https://github.com/soulmachine/machine-learning-cheat-sheet
soulmachine@gmail.com

Machine Learning Cheat Sheet

Classical equations, diagrams and tricks in machine learning

November 20, 2013

Preface

This cheat sheet contains many classical equations and diagrams on machine learning, which will help you quickly recall knowledge and ideas on machine learning.

The cheat sheet will also appeal to someone who is preparing for a job interview related to machine learning.

This cheat sheet has three significant advantages:

1. Strong typed. Compared to programming languages, mathematical formulas are weekly typed. For example, X can be a set, a random variable, or a matrix. This causes difficulty in understanding the meaning of formulas. In this cheat sheet, I try my best to standardize symbols used, see section §.
2. More parentheses. In machine leaning, authors are prone to omit parentheses, brackets and braces, this usually causes ambiguity in mathematical formulas. In this cheat sheet, I use parentheses(brackets and braces) at where they are needed, to make formulas easy to understand.
3. Less thinking jumps. In many books, authors are prone to omit some steps that are trivial in his option. But it often makes readers get lost in the middle way of derivation.

At Tsinghua University, May 2013 *soulmachine*

Contents

List of Contributors

Wei Zhang
PhD candidate at the Institute of Software, Chinese Academy of Sciences (ISCAS), Beijing, P.R.CHINA, e-mail: zh3feng@gmail.com, has written chapters of Naive Bayes and SVM.

Fei Pan
Master at Beijing University of Technology, Beijing, P.R.CHINA, e-mail: example@gmail.com, has written chapters of KMeans, AdaBoost.

Yong Li
PhD candidate at the Institute of Automation of the Chinese Academy of Sciences (CASIA), Beijing, P.R.CHINA, e-mail: liyong3forever@gmail.com, has written chapters of Logistic Regression.

Jiankou Li
PhD candidate at the Institute of Software, Chinese Academy of Sciences (ISCAS), Beijing, P.R.CHINA, e-mail: lijiankoucoco@163.com, has written chapters of BayesNet.

Notation

Introduction

It is very difficult to come up with a single, consistent notation to cover the wide variety of data, models and algorithms that we discuss. Furthermore, conventions difer between machine learning and statistics, and between different books and papers. Nevertheless, we have tried to be as consistent as possible. Below we summarize most of the notation used in this book, although individual sections may introduce new notation. Note also that the same symbol may have diferent meanings depending on the context, although we try to avoid this where possible.

General math notation

Symbol	Meaning
$\lfloor x \rfloor$	Floor of x, i.e., round down to nearest integer
$\lceil x \rceil$	Ceiling of x, i.e., round down to nearest integer
$\boldsymbol{x} \otimes \boldsymbol{y}$	Convolution of $\boldsymbol{x}$ and $\boldsymbol{y}$
$\boldsymbol{x} \odot \boldsymbol{y}$	Hadamard (elementwise) product of $\boldsymbol{x}$ and $\boldsymbol{y}$
$a \wedge b$	logical AND
$a \vee b$	logical OR
$\neg a$	logical NOT
$\mathbb{I}(x)$	Indicator function, $\mathbb{I}(x) = 1$ if x is true, else $\mathbb{I}(x) = 0$
∞	Infinity
$\rightarrow$	Tends towards, e.g., $n \rightarrow \infty$
$\propto$	Proportional to, so $y = ax$ can be written as $y \propto x$
$\lvert x \rvert$	Absolute value
$\lvert \mathcal{S} \rvert$	Size (cardinality) of a set
$n!$	Factorial function
∇	Vector of first derivatives
∇^2	Hessian matrix of second derivatives
$\triangleq$	Defined as
$O(\cdot)$	Big-O: roughly means order of magnitude
$\mathbb{R}$	The real numbers
$1:n$	Range (Matlab convention): $1:n = 1, 2, ..., n$
$\approx$	Approximately equal to
$\arg\max_x f(x)$	Argmax: the value x that maximizes f
$B(a,b)$	Beta function, $B(a,b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$

$B(\boldsymbol{\alpha})$	Multivariate beta function, $\frac{\prod\limits_{k} \Gamma(\alpha_k)}{\Gamma(\sum\limits_{k} \alpha_k)}$
$\binom{n}{k}$	n choose k , equal to $n!/(k!(nk)!)$
$\delta(x)$	Dirac delta function,$\delta(x) = \infty$ if $x = 0$, else $\delta(x) = 0$
$exp(x)$	Exponential function e^x
$\Gamma(x)$	Gamma function, $\Gamma(x) = \int_0^\infty u^{x-1} e^{-u} du$
$\Psi(x)$	Digamma function,$Psi(x) = \frac{d}{dx} \log \Gamma(x)$
$\mathcal{X}$	A set from which values are drawn (e.g.,$\mathcal{X} = \mathbb{R}^D$)

Linear algebra notation

We use boldface lower-case to denote vectors, such as $\boldsymbol{x}$, and boldface upper-case to denote matrices, such as $\boldsymbol{X}$. We denote entries in a matrix by non-bold upper case letters, such as X_{ij}. Vectors are assumed to be column vectors, unless noted otherwise.

