf{x}) = \frac{\partial}{\partial x}(\mathbf{x}^\top \mathbf{b}) = \mathbf{b}$	$\frac{\partial}{\partial x}(\mathbf{x}^\top \mathbf{x}) = 2\mathbf{x}$
$\frac{\partial}{\partial x}(\|\mathbf{Ax} - \mathbf{b}\|_2) = \frac{A^\top(Ax-b)}{\|Ax-b\|_2}$	$\frac{\partial}{\partial x}(\|\mathbf{Ax} - \mathbf{b}\|_2^2) = 2\mathbf{A}^\top(\mathbf{Ax} - \mathbf{b})$

For $f(\mathbf{X}) : \mathbb{R}^{n\times m} \to \mathbb{R}$, $f'(\mathbf{X}) \in \mathbb{R}^{n\times m}$	
$\frac{\partial}{\partial X}(\mathbf{a}^\top \mathbf{X} \mathbf{b}) = \mathbf{a}\mathbf{b}^\top$	$\frac{\partial}{\partial X}(\mathbf{a}^\top \mathbf{X}^\top \mathbf{b}) = \mathbf{b}\mathbf{a}^\top$
$\frac{\partial}{\partial X}\mathrm{Tr}(\mathbf{A}^\top \mathbf{X} \mathbf{B}) = \mathbf{A}\mathbf{B}^\top$	$\frac{\partial}{\partial X}\mathrm{Tr}(\mathbf{A}^\top \mathbf{X}^\top \mathbf{B}) = \mathbf{B}\mathbf{A}^\top$
$\frac{\partial}{\partial X}\lvert\mathbf{X}\rvert = \lvert\mathbf{X}\rvert(\mathbf{X}^{-1})^\top$	

0.1.2 Inverses

Moore-Penrose Inverse

Moore-Penrose pseudo-inverse of $\mathbf{A} \in \mathbb{R}^{n\times m}$ (assuming $\mathbf{A}$ has a SVD as $\mathbf{A} = \mathbf{VDU}^\top$):

$$\mathbf{A}^+ = \left\{(\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top \text{ or } \mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top)^{-1}\right\} = \mathbf{V}\mathbf{D}^+\mathbf{U}^\top \in \mathbb{R}^{m\times n} \tag{1}$$

where $\mathbf{D} = \mathrm{diag}(\sigma_1, \cdots, \sigma_r)$, $\mathbf{D}^+ = \mathrm{diag}(\sigma_1^{-1}, \cdots, \sigma_r^{-1}) \in \mathbb{R}^{r\times r}$ and $r = \mathrm{rank}(\mathbf{A})$.

Identities

Sherman-Morrison Lemma:

$$\left(\mathbf{A} + \mathbf{b}\mathbf{c}^\top\right)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{b}\mathbf{c}^\top\mathbf{A}^{-1}}{1 + \mathbf{c}^\top\mathbf{A}^{-1}\mathbf{b}} \tag{2}$$

Woodbury Identity (and its variations):

$$(\mathbf{A} + \mathbf{BC})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}\left(\mathbf{I} + \mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)^{-1}\mathbf{C}\mathbf{A}^{-1} \tag{3}$$

$$(\mathbf{A} + \mathbf{UCV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}\left(\mathbf{C}^{-1} + \mathbf{V}\mathbf{A}^{-1}\mathbf{U}\right)^{-1}\mathbf{V}\mathbf{A}^{-1} \tag{4}$$

0.1.3 Positive-definite Matrices

For a symmetric matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$,

$$\mathbf{A} \text{ is PD} \iff \mathbf{x}^\top\mathbf{A}\mathbf{x} > 0\ (\mathbf{x} \neq 0) \iff \mathrm{eig}(\mathbf{A}) > 0 \iff \mathbf{A} = \mathbf{B}^\top\mathbf{B} \iff \text{Sylvester's.} \tag{5}$$

Here, *Sylvester's criterion* is to say that all leading principal minors of $\mathbf{A}$ must be positive.

0.2 Statistics and Probability

Property 0.2.1 (Probability Three Axioms)

1. *Normalization:* $p(\Omega) = 1$;
2. *Non-negativity:* $p(A) \geq 0$ for all $A \in \mathcal{F}$;
3. *σ-Additivity:* $\forall A_1, \ldots A_n, \ldots \in \mathcal{F}$ disjoint : $p\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} p\left(A_i\right)$

Definition 0.2.1 (Conditional Probability)

$$p(a \mid b) = \frac{p(a \wedge b)}{p(b)}, \text{ if } p(b) \neq 0 \tag{6}$$

Property 0.2.2 (Joint Distributions)

1. *Sum Rule (a.k.a Marginalization)*

$$p\left(X_i\right) = \sum_{x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n} p\left(x_1, \ldots, x_{i-1}, X_i, x_{i+1}, \ldots, x_n\right) \tag{7}$$

2. *Chain Rule*

$$p\left(X_1, \ldots, X_n\right) = p\left(X_1\right) p\left(X_2 \mid X_1\right) \ldots p\left(X_n \mid X_1, \ldots, X_{n-1}\right) \tag{8}$$

Definition 0.2.2 (Bayes' Rule)

$$p(X \mid Y) = \frac{p(X)\ p(Y \mid X)}{p(Y)}, \quad \text{``posterior''} = \frac{\text{``prior''} \times \text{``likelihood''}}{\text{``evidence''}} \tag{9}$$

0.2.1 Expectation & Variance

For random variables:

$$\mathbb{E}[\alpha X + c] = \alpha\, \mathbb{E}[X] + c \tag{10}$$

$$\mathrm{Var}[\alpha X] = \alpha^2 \mathrm{Var}[X] \tag{11}$$

$$\mathrm{Cov}[\alpha X, Y] = \alpha^2 \mathrm{Cov}[X, Y] \tag{12}$$

$$\mathrm{Cov}[X_1 + X_2, Y] = \mathrm{Cov}[X_1, Y] + \mathrm{Cov}[X_2, Y] \tag{13}$$

Linear forms:

$$\mathbb{E}[\mathbf{AXB} + \mathbf{C}] = \mathbf{A}\, \mathbb{E}[\mathbf{X}]\, \mathbf{B} + \mathbf{C} \tag{14}$$

$$\mathrm{Var}[\mathbf{Ax}] = \mathbf{A}\mathrm{Var}[\mathbf{x}]\mathbf{A}^\top \tag{15}$$

$$\mathrm{Cov}[\mathbf{Ax}, \mathbf{By}] = \mathbf{A}\mathrm{Cov}[\mathbf{x}, \mathbf{y}]\mathbf{B}^\top \tag{16}$$

Quadratic forms: let $\boldsymbol{\mu} = \mathbb{E}[\mathbf{x}], \Sigma = \mathrm{Var}[\mathbf{x}]$.

$$\mathbb{E}[\mathbf{x}^\top \mathbf{x}] = \mathrm{Tr}(\Sigma) + \boldsymbol{\mu}^\top \boldsymbol{\mu} \qquad \mathbb{E}[\mathbf{x}\mathbf{x}^\top] = \Sigma + \boldsymbol{\mu}\boldsymbol{\mu}^\top \tag{17}$$

$$\mathbb{E}[\mathbf{x}^\top \mathbf{A}\mathbf{x}] = \mathrm{Tr}(\mathbf{A}\Sigma) + \boldsymbol{\mu}^\top \mathbf{A}\boldsymbol{\mu} \tag{18}$$

0.2.2 Gaussian Distribution

The density of $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ is

$$p(\mathbf{x}) = \frac{1}{\sqrt{\det(2\pi\Sigma)}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]. \tag{19}$$

Linear combination of two Gaussians $\mathbf{x}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \Sigma_1), \mathbf{x}_2 \sim \mathcal{N}(\boldsymbol{\mu}_2, \Sigma_2)$:

$$\mathbf{A}\mathbf{x}_1 + \mathbf{B}\mathbf{x}_2 + c \sim \mathcal{N}\left(\mathbf{A}\boldsymbol{\mu}_1 + \mathbf{B}\boldsymbol{\mu}_2, \mathbf{A}\Sigma_1\mathbf{A}^\top + \mathbf{B}\Sigma_2\mathbf{B}^\top\right). \tag{20}$$

The products of two Gaussian densities

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_1, \Sigma_1) \cdot \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_2, \Sigma_2) \propto \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}', \Sigma'), \quad \text{where} \tag{21}$$

$$\Sigma' = (\Sigma_1^{-1} + \Sigma_2^{-1})^{-1} \tag{22}$$

$$\boldsymbol{\mu}' = \Sigma'(\Sigma_1^{-1}\boldsymbol{\mu}_1 + \Sigma_2^{-1}\boldsymbol{\mu}_2) \tag{23}$$

Rearranging quadratic form into squared form: (assume $\mathbf{A}$ is symmetric)

$$-\frac{1}{2}\mathbf{x}^\top\mathbf{A}\mathbf{x} + \mathbf{b}^\top\mathbf{x} = -\frac{1}{2}(\mathbf{x} - \mathbf{A}^{-1}\mathbf{b})^\top\mathbf{A}(\mathbf{x} - \mathbf{A}^{-1}\mathbf{b}) + \frac{1}{2}\mathbf{b}^\top\mathbf{A}^{-1}\mathbf{b}. \tag{24}$$

Chapter 1

Basic Statistical Learning

1.1 Properties of Estimators

For an estimator $\hat{\theta}_n$ over available sample of size n, we have the following properties.

- **Unbiased**: $\mathbb{E}[\hat{\theta} - \theta] = 0$.
- **Consistent**: $\lim\limits_{n\to\infty} p(|\hat{\theta}_n - \theta| > \varepsilon) = 0$, or $\hat{\theta}_n \xrightarrow{p} \theta$.
- **Efficient**: $\hat{\theta}$ achieves equality in CRLB (Sec. 1.3). In other word, $\hat{\theta}$ minimizes $\mathbb{E}\left[(\hat{\theta} - \theta)^2\right]$.
- **Asymptotically Efficient**: $\hat{\theta}_n$ is efficient as $n \to \infty$.
- **Minimum-variance Unbiased Estimator (MVUE)**: An unbiased estimator whose variance is lower than any other unbiased estimator for all possible values of parameter $\hat{\theta}$, i.e., $\text{Var}[\hat{\theta}_{\text{MVUE}}] \leq \text{Var}[\hat{\theta}]$, $\forall \hat{\theta}$.

Towards better understanding of the properties, please note that:

- $\underbrace{\mathbb{E}\left[\left(\hat{\theta} - \theta\right)^2\right]}_{\text{MSE}(\hat{\theta},\theta)} = \underbrace{\mathbb{E}\left[\left(\hat{\theta} - \mathbb{E}[\theta]\right)^2\right]}_{\text{Var}[\hat{\theta}]} + \underbrace{\mathbb{E}\left[\hat{\theta} - \theta\right]^2}_{\text{bias}^2} = \sigma^2 + b^2$.
- $\lim_{n\to\infty} \text{MSE}\left(\hat{\theta}_n, \theta\right) = 0 \Longrightarrow$ Consistency (can be proved by *Chebyshev's* inequality).
- Unbiased & efficient estimator $\Longrightarrow$ minimum-variance, or MVUE.
- Unbiased & minimum-variance $\not\Longrightarrow$ efficient estimator (CRLB is not easily achieved).

Property 1.1.1 (Bias after Averaging) *Assume we have a set of estimators $f_1, \cdots, f_B$. The averaging estimator $\bar{f} = \frac{1}{B}\sum_{i=1}^{B} f_i$ has the bias as*

$$\text{bias}[\bar{f}(\mathbf{x})] = \mathbb{E}_{\text{x},y\sim\mathcal{D}}\left[\bar{f}(\mathbf{x}) - y\right] = \frac{1}{B}\sum_{i=1}^{B}\left(\mathbb{E}_{\text{x},y\sim\mathcal{D}}\left[f_i(\mathbf{x}) - y\right]\right) = \frac{1}{B}\sum_{i=1}^{B}\text{bias}[f_i]. \tag{1.1}$$

Remark. An unbiased estimator remains unbiased after averaging.

Property 1.1.2 (variance after Averaging) *Assume we have a set of estimators $f_1, \cdots, f_B$. The averaging estimator $\bar{f} = \frac{1}{B}\sum_{i=1}^{B} f_i$ has the variance as*

$$\begin{aligned}
\text{Var}[\bar{f}(\mathbf{x})] &= \mathbb{E}_{\text{x},y\sim\mathcal{D}}\left[\left(\bar{f}(\mathbf{x}) - \mathbb{E}_{\text{x}\sim\mathcal{D}}\left[\bar{f}(\mathbf{x})\right]\right)^2\right] && (1.2)\\
&= \mathbb{E}_{\text{x},y\sim\mathcal{D}}\left[\left(\frac{1}{B}\sum_{i=1}^{B}\left(f_i(\mathbf{x}) - \mathbb{E}_{\text{x}\sim\mathcal{D}}\left[f_i(\mathbf{x})\right]\right)\right)^2\right] && (1.3)\\
&= \frac{1}{B^2}\sum_{i=1}^{B}\text{Var}[f_i(\mathbf{x})] + \frac{1}{B^2}\sum_{i=1}^{B}\sum_{j=1,j\neq i}^{B}\text{Cov}[f_i(\mathbf{x}), f_j(\mathbf{x})] && (1.4)
\end{aligned}$$

Remark. If the covariances are neglectable, we can approximate it as $\text{Var}[\bar{f}(\mathbf{x})] \approx \sigma^2/B$, which means the variance will reduced by a factor of $1/B$.

1.2 Bayesianism and Frequentism

	Bayesian method	**Frequentist method**
Function	provides a **predictive distribution**	provides a **single-point** estimator
Method	Bayesian inference	Maximum Likelihood Estimator (MLE)
Assumption	a prior distribution $p(\theta)$	a parametric model θ
Likelihood	conditional distribution $p(y \mid \theta)$	likelihood function $L(\theta) = \prod_i p(y_i \mid \theta)$
Estimator	posterior distribution: $p(\theta \mid y) = \dfrac{p(\theta)\, p(y \mid \theta)}{p(y)}$	maximum-likelihood estimator: $\hat{\theta}_{MLE} = \text{argmax}_\theta\, L(\theta)$
Remarks	priors induce a regularization term	consistency: $\hat{\theta}_n \xrightarrow{p} \theta$ asym. normal: $\sqrt{n}(\theta - \hat{\theta}_n) \longrightarrow \mathcal{N}(0, \mathcal{I}(\theta)^{-1})$ asym. efficient: achieves CRLB when $n \to \infty$

Some useful facts of Maximum Likelihood Estimator (MLE):

- An unbiased MLE has the minimum variance as $n \to \infty$ compared to all other **unbiased** estimators.
- Biased estimators may have lower variance than MLE as $n \to \infty$, e.g. the *James-Stein* estimator.
- MLE is **not** always unbiased, e.g. the MLE for θ in Uniform$(0, \theta)$ is $\frac{1}{2}\max\{X_i\} < \frac{1}{2}\theta$.
- MLE of *i.i.d.* observations is always consistent and asymptotically normal.
- If $\hat{\theta}$ is the MLE of θ, for any measurable function $g(\cdot)$, $g(\hat{\theta})$ is also the MLE of $g(\theta)$.

Most nice properties of MLE require $n \to \infty$. For a small n, MLE may not be a good estimator.

1.3 Cramér-Rao Lower Bound

Definition 1.3.1 (Fisher Information) *Given the likelihood $p(x \mid \theta)$ for $\theta \in \Theta$, the Fisher Information $\mathcal{I}(\theta)$ for θ is*

$$\mathcal{I}(\theta) = -\mathbb{E}_{x|\theta}\left[\frac{\partial^2 \log p(x \mid \theta)}{\partial \theta^2}\right] = \mathbb{E}_{x|\theta}\left[\left(\frac{\partial \log p(x \mid \theta)}{\partial \theta}\right)^2\right]$$

Proof. We need to prove that the values of the expectation are the same. For similarity, we introduce the Score Function $\Lambda(\theta)$, which is the deviation of the log likelihood,

$$\Lambda(x; \theta) = \frac{\partial}{\partial \theta}\log p(x \mid \theta) = \frac{1}{p(x \mid \theta)}\frac{\partial}{\partial \theta}p(x \mid \theta) \tag{1.5}$$

Obviously, the expectation of $\Lambda(x; \theta)$ is

$$\begin{aligned}\mathbb{E}_{x|\theta}[\Lambda(x; \theta)] &= \int p(x \mid \theta)\frac{1}{p(x \mid \theta)}\frac{\partial}{\partial \theta}p(x \mid \theta)\ \mathrm{d}x & (1.6)\\ &= \frac{\partial}{\partial \theta}\int p(x \mid \theta)\ \mathrm{d}x & (1.7)\\ &= 0 & (1.8)\end{aligned}$$

According to Eq. 1.6,

$$\mathbb{E}_{x|\theta}[\Lambda] = \int p(x \mid \theta)\frac{\partial}{\partial \theta}\log p(x \mid \theta) = 0 \tag{1.9}$$

Differentiate w.r.t. θ and take the derivative inside gives [1] ,

$$\begin{aligned}
0 &= \int \frac{\partial^2 \log p(x \mid \theta)}{\partial \theta^2} p(x \mid \theta) \text{ d}x + \int \frac{\partial \log p(x \mid \theta)}{\partial \theta} \frac{\partial p(x \mid \theta)}{\partial \theta} \text{ d}x & (1.10) \\
&= \int \frac{\partial^2 \log p(x \mid \theta)}{\partial \theta^2} p(x \mid \theta) \text{ d}x + \int \frac{\partial \log p(x \mid \theta)}{\partial \theta} \frac{1}{p(x \mid \theta)} \frac{\partial p(x \mid \theta)}{\partial \theta} p(x \mid \theta) \text{ d}x & (1.11) \\
&= \int \frac{\partial^2 \log p(x \mid \theta)}{\partial \theta^2} p(x \mid \theta) \text{ d}x + \int \left(\frac{\partial \log p(x \mid \theta)}{\partial \theta} \right)^2 p(x \mid \theta) \text{ d}x & (1.12) \\
&= \mathbb{E}_{x|\theta} \left[\frac{\partial^2 \log p(x \mid \theta)}{\partial \theta^2} \right] + \mathbb{E}_{x|\theta} \left[\left(\frac{\partial \log p(x \mid \theta)}{\partial \theta} \right)^2 \right] & (1.13)
\end{aligned}$$

■

The Fisher Information can be also written as the variance.

$$\mathcal{I}(\theta) = \mathbb{E}_{x|\theta} \left[\left(\frac{\partial \log p(x \mid \theta)}{\partial \theta} \right)^2 \right] - \underbrace{\mathbb{E}_{x|\theta} \left[\frac{\partial \log p(x \mid \theta)}{\partial \theta} \right]^2}_{\mathbb{E}[\Lambda]^2 = 0} = \mathbb{V}_{x|\theta} \left[\frac{\partial}{\partial \theta} \log p(x \mid \theta) \right] \quad (1.14)$$