Symbol	Meaning
$\boldsymbol{X} \succ 0$	$\boldsymbol{X}$ is a positive definite matrix
$tr(\boldsymbol{X})$	Trace of a matrix
$det(\boldsymbol{X})$	Determinant of matrix $\boldsymbol{X}$
$\lvert\boldsymbol{X}\rvert$	Determinant of matrix $\boldsymbol{X}$
$\boldsymbol{X}^{-1}$	Inverse of a matrix
$\boldsymbol{X}^{\dagger}$	Pseudo-inverse of a matrix
$\boldsymbol{X}^T$	Transpose of a matrix
$\boldsymbol{x}^T$	Transpose of a vector
$diag(x)$	Diagonal matrix made from vector $\boldsymbol{x}$
$diag(X)$	Diagonal vector extracted from matrix $\boldsymbol{X}$
$\boldsymbol{I}$ or $\boldsymbol{I}_d$	Identity matrix of size $d \times d$ (ones on diagonal, zeros of)
$\mathbf{1}$ or $\mathbf{1}_d$	Vector of ones (of length d)
$\mathbf{0}$ or $\mathbf{0}_d$	Vector of zeros (of length d)
$\lvert\lvert\boldsymbol{x}\rvert\rvert = \lvert\lvert\boldsymbol{x}\rvert\rvert_2$	Euclidean or ℓ_2 norm $\sqrt{\sum\limits_{j=1}^{d} x_j^2}$
$\lvert\lvert\boldsymbol{x}\rvert\rvert_1$	ℓ_1 norm $\sum\limits_{j=1}^{d} \lvert x_j \rvert$
$\boldsymbol{X}_{:,j}$	jth column of matrix
$\boldsymbol{X}_{i,:}$	transpose of ith row of matrix (a column vector)
$\boldsymbol{X}_{i,j}$	Element (i,j) of matrix $\boldsymbol{X}$
$\boldsymbol{x} \otimes \boldsymbol{y}$	Tensor product of $\boldsymbol{x}$ and $\boldsymbol{y}$

Probability notation

We denote random and fixed scalars by lower case, random and fixed vectors by bold lower case, and random and fixed matrices by bold upper case. Occasionally we use non-bold upper case to denote scalar random variables. Also, we use $p()$ for both discrete and continuous random variables

Symbol	Meaning
X,Y	Random variable
$P()$	Probability of a random event

$F()$	Cumulative distribution function(CDF), also called distribution function
$p(x)$	Probability mass function(PMF)
$f(x)$	probability density function(PDF)
$F(x,y)$	Joint CDF
$p(x,y)$	Joint PMF
$f(x,y)$	Joint PDF
$p(X\|Y)$	Conditional PMF, also called conditional probability
$f_{X\|Y}(x\|y)$	Conditional PDF
$X \perp Y$	X is independent of Y
$X \not\perp Y$	X is not independent of Y
$X \perp Y\|Z$	X is conditionally independent of Y given Z
$X \not\perp Y\|Z$	X is not conditionally independent of Y given Z
$X \sim p$	X is distributed according to distribution p
$\boldsymbol{\alpha}$	Parameters of a Beta or Dirichlet distribution
$cov[X]$	Covariance of X
$\mathbb{E}[X]$	Expected value of X
$\mathbb{E}_q[X]$	Expected value of X wrt distribution q
$\mathbb{H}(X)$ or $\mathbb{H}(p)$	Entropy of distribution $p(X)$
$\mathbb{I}(X;Y)$	Mutual information between X and Y
$\mathbb{KL}(p\|\|q)$	KL divergence from distribution p to q
$\ell(\boldsymbol{\theta})$	Log-likelihood function
$L(\boldsymbol{\theta},a)$	Loss function for taking action a when true state of nature is θ
λ	Precision (inverse variance) $\lambda = 1/\sigma^2$
Λ	Precision matrix $\Lambda = \Sigma^{-1}$
$mode[\boldsymbol{X}]$	Most probable value of $\boldsymbol{X}$
μ	Mean of a scalar distribution
$\boldsymbol{\mu}$	Mean of a multivariate distribution
$\boldsymbol{\Phi}$	cdf of standard normal
ϕ	pdf of standard normal
$\boldsymbol{\pi}$	multinomial parameter vector, Stationary distribution of Markov chain
ρ	Correlation coefficient
$\mathrm{sigm}(x)$	Sigmoid (logistic) function, $\frac{1}{1+e^{-x}}$
σ^2	Variance
Σ	Covariance matrix
$\mathrm{var}[x]$	Variance of x
ν	Degrees of freedom parameter
Z	Normalization constant of a probability distribution

Machine learning/statistics notation

In general, we use upper case letters to denote constants, such as C,K,M,N,T, etc. We use lower case letters as dummy indexes of the appropriate range, such as $c = 1:C$ to index classes, $i = 1:M$ to index data cases, $j = 1:N$ to index input features, $k = 1:K$ to index states or clusters, $t = 1:T$ to index time, etc.

We use x to represent an observed data vector. In a supervised problem, we use y or $\boldsymbol{y}$ to represent the desired output label. We use $\boldsymbol{z}$ to represent a hidden variable. Sometimes we also use q to represent a hidden discrete variable.

Symbol	Meaning
C	Number of classes

D	Dimensionality of data vector (number of features)
N	Number of data cases
N_c	Number of examples of class c,$N_c = \sum_{i=1}^{N} \mathbb{I}(y_i = c)$
R	Number of outputs (response variables)
$\mathcal{D}$	Training data $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) \mid i = 1 : N\}$
$\mathcal{D}_{test}$	Test data
$\mathcal{X}$	Input space
$\mathcal{Y}$	Output space
K	Number of states or dimensions of a variable (often latent)
$k(x,y)$	Kernel function
$\boldsymbol{K}$	Kernel matrix
$\mathcal{H}$	Hypothesis space
L	Loss function
$J(\boldsymbol{\theta})$	Cost function
$f(\boldsymbol{x})$	Decision function
$P(y \mid \boldsymbol{x})$	TODO
λ	Strength of ℓ_2 or $\ell_1 regularizer$
$\phi(x)$	Basis function expansion of feature vector $\boldsymbol{x}$
$\boldsymbol{\Phi}$	Basis function expansion of design matrix $\boldsymbol{X}$
$q()$	Approximate or proposal distribution
$Q(\boldsymbol{\theta}, \boldsymbol{\theta}_{old})$	Auxiliary function in EM
T	Length of a sequence
$T(\mathcal{D})$	Test statistic for data
$\boldsymbol{T}$	Transition matrix of Markov chain
$\boldsymbol{\theta}$	Parameter vector
$\boldsymbol{\theta}^{(s)}$	s'th sample of parameter vector
$\hat{\boldsymbol{\theta}}$	Estimate (usually MLE or MAP) of $\boldsymbol{\theta}$
$\hat{\boldsymbol{\theta}}_{MLE}$	Maximum likelihood estimate of $\boldsymbol{\theta}$
$\hat{\boldsymbol{\theta}}_{MAP}$	MAP estimate of $\boldsymbol{\theta}$
$\bar{\boldsymbol{\theta}}$	Estimate (usually posterior mean) of $\boldsymbol{\theta}$
$\boldsymbol{w}$	Vector of regression weights (called $\boldsymbol{\beta}$ in statistics)
b	intercept (called ε in statistics)
$\boldsymbol{W}$	Matrix of regression weights
x_{ij}	Component (i.e., feature) j of data case i ,for $i = 1 : N, j = 1 : D$
$\boldsymbol{x}_i$	Training case, $i = 1 : N$
$\boldsymbol{X}$	Design matrix of size $N \times D$
$\bar{\boldsymbol{x}}$	Empirical mean $\bar{\boldsymbol{x}} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}_i$
$\tilde{\boldsymbol{x}}$	Future test case
$\boldsymbol{x}_*$	Feature test case
$\boldsymbol{y}$	Vector of all training labels $\boldsymbol{y} = (y_1, ..., y_N)$
z_{ij}	Latent component j for case i

Chapter 1
Introduction

1.1 Types of machine learning

$$\text{Machine learning}\begin{cases}\text{Supervised learning}\begin{cases}\text{Classfication}\\ \text{Regression}\end{cases}\\ \text{Unsupervised learning}\begin{cases}\text{Discovering clusters}\\ \text{Discovering latent factors}\\ \text{Discovering graph structure}\\ \text{Matrix completion}\end{cases}\end{cases}$$

1.2 Three elements of a machine learning model

Model = Representation + Evaluation + Optimization[1]

1.2.1 Representation

In supervised learning, a model must be represented as a conditional probability distribution $P(y|\boldsymbol{x})$(usually we call it classifier) or a decision function $f(x)$. The set of classifiers(or decision functions) is called the hypothesis space of the model. Choosing a representation for a model is tantamount to choosing the hypothesis space that it can possibly learn.

1.2.2 Evaluation

In the hypothesis space, an evaluation function (also called objective function or risk function) is needed to distinguish good classifiers(or decision functions) from bad ones.

1.2.2.1 Loss function and risk function

Definition 1.1. In order to measure how well a function fits the training data, a **loss function** $L : Y \times Y \to R \geq 0$ is defined. For training example (x_i, y_i), the loss of predicting the value $\widehat{y}$ is $L(y_i, \widehat{y})$.

[1] Domingos, P. A few useful things to know about machine learning. Commun. ACM. 55(10):7887 (2012).

The following is some common loss functions:

1. 0-1 loss function $L(Y,f(X))=I(Y,f(X))=\begin{cases}1, & Y=f(X)\\ 0, & Y\neq f(X)\end{cases}$
2. Quadratic loss function $L(Y,f(X))=(Y-f(X))^2$
3. Absolute loss function $L(Y,f(X))=|Y-f(X)|$
4. Logarithmic loss function $L(Y,P(Y|X))=-\log P(Y|X)$

Definition 1.2. The risk of function f is defined as the expected loss of f:

$$R_{exp}(f)=E_p\left[L\left(Y,f(X)\right)\right]=\int_{X\times Y}L\left(y,f(x)\right)P(x,y)dxdy \tag{1.1}$$

which is also called expected loss or **risk function**.

Definition 1.3. The risk function $R_{exp}(f)$ can be estimated from the training data as

$$R_{emp}(f)=\frac{1}{N}\sum_{i=1}^{N}L\left(y_i,f(x_i)\right) \tag{1.2}$$

which is also called empirical loss or **empirical risk**.

You can define your own loss function, but if you're a novice, you're probably better off using one from the literature. There are conditions that loss functions should meet[2]:

1. They should approximate the actual loss you're trying to minimize. As was said in the other answer, the standard loss functions for classification is zero-one-loss (misclassification rate) and the ones used for training classifiers are approximations of that loss.
2. The loss function should work with your intended optimization algorithm. That's why zero-one-loss is not used directly: it doesn't work with gradient-based optimization methods since it doesn't have a well-defined gradient (or even a subgradient, like the hinge loss for SVMs has).
   The main algorithm that optimizes the zero-one-loss directly is the old perceptron algorithm(chapter §3).

1.2.2.2 ERM and SRM

Definition 1.4. ERM(Empirical risk minimization)

$$\min_{f\in\mathcal{F}}R_{emp}(f)=\min_{f\in\mathcal{F}}\frac{1}{N}\sum_{i=1}^{N}L\left(y_i,f(x_i)\right) \tag{1.3}$$

Definition 1.5. Structural risk

$$R_{smp}(f)=\frac{1}{N}\sum_{i=1}^{N}L\left(y_i,f(x_i)\right)+\lambda J(f) \tag{1.4}$$

Definition 1.6. SRM(Structural risk minimization)

$$\min_{f\in\mathcal{F}}R_{srm}(f)=\min_{f\in\mathcal{F}}\frac{1}{N}\sum_{i=1}^{N}L\left(y_i,f(x_i)\right)+\lambda J(f) \tag{1.5}$$

[2] http://t.cn/zTrDxLO

1.2.3 Optimization

Finally, we need a **training algorithm**(also called **learning algorithm**) to search among the classifiers in the the hypothesis space for the highest-scoring one. The choice of optimization technique is key to the **efficiency** of the model.

1.3 Cross validation

Definition 1.7. Cross validation, sometimes called *rotation estimation*, is a *model validation* technique for assessing how the results of a statistical analysis will generalize to an independent data set[3].

Common types of cross-validation:

1. K-fold cross-validation. In k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k 1 subsamples are used as training data.
2. 2-fold cross-validation. Also, called simple cross-validation or holdout method. This is the simplest variation of k-fold cross-validation, k=2.
3. Leave-one-out cross-validation(*LOOCV*). k=M, the number of original samples.