If there are n independent observations, then

$$\mathcal{I}_n(\theta) = n \mathcal{I}(\theta) \quad (1.15)$$

Theorem 1.3.1 (Cramér-Rao Lower Bound, CRLB) *Given the likelihood $p(x \mid \theta)$ for $\theta \in \Theta$, the Fisher Information $\mathcal{I}(\theta)$ for θ, then the expected deviation of a estimator $\hat{\theta}$ with bias $b = \mathbb{E}_{x|\theta}[\hat{\theta} - \theta]$ to the true estimator θ satisfies*

$$\mathbb{E}_{x|\theta}[(\hat{\theta} - \theta)^2] \geq \left(\frac{\partial b}{\partial \theta} + 1 \right)^2 \mathcal{I}^{-1}(\theta) + b^2, \quad \textit{where } \mathcal{I}(\theta) = -\mathbb{E}_{x|\theta} \left[\frac{\partial^2}{\partial \theta^2} \log p(x \mid \theta) \right]$$

Proof. We follow the same notation in the Proof. 1.3.1. The expectation of $\Lambda \hat{\theta}$ is

$$\begin{aligned}
\mathbb{E}_{x|\theta}[\Lambda(x; \theta) \ \hat{\theta}(x)] &= \int p(x \mid \theta) \frac{1}{p(x \mid \theta)} \frac{\partial}{\partial \theta} p(x \mid \theta) \, \hat{\theta}(x) \text{ d}x & (1.16) \\
&= \int \hat{\theta}(x) \frac{\partial}{\partial \theta} p(x \mid \theta) \text{ d}x & (1.17) \\
&= \frac{\partial}{\partial \theta} \int \hat{\theta}(x) p(x \mid \theta) \text{ d}x & (1.18) \\
&= \frac{\partial}{\partial \theta} \mathbb{E}_{x|\theta}[\hat{\theta}(x)] & (1.19) \\
&= \frac{\partial}{\partial \theta} \mathbb{E}_{x|\theta}[\hat{\theta}(x) - \theta] + 1 & (1.20)
\end{aligned}$$

Recalling that the *Cauchy-Schwartz* inequality that $\mathbb{E}[XY]^2 \leq \mathbb{E}[X^2] \ \mathbb{E}[Y^2]$, we have

$$\begin{aligned}
\mathbb{E}_{x|\theta}[\Lambda(x; \theta) \ \hat{\theta}(x)]^2 &= \mathbb{E}_{x|\theta} \left[(\Lambda - \mathbb{E}_{x|\theta}[\Lambda]) \, (\hat{\theta} - \mathbb{E}_{x|\theta}[\hat{\theta}]) \right]^2 & (1.21) \\
&\leq \mathbb{E}_{x|\theta} \left[(\Lambda - \mathbb{E}_{x|\theta}[\Lambda])^2 \right] \ \mathbb{E}_{x|\theta} \left[(\hat{\theta} - \mathbb{E}_{x|\theta}[\hat{\theta}])^2 \right] & (1.22) \\
&= \mathbb{E}_{x|\theta}[\Lambda^2] \, \mathbb{E}_{x|\theta} \left[(\hat{\theta} - \mathbb{E}_{x|\theta}[\hat{\theta}])^2 \right] & (1.23)
\end{aligned}$$

[1] Refer to the slides: https://www.stat.tamu.edu/~suhasini/teaching613/inference.pdf

Therefore,

$$\mathbb{E}_{x|\theta}\left[(\hat{\theta}-\mathbb{E}_{x|\theta}[\hat{\theta}])^2\right] \geq \frac{\mathbb{E}_{x|\theta}[\Lambda\,\hat{\theta}]^2}{\mathbb{E}_{x|\theta}[\Lambda^2]} \tag{1.24}$$

$$\mathbb{E}_{x|\theta}[\hat{\theta}^2]-\mathbb{E}_{x|\theta}[\hat{\theta}]^2 \geq \frac{\left(\frac{\partial}{\partial\theta}\mathbb{E}_{x|\theta}[\hat{\theta}(x)-\theta]+1\right)^2}{\mathcal{I}(\theta)} \tag{1.25}$$

$$\mathbb{E}_{x|\theta}[(\hat{\theta}-\theta)^2]-\mathbb{E}[\hat{\theta}-\theta]^2 \geq \frac{\left(\frac{\partial}{\partial\theta}\mathbb{E}_{x|\theta}[\hat{\theta}(x)-\theta]+1\right)^2}{\mathcal{I}(\theta)} \tag{1.26}$$

$$\mathbb{E}_{x|\theta}[(\hat{\theta}-\theta)^2] \geq \frac{\left(\frac{\partial b}{\partial\theta}+1\right)^2}{\mathcal{I}(\theta)}+b^2 \tag{1.27}$$

■

The condition that a estimator $\hat{\theta}$ attains the CRLB is

$$\hat{\theta}=\alpha+\beta\cdot\frac{\partial\log p(x\mid\theta)}{\partial\theta}, \quad \alpha,\beta\in\mathbb{R} \tag{1.28}$$

The maximum likelihood estimator $\hat{\theta}_{MLE}$ is asymptotically efficient, i.e.,

$$\lim_{n\to\infty}\mathbb{E}_{x|\theta}[(\hat{\theta}_{MLE}-\theta)^2]=\frac{1}{n\mathcal{I}(\theta)} \tag{1.29}$$

Chapter 2

Linear Models

$\mathbf{X} \in \mathbb{R}^{n\times d}$	data matrix
$\mathbf{y} \in \mathbb{R}^{n}$	label vector
$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$	a data set with samples and labels
$\beta \in \mathbb{R}^{d}$	weights
$\lambda \in \mathbb{R}_{+}$	regularization scaler

2.1 Linear Regression

Definition 2.1.1 (Linear Regression or OLS) *Let $y = f(\mathbf{x}) = \beta^\top \mathbf{x}$ be the target function. The optimization objective of linear regression is*

$$\min_{\beta} \quad \sum_{i=1}^{n} (y_i - \beta^\top \mathbf{x}_i)^2 \tag{2.1}$$

For the bias term, using this trick: $x' = [x_1, x_2, \cdots, x_n, 1]$

Here, the objective is to obtain the minimal squared error, so it is also called "Ordinary Least Squared" (OLS) method.

Instead of the squared error, other loss functions might also be used here. For example, with the the L_p distance, we have

$$\min_{\beta} \quad \sum_{i=1}^{n} \left\| y_i - \beta^\top \mathbf{x}_i \right\|_p \tag{2.2}$$

where $\|\mathbf{x}\|_p = \sqrt[p]{\sum_{i=1}^{d} |x_i|^p}$. L_p loss is convex when $p \geq 1$. The L_1 loss puts less emphasis on large outliers. For a large p, the results are restricted to a region. For $0 \leq p < 1$, the function becomes non-convex but is even more robust towards outliers.

Derivative of L_p: $\partial \|\vec{w}\|_p^p / \partial w_j = p|w_j|^{p-1} \operatorname{sign}(w_j)$

2.1.1 Ridge Regression

Definition 2.1.2 (Ridge Regression) *To condition the model's complexity, it is common to use the method called "Ridge regression", which introduces a L_2 constraint on the weight.*

$$\min_{\beta} \quad \left\{ \sum_{i=1}^{n} \left(y_i - \beta^\top \mathbf{x}_i \right)^2 \right\}, \quad s.t. \ \sum_{j=1}^{d} \beta_j^2 \leq t. \tag{2.3}$$

Using Lagrange multipliers, it can be re-written into,

$$\widehat{\beta} = \operatorname*{argmin}_{\beta} \left\{ \sum_{i=1}^{n} \left(y_i - \beta^\top \mathbf{x}_i \right)^2 + \lambda \sum_{j=1}^{d} \beta_j^2 \right\} \tag{2.4}$$

Should be $\lambda(\sum_j \beta_j^2 - t)$, but $-\lambda t$ can be omitted since it is a constant.

Note that, here λ and t has a one-to-one relationship. L_2 regression on weights is also called the "weight decay", as it decays weights to zero.

Property 2.1.1 *Ridge regression has a* **closed-form** *solution of*

$$\widehat{\beta} = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y}. \tag{2.5}$$

Proof. Let

$$\widehat{\beta} = \underset{\beta}{\operatorname{argmin}}\ J(\beta; \mathbf{X}, \mathbf{y}) := \underset{\beta}{\operatorname{argmin}}\ (\mathbf{y} - \beta^\top\mathbf{X})^\top(\mathbf{y} - \beta^\top\mathbf{X}) + \lambda\beta^\top\beta \tag{2.6}$$

The derivative of $J(\beta; \mathbf{X}, \mathbf{y})$ is

$$\widehat{\beta} = \underset{\beta}{\operatorname{argmin}}\ (\mathbf{y} - \beta^\top\mathbf{X})^\top(\mathbf{y} - \beta^\top\mathbf{X}) + \lambda\beta^\top\beta \tag{2.7}$$

By differentiating the above equation and equating it to zero, we can get the optimal point.

$$\begin{aligned} 0 &= \frac{\partial J(\beta)}{\partial\beta} \\ &= -2\mathbf{X}^\top\mathbf{Y} + 2\mathbf{X}^\top\mathbf{X}\beta + 2\lambda\beta \end{aligned} \tag{2.8}$$

Therefore,

$$(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})\beta = \mathbf{X}^\top\mathbf{Y} \tag{2.9}$$

Multiplying $(\mathbf{x}^\top\mathbf{x} + \lambda\mathbf{I})^{-1}$ on both sides, we get

$$\hat{\beta} = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{Y} \tag{2.10}$$

■

Alternatively, the gradient descent can also be used to find the global optimum of ridge regression, since its objective is convex. The closed-form solution takes $\mathcal{O}(n^3)$ time. The gradient descent solution takes $\mathcal{O}(nd\log(1/\epsilon))$ time. When n becomes larger, the gradient descent method will be more efficient than closed-form solution.

2.1.2 Lasso Regression

Lasso regression is the linear regression whose weights regularized by L_1. The objective then becomes,

$$\min_{\beta}\ \sum_{i=1}^{n}(y_i - w^T\mathbf{x}_i)^2 + \lambda\left\|\beta\right\|_1$$

The L_1 penalty, in theory, encourages coefficients to be exactly zero. In ridge regression, weights decay relatively smoothly; in lasso regression, some weights may heavily increase with increasing λ. This results in some form in automatic feature selection. However, there is **no** closed-form solution for lasso regression.

2.2 Bias and Variance Trade-off

2.2.1 From Optimization Perspective

Expectation of Bias Ridge regression produces a **biased estimator** of the true parameter β^*. The expectation of $\widehat{\beta}$ is,

$$\begin{aligned} \mathbb{E}\left[\widehat{\beta} \mid \mathbf{X}\right] &= (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbb{E}[\mathbf{y} \mid \mathbf{X}] &(2.11)\\ &= (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{X}\beta^* \quad \text{(recall that } \mathbf{y} = \mathbf{X}\beta^*) &(2.12)\\ &= (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I} - \lambda\mathbf{I})\beta^* &(2.13)\\ &= [\mathbf{I} - \lambda(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}]\beta^* &(2.14) \end{aligned}$$

So, the expectation of bias is,

$$\mathbb{E}\left[\beta^* - \widehat{\beta} \mid \mathbf{X}\right] = \lambda(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\beta^*. \tag{2.15}$$

Bias under Noisy Label Let $\mathbf{y} = \mathbf{X}\beta^* + \xi$, $\xi \sim N(0, \sigma^2 \mathbf{I})$. Conduct SVD decomposing on $\mathbf{X} \in \mathbb{R}^{n \times d}$,

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^\top. \tag{2.16}$$

When $\lambda = 0$, the solution of $\widehat{\beta}$ is

$$\begin{aligned} \widehat{\beta} &= (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} && (2.17) \\ &= (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top (\mathbf{X}\beta^* + \xi) && (2.18) \\ &= \beta^* + (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \xi. && (2.19) \end{aligned}$$

Insert Eq. 2.16, we have

$$\begin{aligned} \widehat{\beta} - \beta^* &= (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \xi && (2.20) \\ &= \mathbf{V}\mathbf{D}^+ \mathbf{U}^\top \xi. && (2.21) \end{aligned}$$

Here, $\mathbf{D}^+$ is the Moore-Penrose pseudo inverse matrix of $\mathbf{D}$.

$(\mathbf{X}^\top\mathbf{X})^{-1} =$ $(\mathbf{V}\mathbf{D}^\top\mathbf{U}^\top\mathbf{U}\mathbf{D}\mathbf{V}^\top)^{-1}$ $= (\mathbf{V}\mathbf{D}^\top\mathbf{D}\mathbf{V}^\top)^{-1}$ $= (\mathbf{V}^\top)^{-1}(\mathbf{D}^\top\mathbf{D})^{-1}\mathbf{V}^{-1}$ $= \mathbf{V}(\mathbf{D}^\top\mathbf{D})^{-1}\mathbf{V}^{-1}$. Thus, $(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top =$ $\mathbf{V}(\mathbf{D}^\top\mathbf{D})^{-1}\mathbf{V}^{-1}\mathbf{V}\mathbf{D}^\top\mathbf{U}^\top$ $= \mathbf{V}(\mathbf{D}^\top\mathbf{D})^{-1}\mathbf{D}^\top\mathbf{U}^\top$ $= \mathbf{V}\mathbf{D}^+\mathbf{U}^\top$.

2.2.2 From Bayesian Perspective

Reference: Probabilistic AI, Lecture 2

Equivalence to Bayes Assume the prior distribution of β is $p(\beta) = \mathcal{N}(0, \sigma_p^2 \cdot \mathbf{I})$, and it is independent of $\mathbf{x}_i$. Also, we assume the labels y_i have some independent Gaussian noise, i.e. $y_i = \beta^\top \mathbf{x}_i + \xi_i$, $\xi \sim \mathcal{N}(0, \sigma_n^2)$. Then, the Ridge regression can be viewed as a **Bayesian inference**, where

$$\lambda = \sigma_n^2 / \sigma_p^2 \tag{2.22}$$

Proof. According to the noise assumption, the likelihood of $y_{1:n}$ is

$$p(y_{1:n} \mid \beta, \mathbf{x}_{i:n}) = \prod_{i=1}^{n} p(y_i \mid w_1, x_i) = \mathcal{N}(y_i; \beta^\top \mathbf{x}, \sigma_n^2) \tag{2.23}$$

First, using Bayes' rule, we obtain the posterior of β,

$$\begin{aligned} p(\beta \mid \mathbf{x}_{1:n}, y_{1:n}) &= \frac{1}{Z} p(\beta \mid \mathbf{x}_{1:n}) p(y_{1:n} \mid \beta, \mathbf{x}_{1:n}) \\ &= \frac{1}{Z} p(\beta) \prod_i p(y_i \mid \beta, \mathbf{x}_i) \\ &= \frac{1}{Z} \frac{1}{Z_p} \exp\left(-\frac{1}{2\sigma_p^2} ||\beta||^2\right) \cdot \frac{1}{Z_l} \prod_i \exp\left(-\frac{1}{\sigma_n^2} (y_i - \beta^\top \mathbf{x}_i)^2\right) \\ &= \frac{1}{Z Z_p Z_l} \exp\left(-\frac{1}{2\sigma_p^2} ||\beta||^2 - \frac{1}{2\sigma_n^2} \sum_{i=1}^{n} (y_i - \beta^\top \mathbf{x}_i)^2\right) \end{aligned} \tag{2.24}$$

Here, Z, Z_p, Z_l are the normalizers: $Z = p(y_{1:n} \mid \mathbf{x}_{1:n})$, $Z_p = \sqrt{2\pi}\sigma_p$, $Z_l = (2\pi)^{(n/2)} \sigma_n$.

Next, we maximize the posterior of β,

$$\operatorname*{argmax}_{\beta} \; p(\beta \mid \mathbf{x}_{1:n}, y_{1:n}) = \operatorname*{argmax}_{\beta} \; \underbrace{\sum_{i=1}^{n} (y_i - \beta^\top \mathbf{x}_i)^2}_{\text{OLS}} + \underbrace{\frac{\sigma_n^2}{\sigma_p^2} \|\beta\|^2}_{\lambda \cdot L_2 \text{Reg.}} \tag{2.25}$$

Note that the posterior has the same form of the Ridge Regression when λ has the value of σ_n^2 / σ_p^2. Hence, the Ridge Regression (Eq. **??**) can be viewed as a Bayesian Inference process theoretically. ■

Uncertainty From a Bayesian perspective, we can not only know the optimal solution, we can also obtain the uncertainty for such solution. The uncertainty is denoted by variance of the posterior $p(\beta \mid \mathbf{x}_{1:n}, y_{1:n}) = \mathcal{N}(\bar{\mu}, \bar{\Sigma})$. The mean and variance of the posterior are,

$$\begin{aligned} \bar{\mu} &= \left(\mathbf{x}^\top \mathbf{x} + \sigma_n^2 \mathbf{I}\right)^{-1} \mathbf{x}^\top \mathbf{y} \\ \bar{\Sigma} &= \left(\sigma_n^{-2} \mathbf{x}^\top \mathbf{x} + \mathbf{I}\right)^{-1} \end{aligned} \tag{2.26}$$

Proof. We can get the close-form of the posterior.[1].

$\mathbf{u}^\top \mathbf{A} \mathbf{u} - 2\alpha^\top \mathbf{u} = (\mathbf{u} - \mathbf{A}^{-1}\alpha)^\top \mathbf{A}(\mathbf{u} - \mathbf{A}^{-1}\alpha)$

$$\begin{aligned} p(\beta \mid \mathbf{x}_{1:n}, y_{1:n}) &= \frac{1}{Z Z_p Z_l} \exp\left(-\frac{1}{2\sigma_p^2}||\beta||^2 - \frac{1}{2\sigma_n^2}\sum_i ||y_i - \beta^\top \mathbf{x}_i||^2\right) \\ &= \frac{1}{Z Z_p Z_l} \exp\left(-\frac{1}{2\sigma_p^2}\beta^\top \beta - \frac{1}{2\sigma_n^2}(\mathbf{y}^\top \mathbf{y} \ - 2\mathbf{y}\beta^\top \mathbf{x} + \beta^\top \mathbf{x}^\top \mathbf{x}\beta)\right) \\ &\propto \exp\left(-\frac{1}{2}\left(\beta^\top(\frac{1}{\sigma_n^2}\mathbf{x}^\top \mathbf{x} + \frac{1}{\sigma_p^2}\mathbf{I})\beta - 2(\mathbf{x}^\top \mathbf{y})^\top \beta\right)\right) \\ &= \exp\left(-\frac{1}{2}(\beta - \mu')^\top \Sigma'^{-1}(\beta - \mu')\right) \end{aligned} \tag{2.27}$$

Here,

$$\begin{aligned} \mu' &= (\sigma_n^{-2}\mathbf{x}^\top \mathbf{x} + \sigma_p^{-2}\mathbf{I})^{-1}\mathbf{x}^\top \mathbf{y} \\ \Sigma' &= (\sigma_n^{-2}\mathbf{x}^\top \mathbf{x} + \sigma_p^{-2}\mathbf{I})^{-1} \end{aligned} \tag{2.28}$$

After normalization, we can get the answer

$$\begin{aligned} \bar{\mu} &= \left(\mathbf{x}^\top \mathbf{x} + \sigma_n^2 \mathbf{I}\right)^{-1} \mathbf{x}^\top \mathbf{y} \\ \bar{\Sigma} &= \left(\sigma_n^{-2} \mathbf{x}^\top \mathbf{x} + \mathbf{I}\right)^{-1} \end{aligned} \tag{2.29}$$

■

Uncertainty in Prediction Define $f^* = \beta^\top \mathbf{x}^*$ as the model's output at $\mathbf{x}^*$. The uncertainty of the prediction is

$$\mathbf{x}^{*\top} \bar{\Sigma} \mathbf{x}^* + \sigma_n^2.$$

Proof.

$$p(f^* \mid \mathbf{x}_{1:n}, y_{1:n}, \mathbf{x}^*) = \int p(f^* \mid \beta, \mathbf{x}^*) \ p(\beta \mid \mathbf{x}_{1:n}, y_{1:n}, \mathbf{x}^*) \mathrm{d}\beta \tag{2.30}$$

Since $\beta \sim \mathcal{N}(\bar{\mu}, \bar{\Sigma})$,

$$p(f^* \mid \mathbf{x}_{1:n}, y_{1:n}, \mathbf{x}^*) = \mathcal{N}(\bar{\mu}^\top \mathbf{x}^*, \mathbf{x}^{*\top} \bar{\Sigma} \mathbf{x}^*) \tag{2.31}$$

Since $y^* = f^* + \xi$, $\xi \sim \mathcal{N}(0, \sigma_n^2)$,

$$p(y^* \mid \mathbf{x}_{1:n}, y_{1:n}, \mathbf{x}^*) = \mathcal{N}(\bar{\mu}^\top \mathbf{x}^*, \mathbf{x}^{*\top} \bar{\Sigma} \mathbf{x}^* + \sigma_n^2) \tag{2.32}$$

Here, $\bar{\mu}$ and $\bar{\Sigma}$ are the mean and variance of posterior, respectively. ■

Moreover, we can distinguish two forms of uncertainty:

1. **Epistemic uncertainty** ($\mathbf{x}^{*\top} \bar{\Sigma} \mathbf{x}^*$): Uncertainty about the model due to the lack of data.
2. **Aleatoric uncertainty** (σ_n^2): Irreducible noise from measurement.