1.4 Linear Regression

Given

$$\begin{aligned}
&\mathcal{D} = \{(\boldsymbol{x}_i, y_i)|i=1:M\} \\
&\mathcal{H} = \left\{f(\boldsymbol{x_i}) = \boldsymbol{w}^T\boldsymbol{x_i}+b|i=1:M\right\} \\
&L(\boldsymbol{w},b) = \sum_{i=1}^{M}\left(y_i - f(\boldsymbol{x}_i)-b\right)^2
\end{aligned} \tag{1.6}$$

Let $\widehat{\boldsymbol{w}} = \left(\boldsymbol{w}^T, b\right)^T$, and

$$\widehat{\boldsymbol{X}} = \begin{pmatrix} \widehat{\boldsymbol{x}}_1^T \\ \widehat{\boldsymbol{x}}_2^T \\ \vdots \\ \widehat{\boldsymbol{x}}_M^T \end{pmatrix}, where\ \widehat{\boldsymbol{x}}_i = \left(\boldsymbol{x}_i^T, 1\right)^T \tag{1.7}$$

We can get

$$\begin{aligned}
&L(\widehat{\boldsymbol{w}}) = \left(\boldsymbol{y} - \widehat{\boldsymbol{X}}\widehat{\boldsymbol{w}}\right)^T\left(\boldsymbol{y} - \widehat{\boldsymbol{X}}\widehat{\boldsymbol{w}}\right) \\
&\frac{\partial L}{\partial \widehat{\boldsymbol{w}}} = -2\widehat{\boldsymbol{X}}^T\boldsymbol{y} + 2\widehat{\boldsymbol{X}}^T\widehat{\boldsymbol{X}}\widehat{\boldsymbol{w}} = 0 \\
&\widehat{\boldsymbol{X}}^T\boldsymbol{y} = \widehat{\boldsymbol{X}}^T\widehat{\boldsymbol{X}}\widehat{\boldsymbol{w}} \\
&\widehat{\boldsymbol{w}} = \left(\widehat{\boldsymbol{X}}^T\widehat{\boldsymbol{X}}\right)^{-1}\widehat{\boldsymbol{X}}^T\boldsymbol{y}
\end{aligned} \tag{1.8}$$

If $\widehat{\boldsymbol{X}}^T\widehat{\boldsymbol{X}}$ is singular, the pseudo-inverse can be used, or else the technique of ridge regression described below can be applied.

[3] http://en.wikipedia.org/wiki/Cross-validation_(statistics)

Chapter 2
Probability

2.1 Frequentists vs. Bayesians

what is probability? **frequentist** interpretation vs. **Bayesian** interpretation.

One big advantage of the Bayesian interpretation is that it can be used to model our uncertainty about events that do not have long term frequencies.

Therefore most machine learning books adopt the Bayesian interpretation. Fortunately, the basic rules of probability theory are the same, no matter which interpretation is adopted.

2.2 A brief review of probability theory

2.2.1 Basic concepts

We denote a random event by defining a **random variable** X.

Descrete random variable: X can take on any value from a finite or countably infinite set.

Continuous random variable: the value of X is real-valued.

2.2.1.1 CDF

$$F(x) \triangleq P(X \leq x) = \begin{cases} \sum_{u \leq x} p(u), & \text{descrete random variable} \\ \int_{-\infty}^{x} f(u)du, & \text{continuous random variable} \end{cases} \tag{2.1}$$

2.2.1.2 PMF and PDF

For descrete random variable, We denote the probability of the event that $X = x$ by $P(X = x)$, or just $p(x)$ for short. Here $p(x)$ is called a **probability mass function** or **PMF**.A probability mass function is a function that gives the probability that a discrete random variable is exactly equal to some value[4]. This satisfies the properties $0 \leq p(x) \leq 1$ and $\sum_{x \in \mathcal{X}} p(x) = 1$.

For continuous variable, in the equation $F(x) = \int_{-\infty}^{x} f(u)du$, the function $f(x)$ is called a **probability density function** or **PDF**. A probability density function is a function that describes the relative likelihood for this random variable to take on a given value[5].This satisfies the properties $f(x) \geq 0$ and $\int_{-\infty}^{\infty} f(x)dx = 1$.

[4] http://en.wikipedia.org/wiki/Probability_mass_function

[5] http://en.wikipedia.org/wiki/Probability_density_function

2.2.2 Mutivariate random variables

2.2.2.1 Joint CDF

We denote joint CDF by $F(x,y) \triangleq P(X \leq x \cap Y \leq y) = P(X \leq x, Y \leq y)$.

$$F(x,y) \triangleq P(X \leq x, Y \leq y) = \begin{cases} \sum_{u \leq x, v \leq y} p(u,v), & \text{descrete} \\ \int_{-\infty}^{x} \int_{-\infty}^{y} f(u,v) du dv, & \text{continuous} \end{cases} \tag{2.2}$$

product rule:

$$p(X,Y) = P(X|Y)P(Y) \tag{2.3}$$

Chain rule:

$$p(X_{1:N}) = p(X_1)p(X_2|X_1)p(X_3|X_2,X_1)...p(X_N|X_{1:N-1}) \tag{2.4}$$

2.2.2.2 Marginal distribution

Marginal CDF:

$$\begin{cases} F_X(x) \triangleq F(x,+\infty) = \begin{cases} \sum_{x_i \leq x} P(X = x_i) = \sum_{x_i \leq x} \sum_{j=1}^{+\infty} P(X = x_i, Y = y_j), & \text{descrete} \\ \int_{-\infty}^{x} f_X(u) du = \int_{-\infty}^{x} \int_{-\infty}^{+\infty} f(u,v) du dv, & \text{continuous} \end{cases} \\ F_Y(y) \triangleq F(+\infty,y) = \begin{cases} \sum_{y_j \leq y} p(Y = y_j) = \sum_{i=1}^{+\infty} \sum_{y_j \leq y} P(X = x_i, Y = y_j), & \text{descrete} \\ \int_{-\infty}^{y} f_Y(v) dv = \int_{-\infty}^{+\infty} \int_{-\infty}^{y} f(u,v) du dv, & \text{continuous} \end{cases} \end{cases} \tag{2.5}$$

Marginal PMF or PDF:

$$\begin{cases} \begin{cases} P(X = x_i) = \sum_{j=1}^{+\infty} P(X = x_i, Y = y_j), & \text{descrete} \\ f_X(x) = \int_{-\infty}^{+\infty} f(x,y) dy, & \text{continuous} \end{cases} \\ \begin{cases} p(Y = y_j) = \sum_{i=1}^{+\infty} P(X = x_i, Y = y_j), & \text{descrete} \\ f_Y(y) = \int_{-\infty}^{+\infty} f(x,y) dx, & \text{continuous} \end{cases} \end{cases} \tag{2.6}$$

2.2.2.3 Conditional distribution

Conditional PMF:

$$p(X = x_i|Y = y_j) = \frac{p(X = x_i, Y = y_j)}{p(Y = y_j)} \text{ if } p(Y) > 0 \tag{2.7}$$

The pmf $p(X|Y)$ is called **conditional probability**.

Conditional PDF:

$$f_{X|Y}(x|y) = \frac{f(x,y)}{f_Y(y)} \tag{2.8}$$

2.2.3 Bayes rule

$$p(Y = y|X = x) = \frac{p(X = x, Y = y)}{p(X = x)} = \frac{p(X = x|Y = y)p(Y = y)}{\sum_{y'} p(X = x|Y = y')p(Y = y')} \tag{2.9}$$

2.2.3.1 Example: Generative classifiers

$$p(y=c|X,\boldsymbol{\theta}) = \frac{p(X|y=c,\boldsymbol{\theta})p(y=c|\boldsymbol{\theta})}{\sum_{c'} p(X|y=c',\boldsymbol{\theta})p(y=c'|\boldsymbol{\theta})} \tag{2.10}$$

This is called a **generative classifier**, since it specifies how to generate the data using the class conditional density $p(x|y=c)$ and the class prior $p(y=c)$. An alternative approach is to directly fit the class posterior, $p(y=c|x)$;this is known as a **discriminative classifier**.

2.2.4 Independence and conditional independence

We say X and Y are unconditionally independent or marginally independent, denoted $X \perp Y$, if we can represent the joint as the product of the two marginals, i.e.,

$$X \perp Y = P(X,Y) = P(X)P(Y) \tag{2.11}$$

We say X and Y are conditionally independent(CI) given Z if the conditional joint can be written as a product of conditional marginals:

$$X \perp Y|Z = P(X,Y|Z) = P(X|Z)P(Y|Z) \tag{2.12}$$

2.2.5 Quantiles

Since the cdf F is a monotonically increasing function, it has an inverse; let us denote this by F^{-1}. If F is the cdf of X , then $F^{-1}(\alpha)$ is the value of x_α such that $P(X \leq x_\alpha) = \alpha$; this is called the α quantile of F. The value $F^{-1}(0.5)$ is the **median** of the distribution, with half of the probability mass on the left, and half on the right. The values $F^{-1}(0.25)$ and $F^{1}(0.75)$are the lower and upper **quartiles**.

2.2.6 Mean and variance

The most familiar property of a distribution is its **mean**,or **expected value**, denoted by μ. For discrete rvs, it is defined as $\mathbb{E}[X] \triangleq \sum_{x \in \mathcal{X}} xp(x)$, and for continuous rvs, it is defined as $\mathbb{E}[X] \triangleq \int_{\mathcal{X}} xp(x)dx$. If this integral is not finite, the mean is not defined (we will see some examples of this later).

The **variance** is a measure of the spread of a distribution, denoted by σ^2. This is defined as follows:

$$var[X] = \mathbb{E}[(X-\mu)^2] = \int (x-\mu)^2 p(x)dx \tag{2.13}$$

$$= \int x^2 p(x)dx \int \mu^2 p(x)dx - 2\mu \int xp(x)dx = \mathbb{E}[X^2] - \mu^2 \tag{2.14}$$

from which we derive the useful result

$$\mathbb{E}[X^2] = \sigma^2 + \mu^2 \tag{2.15}$$

The **standard deviation** is defined as

$$std[X] \triangleq \sqrt{var[X]} \tag{2.16}$$

This is useful since it has the same units as X itself.

2.3 Some common discrete distributions

In this section, we review some commonly used parametric distributions defined on discrete state spaces, both finite and countably infinite.

2.3.1 The binomial and Bernoulli distributions

2.3.2 The multinomial and multinoulli distributions

2.3.3 The Poisson distribution

2.3.4 The empirical distribution

2.4 Some common continuous distributions

In this section we present some commonly used univariate (one-dimensional) continuous probability distributions.

2.4.1 Gaussian (normal) distribution

2.4.2 Degenerate pdf

2.4.3 The Laplace distribution

2.4.4 The gamma distribution

2.4.5 The beta distribution

2.4.6 Pareto distribution

2.5 Information theory

2.5.1 Entropy

The entropy of a random variable X with distribution p, denoted by $\mathbb{H}(X)$ or sometimes $\mathbb{H}(p)$, is a measure of its uncertainty. In particular, for a discrete variable with K states, it is defined by

$$\mathbb{H}(X) \triangleq -\sum_{k=1}^{K} p(X=k)\log_2 p(X=k) \tag{2.17}$$

Usually we use log base 2, in which case the units are called **bits**(short for binary digits). If we use log base e , the units are called **nats**.

The discrete distribution with maximum entropy is the uniform distribution (see Section XXX for a proof). Hence for a K-ary random variable, the entropy is maximized if $p(x=k)=1/K$; in this case, $\mathbb{H}(X)=\log_2 K$.

Conversely, the distribution with minimum entropy (which is zero) is any **delta-function** that puts all its mass on one state. Such a distribution has no uncertainty.