[1] For more details, please check here https://en.wikipedia.org/wiki/Bayesian_linear_regression

2.3 Linear Discriminative Analysis (LDA)

2.3.1 Fisher's Linear Discriminant

To separate two classes of data points, Fisher's idea is to find a hyper-plane, on which the data points project with maximal distance between the centers and minimal variance within the class.

Definition 2.3.1 (Separation) *Suppose two classes of observations have mean $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$ and covariances Σ_1, Σ_2. The separation between these two distributions is*

$$S(\mathbf{w}) = \frac{\sigma^2_{between}}{\sigma^2_{within}} = \frac{(\mathbf{w}\cdot\boldsymbol{\mu}_1 - \mathbf{w}\cdot\boldsymbol{\mu}_2)^2}{\mathbf{w}^\top\Sigma_1\mathbf{w} + \mathbf{w}^\top\Sigma_2\mathbf{w}} = \frac{(\mathbf{w}^\top(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2))^2}{\mathbf{w}^\top(\Sigma_1 + \Sigma_2)\mathbf{w}} := \frac{\mathbf{w}^\top\mathbf{S}_b\mathbf{w}}{\mathbf{w}^\top\mathbf{S}_w\mathbf{w}}$$

Here, $\mathbf{S}_b = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\top$ *and* $\mathbf{S}_w = \Sigma_1 + \Sigma_2$.

Theorem 2.3.1 (Solution) *The optimal solution of* $\max_\mathrm{w} S(\mathbf{w})$ *satisfies*

$$\mathbf{w}^* \propto \mathbf{S}_w^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2).$$

Proof. Setting the derivatives of S to 0, we have

$$0 = \frac{\partial}{\partial\mathbf{w}}S(\mathbf{w}) = \frac{\mathbf{w}^\top\mathbf{S}_w\mathbf{w}\cdot 2\mathbf{S}_b\mathbf{w} - \mathbf{w}^\top\mathbf{S}_b\mathbf{w}\cdot 2\mathbf{S}_w\mathbf{w}}{(\mathbf{w}^\top\mathbf{S}_w\mathbf{w})^2} \tag{2.33}$$

$$\implies \mathbf{w}^\top\mathbf{S}_w\mathbf{w}\cdot\mathbf{S}_b\mathbf{w} - \mathbf{w}^\top\mathbf{S}_b\mathbf{w}\cdot\mathbf{S}_w\mathbf{w} = 0 \tag{2.34}$$

$$\implies \mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{w} = \frac{\mathbf{w}^\top\mathbf{S}_b\mathbf{w}}{\mathbf{w}^\top\mathbf{S}_w\mathbf{w}}\cdot\mathbf{w} \tag{2.35}$$

$$\implies \mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{w} = \lambda\mathbf{w}, \quad \text{where } \lambda = \frac{\mathbf{w}^\top\mathbf{S}_b\mathbf{w}}{\mathbf{w}^\top\mathbf{S}_w\mathbf{w}}. \tag{2.36}$$

$$\tag{2.37}$$

This is an eigenvalue problem. However, consider the fact that $\mathbf{S}_b\mathbf{w} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\top\mathbf{w}$, we have

$$\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{w} = \mathbf{S}_w^{-1}[\beta(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)] = \beta[\mathbf{S}_w^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)], \quad \text{where } \beta = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\top\mathbf{w}, \tag{2.38}$$

which means the solution of the eigenvalue problem is just $\mathbf{w}^* = \beta\mathbf{S}_w^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$. ■

2.3.2 LDA with Gaussian Prior

Definition 2.3.2 *One way to employ linear model to classification problem is Linear Discriminative Analysis (LDA). It given the probability of a sample* $\mathbf{x}$ *lying in the positive class,*

$$p(y = 1 \mid \mathbf{x}) = \sigma(w^\top\mathbf{x} + w_0), \quad \textit{where } \sigma(\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{x})} \tag{2.39}$$

Proof.

$$p(y = 1 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid y = 1)\, p(y = 1)}{p(\mathbf{x} \mid y = 1)\, p(y = 1) + p(\mathbf{x} \mid y = 0)\, p(y = 0)} \tag{2.40}$$

$$= \frac{1}{1 + \frac{p(\mathrm{x}|y=0)\,p(y=0)}{p(\mathrm{x}|y=1)\,p(y=1)}} \tag{2.41}$$

$$= \frac{1}{1 + \exp\left(-\log\frac{p(\mathrm{x}|y=1)\,p(y=1)}{p(\mathrm{x}|y=0)\,p(y=0)}\right)} \tag{2.42}$$

If we assume that all $p(\mathbf{x} \mid y = i)$ are Gaussian distributions with the same variance Σ, we have

$$\log p(\mathbf{x} \mid y = i) = -\frac{1}{2} \log |2\pi\Sigma| - \frac{1}{2}\mathbf{x}^\top \Sigma^{-1}\mathbf{x} + \mathbf{x}^\top \Sigma^{-1}\mu_i - \frac{1}{2}\mu_i^\top \Sigma^{-1}\mu_i \tag{2.43}$$

Then,

$w^\top \mathbf{x} = \mathbf{x}^\top w \in \mathbb{R}$

$$\log \frac{p(\mathbf{x} \mid y = 1)\, p(y = 1)}{p(\mathbf{x} \mid y = 0)\, p(y = 0)} = \underbrace{\mathbf{x}^\top \Sigma^{-1}(\mu_1 - \mu_0)}_{\mathrm{x}^\top w} \underbrace{- \frac{1}{2}(\mu_1\Sigma^{-1}\mu_1 - \mu_0\Sigma^{-1}\mu_0) + \log \frac{p(y = 1)}{p(y = 0)}}_{w_0} \tag{2.44}$$

Insert Eq. 2.44 to Eq. 2.42, we have

$$p(y = 1 \mid \mathbf{x}) = \frac{1}{1 + \exp(w^\top \mathbf{x} + w_0)} \tag{2.45}$$

This proof provides some intuition of the *sigmoid* function σ. ■

2.3.3 Quadratic Discriminative Analysis

Definition 2.3.3 *We can extend LDA by using a quadratic function, named Quadratic Discriminative Analysis (QDA). It can model the clusters with different variance. It given the probability of a sample* $\mathbf{x}$ *lying in the positive class as,*

$$p(y = 1 \mid \mathbf{x}) = \sigma(\mathbf{x}^\top W\mathbf{x} + w^\top \mathbf{x} + w_0), \quad \textit{where } \sigma(\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{x})} \tag{2.46}$$

Proof. Similar to LDA, but $p(\mathbf{x} \mid y = i)$ are Gaussian distributions with different variance of Σ_i. ■

Chapter 3

Support Vector Machine

3.1 Lagrange Duality & KKT Conditions

Considering the following optimization problem for $\mathbf{x} \in \mathbb{R}^d$ with constraints $g(\mathbf{x}) \leq 0$ and $h(\mathbf{x}) = 0$,

$$\min_{\mathbf{x}} \; f(\mathbf{x}), \quad s.t. \; g_i(\mathbf{x}) \leq 0, h_j(\mathbf{x}) = 0 \quad (\forall i = [m], j = [n]), \tag{3.1}$$

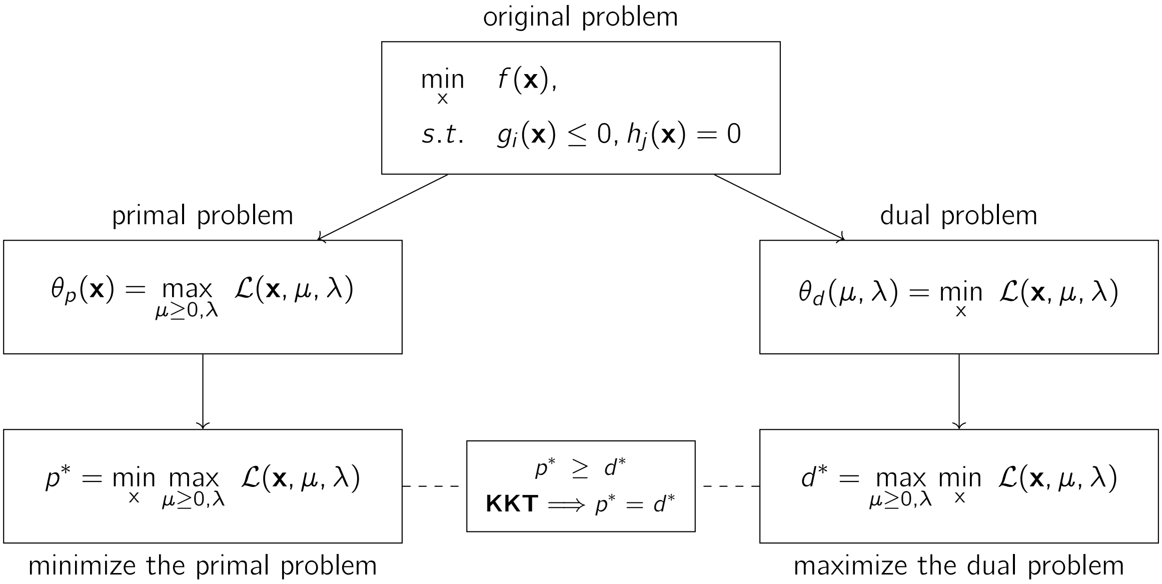
One can form the **generalized Lagrangian** as

$$\mathcal{L}(\mathbf{x}, \mu, \lambda) = f(\mathbf{x}) + \sum_i \mu_i g_i(\mathbf{x}) + \sum_j \lambda_j h_j(\mathbf{x}), \quad \mu_i \geq 0. \tag{3.2}$$

Here, μ_i and λ_j $(i = [m], j = [n])$ are the Lagrange multipliers.

3.1.1 Primal and Dual Problem

With the generalized Lagrangian, the original problem can be turned into an optimization problem without constraints. There are two ways for doing so, which pose the "primal" and "dual" problem, respectively.


original problem
$\min_{\mathbf{x}} \; f(\mathbf{x}),$
$s.t. \; g_i(\mathbf{x}) \leq 0, h_j(\mathbf{x}) = 0$
primal problem
dual problem
$\theta_p(\mathbf{x}) = \max_{\mu \geq 0, \lambda} \mathcal{L}(\mathbf{x}, \mu, \lambda)$
$\theta_d(\mu, \lambda) = \min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \mu, \lambda)$
$p^* = \min_{\mathbf{x}} \max_{\mu \geq 0, \lambda} \mathcal{L}(\mathbf{x}, \mu, \lambda)$
$p^* \geq d^*$
$\textbf{KKT} \Longrightarrow p^* = d^*$
$d^* = \max_{\mu \geq 0, \lambda} \min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \mu, \lambda)$
minimize the primal problem
maximize the dual problem

The primal problem and dual problem have the same objective, except that the order of the "max" and the "min" are exchanged.

3.1.2 KKT Conditions

The Karush-Kuhn-Tucker (KKT) theorem state a **necessary condition** that there must exist $(\mathbf{x}^*, \mu^*, \lambda^*)$ that is the solution for both primal problem and dual problem, and moreover

$p^* = d^* = \mathcal{L}(\mathbf{x}^*, \mu^*, \lambda^*)$. The KKT conditions are as follows,

$$\frac{\partial}{\partial x_i}\mathcal{L}(\mathbf{x}^*, \mu^*, \lambda^*) = 0, \quad i = [d] \tag{3.3}$$

$$\frac{\partial}{\partial \lambda_j}\mathcal{L}(\mathbf{x}^*, \mu^*, \lambda^*) = 0, \quad j = [n] \tag{3.4}$$

$$\mu_i^* g_i(\mathbf{x}^*) = 0, \quad i = [m] \qquad (\text{“dual complementarity”}) \tag{3.5}$$

$$g_i(\mathbf{x}^*) \leq 0, \quad i = [m] \tag{3.6}$$

$$\mu_i^* \geq 0, \quad i = [m] \tag{3.7}$$

3.2 Original SVM

3.2.1 Hard-margin SVM

Definition 3.2.1 (SVM - Primal Problem) *Given a training dataset of n points of the form* $(\mathbf{x}_i, y_i)$, $y_i = \{1, -1\}$, *the SVM algorithm finds the optimal hyper-plan* $\mathbf{w}^\top \mathbf{x} + w_0 = 0$ *that separates the positive and negative points with the maximum margin. More formally,*

$$\min_{\mathrm{w}, w_0} \quad \frac{1}{2}\|\mathbf{w}\|^2 \tag{3.8}$$

$$s.t. \quad y_i(\mathbf{w}^\top \mathbf{x}_i + w_0) \geq 1, \quad i = [n] \tag{3.9}$$

Definition 3.2.2 (SVM - Dual Problem) *The Lagrangian dual problem of SVM is given by*

$$\max_{\alpha} \quad \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \tag{3.10}$$

$$s.t. \quad \alpha_i \geq 0, \ \sum_i \alpha_i y_i = 0 \tag{3.11}$$

Proof. We show that how SVM's primal problem can be turned into its dual problem. We first construct the Lagrangian

$$\mathcal{L}(\mathbf{w}, w_0; \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i \leq n} \alpha_i [y_i(\mathbf{w}^\top \mathbf{x} + w_0) - 1]. \tag{3.12}$$

To get the dual problem $\theta_d(\alpha)$, we need to minimize the Lagrangian by setting its derivatives to 0,

$$\mathbf{0} = \nabla_{\mathrm{w}} \mathcal{L}(\mathbf{w}, w_0; \alpha) = \mathbf{w} - \sum_{i \leq n} \alpha_i y_i \mathbf{x}_i \quad \Longrightarrow \quad \mathbf{w}^* = \sum_{i \leq n} \alpha_i y_i \mathbf{x}_i, \tag{3.13}$$

$$0 = \frac{\partial}{\partial w_0}\mathcal{L}(\mathbf{w}, w_0; \alpha) = \sum_{i \leq n} \alpha_i y_i \quad \Longrightarrow \quad \sum_{i \leq n} \alpha_i y_i = 0. \tag{3.14}$$

If we plug them back into Eg. 3.12, we have

$$\mathcal{L}(\mathbf{w}^*, w_0^*; \alpha) = \sum_{i \leq n} \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j - \underbrace{w_0 \sum_{i \leq n} \alpha_i y_i}_{=0}, \tag{3.15}$$

Then the dual problem is achieved as

$$\max_{\alpha} \quad \mathcal{L}(\mathbf{w}^*, w_0^*; \alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \tag{3.16}$$

$$s.t. \quad \alpha_i \geq 0, \ \sum_i \alpha_i y_i = 0 \tag{3.17}$$

■

Remark. The SVM can be more efficiently calculated in its dual form, because we can easily find the support vectors (whose $\alpha_i = 0$) and then derive the classification results directly by

$$f(\mathbf{x}_*) = \mathbf{w}^\top \mathbf{x}_* + w_0 = w_0 + \sum_{\{i|\alpha_i > 0\}} \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x}_* \rangle. \tag{3.18}$$

It should be noticed that the hard-margin SVM requires both assumptions to work:

- The data points are linearly separable;
- There exists a separating hyperplane with non-zero margin.

3.2.2 Soft-margin SVM

Definition 3.2.3 (SoftSVM - Primal Problem)

$$\min_{\mathrm{w}, w_0} \quad \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i \le n} \xi_i \tag{3.19}$$

$$s.t. \quad y_i(\mathbf{w}^\top \mathbf{x}_i + w_0) \ge 1 - \xi_i, \quad \xi_i > 0 \tag{3.20}$$

Definition 3.2.4 (SoftSVM - Dual Problem)

$$\min_{\alpha} \quad \frac{1}{2} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \tag{3.21}$$

$$s.t. \quad 0 \le \alpha_i \le C, \ \sum_i \alpha_i y_i = 0 \tag{3.22}$$

Proof. Similar to hard-margin SVM, given $\alpha_i, \beta_i \ge 0$, we have

$$\mathcal{L}(\mathbf{w}, w_0, \xi; \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i \le n} \xi_i - \sum_{i \le n} \alpha_i [y_i(\mathbf{w}^\top \mathbf{x} + w_0) - 1 + \xi_i] - \sum_{i \le n} \beta_i \xi_i. \tag{3.23}$$

Set its derivatives of Lagrangian to 0:

$$\mathbf{0} = \nabla_{\mathrm{w}} \mathcal{L}(\mathbf{w}, w_0, \xi; \alpha) = \mathbf{w} - \sum_i \alpha_i y_i \mathbf{x}_i \quad \Longrightarrow \quad \mathbf{w}^* = \sum_i \alpha_i y_i \mathbf{x}_i; \tag{3.24}$$

$$0 = \frac{\partial}{\partial w_0} \mathcal{L}(\mathbf{w}, w_0, \xi; \alpha) = \sum_i \alpha_i y_i \quad \Longrightarrow \quad \sum_i \alpha_i y_i = 0; \tag{3.25}$$

$$0 = \frac{\partial}{\partial \xi_i} \mathcal{L}(\mathbf{w}, w_0, \xi; \alpha) = C - \alpha_i + \beta_i \quad \Longrightarrow \quad \alpha_i = C - \beta_i \ \ (i = [n]). \tag{3.26}$$

If we plug them back into Eg. 3.23, we get

$$\mathcal{L}(\mathbf{w}^*, w_0^*; \alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j - \underbrace{w_0 \sum_i \alpha_i y_i}_{=0}, \quad \text{where} \ \ \alpha_i \le C. \tag{3.27}$$

Then we can obtain the dual problem. ■

The optimal solution is given by

$$\mathbf{w}^* = \sum_i \alpha_i^* y_i \mathbf{x}_i \tag{3.28}$$

$$\xi_i^* = \max(0, 1 - y_i(\mathbf{w}^{*\top} \mathbf{x}_i + w_0^*)) \tag{3.29}$$

Remark. Here, C is a trade-off hyper-parameter. Larger C means narrower margin & few neglected samples. When $C \to \infty$, then the solution is approaching to the solution from a hard-margin SVM.