2.5.2 KL divergence

One way to measure the dissimilarity of two probability distributions, p and q , is known as the **Kullback-Leibler divergence**(**KL divergence**)or **relative entropy**. This is defined as follows:

$$\mathbb{KL}(P||Q) \triangleq \sum_x p(x) \log_2 \frac{p(x)}{q(x)} \tag{2.18}$$

where the sum gets replaced by an integral for pdfs[6]. The KL divergence is only defined if P and Q both sum to 1 and if $q(x) = 0$ implies $p(x) = 0$ for all x(absolute continuity). If the quantity $0 \ln 0$ appears in the formula, it is interpreted as zero because $\lim\limits_{x \to 0} x \ln x$. We can rewrite this as

$$\mathbb{KL}(p||q) \triangleq \sum_x p(x) \log_2 p(x) - \sum_{k=1}^{K} p(x) \log_2 q(x) = \mathbb{H}(p) - \mathbb{H}(p,q) \tag{2.19}$$

where $\mathbb{H}(p,q)$ is called the **cross entropy**,

$$\mathbb{H}(p,q) = \sum_x p(x) \log_2 q(x) \tag{2.20}$$

One can show (Cover and Thomas 2006) that the cross entropy is the average number of bits needed to encode data coming from a source with distribution p when we use model q to define our codebook. Hence the regular entropy $\mathbb{H}(p) = \mathbb{H}(p,p)$, defined in section §2.5.1,is the expected number of bits if we use the true model, so the KL divergence is the diference between these. In other words, the KL divergence is the average number of *extra* bits needed to encode the data, due to the fact that we used distribution q to encode the data instead of the true distribution p.

The extra number of bits interpretation should make it clear that $\mathbb{KL}(p||q) \geq 0$, and that the KL is only equal to zero if $q = p$. We now give a proof of this important result.

Theorem 2.1. ***(Information inequality)*** $\mathbb{KL}(p||q) \geq 0$ *with equality iff* $p = q$.

One important consequence of this result is that *the discrete distribution with the maximum entropy is the uniform distribution*.

2.5.3 Mutual information

Definition 2.1. Mutual information or **MI**, is defined as follows:

$$\mathbb{I}(X;Y) \triangleq \mathbb{KL}(P(X,Y)||P(X)P(X)) = \sum_x \sum_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \tag{2.21}$$

We have $\mathbb{I}(X;Y) \geq 0$ with equality if $P(X,Y) = P(X)P(Y)$. That is, the MI is zero if the variables are independent.

To gain insight into the meaning of MI, it helps to re-express it in terms of joint and conditional entropies. One can show that the above expression is equivalent to the following:

$$\begin{aligned}
\mathbb{I}(X;Y) &= \mathbb{H}(X) - \mathbb{H}(X|Y) && (2.22)\\
&= \mathbb{H}(Y) - \mathbb{H}(Y|X) && (2.23)\\
&= \mathbb{H}(X) + \mathbb{H}(Y) - \mathbb{H}(X,Y) && (2.24)\\
&= \mathbb{H}(X,Y) - \mathbb{H}(X|Y) - \mathbb{H}(Y|X) && (2.25)
\end{aligned}$$

[6] The KL divergence is not a distance, since it is asymmetric. One symmetric version of the KL divergence is the **Jensen-Shannon divergence**, defined as $JS(p_1, p_2) = 0.5\mathbb{KL}(p_1||q) + 0.5\mathbb{KL}(p_2||q)$,where $q = 0.5p_1 + 0.5p_2$

where $\mathbb{H}(X)$ and $\mathbb{H}(Y)$ are the **marginal entropies**, $\mathbb{H}(X|Y)$ and $\mathbb{H}(Y|X)$ are the **conditional entropies**, and $\mathbb{H}(X,Y)$ is the **joint entropy** of X and Y, see Fig. 2.1[7].

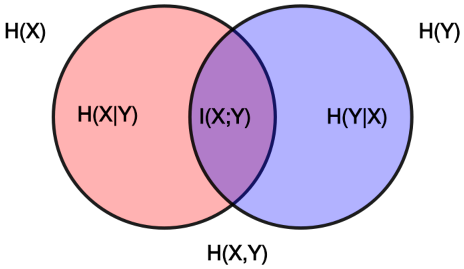

Fig. 2.1: Individual $\mathbb{H}(X), \mathbb{H}(Y)$, joint $\mathbb{H}(X,Y)$, and conditional entropies for a pair of correlated subsystems X,Y with mutual information $\mathbb{I}(X;Y)$.

Intuitively, we can interpret the MI between X and Y as the reduction in uncertainty about X after observing Y, or, by symmetry, the reduction in uncertainty about Y after observing X.

A quantity which is closely related to MI is the **pointwise mutual information** or **PMI**. For two events (not random variables) x and y, this is defined as

$$PMI(x,y) \triangleq \log \frac{p(x,y)}{p(x)p(y)} = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)} \tag{2.26}$$

This measures the discrepancy between these events occuring together compared to what would be expected by chance. Clearly the MI of X and Y is just the expected value of the PMI. Interestingly, we can rewrite the PMI as follows:

$$PMI(x,y) = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)} \tag{2.27}$$

This is the amount we learn from updating the prior $p(x)$ into the posterior $p(x|y)$, or equivalently, updating the prior $p(y)$ into the posterior $p(y|x)$.

[7] http://en.wikipedia.org/wiki/Mutual_information

Chapter 3
Perceptron

3.1 Representation

$$\mathcal{H} : y = f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}^T \boldsymbol{x} + b) \tag{3.1}$$

where $\text{sign}(x) = \begin{cases} +1, & x \geq 0 \\ -1, & x < 0 \end{cases}$, see Fig. 3.1[8].

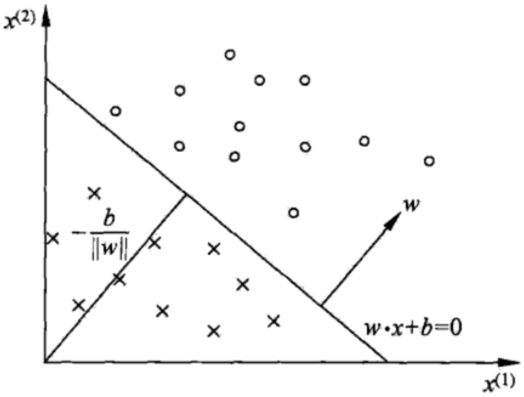

Fig. 3.1: Perceptron

3.2 Evaluation

$$L(\boldsymbol{w}, b) = -y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \tag{3.2}$$
$$R_{emp}(f) = -\sum_i y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \tag{3.3}$$
$$\tag{3.4}$$

[8] https://en.wikipedia.org/wiki/Perceptron

3.3 Optimization

3.3.1 Primal form

Stochastic gradient descent, the pseudo code is as follows:

```
$\boldsymbol{w} \leftarrow 0$; $b \leftarrow 0$; $k \leftarrow 0$;
while no mistakes made within the for loop do
    for $i \leftarrow 1$ to $N$ do
        if $y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \leq 0$ then
            $\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta y_i \boldsymbol{x}_i$;
            $b \leftarrow b + \eta y_i$;
            $k \leftarrow k + 1$;
        end
    end
end
```

Algorithm 1: Perceptron learning algorithm, primal form

3.3.1.1 Convergency

Theorem 3.1. *(Novikoff) If traning data set $\mathcal{D}$ is linearly separable, then*

1. *There exists a hyperplane denoted as $\widehat{\boldsymbol{w}}_{opt} \cdot \boldsymbol{x} + b_{opt} = 0$ which can correctly seperate all samples, and*

$$\exists \gamma > 0,\ \forall i,\ y_i(\boldsymbol{w}_{opt} \cdot \boldsymbol{x}_i + b_{opt}) \geq \gamma \tag{3.5}$$

2.

$$k \leq \left(\frac{R}{\gamma}\right)^2, \text{ where } R = \max_{1 \leq i \leq N} ||\widehat{\boldsymbol{x}}_i|| \tag{3.6}$$

Proof. (1) let $\gamma = \min_i y_i(\boldsymbol{w}_{opt} \cdot \boldsymbol{x}_i + b_{opt})$, then we get $y_i(\boldsymbol{w}_{opt} \cdot \boldsymbol{x}_i + b_{opt}) \geq \gamma$.

(2) The algorithm start from $\widehat{\boldsymbol{x}}_0 = 0$, if a instance is misclassified, then update the weight. Let $\widehat{\boldsymbol{w}}_{k-1}$ denotes the extended weight before the k-th misclassified instance, then we can get

$$y_i(\widehat{\boldsymbol{w}}_{k-1} \cdot \widehat{\boldsymbol{x}}_i) = y_i(\boldsymbol{w}_{k-1} \cdot \boldsymbol{x}_i + b_{k-1}) \leq 0 \tag{3.7}$$

$$\widehat{\boldsymbol{w}}_k = \widehat{\boldsymbol{w}}_{k-1} + \eta y_i \widehat{\boldsymbol{x}}_i \tag{3.8}$$

We could infer the following two equations, the proof procedure are omitted.

1. $\widehat{\boldsymbol{w}}_k \cdot \widehat{\boldsymbol{w}}_{opt} \geq k\eta\gamma$
2. $||\widehat{\boldsymbol{w}}_k||^2 \leq k\eta^2 R^2$

From above two equations we get

$$\begin{aligned} & k\eta\gamma \leq \widehat{\boldsymbol{w}}_k \cdot \widehat{\boldsymbol{w}}_{opt} \leq ||\widehat{\boldsymbol{w}}_k|| \, ||\widehat{\boldsymbol{w}}_{opt}|| \leq \sqrt{k}\eta R \\ & k^2\gamma^2 \leq kR^2 \\ & \text{i.e. } k \leq \left(\frac{R}{\gamma}\right)^2 \end{aligned}$$

3.3.2 Dual form

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i \tag{3.9}$$

$$b = \sum_{i=1}^{N} \alpha_i y_i \tag{3.10}$$

$$f(\boldsymbol{x}) = \operatorname{sign}\left(\sum_{j=1}^{N} \alpha_j y_j \boldsymbol{x}_j \cdot \boldsymbol{x} + b\right) \tag{3.11}$$

```
$\boldsymbol{\alpha} \leftarrow 0$; $b \leftarrow 0$; $k \leftarrow 0$;
**while** *no mistakes made within the for loop* **do**
    **for** $i \leftarrow 1$ **to** $N$ **do**
        **if** $y_i \left(\sum_{j=1}^{N} \alpha_j y_j \boldsymbol{x}_j \cdot \boldsymbol{x}_i + b\right) \leq 0$ **then**
            $\boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha} + \eta$;
            $b \leftarrow b + \eta y_i$;
            $k \leftarrow k + 1$;
        **end**
    **end**
**end**
```

Algorithm 2: Perceptron learning algorithm, dual form

Chapter 4
K-Nearest Neighbors

4.1 Representation

$$y = f(\boldsymbol{x}) = \arg\min_{c} \sum_{\boldsymbol{x}_i \in N_k(\boldsymbol{x})} \mathbb{I}(y_i = c) \tag{4.1}$$

where $N_k(\boldsymbol{x})$ is the set of k points that are closest to point $\boldsymbol{x}$.

Usually use **k-d tree** to accelerate the process of finding k nearest points.

4.2 Evaluation

No training is needed.

4.3 Optimization

No training is needed.

Chapter 5
K-Means Clustering

5.1 Representation

$$y_j = k \text{ if } \|\boldsymbol{x}_j - \boldsymbol{\mu}_k\|_2^2 \text{ is minimal} \tag{5.1}$$

where $\boldsymbol{\mu}_k$ is the centroid of cluster k.