Property 3.2.1 (Hinge Loss Form) *The soft-margin SVM can be also achieved via hinge loss $\ell_{hinge}(\cdot,\cdot)$ as*

$$\min_{\mathrm{w},w_0} \quad \sum_{i\le n} \ell_{hinge}(\mathbf{x}_i, y_i) + \frac{1}{2C}\|\mathbf{w}\|^2, \quad \textit{where} \;\; \ell_{hinge}(\mathbf{x}_i, y_i) = \max\left\{0, 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + w_0)\right\}$$

Remark. Since hinge loss is convex and its derivative is known, we can solve the soft-margin SVM directly by gradient descent methods. Note that there is **no** such loss for hard-margin SVM, because no misclassifications, by definition, will occur in hard-margin SVM.

3.3 Extended SVM

3.3.1 Multi-Class SVM

SVMs can be generalized to a multi-class scenario. The key idea is to maintain c weight vectors $w^{(i)}$, one for each class. The prediction result is then

$$\hat{y} = \operatorname*{argmax}_i \quad w^{(i)\top}\mathbf{x} \tag{3.30}$$

For each data point, it should hold that the prediction for the true class is separated by a margin from the class which has the second highest prediction, i.e.

$$w^{(y)\top}\mathbf{x} \ge \max_{i\neq y} \quad w^{(i)\top}\mathbf{x} + 1 \tag{3.31}$$

Thus, the multi-class Hinge loss is given as

$$\ell_{hinge}(w^{(1)}, \cdots, w^{(c)}; \mathbf{x}, y) = \max\{0, 1 + \max_{j\neq y} \; w^{(j)\top}\mathbf{x} - w^{(y)\top}\mathbf{x}\}$$

3.3.2 Structured SVM

Structured SVM generalizes the SVM, which maximizes the margin between the score of the correct class and the score of the highest-scoring incorrect runner-up class (a.k.a. the hard nagetives).

Definition 3.3.1 (StructuredSVM - Primal Problem)

$$\min_{w,w_0,\xi} \quad \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{1\le i\le n} \xi_i \tag{3.32}$$

$$s.t. \quad \mathbf{w}^\top\Psi(\mathbf{x}_i, y_i) - \mathbf{w}^\top\Psi(x_i, y) \ge \Delta(y_i, y) - \xi_i, \quad \xi_i > 0, \quad \forall i \le n, y \neq y_i, \tag{3.33}$$

where $\Psi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^{d_1} \times \mathbb{Z}^{d-d_1}$ is a joint-feature map function. $\Delta(\cdot,\cdot) : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ is function measuring the distance between two labels, which satisfies $\Delta(y, y) = 0$. $\Delta(\cdot,\cdot)$ defines the margin of each pair of samples.

Replacing the margin 1 with a general function $\Delta(y, y')$.

Definition 3.3.2 (StructuredSVM - Dual Problem) *To simplify the notation, let $\Psi_i(y) := \Psi(y_i, \mathbf{x}_i) - \Psi(y, \mathbf{x}_i)$ and $\Delta_i(y) := \Delta(y, y_i)$. Then, the dual problem is*

$$\min_\alpha \quad -\frac{1}{2}\left\|\sum_{i=1}^n \sum_{y_j\in\mathbb{K}_i} \alpha_{ij}\Psi_i(y_j)\right\|^2 + \sum_{i=1}^n \sum_{y_j\in K_i} \alpha_{ij}\Delta_i(y_j) \tag{3.34}$$

$$s.t. \quad 0 \le \sum_{y_j\in\mathbb{K}_i} \alpha_{ij} \le C, \quad \alpha_{ij} \ge 0. \tag{3.35}$$

Here, $\mathbb{K}_i = \mathcal{Y}/\{y_i\}$.

Proof. Let $\mathbb{K}_i = \mathcal{Y}/\{y_i\}$. The Lagrangian is

This is taken from the Ex.5-1.

$$\mathcal{L}(\mathbf{w}, \xi, \alpha, \beta) = \frac{1}{2}\mathbf{w}^\top\mathbf{w} + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\sum_{y_j \in \mathbb{K}_i}\alpha_{ij}\left(\mathbf{w}^\top\Psi_i\left(y_j\right) - \Delta_i\left(y_j\right) + \xi_i\right) - \sum_{i=1}^{n}\beta_i\xi_i \tag{3.36}$$

According to stationary conditions, we have

$$0 = \nabla_{\mathrm{w}}\mathcal{L} = \mathbf{w} - \sum_{i=1}^{n}\sum_{y_j \in \mathbb{K}_i}\alpha_{ij}\Psi_i(y_j), \tag{3.37}$$

$$0 = \frac{\partial}{\partial \xi_i}\mathcal{L} = C - \sum_{y_j \in \mathbb{K}_i}\alpha_{ij} - \beta_i. \tag{3.38}$$

Plug them into the Lagrangian, we get

$$\mathcal{L}(\alpha) = -\frac{1}{2}\left\|\sum_{i=1}^{n}\sum_{y_j \in \mathbb{K}_i}\alpha_{ij}\Psi_i\left(y_j\right)\right\|^2 + \sum_{i=1}^{n}\sum_{y_j \in \mathbb{K}_i}\alpha_{ij}\Delta_i\left(y_j\right) \tag{3.39}$$

Note that, Eq. 3.38 also implies $\sum_{y_j \in \mathbb{K}_i}\alpha_{ij} \le C$. This concludes the proof. ■

Remark. In the dual form, constraints are separable in blocks which is favorable for optimization.

Property 3.3.1 (Similarity with CRF) *The structured SVM can be written into the loss form as*

$$\min \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{1\le i\le n}\left[\max_{y\in\mathcal{Y}}\ \Delta(y_i, y) - \mathbf{w}^\top\Psi(\mathbf{x}_i, y_i) + \mathbf{w}^\top\Psi(x_i, y)\right]$$

The CRF can be formulated as

$$\min \frac{\|\mathbf{w}\|^2}{2\sigma^2} + \sum_{1\le i\le n}\left[\log\sum_{y\in\mathcal{Y}}\exp\left(\mathbf{w}^\top\Psi(\mathbf{x}_i, y_i) + \mathbf{w}^\top\Psi(x_i, y)\right)\right]$$

Remark. They both do regularized risk minimization and $\log\sum_y \exp(\cdot)$ can be interpreted as the `softmax` function.

Table 3.1: Summary of some unstructured and structured models

Training Criteria	**Unstructured**	**Structured**
Max posterior's likelihood	Naive Bayes $p(\mathbf{x} \mid y)$	Hidden Markov Model $p(\mathbf{x} \mid \mathbf{y})$
Max conditional likelihood	Logistic / NN $p(y \mid \mathbf{x})$	Conditional Random Field $p(\mathbf{y} \mid \mathbf{x})$
Max margin	SVM $\alpha^\top\phi(\mathbf{x})$	Structured SVM $\alpha^\top\psi(\mathbf{x}, \mathbf{y})$

Chapter 4

Kernel Tricks

To handle the linear unseparable data, one common method is to use the "Kernel Trick" that maps the data into a new high-dimension space.

4.1 Properties of Kernels

One of the fundamental mathematical results underlying learning theory with kernels is Mercer's theorem.

Definition 4.1.1 (Gram Matrix) *Let $\mathcal{X}$ be a closed subset of $\mathbb{R}^n$ ($n \in \mathbb{N}$) and $S = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\} \subset \mathcal{X}$. For any kernel function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and its corresponding feature map function $\phi(\mathbf{x}) : \mathcal{X} \to \mathcal{H}$, the Gram matrix $\mathbf{K}$ on S is defined as*

$$\mathbf{K} = \begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & \cdots & K(\mathbf{x}_1, \mathbf{x}_n) \\ \vdots & & \vdots \\ K(\mathbf{x}_n, \mathbf{x}_1) & \cdots & K(\mathbf{x}_n, \mathbf{x}_n) \end{bmatrix} = \begin{bmatrix} \phi(\mathbf{x}_1) \\ \vdots \\ \phi(\mathbf{x}_n) \end{bmatrix} [\phi(\mathbf{x}_1), \cdots, \phi(\mathbf{x}_n)],$$

which must be positive semi-definite.

Theorem 4.1.1 (Mercer's Theorem) *Let $\mathcal{X}$ be a closed subset of $\mathbb{R}^n$ ($n \in \mathbb{N}$), μ a Borel measure on $\mathcal{X}$, and $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a symmetric function, i.e., $K(\mathbf{x}, \mathbf{y}) = K(\mathbf{y}, \mathbf{x})$.*

1. Continuous version. *For any $f \in L^2(\mathcal{X}, \mu)$,*

$$\iint f(\mathbf{x}) K(\mathbf{x}, \mathbf{y}) f(\mathbf{y}) \, \mathrm{d}\mathbf{x} \, \mathrm{d}\mathbf{y} \geq 0 \iff K(\mathbf{x}, \mathbf{y}) \textit{ is a kernel function,}$$

2. Discrete version. *For any finite set of points $\{x_1, \cdots, x_N\} \subset \mathcal{X}$ and $f : \mathcal{X} \to \mathbb{R}$,*

$$\sum_{i,j \leq N} f(\mathbf{x}_i) K(\mathbf{x}_i, \mathbf{x}_j) f(\mathbf{x}_j) \geq 0 \iff \mathbf{f}^\top \mathbf{K} \mathbf{f} \geq 0 \iff \mathbf{K} \in \mathbf{SP}_+ \textit{ is a kernel matrix,}$$

where $\mathbf{f} = [f(x_1), \cdots, f(x_N)]^\top$ and $\mathbf{K}$ is the Gram matrix.

Remark. To become a kernel, the matrix $\mathbf{K}$ must be

- square and symmetric;
- semi-positive definite (can be judged via Sylvester's criterion).

Property 4.1.1 (Kernel Combinations) *Let $K_1 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and $K_2 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be two kernel functions, $c > 0$ be a scalar, $f : \mathbb{R} \to \mathbb{R}$ be either a polynomial with positive coefficients or the exponential function and $\mathcal{V} : \mathcal{Z} \to \mathcal{X}$ be a mapping. Then, the following combinations are also valid kernels:*

- $K(\mathbf{x}, \mathbf{y}) = K_1(\mathbf{x}, \mathbf{y}) + K_2(\mathbf{x}, \mathbf{y})$
- $K(\mathbf{x}, \mathbf{y}) = c \cdot K_1(\mathbf{x}, \mathbf{y}) \quad (c > 0)$
- $K(\mathbf{x}, \mathbf{y}) = K_1(\mathbf{x}, \mathbf{y}) \cdot K_2(\mathbf{x}, \mathbf{y})$

- $K(\mathbf{x},\mathbf{y}) = g(K_1(\mathbf{x},\mathbf{y}))$ *(g is a positive-coef. polynomial or the exponential function)*
- $K(\mathbf{z},\mathbf{z}') = K_1(\mathcal{V}(\mathbf{z}),\mathcal{V}(\mathbf{z}'))$

Proof. Here we just prove some of them and only consider the matrix form.

1. For any $\mathbf{f}$,

$$\mathbf{f}^\top(\mathbf{K}_1+\mathbf{K}_2)\mathbf{f} = \mathbf{f}^\top\mathbf{K}_1\mathbf{f} + \mathbf{f}^\top\mathbf{K}_2\mathbf{f} \geq 0. \tag{4.1}$$

2. For any $\mathbf{f}$,

$$\mathbf{f}^\top(c\mathbf{K}_1)\mathbf{f} = c\ \mathbf{f}^\top\mathbf{K}_1\mathbf{f} \geq 0. \tag{4.2}$$

3. Let $K_1(\mathbf{x},\mathbf{y}) = \phi_1(\mathbf{x})^\top\phi_1(\mathbf{y})$, $K_2(\mathbf{x},\mathbf{y}) = \phi_2(\mathbf{x})^\top\phi_2(\mathbf{y})$,

$$\begin{aligned}
K_1(\mathbf{x},\mathbf{y})K_2(\mathbf{x},\mathbf{y}) &= \phi_1(\mathbf{x})^\top\phi_1(\mathbf{x})\phi_2(\mathbf{x})^\top\phi_2(\mathbf{x}) && (4.3)\\
&= \sum_{m=1}^{M}\phi_1^{(m)}(\mathbf{x})\phi_1^{(m)}(\mathbf{y})\sum_{n=1}^{N}\phi_2^{(n)}(\mathbf{x})\phi_2^{(n)}(\mathbf{y}) && (4.4)\\
&= \sum_{m=1}^{M}\sum_{n=1}^{N}\phi_1^{(m)}(\mathbf{x})\phi_1^{(m)}(\mathbf{y})\phi_2^{(n)}(\mathbf{x})\phi_2^{(n)}(\mathbf{y}) && (4.5)\\
&= \sum_{k=1}^{MN}\psi_k(\mathbf{x})^\top\psi_k(\mathbf{y}) && (4.6)
\end{aligned}$$

where $\psi_k(\mathbf{x}) = \mathrm{vec}(\phi_1(\mathbf{x})\phi_2(\mathbf{x})^\top)_k = \phi_1^{(\lceil k-1/N\rceil)}(\mathbf{x})\phi_1^{(k-1\%N)}(\mathbf{x})$.

here, $\lceil\cdot\rceil$ is for ceiling and % is for remainder.

4. Can be proved using previous three conclusions. ■

Exercise 4.1.1 (Ex.4-3: Kernel Function) *Assume we are given a probability density function $p(\mathbf{x},h)$, where $\mathbf{x}\in\mathcal{X}$ and $h\in\mathcal{H}$ (finite sets). Consider a kernel $k((\mathbf{x},h),(\mathbf{x}',h'))$ defined on pairs $(\mathbf{x},h)\in\mathcal{X}\times H$. Prove that following function $k_m:\mathcal{X}\times\mathcal{X}\to\mathbb{R}$ defines a kernel.*

$$k_m(\mathbf{x},\mathbf{y}) = \sum_{h\in\mathcal{H}}\sum_{h'\in\mathcal{H}} k\left((\mathbf{x},h),(\mathbf{y},h')\right)p(h\mid\mathbf{x})p\left(h'\mid\mathbf{y}\right)$$

Solution Let $k((\mathbf{x},h),(\mathbf{x}',h')) = \phi(\mathbf{x},h)^\top\phi(\mathbf{x}',h')$.

$$\begin{aligned}
k_m(\mathbf{x},\mathbf{y}) &= \sum_{h\in\mathcal{H}}\sum_{h'\in\mathcal{H}}\phi(\mathbf{x},h)^\top\phi(\mathbf{x}',h')p(h\mid\mathbf{x})p\left(h'\mid\mathbf{y}\right) && (4.7)\\
&= \left[\sum_{h\in\mathcal{H}}p(h\mid\mathbf{x})\phi(\mathbf{x},h)\right]^\top\left[\sum_{h'\in\mathcal{H}}p(h'\mid\mathbf{y})\phi(\mathbf{y},h')\right] && (4.8)
\end{aligned}$$

This means $k_m(\mathbf{x},\mathbf{y})$ can be decomposed into $\psi(\mathbf{x})^\top\psi(\mathbf{y})$. ■

4.2 Useful Kernels

4.2.1 Polynomial Kernel

Definition 4.2.1 (Poly Kernel) *Polynomial kernels of degree d over $\mathbf{x}\in\mathbb{R}^N$ are defined as*

$$K(\mathbf{x},\mathbf{y}) = (c+\mathbf{x}\cdot\mathbf{y})^d.$$

A polynomial kernel can represent the inner product of two polynomial mappings $\phi(\mathbf{x}), \phi(\mathbf{y}) : \mathbb{R}^N \to \mathbb{R}^d$ by

$$\phi(\mathbf{x})^\top \phi(\mathbf{y}) := K(\mathbf{x}, \mathbf{y}) = \sum_{i=0}^{d} \binom{d}{i} c^{d-i} (\mathbf{x} \cdot \mathbf{y})^i, \tag{4.9}$$

This mapping function $\phi(\mathbf{x})$ is not unique, and may not be the smallest.

where

$$\phi(\mathbf{x}) = \left[\sqrt{c^d}, \sqrt{dc^{d-1}}\mathbf{x}, \cdots, \sqrt{\binom{d}{i}}\mathbf{x}^d\right]^\top, \quad \phi(\mathbf{y}) = \left[\sqrt{c^d}, \sqrt{dc^{d-1}}\mathbf{y}, \cdots, \sqrt{\binom{d}{i}}\mathbf{y}^d\right]^\top$$

Property 4.2.1 (Dimension of Feature Space) [1] *The smallest dimension of the feature space $\phi(\mathbf{x})$ associated to the polynomial kernel $K(\mathbf{x}, \mathbf{y}) : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$ of degree d is*

$$\binom{N+d}{d}.$$

Proof. The dimension of the feature space $\phi(\mathbf{x})$ is thus the number of such monomials $f(N, d) \in \mathbb{Z}$, that is the number of ways of adding N non-negative integers to obtain a sum of at most d. We have

$$f(N, d) = \underbrace{f(N-1, d)}_{N \text{ integers with sum of } d} + \underbrace{f(N, d-1)}_{N \text{ integers with sum at most } d-1} \tag{4.10}$$

The result then follows by induction on $N+d$, using initial the conditions $f(1, 0) = f(0, 1) = 1$. ■

4.2.2 RBF Kernel

The Radial-basis Function (RBF) kernel, also called the Gaussian kernel or squared exponential kernel, is a popular kernel that is in the form of a radial basis function.

Definition 4.2.2 (RBF Kernel) *With hyperparameter of scale h and variance σ^2, the RBF kernel is defined as*

$$k_{RBF}(\mathbf{x}, \mathbf{y}) = \sigma^2 \exp\left(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2h^2}\right).$$

The RBF kernel can project vectors into an infinite dimensional space, i.e. $\phi_{RBF} : \mathbb{R}^d \to \mathbb{R}^\infty$. Without loss of generality, after assuming $\sigma = 1, h = 1/4$, we have

$$\exp\left(-\|\mathbf{x}-\mathbf{y}\|^2\right) = \sum_{k=0}^{\infty} \underbrace{\left(\sqrt{\frac{1}{k!}} \exp\left(-\|\mathbf{x}\|^2\right) \phi_{\text{poly}}(\mathbf{x})\right)}_{\text{row } k \text{ in } \phi_{\text{RBF}}(\text{x})} \cdot \underbrace{\left(\sqrt{\frac{1}{k!}} \exp\left(-\|\mathbf{y}\|^2\right) \phi_{\text{poly}}(\mathbf{y})\right)}_{\text{row } k \text{ in } \phi_{\text{RBF}}(\text{y})} \tag{4.11}$$

Property 4.2.2 (RBF Kernel is Stationary) *RBF is a stationary kernel, since*

$$k_{RBF}(\mathbf{x}, \mathbf{y}) = k_{RBF}(\mathbf{x}+\Delta, \mathbf{y}+\Delta) = g(\mathbf{y}-\mathbf{x}). \tag{4.12}$$

[1] See slides here: https://cs.nyu.edu/~mohri/ml/ml10/sol3.pdf

4.2.3 Periodic Kernel

Definition 4.2.3

$$k_{Per}(\mathbf{x}, \mathbf{y}) = \sigma^2 \exp\left(-\frac{2\sin^2\left(\pi\left|\mathbf{x}-\mathbf{y}\right|/p\right)}{h^2}\right)$$

Here, the period p simply determines the distance between repetitions of the peaks, and the h determines the length scale function in the same way as in the RBF kernel.

Property 4.2.3 (Periodic Kernel is Stationary) *Periodic is a stationary kernel, since*

$$k_{Per}(\mathbf{x}, \mathbf{y}) = k_{Per}(\mathbf{x}+\Delta, \mathbf{y}+\Delta) = g(\mathbf{y}-\mathbf{x}). \tag{4.13}$$

Chapter 5

Gaussian Process

5.1 Properties of Gaussian Distribution

Definition 5.1.1 (Multivariate Gaussian Distribution) *We say $\mathbf{x} = (x_1, \cdots, x_n)$ has a multivariate Gaussian distribution if and only if every linear combination of its components has a (multivariate) Gaussian distribution. Formally,*

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \Leftrightarrow A\mathbf{x} \sim \mathcal{N}(A\boldsymbol{\mu}, A\boldsymbol{\Sigma}A^\top), \ \forall A.$$

Property 5.1.1 (Marginal Distribution) *From the definition, any subset of random variables $\mathbf{x}' \subset \mathbf{x}$ are themselves normally distributed, and the mean and covariance is given by simply ignoring all elements that are not in $\mathbf{x}'$.*

Property 5.1.2 (Conditional Distribution) *For a joint Gaussian distribution, if we given the values of some RVs $\mathbf{y}$, then the conditional distribution of the remaining RVs $\mathbf{y}_*$ is*

$$p(\mathbf{y}_* \mid \mathbf{y}) = \mathcal{N}(\boldsymbol{\mu} + \boldsymbol{\Sigma}_* \boldsymbol{\Sigma}_{**}^{-1}(\mathbf{y}_* - \boldsymbol{\mu}_*), \boldsymbol{\Sigma} - \boldsymbol{\Sigma}_* \boldsymbol{\Sigma}_{**}^{-1} \boldsymbol{\Sigma}_*^\top),$$

where

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{y}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu}_* \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma}_* \\ \boldsymbol{\Sigma}_*^\top & \boldsymbol{\Sigma}_{**} \end{bmatrix} \right).$$

Proof. By definition, the conditional distribution for $\mathbf{y}_*$ given $\mathbf{y}$ is

$$\begin{aligned} p(\mathbf{y}_* \mid \mathbf{y}) &= \frac{p(\mathbf{y}, \mathbf{y}_*)}{p(\mathbf{y})} && (5.1) \\ &\propto \exp\left(-\frac{1}{2} [\mathbf{y} - \boldsymbol{\mu}, \mathbf{y}_* - \boldsymbol{\mu}_*] \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma}_* \\ \boldsymbol{\Sigma}_*^\top & \boldsymbol{\Sigma}_{**} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{y} - \boldsymbol{\mu} \\ \mathbf{y}_* - \boldsymbol{\mu}_* \end{bmatrix} \right) && (5.2) \end{aligned}$$

To get the inverse of covariance matrix, we need to diagonalize it,

$$\begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma}_* \\ \boldsymbol{\Sigma}_*^\top & \boldsymbol{\Sigma}_{**} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} & -\boldsymbol{\Sigma}_* \boldsymbol{\Sigma}_{**}^{-1} \\ 0 & \boldsymbol{I} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{\Sigma} - \boldsymbol{\Sigma}_* \boldsymbol{\Sigma}_{**}^{-1} \boldsymbol{\Sigma}_*^\top & 0 \\ 0 & \boldsymbol{\Sigma}_{**} \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & 0 \\ -\boldsymbol{\Sigma}_{**}^{-1} \boldsymbol{\Sigma}_*^\top & \boldsymbol{I} \end{bmatrix}^{-1} \quad (5.3)$$

Then, we get the inverse of the covariance matrix,

$$\begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma}_* \\ \boldsymbol{\Sigma}_*^\top & \boldsymbol{\Sigma}_{**} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{I} & 0 \\ -\boldsymbol{\Sigma}_{**}^{-1} \boldsymbol{\Sigma}_*^\top & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} (\boldsymbol{\Sigma} - \boldsymbol{\Sigma}_* \boldsymbol{\Sigma}_{**}^{-1} \boldsymbol{\Sigma}_*^\top)^{-1} & 0 \\ 0 & \boldsymbol{\Sigma}_{**}^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & -\boldsymbol{\Sigma}_* \boldsymbol{\Sigma}_{**}^{-1} \\ 0 & \boldsymbol{I} \end{bmatrix} \quad (5.4)$$

Plug it back into the conditional distribution, we get

This can be viewed as the product of two Gaussians.

$$\begin{aligned} & [\mathbf{y} - \boldsymbol{\mu}, \mathbf{y}_* - \boldsymbol{\mu}_*] \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma}_* \\ \boldsymbol{\Sigma}_*^\top & \boldsymbol{\Sigma}_{**} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{y} - \boldsymbol{\mu} \\ \mathbf{y}_* - \boldsymbol{\mu}_* \end{bmatrix} && (5.5) \\ &= \left(\mathbf{y} - \boldsymbol{\mu}_1 - \boldsymbol{\Sigma}_* \boldsymbol{\Sigma}_{**}^{-1} (\mathbf{y}_* - \boldsymbol{\mu}_2) \right)^\top \left(\boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_* \boldsymbol{\Sigma}_{**}^{-1} \boldsymbol{\Sigma}_{21} \right)^{-1} \left(\mathbf{y} - \boldsymbol{\mu}_1 - \boldsymbol{\Sigma}_* \boldsymbol{\Sigma}_{**}^{-1} (\mathbf{y}_* - \boldsymbol{\mu}_2) \right) && (5.6) \\ &\quad + (\mathbf{y}_* - \boldsymbol{\mu}_2)^\top \boldsymbol{\Sigma}_{**}^{-1} (\mathbf{y}_* - \boldsymbol{\mu}_2) && (5.7) \end{aligned}$$

Therefore, the conditional distribution is

$$p(\mathbf{y}_* \mid \mathbf{y}) = \mathcal{N}(\boldsymbol{\mu} + \boldsymbol{\Sigma}_* \boldsymbol{\Sigma}_{**}^{-1}(\mathbf{y}_* - \boldsymbol{\mu}_*), \boldsymbol{\Sigma} - \boldsymbol{\Sigma}_* \boldsymbol{\Sigma}_{**}^{-1} \boldsymbol{\Sigma}_*^\top) \tag{5.8}$$

■

5.1.1 Prediction using Gaussian Processes

Definition 5.1.2 (Gaussian Process) *A Gaussian Process (GP) is a (potentially infinite) collection of random variables (RV) such that the joint distribution of every finite subset of RVs is multivariate Gaussian:*

$$f \sim \mathrm{GP}(\mu(\mathbf{x}), K(\mathbf{x}, \mathbf{x}'))$$

where $\mu(\mathbf{x}) : \mathcal{X} \to \mathcal{X}$ *and* $K(\mathbf{x}, \mathbf{x}') : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *are the mean and covariance function, respectively.*

Now, in order to model the predictive distribution $P(\mathbf{f}_* \mid \mathbf{x}_*, D)$ we can use a Bayesian approach by using a GP prior: $p(\mathbf{f} \mid \mathbf{x}) \sim \mathcal{N}(\mu, \boldsymbol{\Sigma})$ and condition it on the training data D to model the joint distribution of $\mathbf{f} = f(\mathbf{x})$ (vector of training observations) and $\mathbf{f}_* = f(\mathbf{x}_*)$ (prediction at test input).

Definition 5.1.3 (Predictive Distribution under Noise) *Let* $y_i = f_i + \epsilon_i$, *where* $\mathbb{E}[y_i] = 0$ *and* $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$. *The predictive density* $p(y_{n+1} \mid \mathbf{x}_{n+1}, \mathbf{X}, \mathbf{y})$ *for a new data point* $\mathbf{x}_{n+1}$ *can be obtained analytically in Gaussian Process by deriving the joint distribution*

$$p\left(\left[\begin{array}{c} \mathbf{y} \\ y_{n+1} \end{array}\right] \middle| \mathbf{x}_{n+1}, \mathbf{X}, \sigma\right) = \mathcal{N}\left(\left[\begin{array}{c} \mathbf{y} \\ y_{n+1} \end{array}\right] \middle| \mathbf{0}, \left[\begin{array}{cc} \mathbf{K} + \sigma^2 & \mathbf{k}_* \\ \mathbf{k}_*^\top & k + \sigma^2 \end{array}\right]\right)$$

where

$$\mathbf{K} = [K(\mathbf{x}_i, \mathbf{x}_j)]_{i=j=1}^n, \quad \mathbf{k}_* = [K(\mathbf{x}_{n+1}, \mathbf{x}_i)]_{i=1}^n, \quad k = K(\mathbf{x}_{n+1}, \mathbf{x}_{n+1}).$$

Using the conditional distribution of Gaussian, we have the closed-form solution of it,

$$p(y_{n+1} \mid \mathbf{x}_{n+1}, \mathbf{X}, \mathbf{y}) = \mathcal{N}(y_{n+1} \mid \underbrace{\mathbf{k}^\top(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y}}_{\text{mean follows observations}}, \underbrace{k - \mathbf{k}^\top(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{k}}_{\text{variance shrinks given more data}}). \tag{5.9}$$

Property 5.1.3 (Model Selection in GP) *A benefit of Gaussian Processes is that we can compute the marginal likelihood of the data given a model. This gives us a principled way of comparing different models.*

5.1.2 Kernels in Gaussian Processes

Fig. 5.1 and 5.2 give intuitive examples of the behavior of the kernels in GP. [1]

[1] Figures are taken from D. K. Duvenaud's Thesis: Automatic Model Construction with Gaussian Processes, Cambridge, 2014

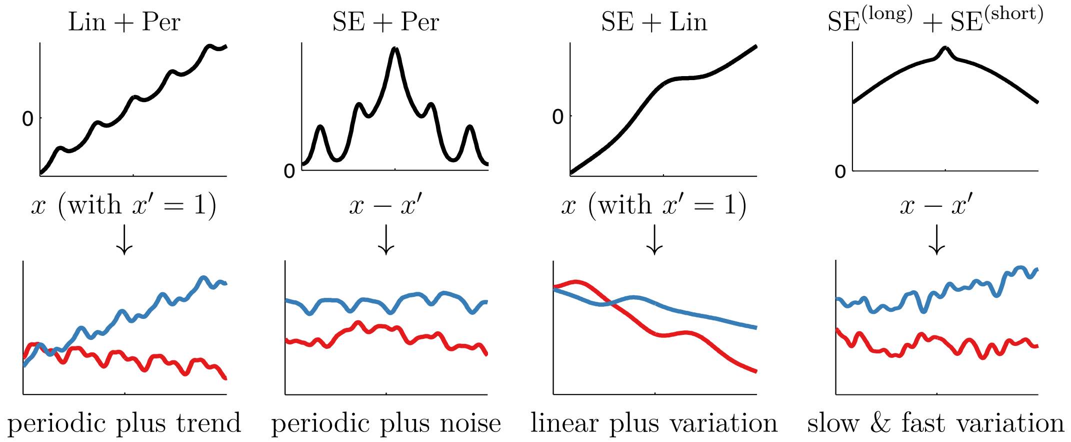


Figure 5.1: Sum of different kernels

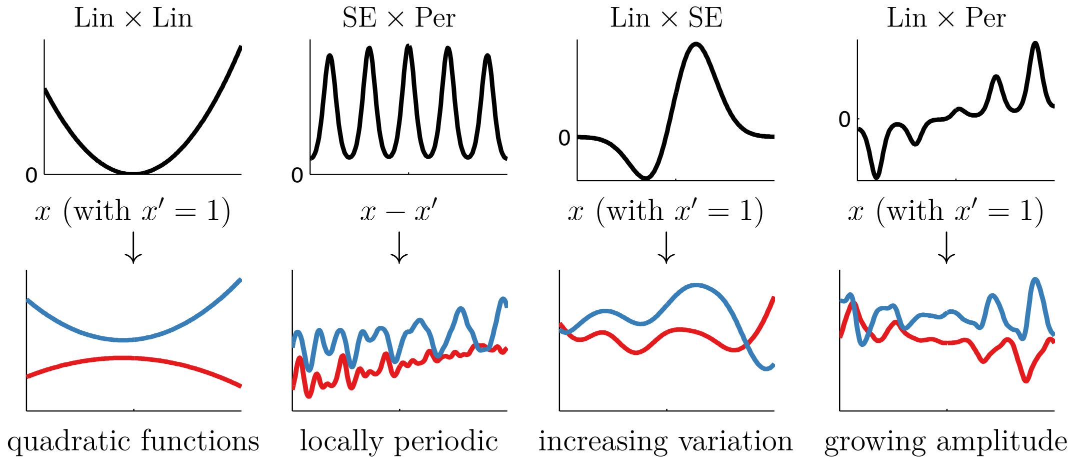


Figure 5.2: Product of different kernels

Chapter 6

Ensamble Methods

(x, y)	a random example
Z	a random training set
$Z_1, \cdots Z_M$	bootstrap sets from Z
b	a base model trained on a bootstrap set
$b^{(M)}$	an ensemble model trained via bagging on M bootstrap sets

6.1 Bagging

In bagging methods, each classifier trained on a different bootstrap sample of the training data, and the final predictions given by the majority voting.

Definition 6.1.1 (Bagging) *Bagging = Bootstrap + Aggregation. For a training set Z,*

1. *Draw M bootstrap datasets, $Z_1, \cdots Z_M$, where $Z_i \subset Z$;*
2. *Train M base models, $b^{(1)}, \cdots, b^{(M)}$, independently;*
3. *Aggregate base models with*

$$\hat{b}(\mathbf{x}) = \begin{cases} \frac{1}{M} \sum_{t \leq M} b^{(t)}(\mathbf{x}) & \text{regression} \\ \text{vote}\left\{b^{(1)}, \cdots, b^{(M)}\right\} & \text{classification} \end{cases}$$

Theorem 6.1.1 (Variance Reducing) *For $\mathbf{x} \in X$, if $range(y) := \max y - \min y < \infty$, then there is a sufficiently large M s.t.*

$$\mathbb{E}_{Z,Z'_1,\cdots,Z'_M,y|\mathrm{x}}\left[(y - b^{(M)}(\mathbf{x}))^2\right] \leq \mathbb{E}_{Z,Z',y|\mathrm{x}}\left[(y - b(\mathbf{x}))^2\right] \tag{6.1}$$

Proof Sketch.

See lecture slide 9.

$$\begin{aligned} \mathbb{E}_{Z,Z',y|\mathrm{x}}\left[(y - b(\mathbf{x}))^2\right] &= \mathbb{E}_{Z,Z',y|\mathrm{x}}\left[(y - \mathbb{E}_{Z,Z'}\left[b(\mathbf{x})\right] + \mathbb{E}_{Z,Z'}\left[b(\mathbf{x})\right] - b(\mathbf{x}))^2\right] && (6.2) \\ &= \mathbb{E}_{Z,Z',y|\mathrm{x}}\left[(y - \mathbb{E}_{Z,Z'}\left[b(\mathbf{x})\right])^2\right] + \underbrace{\mathbb{E}_{Z,Z'}\left[(\mathbb{E}_{Z,Z'}\left[b(\mathbf{x})\right] - b(\mathbf{x})^2\right]}_{Var[b(\mathrm{x})] \geq 0} && (6.3) \\ &\geq \mathbb{E}_{Z,Z'_1,\cdots,Z'_M,y|\mathrm{x}}\left[(y - \mathbb{E}_{Z,Z'}\left[\hat{b}(\mathbf{x})\right])^2\right] && (6.4) \end{aligned}$$

■

Remark. The main differences between bagging and boosting:

- Bagging builds learners independently, while boosting tries to add new models that do well where previous models fail.

- Bagging uniformly samples data, while boosting samples data based on weights.
- Bagging uses majority voting, while boosting uses weighted voting (more weight to those with better performance on training data).
- Boosting tries to reduce both bias and variance; while bagging only solve the over-fitting problem (high variance).

6.2 Boosting

6.2.1 AdaBoost

- AdaBoost = Adaptive Boosting
- Sequentially add *week* predictors to the ensemble, where each new predictor **improves** its predecessor by paying more attention to the hard cases.
- Final prediction is made via a weighted majority voting scheme.

Given the training samples $\{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)\} \in \mathcal{X} \times \{-1, +1\}$, and the base model $b(\mathbf{x}) : \mathcal{X} \to \{-1, +1\}$, the AdaBoost algorithm is shown in Alg. 1.

Algorithm 1: AdaBoost method (from Slides 6 & Toturial 8)

```
$w_i^{(t)} = 1/n$ for $i = 1, \cdots, n$;
for $t = 1, \cdots, M$ do
    $b_t(\mathbf{x}) = \operatorname*{argmin}_b \sum_{i \leq n} w_i^{(t)} \mathbb{1}\{ y_i \neq b(\mathbf{x}_i)\}$;
    $\epsilon_t = \left( \sum_{i \leq n} w_i^{(t)} \mathbb{1}\{ y_i \neq b_t(x_i)\} \right) \Big/ \left( \sum_{i \leq n} w_i^{(t)} \right)$;    ▷ weighted error rate
    $\alpha_t = \log \frac{1 - \epsilon_t}{\epsilon_t}$;    ▷ correct and wrong samples both get half of the weights
    $w_i^{(t+1)} = w_i^{(t)} \exp(\alpha_t \mathbb{1}\{ y_i \neq b_t(\mathbf{x}_i)\})$;
end
Output: $\widehat{b}(x) = \operatorname{sign}\left( \sum_{t \leq M} \alpha_t b_t(\mathbf{x}) \right), \forall \mathbf{x} \in \mathcal{X}$
```

Note that there is another version of AdaBoost typically found in literatures. The difference lies in how to handle correct samples. In the previous version, correct samples will have unchanged weights; whereas in the second version, they will have weights of $\exp(-\alpha_t)$.

Algorithm 2: AdaBoost method (another version)

```
1 $w_i^{(t)} = 1/n$ for $i = 1, \cdots, n$;
2 for $t = 1, \cdots, M$ do
3     $b_t(\mathbf{x}) = \operatorname*{argmin}_b \sum_{i \le n} w_i^{(t)} \mathbb{1}\{y_i \neq b(\mathbf{x}_i)\}$;
4     $\epsilon_t = \sum_{i \le n} w_i^{(t)} \mathbb{1}\{y_i \neq b_t(x_i)\}$;    ▷ weighted error rate
5     $\alpha_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}$;    ▷ note the additional 1/2
6     $Z_t = 2\sqrt{\epsilon_t(1 - \epsilon_t)}$;    ▷ normalizer
7     $w_i^{(t+1)} = w_i^{(t)} \exp(-\alpha_t y_i b_t(\mathbf{x}_i)) / Z_t$;
8 end
  Output: $\widehat{b}(x) = \text{sign}\left(\sum_{t \le M} \alpha_t b_t(\mathbf{x})\right), \forall \mathbf{x} \in \mathcal{X}$
```

Remark. Here we only consider binary classification problem, where the base models are assumed to have accuracy higher than 50%. Thus, Line 5 will not decrease the weight of mis-classified samples.

Theorem 6.2.1 (Empirical Error of AdaBoost) *The empirical error of the classifier returned by AdaBoost verifies*

$$\widehat{\text{Err}}(b) \le \exp\left[-2 \sum_{t \le M} (\frac{1}{2} - \epsilon_t)^2\right].$$

Proof.

$$\widehat{\text{Err}}(b) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{y_i \neq b(\mathbf{x}_i)\} \le \frac{1}{n} \sum_{i=1}^{n} e^{-y_i b(\mathbf{x}_i)} = \frac{1}{n} \sum_{i=1}^{n} \left[n \prod_{t=1}^{M} Z_t\right] w_i^{(M+1)} = \prod_{t=1}^{M} Z_t. \quad (6.5)$$

■

6.2.2 Gradient Boosting

Gradient Boosting is yet another popular and efficient approach. It uses gradients (or residuals) instead of sample weights. The final predictions are made from weighted sum of weak models. It works for arbitrary differentiable loss functions $L(y, f(\mathbf{x}))$.

Algorithm 3: Gradient Boosting

```
for $t = 1, \cdots, M$ do
    Compute gradient: $g_t(\mathbf{x}_i) = -\left[\frac{\partial L(y_i; f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)}\right]_{f = \widehat{f}_{t-1}} \quad \forall i \le n$;
    Fit weak learner to the gradients: $h_t = \operatorname*{argmin}_h \sum_{i \le n} (-g_t(\mathbf{x}_i) - h(\mathbf{x}_i))^2$;
    Fit the step length via line search: $\beta_t = \operatorname*{argmin}_\beta \sum_{i \le n} L(y_i, \widehat{f}_{t-1}(\mathbf{x}_i) + \beta h_t(\mathbf{x}_i))$;
    Update $\widehat{f}_t(\mathbf{x}) = \widehat{f}_{t-1}(\mathbf{x}) + \beta_t h_t(\mathbf{x})$;
end
Output: $\widehat{f}_M(\mathbf{x})$
```

6.2.3 Theoretical Insights

Forward Stage-wise Additive Estimator with the Exponential Loss

Let $L(y, y') = \exp(-yy')$ be an exponential loss, then AdaBoost can be translated into the following form.

Algorithm 4: AdaBoost (FSAE explanation)

```
$f_0(\mathbf{x}) = 0$, for all $\mathbf{x} \in \mathbb{R}^D$;
for $t = 1, \cdots, M$ do
    $(\alpha_t, b^{(t)}) = \underset{\alpha,b}{\text{argmin}} \sum_{i=1}^{n} L(y_i, \alpha b(\mathbf{x}_i), +f_{t-1}(\mathbf{x}_i))$;
    $f_t(\mathbf{x}) = \alpha_t b^{(t)}(\mathbf{x}_i) + f_{t-1}(\mathbf{x}_i))$for all $\mathbf{x} \in \mathbb{R}^D$;
end
Output: $\widehat{f}_M(\mathbf{x})$
```

The proof of the equivalence can be found in Ex.6 - 1.

AdaBoost trains max-margin classifiers

$$b^{(M)}(\mathbf{x}) = \text{sign}\left(\sum_{t \leq M} \alpha_t b^{(t)}(\mathbf{x})\right) \qquad w.l.o.g. \sum_{t \leq M} \alpha_t = 1 \tag{6.6}$$

$$margin(\mathbf{x}_i) := y_i \sum_{t \leq M} \alpha_t b^{(t)}(\mathbf{x}_i) \longrightarrow 0 \quad (M \to \infty) \tag{6.7}$$

Chapter 7

Non-parametric Methods

7.1 Expectation Maximization (EM)

The Expectation Maximization (EM) algorithm is one approach to unsupervised, semi-supervised, or lightly supervised learning. Given a model p (parameterized by θ) of observable variables $\mathbf{x}_1, \cdots, \mathbf{x}_n$ and latent variables $\mathbf{z}_1, \cdots, \mathbf{z}_m$, EM is an efficient method that **approximately** solves the following optimization problem,

$$\theta^* = \operatorname*{argmax}_{\theta} \max_{\mathrm{z}_1, \cdots, \mathrm{z}_n} p(\mathbf{x}_1, \cdots, \mathbf{x}_n, \mathbf{z}_1, \cdots, \mathbf{z}_m \mid \theta) \tag{7.1}$$

Definition 7.1.1 (EM Principle) *Let $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ be the observable variables, $\mathcal{X}_L = \{\mathbf{z}_1, \cdots, \mathbf{z}_m\}$ be the latent variables. The EM method is to repeat the following procedure until the convergence of θ.*

1. **Expectation step**: *Calculate a function Q of θ, given previous-estimated $\theta^{(j)}$,*
$$Q\left(\theta; \theta^{(j)}\right) := \mathbb{E}_{\mathcal{X}_L}\left[L\left(\mathcal{X}, \mathcal{X}_L \mid \theta\right) \mid \mathcal{X}, \theta^{(j)}\right]$$
Here, $L(\mathcal{X}, \mathcal{X}_L \mid \theta) = \log p(\mathcal{X}, \mathcal{X}_L \mid \theta)$ is the log-likelihood function.
2. **Maximization step**: *Estimate new parameter $\theta^{(j+1)}$ by maximizing the function Q,*
$$\theta^{(j+1)} = \operatorname*{argmax}_{\theta} Q\left(\theta; \theta^{(j)}\right).$$

Intuitively, EM is to optimize two group of coordinates alternatively. When optimizing one group, the other group is fixed. This is called "coordinates descent".

Theorem 7.1.1 (EM's Correctness) *EM method is guaranteed to converge to a point with gradient of* 0.

Proof Sketch.

7.1.1 K-Means Clustering

Definition 7.1.2 (K-Means Problem) *Given d-dimensional sample vectors $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ and an assignment function $c(\mathbf{x}) : \mathbb{R}^d \to \{1, \cdots, k\}$ with prototypes $\boldsymbol{\mu}_{c(\cdot)} \in \mathcal{Y} \subset \mathbb{R}^d$. The k-means finds the $c(\cdot)$ and $\mathcal{Y}$ that minimize*

$$R(c, \mathcal{Y}) = \sum_{\mathrm{x} \in \mathcal{C}} \left\| \mathbf{x} - \boldsymbol{\mu}_{c(\mathrm{x})} \right\|_2^2.$$

The k-means problem is a mixture problem of combinatorial and continuous optimization, which is hard to optimize directly. Practically, K-Means relies on the **Hard-EM** technique, as shown in Alg. 5.

Algorithm 5: K-Means, an example of Hard-EM

```
while $c(\mathbf{x})$ and $\boldsymbol{\mu}_c$ keep changing do
    $c(\mathbf{x}) = \operatorname{argmin}_{c \in \{1, \cdots, k\}} \|\mathbf{x} - \boldsymbol{\mu}_c\|_2^2$;   ▷ E-step: assign x to the nearest prototypes
    $\boldsymbol{\mu}_\alpha = \frac{1}{|N_\alpha|} \sum_{\mathrm{x} \in N_\alpha} \mathbf{x}$, where $N_\alpha = \{\mathbf{x} \mid c(\mathbf{x}) = \alpha\}$ ;   ▷ M-step: update prototypes
end
Output: $c(\mathbf{x}), \forall \mathbf{x} \in \mathcal{X}$
```

7.1.2 Gaussian Mixture Models (GMM)

In mixture models, data are assumed to be generated by a mixture of distribution $p(\mathbf{x} \mid \theta)$. Mixture models are generative, which tries to describe all the data.

Definition 7.1.3 (GMM) *A Gaussian mixture is a convex combination of k Gaussian distributions,*

$$p\left(\mathbf{x} \mid \pi_1, \ldots, \pi_k, \theta_1, \ldots, \theta_k\right) = \sum_{c \leq k} \pi_c p\left(\mathbf{x} \mid \theta_c\right), \quad \textit{where} \;\; p(\mathbf{x} \mid \theta_c) = \mathcal{N}(\mathbf{x}; \mu_c, \Sigma_c).$$

Here, $\pi_c > 0$ is the mixture weight, denoting the prior probability that a sample is generated by the mixture Gaussian component c with parameters $\theta_c = \{\mu_c, \Sigma_c\}$.

To estimate parameters θ_c, we can maximize its likelihood of sample feature vectors $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$. The log-likelihood is often computationally preferable, as

$$L(\mathcal{X}; \pi_1, \ldots, \pi_k, \theta_1, \ldots, \theta_k) = \sum_{\mathrm{x} \in \mathcal{X}} \log \sum_{c \leq k} \pi_c p(\mathbf{x} \mid \theta_c). \tag{7.2}$$

However, to directly maximize it is still intractable, because the logarithm within the sum makes it a non-convex problem. Alg. 6 shows the Soft-EM method that solves the GMM problem.

Algorithm 6: GMM, an example of Soft-EM

```
while $c(\mathbf{x})$ and $\boldsymbol{\mu}_c$ keep changing do
    $\gamma_{\mathrm{x},c} = \frac{p(\mathbf{x} \mid c, \theta^{(j)}) \; p(c \mid \theta^{(j)})}{p(\mathbf{x} \mid \theta^{(j)})}$;    ▷ E-step: soft-assign sample $\mathbf{x}$ to clusters
    $\boldsymbol{\mu}_c^{(j+1)} = \frac{\sum_{\mathrm{x}} \gamma_{\mathrm{x},c} \mathbf{x}}{\sum_{\mathrm{x}} \gamma_{\mathrm{x},c}}$, $\Sigma_c^{(j+1)} = \frac{\sum_{\mathrm{x}} \gamma_{\mathrm{x},c} (\mathbf{x} - \boldsymbol{\mu}_c)^2}{\sum_{\mathrm{x}} \gamma_{\mathrm{x},c}}$;    ▷ M-step: update parameters
    $\pi_c^{(j+1)} = \frac{1}{|\mathcal{X}|} \sum_{\mathrm{x}} \gamma_{\mathrm{x},c}$;
end
Output: $c(\mathbf{x}), \forall \mathbf{x} \in \mathcal{X}$
```

Here, $\gamma_{\mathrm{x},c}$ is the "responsible probability", denoting the probability that $\mathbf{x}$ belongs to component c, i.e. $\gamma_{\mathrm{x},c} = p(y = c \mid \mathbf{x}, \theta)$.

Next, we will proof the correctness of *Proof.*

■

7.2 Dirichlet Process

Definition 7.2.1 (Beta Distribution)

$$\mathrm{Beta}(\theta; a, b) = \frac{1}{\beta(a, b)} \theta^{a-1} (1 - \theta)^{b-1}, \quad \textit{where} \;\; \beta(a, b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a + b)}$$

Here, $\Gamma(z)$ is the Gamma function, $\Gamma(z) = \int_0^\infty x^{z-1} e^{-x} \, \mathrm{d}x$.

Remark. Beta distribution is the probability θ of a Bernoulli process after observing $a - 1$ successes and $b - 1$ failures.

Please note that the Binomial distribution $\mathrm{Bin}(k; n, p) = C(n, k) p^k (1-p)^{n-k}$ is a function of number of success trials $k \in \mathbb{N}$, while the Beta distribution $\mathrm{Beta}(\theta; a, b)$ is a function of the success probability $\theta \in [0, 1]$.

Definition 7.2.2 (Dirichlet Distribution) *Dirichlet distribution is the multivariate generalization of the beta distribution. Given* $\mathbf{x} = \{x_1, \cdots, x_n\}$ $(x_i \in [0,1])$ *and* $\boldsymbol{\alpha} = \{\alpha_1, \cdots, \alpha_n\}$ $(\alpha_i > 0)$*,*

$$\mathrm{Dir}(\mathbf{x}; \boldsymbol{\alpha}) = \frac{1}{\beta(\boldsymbol{\alpha})} \prod_{k \leq n} x_k^{\alpha_k - 1}, \quad \textit{where } \beta(\boldsymbol{\alpha}) = \frac{\prod_{k \leq n} \Gamma(\alpha_k)}{\Gamma(\sum_{k \leq n} \alpha_k)}$$

Definition 7.2.3 (Dirichlet Process) *A Dirichlet Process* $DP(\alpha, H)$ *is a stochastic process whose sample paths are probability distributions on a space* Θ*. Here,* $\alpha > 0$ *is the "concentration parameter", and* H *is the base measure on* Θ*. For any measurable finite partition of* Θ*, denoted by* $\{B_1, \cdots, B_n\}$*, if* $G \sim DP(\alpha, H)$*, then*

$$(G(B_1), \cdots, G(B_n)) \sim \mathrm{Dir}(\alpha H(B_1), \cdots, \alpha H(B_n))$$

Remark. Note that H is continuous, so the probability that any two samples are equal is precisely zero. However, G is a discrete distribution, made up of a countably infinite number of point masses, and thus there is a non-zero probability of two samples colliding.

Theorem 7.2.1 (De Finetti's Theorem) *Let* $(X_1, X_2, \cdots)$ *be an infinitely exchangeable sequence of random variables. Then,* $\forall n$*:*

$$p(X_1, \cdots, X_n) = \int (\prod_i p(x_i \mid G)) \, \mathrm{d}P(G),$$

for some random variable G*.*

Remark. An infinitely exchangeable sequence can be represented by a product of conditionally independent random variables.

7.2.1 Stick-Breaking Process

Stick-breaking process provides a constructive way to draw samples from $DP(\alpha, H)$.

Algorithm 7: Stick-Breaking Process

```
for $k = 1, 2 \cdots$ do
    $\beta_k \sim \mathrm{Beta}(1, \alpha)$;
    $\rho_k = \beta_k \prod_{i=1}^{k-1}(1 - \beta_i)$ ;        ▷ alternatively, $\rho_k = \beta_k(1 - \sum_{i=1}^{k-1} \rho_i)$
    $\theta_k \sim H$;
end
Output: $G(\theta) = \sum_{k=1}^{\infty} \rho_k \delta_{\theta_k}(\theta)$
```

Here, $\delta_t(\theta)$ is the Dirac function.

Definition 7.2.4 (GEM Distribution) *The* $\{\rho_1, \rho_2, \cdots\}$*'s distribution in the Stick-breaking process is called GEM distribution, named after Griffiths-Engen-McCloskey.*

$$\{\rho_1, \rho_2, \cdots\} \sim \mathrm{GEM}(\alpha) \quad \Longrightarrow \quad \forall k, \rho_k = \beta_k(1 - \sum_{i=1}^{k-1} \rho_i), \quad \textit{where } \beta_k \sim \mathrm{Beta}(1, \alpha).$$

7.2.2 Chinese Restaurant Process (CRP)

The Chinese restaurant process (CRP) is a metaphor of Dirichlet process.

Definition 7.2.5 (CRP) *Let $\mathcal{P} = \{\tau_1, \cdots, \tau_k\}$ denote a k-partition over the integers $\{1, \cdots, n\}$. The partition $\mathcal{P}$ represents the table assignment, i.e., $|\tau_i|$ is the number of people sitting at table i.*

When a new person arrives, he can either join an existing table i $(1 \leq i \leq k)$ with probability proportional to $|\tau_i|$, or start a new table with probability proportional to α. More formally,

$$p(n+1 \textit{ joins table } \tau \mid \mathcal{P}) = \begin{cases} \dfrac{|\tau|}{\alpha + n} & \tau \in \mathcal{P}, \qquad\qquad (\textit{to join table } i) \\ \dfrac{\alpha}{\alpha + n} & \tau \notin P. \qquad (\textit{to start a new table}) \end{cases}$$

Remark The larger the α is, the greater the number of clusters (tables) is.

Property 7.2.1 (CRP is Exchangeable) *No matter in which order people come, the probability of a given partition $\mathcal{P}$ is the same, i.e.,*

$$p(\mathcal{P}) = \frac{\alpha^k}{\alpha(\alpha+1)\cdots(\alpha+n-1)} \prod_{1 \leq i \leq k} (|\tau_i| - 1)! \tag{7.3}$$

Property 7.2.2 (Number of Occupied Tables in CRP) *The number of occupied tables in CRP after N customers is*

$$S(N) = \sum_{1 \leq i \leq N} \frac{\alpha}{\alpha + i - 1} \sim \mathcal{O}(\alpha \log N). \tag{7.4}$$

Proof. Let $\mathcal{P}_i$ be the partition after i customers. The The number of occupied tables after N customers can be written as,

$$\begin{aligned} S(N) &= \mathbb{E}\left[\sum_{1 \leq i \leq N} \mathbb{1}\{\tau_i \notin \mathcal{P}_i\}\right] && (7.5) \\ &= \sum_{1 \leq i \leq N} \mathbb{E}\left[\, \mathbb{1}\{\tau_i \notin \mathcal{P}_i\}\right] && (7.6) \\ &= \sum_{1 \leq i \leq N} p(\text{start a new table} \mid n = i) && (7.7) \\ &= \sum_{1 \leq i \leq N} \frac{\alpha}{\alpha + i - 1} && (7.8) \end{aligned}$$

■

Exercise 7.2.1 (Ex.8-2: Dirichlet Process) *Consider the following algorithm for sampling from the Dirichlet process with base distribution F_0 and concentration parameter α.*

1. *Draw the first sample $X_1 \sim F_0$.*
2. *For $i = 2, 3, \cdots$, draw*

$$X_i \mid Z_1, \ldots, X_{i-1} = \begin{cases} X \sim \widehat{F}_{i-1}, & \textit{with probability } p = \frac{i-1}{\alpha+i-1} \\ X \sim F_0, & \textit{with probability } p = \frac{\alpha}{\alpha+i-1} \end{cases} \tag{7.9}$$

where $\widehat{F}_{i-1}$ is the empirical distribution of $X_1, \cdots, X_{i-1}$.

Find the asymptotics of the expected number of distinct samples drawn, as a function of the total number of samples drawn: $X_1, \cdots, X_n$. Or equivalently, the number of occupied tables in the Chinese restaurant process metaphor.

Solution Note that the base distribution F_0 is a continuous, so the probability of sampling a sample X' that equal to any number of finite sample $X_1, \cdots X_i$ is 0. The distinct sample after n samples drawn,

$$S(n) = \mathbb{E}\left[\sum_{i \leq n} \mathbb{1}\{x_i \cup \{X_1, \cdots, X_{i-1}\} = \phi\}\right] \tag{7.10}$$

$$= \sum_{i \leq n} p(X_i \sim F_0)\ p(x_i \cup \{X_1, \cdots, X_{i-1}\} = \phi \mid X_k \sim F_0) \tag{7.11}$$

$$= \sum_{i \leq n} \frac{\alpha}{\alpha + i - 1} \tag{7.12}$$

Then, it is easy to prove that $S(n) \sim O(\alpha \log n)$.

■

7.2.3 DP Mixture Model

Definition 7.2.6 *Let Θ be a set that parameterizes a set of probability distributions, and fix a base measure H on Θ. Here, we assume that $\Theta = \mathbb{R}$ and $H = \mathcal{N}(\mu_0, \sigma_0)$ for some fixed $\mu_0 \in \mathbb{R}, \sigma_0 \in \mathbb{R}_+$. The DP Mixture model is defined as*

- *Probabilities of clusters ("mixture weights"): $\boldsymbol{\rho} = (\rho_1, \rho_2, \cdots) \sim GEM(\alpha)$,*
- *Centers of the clusters: $\mu_k \sim \mathcal{N}(\mu_0, \sigma_0),\ k = 1, 2, \cdots,$*
- *Assignments of data points to clusters: $z_i \sim \text{Categorical}(\rho),\ i = 1, 2, \cdots$*

7.2.4 Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA) is one of the most popular non-parametric model.

Definition 7.2.7 *Given K topics and V words in the vocabulary, for M documents with N words each.*

- *Distribution of topics in document d: $\boldsymbol{\theta}_d \sim \text{Dir}(\boldsymbol{\alpha})$;*
- *What topic the word w belongs to in document d: $z_{d,w} \sim \text{Categorical}(\boldsymbol{\theta}_d)$;*
- *Distribution of words in topic k: $\boldsymbol{\psi}_k \sim \text{Dir}(\boldsymbol{\beta})$;*
- *What word w in document d: $w_d \sim \text{Categorical}(\boldsymbol{\psi}_{z_{d,w}})$*

7.3 Sampling Methods

7.3.1 Markov-chain Monte Carlo (MCMC)

Markov-chain Monte Carlo (MCMC) is a powerful framework, allowing sampling from a large class of distributions.

7.3.2 Gibbs Sampling

However, the converge of MCMC is typically slow. Gibbs sampling is a simple and faster method that samples one random variable from conditional distribution at a time, while the

Algorithm 8: MCMC Sampling

```
for $\tau = 1, 2 \cdots$ do
    $\mathbf{z}^* \sim p(\mathbf{z} \mid \mathbf{z}^{(\tau)})$;                          ▷ proposal distribution
    $\alpha \sim \text{Uniform}(0, 1)$;                                ▷ acceptance threshold
    if $A(\mathbf{z}^*, \mathbf{z}^{(\tau)}) = p(\mathbf{z}^*)\ p(\mathbf{z}^{(\tau)} \mid \mathbf{z}^*) > \alpha$ then
        $\mathbf{z}^{(\tau+1)} = \mathbf{z}^*$
    else
        $\mathbf{z}^{(\tau+1)} = \mathbf{z}^{(\tau)}$
    end
end
Output: $\{\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \cdots\}$
```

remaining variables fixed to their current values. The theory of MCMC guarantees that the stationary distribution of the samples generated is the target joint posterior.

Algorithm 9: Gibbs Sampling

```
Initialize: $\mathbf{z}^{(0)} \sim q(\mathbf{z})$;
for $\tau = 1, 2 \cdots$ do
    $\mathbf{z}_1^{(\tau)} \sim p\left(Z_1 = \mathbf{z}_1 \mid Z_2 = \mathbf{z}_2^{(\tau-1)}, Z_3 = \mathbf{z}_3^{(\tau-1)}, \cdots, Z_D = \mathbf{z}_D^{(\tau-1)}\right)$;
    $\mathbf{z}_2^{(\tau)} \sim p\left(Z_2 = \mathbf{z}_2 \mid Z_1 = \mathbf{z}_1^{(\tau)}, Z_3 = \mathbf{z}_3^{(\tau-1)}, \cdots, Z_D = \mathbf{z}_D^{(\tau-1)}\right)$;
    $\vdots$
    $\mathbf{z}_D^{(\tau)} \sim p\left(Z_D = \mathbf{z}_D \mid Z_1 = \mathbf{z}_2^{(\tau)}, Z_2 = \mathbf{z}_2^{(\tau)}, \cdots, Z_{D-1} = \mathbf{z}_{D-1}^{(\tau)}\right)$;
end
Output: $\{\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \cdots\}$
```

Property 7.3.1 (Gibbs Samples is a Special Case of Metropolis-Hastings) *We can view the Gibbs sampling as a special case of Metropolis-Hastings, where the acceptance of Gibbs' samples is always* 1.

Proof. Let $\mathbf{z}_k$ denotes the the sample's projection at k-th dimension and $\mathbf{z}_{\setminus k}$ denotes the sample's projection at dimensions other than k. In Gibbs sampling, we have $q_k(\mathbf{z}^* \mid \mathbf{z}) = p(\mathbf{z}_k^* \mid \mathbf{z}_{\setminus k})$ (every time we sample one variable, and fix others). Therefore,

$$A\left(\mathbf{z}^*, \mathbf{z}\right) = \frac{p\left(\mathbf{z}^*\right) q_k\left(\mathbf{z} \mid \mathbf{z}^*\right)}{p(\mathbf{z}) q_k\left(\mathbf{z}^* \mid \mathbf{z}\right)} = \frac{p\left(z_k^* \mid \mathbf{z}_{\setminus k}^*\right) p\left(\mathbf{z}_{\setminus k}^*\right) p\left(z_k \mid \mathbf{z}_{\setminus k}^*\right)}{p\left(z_k \mid \mathbf{z}_{\setminus k}\right) p\left(\mathbf{z}_{\setminus k}\right) p\left(z_k^* \mid \mathbf{z}_{\setminus k}\right)} = 1 \tag{7.13}$$

■

Table 7.1: Summary of some useful sampling methods

Method	**Metropolis**	**Metropolis-Hastings**	**Gibbs**
Proposal dist.	$q(\mathbf{z} \mid \mathbf{z}^{(\tau)})$	$q(\mathbf{z} \mid \mathbf{z}^{(\tau)})$	$p(\mathbf{z}_k \mid \mathbf{z}_{\setminus k})$
Assumption	$q(\mathbf{z}_A \mid \mathbf{z}_B) = q(\mathbf{z}_B \mid \mathbf{z}_A)$	-	-
Accept prob. $A_k\left(\mathbf{z}^\star, \mathbf{z}^{(\tau)}\right)$	$\min\left\{1, \frac{\tilde{p}\left(\mathbf{z}^\star\right)}{\tilde{p}\left(\mathbf{z}^{(\tau)}\right)}\right\}$	$\min\left\{1, \frac{\tilde{p}\left(\mathbf{z}^\star\right) q_k\left(\mathbf{z}^{(\tau)} \mid \mathbf{z}^\star\right)}{\tilde{p}\left(\mathbf{z}^{(\tau)}\right) q_k\left(\mathbf{z}^\star \mid \mathbf{z}^{(\tau)}\right)}\right\}$	1

Chapter 8

Deep Learning

8.1 Neural Network

Definition 8.1.1 *A d-layer neural network (NN) is defined as the composition of the following components:*

- *Linear functions $L(\mathbf{x}) = \mathbf{W}\mathbf{x} + b : \mathbb{R}^n \to \mathbb{R}^m$, $\mathbf{W} \in \mathbb{R}^{n\times m}$, $b \in \mathbb{R}$;*
- *Activation function $\alpha(\mathbf{x}) : \mathbb{R}^m \to \mathbb{R}^m$.*

alternatively applied to the input data $\mathbf{x} \in \mathcal{X}$, i.e.,

$$NN(\mathbf{x}) = \alpha^{(d)} \circ L^{(d)} \circ \cdots \circ \alpha^{(1)} \circ L^{(1)}(\mathbf{x}).$$

Here, $\circ$ is the composition operator.

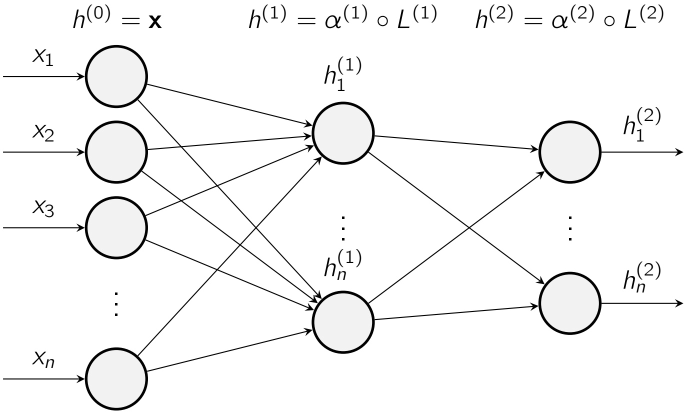


Figure 8.1: Illustration of a 2-layer neural network

8.1.1 Gradients in NN

(Omitted here. Please see the Ex. 6 for details.)

8.2 Variational Autoencoder

8.2.1 InfoMax Principle

Let X and Z be a measurement and a representation space, respectively. Let $F = \{\text{enc}_\theta : \theta \in \Theta\}$ be a parametric family of function with $\text{enc}_\theta : X \to Z$.

Definition 8.2.1 (Mutual Information) *The mutual information between X and Z is*

$$I(X; Z) = \mathbb{E}_{X,Z}\left[\log \frac{p(X, Z)}{p(X)p(Z)}\right]$$

Definition 8.2.2 (InfoMax Principle) *Given a training set* $\{x_1, \cdots, x_n\} \in X$*, we can do the following approximation*

$$\begin{aligned}\underset{\theta}{\operatorname{argmax}}\ I(X;Z) &= \underset{\theta}{\operatorname{argmax}}\ \mathbb{E}_{X,Z}\left[\log \frac{p(X,Z)}{p(X)p(Z)}\right] \\ &= \underset{\theta}{\operatorname{argmax}}\ \mathbb{E}_{X,Z}\left[\log p(X \mid Z)\right] - \mathbb{E}_X\left[\log p(X)\right] \\ &= \underset{\theta}{\operatorname{argmax}}\ \mathbb{E}_X \mathbb{E}_{Z|X}\left[\log p(X_i \mid Z)\right] \\ &\approx \underset{\theta}{\operatorname{argmax}}\ \sum_{i \le n} \mathbb{E}_{Z|X}\left[\log p(X_i \mid Z)\right]\end{aligned}$$

8.2.2 Variational Autoencoder

$$\underset{\theta,\theta',\phi}{\operatorname{argmax}}\ \sum_{i\le n} \log p_{\theta',\theta}(x_i) \tag{8.1}$$

$$= \underset{\theta,\theta',\phi}{\operatorname{argmax}}\ \mathbb{E}_{Z\sim q_\phi(\cdot|x_i)}[\log p_{\theta',\theta}(x_i)] \tag{8.2}$$

$$= \underset{\theta,\theta',\phi}{\operatorname{argmax}}\ \mathbb{E}_{Z\sim q_\phi(\cdot|x_i)}\left[\log\left(\frac{p_{\theta',\theta}(x_i,Z)}{p_{\theta',\theta}(x_i \mid Z)}\frac{q_\phi(Z \mid x_i)}{q_\phi(Z \mid x_i)}\right)\right] \tag{8.3}$$

$$= \underset{\theta,\theta',\phi}{\operatorname{argmax}}\ \underbrace{\mathbb{E}_{Z\sim q_\phi(\cdot|x_i)}\left[\log\left(\frac{p_{\theta',\theta}(x_i,Z)}{q_\phi(Z \mid x_i)}\right)\right]}_{\text{ELBO of } \theta',\theta,\phi} + \underbrace{\mathbb{E}_{Z\sim q_\phi(\cdot|x_i)}\left[\log\left(\frac{q_\phi(Z \mid x_i)}{p_{\theta',\theta}(Z \mid x_i)}\right)\right]}_{KL\left[q_\phi(\cdot|x_i)\ ||\ p_{\theta,\theta'}(\cdot|x_i)\right] \ge 0} \tag{8.4}$$

Note that $p_{\theta',\theta}(x_i, Z) = p_\theta(x_i \mid Z)\, p_{\theta'}(Z)$. we have

$$\text{ELBO} = \underbrace{\mathbb{E}_{Z\sim q_\phi(\cdot|x_i)}[\log p_\theta(x_i \mid Z)]}_{\text{Mutual Information: } I(X;Z)} + \underbrace{\mathbb{E}_{Z\sim q_\phi(\cdot|x_i)}\left[\log\left(\frac{p_{\theta'}(Z)}{q_\phi(Z \mid x_i)}\right)\right]}_{-KL\left[q_\phi(\cdot|x_i)\ ||\ p_{\theta'}\right]} \tag{8.5}$$

Remark. The Mutual Information term encourage the representation to be infomrative, and KL divergence term tries to disentangle the encoder and decoder.

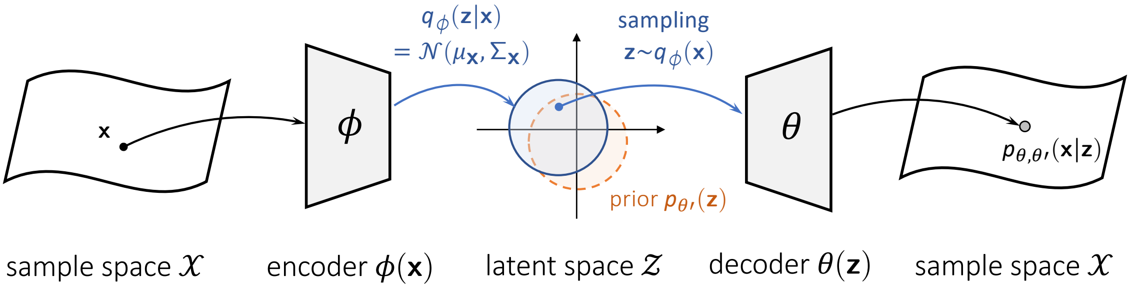

$q_\phi(\mathbf{z}|\mathbf{x})$
$= \mathcal{N}(\mu_\mathbf{x}, \Sigma_\mathbf{x})$
sampling
$\mathbf{z}\sim q_\phi(\mathbf{x})$
$\mathbf{x}$
ϕ
θ
$p_{\theta,\theta'}(\mathbf{x}|\mathbf{z})$
prior $p_{\theta'}(\mathbf{z})$
sample space $\mathcal{X}$
encoder $\phi(\mathbf{x})$
latent space $\mathcal{Z}$
decoder $\theta(\mathbf{z})$
sample space $\mathcal{X}$

Figure 8.2: Illustration for Variational Autoencoder

8.3 Optimization Methods

8.3.1 Newton's Method

Newton's method was originally created to find a root $f(\mathbf{x}^*) = 0$ of a given function $f(\mathbf{x})$ via the iteration.

Definition 8.3.1 (Newton's Method) *Given a function $f : \mathcal{X} \to \mathbb{R}$ and a initial value of $\mathbf{x}_0 \in \mathcal{X}$, Newton's method finds a root of $f(\mathbf{x}^*) = 0$ near $\mathbf{x}_0$ by iteration steps as*

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \frac{f(\mathbf{x}_n)}{f'(\mathbf{x}_n)},$$

where $f'(\mathbf{x})$ denotes the derivative of f with respect to $\mathbf{x}$.

In optimization we are usually interested in finding the minimum of a function. This can be achieved using Newton's method to find a root of the first derivative $f'(\mathbf{x}^*) = 0$, the optimization step then becomes

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \frac{f'(\mathbf{x}_n)}{f''(\mathbf{x}_n)}. \tag{8.6}$$

When applying in the high-dimensions, we have the form as

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \mathbf{H}_f^{-1}(\mathbf{x}_n)\nabla_f(\mathbf{x}_n) \tag{8.7}$$

Property 8.3.1 (Convergence of Newton's Method) *We call a smooth function f with one root $f(x^*) = 0$ has order k, if its all derivatives $f^{(i)}(x^*) = 0, \forall i > k$ and $f^{(k)}(x^*) \neq 0$. Newton's method converges linearly ($m = 1$) for the function f of order $k > 1$ in a region around the point x^*.*

A sequence x_n converges with order m towards x^*, if there exists a constant C, such that $|x_n - x^*| \leq C|x_n - x^*|^m$

Property 8.3.2 (Limits of Newton's Method) *Newton's Method does not always work. For example, it will never converge for $f(x) = \sqrt[3]{x}$ with $x_0 \neq 0$.*

Proof. Simply, we have

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = x_n - \frac{x_n^{1/3}}{x_n^{-2/3}/3} = -2x_n \tag{8.8}$$

It will not converge, since $\lim_{n\to\infty} |x_n| = \infty, \forall x_0 \neq 0$. ∎

8.3.2 Gradient Descent

Definition 8.3.2 *In gradient descent (GD), we iteratively estimate $\min_{\mathrm{w}} f(\mathbf{w})$ by computing a sequence of estimates $\mathbf{w}^{(0)}, \cdots, \mathbf{w}^{(k)}, \cdots$. Each estimate $\mathbf{w}^{(k+1)}$ is obtained from the previous by adding an update in the direction against gradients $-\nabla_f(\mathbf{w}^{(k)})$ with step length of η_k, i.e.,*

$$\mathbf{w}^{(k+1)} \leftarrow \mathbf{w}^{(k)} - \eta_k \nabla_f(\mathbf{w}^{(k)}). \tag{8.9}$$

Property 8.3.3 (Optimal Update of GD) *Assume that f's Hessian is invertible when evaluated at any point, then the optimal update is*

$$\Delta_k = -\mathbf{H}_f^{-1}(\mathbf{w}^{(k)})\nabla_f(\mathbf{w}^{(k)}),$$

where $\mathbf{H}_f$ is the Hessian matrix of f.

This is very similar to Newton's Method.

Proof. Let the optimal update of $k+1$ step is $\mathbf{w}^{(k+1)} = \operatorname{argmin}_{\mathrm{w}} f(\mathbf{w})$, where $\mathbf{w} \in U(\mathbf{w}^{(k)})$. Using Taylor's expansion of f at $\mathbf{w}^{(k)}$, we have

$$\begin{aligned} f(\mathbf{w}^{(k+1)}) = f(\mathbf{w}^{(k)}) &+ (\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)})\nabla_f(\mathbf{w}^{(k)}) && (8.10) \\ &+ \frac{1}{2}(\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)})^\top \mathbf{H}_f(\mathbf{w}^{(k)})(\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)}) + o. && (8.11) \end{aligned}$$

Set $\frac{\partial}{\partial \mathrm{w}} f(\mathbf{w}) = 0$, we get

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \mathbf{H}_f^{-1}(\mathbf{w}^{(k)})\nabla_f(\mathbf{w}^{(k)}) \tag{8.12}$$

■

Remark. When applying optimal update, GD is equivalent to Newton's method. The drawback is that Newton's method relies on the inverse of Hessian matrix, which may be intractable in practical.

Property 8.3.4 (Optimal Learning Rate) *The optimal learning rate η_k is the learning rate such that $\mathbf{w}^{(k)} - \eta_k \nabla_f(\mathbf{w}^{(k)})$ takes the minimal value.*

Proof. Similar to the previous one, we have the Taylor's expansion of f at $\mathbf{w}^{(k)}$ as

$$\begin{aligned} f(\mathbf{w}^{(k+1)}) &\approx f(\mathbf{w}^{(k)}) + (\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)})\nabla_f(\mathbf{w}^{(k)}) \qquad (8.13)\\ &+ \frac{1}{2}(\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)})^\top \mathbf{H}_f(\mathbf{w}^{(k)})(\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)}) \qquad (8.14)\\ &= f(\mathbf{w}^{(k)}) - \eta_k \nabla_f(\mathbf{w}^{(k)})^\top \nabla_f(\mathbf{w}^{(k)}) + \frac{1}{2}\eta_k^2 \nabla_f(\mathbf{w}^{(k)})^\top \mathbf{H}_f(\mathbf{w}^{(k)})\nabla_f(\mathbf{w}^{(k)}) \qquad (8.15) \end{aligned}$$

Set $\frac{\partial}{\partial \eta_k} f(\mathbf{w}) = 0$, we get

$$0 = \frac{\partial}{\partial \eta_k} f(\mathbf{w}) = -\left\|\nabla_f(\mathbf{w}^{(k)})\right\|^2 + \eta_k \nabla_f(\mathbf{w}^{(k)})^\top \mathbf{H}_f(\mathbf{w}^{(k)})\nabla_f(\mathbf{w}^{(k)}) \tag{8.16}$$

Therefore, the optimal learning rate is

$$\eta_k = \frac{\left\|\nabla_f(\mathbf{w}^{(k)})\right\|^2}{\nabla_f(\mathbf{w}^{(k)})^\top \mathbf{H}_f(\mathbf{w}^{(k)})\nabla_f(\mathbf{w}^{(k)})}. \tag{8.17}$$

■

8.3.3 Robbins-Monro Algorithm

Definition 8.3.3 *Given a function $f : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}$ and a random variables $Z \in \mathbb{R}^m$ whose distribution is unknown. The goal is to compute θ^* such that $\mathbb{E}_{z\sim Z}[f(\mathbf{z};\theta)] = 0$ via iteration as*

$$\theta^{(k+1)} = \theta^{(k)} - \eta^{(k+1)} f(\mathbf{z}_{k+1};\theta^{(k)}), \tag{8.18}$$

where $\eta^{(k)}$ denotes the learning rate at each step, and samples $\mathbf{z}_1, \cdots, \mathbf{z}_k \sim Z$.

Theorem 8.3.1 (Convergence) *If $\mathbb{E}_{z\sim Z}[f(\mathbf{z};\theta)]$ satisfies following regularity conditions*

$$\eta^{(k)} \geq 0, \quad \sum_k \eta^{(k)} = \infty, \quad \sum_k \eta^{(k)2} < \infty,$$

then Robbins-Monro algorithm will converge to θ^* with probability 1.

Proof. The proof can be found in Ex.4 - 4, or at [1]. ■

Definition 8.3.4 (Regularity Conditions) *If certain sufficient conditions are meet, then Robbins-Monro algorithm will converge to θ^*. These conditions are called regularity conditions, which are*

$$\begin{aligned} \mathbb{E}_Z[f(Z;\theta^*)] < \mathbb{E}_Z[f(Z;\theta)], \forall\theta > \theta^*,\\ \mathbb{E}_Z[f(Z;\theta^*)] > \mathbb{E}_Z[f(Z;\theta)], \forall\theta < \theta^*. \end{aligned}$$

[1] K. Fukunaga. Introduction to Statistical Pattern Recognition.

8.3.4 Stochastic Gradient Descent

Stochastic Gradient Descent (SGD) is an example of Robbins-Monro algorithm.

Definition 8.3.5

$$\min_{\theta} \sum_{i \leq n} \mathcal{L}(y_i, NN_{\theta}(\mathbf{x}_i)) \tag{8.19}$$

	Batch GD	**Mini-batch GD**
Gradient precision	High	Low
Handling large training set	Bad	Good
Improvement	Slow	Fast
Escaping local minimal	Not likely	Likely
Generalization error	High	Low

Chapter 9

PAC Learning

9.1 Empirical Risk Minimization

Definition 9.1.1 (Generalization & Empirical Error)

$$(\textit{Generalization}) \quad \mathcal{R}(\widehat{c}) = p(\widehat{c}(\mathbf{x}) \neq c(\mathbf{x})) \tag{9.1}$$

$$(\textit{Empirical}) \quad \widehat{\mathcal{R}}(\widehat{c}) = \frac{1}{n}\sum_{i} \mathbb{1}\{\, c(\mathbf{x}_i) \neq y_i\} \tag{9.2}$$

Definition 9.1.2 (ERM) *Empirical Risk Minimization (ERM) means selecting the classifier $\widehat{c}_n \in \mathcal{C}$ with the smallest error on the training data $Z = \{(\mathbf{x}_1.y_1), \cdots, (\mathbf{x}_n.y_n)\}$, i,e,,*

$$\widehat{c}_n = \operatorname*{argmin}_{c\in\mathcal{C}} \widehat{\mathcal{R}}_n(c), \quad \textit{where} \;\; \widehat{\mathcal{R}}_n(c) = \frac{1}{n}\sum_{i} \mathbb{1}\{\, c(\mathbf{x}_i) \neq y_i\}\,.$$

Here, $\widehat{\mathcal{R}}_n(c)$ is called the **empirical error** *of c.*

Theorem 9.1.1 (Vapnik & Chervonenkis) *Let $c^* = \operatorname{argmin}_c \mathcal{R}(c)$,*

$$\mathcal{R}(\widehat{c}^*_n) - \inf_{c\in\mathcal{C}} \mathcal{R}(c) \leq 2\sup_{c\in\mathcal{C}} |\widehat{\mathcal{R}}_n(c) - \mathcal{R}_n(c)|.$$

Proof.

$$\begin{aligned}
\mathcal{R}(\widehat{c}^*_n) - \inf_{c\in\mathcal{C}} \mathcal{R}(c) &= \mathcal{R}(\widehat{c}^*_n) - \widehat{\mathcal{R}}(\widehat{c}^*_n) + \widehat{\mathcal{R}}(\widehat{c}^*_n) - \inf_{c\in\mathcal{C}} \mathcal{R}(c) && (9.3)\\
&\leq \mathcal{R}(\widehat{c}^*_n) - \widehat{\mathcal{R}}(\widehat{c}^*_n) + \widehat{\mathcal{R}}(c^*_n) - \mathcal{R}(c^*) && (9.4)\\
&\leq \sup_{c\in\mathcal{C}} |\mathcal{R}(c) - \widehat{\mathcal{R}}(c)| + \sup_{c\in\mathcal{C}} |\widehat{\mathcal{R}}(c) - \mathcal{R}(c)| && (9.5)\\
&\leq 2\sup_{c\in\mathcal{C}} |\mathcal{R}(c) - \widehat{\mathcal{R}}(c)|. && (9.6)
\end{aligned}$$

■

9.2 PAC Learning Model

A learning algorithm $\mathcal{A}$ can learn a concept class $\mathcal{C}$ from $\mathcal{H}$ if, given as input a sufficiently large sample, it outputs a hypothesis that generalizes well with high probability.

Definition 9.2.1 (PCA Learnable) *A learning algorithm $\mathcal{A}$ can learn a concept class $\widehat{c}^*_n \in \mathcal{C}$ from $\mathcal{H}$, if there is a polynomial function $\operatorname{poly}(\cdot,\cdot,\cdot)$ such that: (1) for any distribution $\mathcal{D}$ on $\mathcal{X} \times \{0,1\}$ and (2) for any $0 < \epsilon < 1/2$, $0 < \delta < 1/2$,*

$$p_{Z\sim\mathcal{D}^n}\left(\mathcal{R}(\widehat{c}^*_n) - \inf_{c\in\mathcal{C}} \mathcal{R}(c) \leq \epsilon\right) \geq 1-\delta, \quad \textit{where} \;\; n \geq \operatorname{poly}(1/\epsilon, 1/\delta, dim(\mathcal{X})). \tag{9.7}$$

Here, $\mathcal{D}^n$ means the training set of n samples, i.e., $\mathcal{D}^n = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)\}$.

Property 9.2.1 (Universal Concept Class) *Universal concept class is not PAC learnable. Let $\mathcal{X} = \{0,1\}^*$ be the set of all finite binary sequences. The concept class $\mathcal{C}$ formed by all subsets of $\mathcal{X}$ is not PAC learnable from C.*

Definition 9.2.2 (Efficient PAC Learning) *If $\mathcal{A}$ runs in* **polynomial** *time in $1/\epsilon$ and $1/\delta$, we say that $\mathcal{A}$ is an efficient PAC learning algorithm.*

9.2.1 Finite Hypothesis Classes

In this section, we assume that the hypothesis class is finite. Let $|\mathcal{H}|$ denote the cardinality of $\mathcal{H}$.

Theorem 9.2.1 (Error Bound - Consistent Hypothesis Classes) *Let $\mathcal{C}$ be a finite concept class, $\mathcal{H}$ be a consistent hypothesis class $\mathcal{H} = \mathcal{C}$ and $\mathcal{A}$ be an algorithm that returns a consistent hypothesis $\widehat{c}$ (i.e., $\forall n < \infty : \widehat{\mathcal{R}}_n(\widehat{c}) = 0$).*

For any target concept $c \in \mathcal{C}$ and any i.i.d. sample Z, for any $\epsilon > 0$, there exists $\delta > 0$ such that

$$p(\mathcal{R}(\widehat{c}) \leq \epsilon) \geq 1 - \delta, \quad \textit{where} \;\; n \geq \frac{1}{\epsilon}\left(\log|\mathcal{H}| + \log\frac{1}{\delta}\right). \tag{9.8}$$

Proof. In consistent case, we have assumed that the empirical error $\widehat{\mathcal{R}}_n(\widehat{c}) = 0$.

$$\begin{aligned} p\left(\sup_{c\in\mathcal{C}} \mathcal{R}(\widehat{c}) > \epsilon\right) &\leq \sum_{c\in\mathcal{C}} p\left(\mathcal{R}(\widehat{c}) > \epsilon \cap \widehat{\mathcal{R}}_n(\widehat{c}) = 0\right) && (9.9)\\ &\leq \sum_{c\in\mathcal{C}} p\left(\widehat{\mathcal{R}}_n(\widehat{c}) = 0 \mid \mathcal{R}(\widehat{c}) > \epsilon\right) && (9.10)\\ &\leq |\mathcal{H}|(1-\epsilon)^n && (9.11)\end{aligned}$$

Then, we can solve the n from this inequality, which brings us to the conclusion. ■

Theorem 9.2.2 (Error Bound - Inconsistent Hypothesis Classes) *Let $\mathcal{C}$ be a finite concept class, $\mathcal{H}$ be a inconsistent hypothesis class $\mathcal{H} \neq \mathcal{C}$ and $\mathcal{A}$ be an algorithm that returns a consistent hypothesis $\widehat{c}$ (i.e., $\forall n < \infty : \widehat{\mathcal{R}}_n(\widehat{c}) = 0$).*

For any target concept $c \in \mathcal{C}$ and any i.i.d. sample Z, for any $\epsilon > 0$, there exists $\delta > 0$ such that

$$p\left(\sup_{c\in\mathcal{C}} |\widehat{\mathcal{R}}_n(\widehat{c}) - \mathcal{R}(\widehat{c})| \leq \epsilon\right) \geq 1 - \delta, \quad \textit{where} \;\; n \geq \frac{1}{2\epsilon^2}\left(\log|\mathcal{H}| + \log\frac{2}{\delta}\right). \tag{9.12}$$

Proof. Using the Hoeffding inequality, we have

$$\begin{aligned} p\left(\sup_{c\in\mathcal{C}} |\widehat{\mathcal{R}}_n(\widehat{c}) - \mathcal{R}(\widehat{c})| \leq \epsilon\right) &\leq \sum_{c\in\mathcal{C}} p\left(|\widehat{\mathcal{R}}_n(\widehat{c}) - \mathcal{R}(\widehat{c})| \leq \epsilon\right) && (9.13)\\ &\leq 2|\mathcal{H}|\exp(-2n\epsilon^2) && (9.14)\end{aligned}$$

Then, we can solve the n from this inequality, which brings us to the conclusion. ■

Remark. This theorem can be also written into the following form. The variance depends on sample size as $1/\sqrt{n}$, but only logarithmically on the size of the hypothesis class as $\log|\mathcal{H}|$.

$$\underbrace{\mathcal{R}(\widehat{c})}_{\text{expected error}} \leq \underbrace{\widehat{\mathcal{R}}_n(\widehat{c})}_{\text{empirical error}} + \underbrace{\sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta}}{2n}}}_{\text{variance}}, \quad \forall c \in \mathcal{C}. \tag{9.15}$$

9.2.2 Example: Learning Axis-aligned Rectangles

Let $\mathcal{C}$ be the concept of all axis-aligned rectangles R. We show that $\mathcal{C}$ can be learned from $\mathcal{H} = \mathcal{C}$.

See Fig. 9.1, consider the algorithm $\mathcal{A}$ finds the smallest rectangle R' containing all positive points. [1] We will show that $\mathcal{A}$ can learn any concept of $R \in \mathcal{C}$.

[1] This figure is taken from M. Mohri, Foundations of Machine Learning, 2018

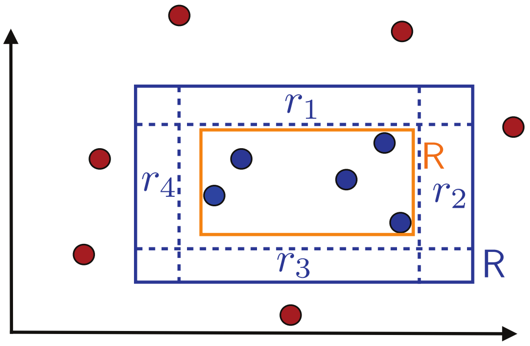


Figure 9.1: Illustration of rectangle R and “RIG” regions.

Theorem 9.2.3 (Axis-aligned Rectangles are PCA Learnable) *Given a dataset with n points, for any $\delta > 0, \epsilon > 0$, to ensure that $p(\mathcal{R}(R') > \epsilon) \leq \delta$, we can impose a constraint that*

$$n \geq \text{poly}(1/\epsilon, 1/\delta, \text{size}(R') = 4) = \frac{4}{\epsilon} \log \frac{4}{\delta}.$$

Proof. Given a prediction R', its risk $\mathcal{R}(R')$ is the sum of probability of false negative and false positive. More precisely,

$$\mathcal{R}(R') = p\left(\underbrace{(R - R')}_{\text{false neg.}} \cup \underbrace{(R' - R)}_{falsepos.}\right) = p(R - R'). \tag{9.16}$$

We define four stripes of the R, r_1, r_2, r_3, r_4, as shown in Fig. 9.1. Each of the stripe has probability at least $\epsilon/4$. If the risk exceed ϵ, then R' does not hit at least one of the stripes. We can write,

$$p(\mathcal{R}(R') \geq \epsilon) \leq p_D\left(\bigcup_{i \leq 4} R' \cap r_i = \emptyset\right) \tag{9.17}$$

$$\leq \sum_{i \leq 4} p_D(R' \cap r_i = \emptyset) \tag{9.18}$$

$$\leq 4(1 - \epsilon/4)^n \tag{9.19}$$

$$\leq 4 \exp(-n\epsilon/4). \qquad (1 - x \leq e^{-x}) \tag{9.20}$$

To ensure $p(\mathcal{R}(R') \geq \epsilon) \leq \delta$, we have

$$4 \exp(-n\epsilon/4) \leq \delta \iff n \geq \frac{4}{\epsilon} \log \frac{4}{\delta} \tag{9.21}$$

■

9.3 VC Dimension

In Vapnik–Chervonenkis theory, the Vapnik–Chervonenkis (VC) dimension is a measure of the capacity (complexity) of a set of functions that can be learned by a **binary** classification algorithm.

Definition 9.3.1 *VC dimension of $\mathcal{H}$ is defined as the cardinality of the largest set of points on $\mathcal{H}$ that the algorithm can shatter.*

Table 9.1: Some interesting examples of VC dimension.

Classifier		**VC-dim**
stump classifier	$(-\infty, a]$, $a \in \mathbb{R}$	1
all intervals in $\mathbb{R}$	$\{[a, b] \mid a, b \in \mathbb{R}\}$	2
all unions of k intervals in $\mathbb{R}$	$\{\bigcup_{i=1}^{k}[a_i, b_i] \mid a_i, b_i \in \mathbb{R}\}$	$2k$
all half-planes in $\mathbb{R}^2$	$\{(x, y) \mid ax + by + c \geq 0, a, b, c \in \mathbb{R}\}$	3
all convex polygons in $\mathbb{R}^2$	-	∞
all convex polygons with at most k vertices in $\mathbb{R}^2$	-	$2k + 1$

9.4 Concentration Inequalities

Theorem 9.4.1 (Markov Inequality) *Let X be a non-negative random variable. Then*

$$p(X \geq \epsilon) \leq \frac{\mathbb{E}[X]}{\epsilon} \tag{9.22}$$

Theorem 9.4.2 (Hoeffding inequality) *Let $X_1, ..., X_m$ be independent random variables with X_i taking values in $[a_i, b_i]$ for all $i \in [m]$. Then, for any $\epsilon > 0$, the following inequalities hold for $S_m = \sum_{i=1}^{m} X_i$:*

$$p(S_m - \mathbb{E}[S_m] \geq \epsilon) \leq \exp\left(\frac{-2\epsilon^2}{\sum_{i=1}^{m}(b_i - a_i)^2}\right) \tag{9.23}$$

$$p(S_m - \mathbb{E}[S_m] \leq -\epsilon) \leq \exp\left(\frac{-2\epsilon^2}{\sum_{i=1}^{m}(b_i - a_i)^2}\right) \tag{9.24}$$

Theorem 9.4.3 (McDiarmid inequality) *Let $X_1, \cdots, X_m \in \mathcal{X}^m$ be a set of $m \geq 1$ independent random variables and assume that there exist $c_1, \cdots, c_m > 0$ such that $f : \mathcal{X}^m \to \mathbb{R}$ satisfies the following conditions:*

$$|f(x_1, ..., x_i, ..., x_m) - f(x_1, ..., x_i', ...x_m)| \leq c_i,$$

for all $i \in [m]$ and any points $x_1, ..., x_m, x_i' \in \mathcal{X}$. Let $f(S)$ denote $f(X_1, ..., X_m)$, then, for all $\epsilon > 0$, the following inequalities hold:

$$p(f(S) - \mathbb{E}[f(S)] \geq \epsilon) \leq \exp\left(\frac{-2\epsilon^2}{\sum_{i=1}^{m} c_i^2}\right) \tag{9.25}$$

$$p(f(S) - \mathbb{E}[f(S)] \leq -\epsilon) \leq \exp\left(\frac{-2\epsilon^2}{\sum_{i=1}^{m} c_i^2}\right) \tag{9.26}$$

Note that *Hoeffding inequality* is a special instance of *McDiarmid inequality* where f is defined by $f(S) = \frac{1}{m}\sum_{i=1}^{m} x_i$.