5.2 Evaluation

$$\arg\min_{\boldsymbol{\mu}} \sum_{j=1}^{N} \sum_{k=1}^{K} \gamma_{jk} \|\boldsymbol{x}_j - \boldsymbol{\mu}_k\|_2^2 \tag{5.2}$$

The hidden variable is γ_{jk}, which's meanining is:

$$\gamma_{jk} = \begin{cases} 1, & \text{if } \|\boldsymbol{x}_j - \boldsymbol{\mu}_k\|_2 \text{ is minimal for } \boldsymbol{\mu}_k \\ 0, & \text{otherwise} \end{cases}$$

5.3 Optimization

E-Step:

$$\gamma_{jk}^{(i+1)} = \begin{cases} 1, & \text{if } \|\boldsymbol{x}_j - \boldsymbol{\mu}_k^{(i)}\|_2 \text{ is minimal for } \boldsymbol{\mu}_k^{(i)} \\ 0, & \text{otherwise} \end{cases} \tag{5.3}$$

M-Step:

$$\boldsymbol{\mu}_k^{(i+1)} = \frac{\sum_{j=1}^{N} \gamma_{jk}^{(i+1)} \boldsymbol{x}_j}{\sum \gamma_{jk}^{(i+1)}} \tag{5.4}$$

5.4 Tricks

5.4.1 Choosing k

5.4.2 Choosing the initial centroids(seeds)

5.4.2.1 K-means++

The intuition that spreading out the k initial cluster centers is a good thing is behind this approach: the first cluster center is chosen uniformly at random from the data points that are being clustered, after which each subsequent cluster center is chosen from the remaining data points with probability proportional to its squared distance from the point's closest existing cluster center[9].

The exact algorithm is as follows:

1. Choose one center uniformly at random from among the data points.
2. For each data point $\boldsymbol{x}$, compute $D(\boldsymbol{x})$, the distance between $\boldsymbol{x}$ and the nearest center that has already been chosen.
3. Choose one new data point at random as a new center, using a weighted probability distribution where a point $\boldsymbol{x}$ is chosen with probability proportional to $D(\boldsymbol{x})^2$.
4. Repeat Steps 2 and 3 until k centers have been chosen.
5. Now that the initial centers have been chosen, proceed using standard k-means clustering.

5.5 Reference

1. cheat-sheet: Algorithm for supervised and unsupervised learning by Emanuel Ferm http://t.cn/hD0Stf

[9] http://en.wikipedia.org/wiki/K-means++

Chapter 6
Naive Bayes

6.1 Representation

$$y_i = f(\boldsymbol{x}_i) = \arg\max_{c_k} P(c_k|\boldsymbol{x}_i) = P(y = c_k)\prod_{j=1}^{D} P(x_{ij}|y = c_k) \tag{6.1}$$

6.2 Evaluation

$$R_{exp}(f) = E_{(X,Y)}\sum_{k=1}^{K}\left[L\left(Y, f(X)\right)\right]P(c_k|X) \tag{6.2}$$

where $L\left(Y, f(X)\right)$ is 0-1 loss function.

6.3 Optimization

The likelihood function is:

$$\begin{aligned}P(\mathcal{D}|\boldsymbol{\pi},\boldsymbol{\theta}) &= \prod_{i=1}^{N} P(\boldsymbol{x}_i, y_i)\\ &= \prod_{i=1}^{N}\left[P(y_i)\prod_{j=1}^{D} P(x_{ij}|y_i)\right]\end{aligned}$$

For discrete distributions, we have MLE solution as follows:

$$\pi_k = P(Y = c_k) = \frac{\sum_{i=1}^{N}\mathbb{I}(y_i = c_k)}{N} \tag{6.3}$$

$$\theta_{kij} = P(x_{ij} = a_{ij}|Y = c_k) = \frac{\sum_{i=1}^{N}\mathbb{I}(x_{ij} = a_{ij}, y_i = c_k)}{\sum_{i=1}^{N}\mathbb{I}(y_i = c_k)} \tag{6.4}$$

Chapter 7
Decision Tree

Chapter 8
Linear Regression

8.1 Representation

$$y = f(x) = \boldsymbol{w}^T\boldsymbol{x} + b \quad , \boldsymbol{x} \in \mathbb{R}^D, y \in \mathbb{R} \tag{8.1}$$

If we extend $\boldsymbol{x}^T = (1, x_1, x_2, \cdots, x_D), \boldsymbol{w}^T = (b, w_1, w_2, \cdots, w_D)$, then

$$y = f(x) = \boldsymbol{w}^T\boldsymbol{x} \tag{8.2}$$

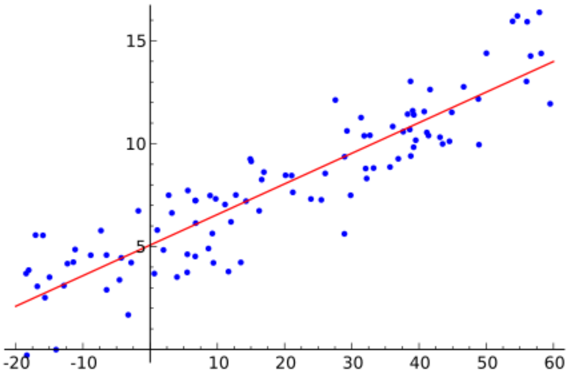

Fig. 8.1: Example of simple linear regression, which has one independent variable, from Wikipedia

8.2 Evaluation

$$R_{emp}(f) = \frac{1}{2N}\sum_{i=1}^{N}(y - f(\boldsymbol{x}))^2 = \frac{1}{2N}\sum_{i=1}^{N}\left(y - \boldsymbol{w}^T\boldsymbol{x}\right)^2 \tag{8.3}$$

$$J(\boldsymbol{w}) \triangleq R_{emp}(f) \tag{8.4}$$

Prerequisite: define error term $\varepsilon_i \triangleq y_i - \boldsymbol{w}^T\boldsymbol{x}_i$, ε_i are IID (independently and identically distributed) according to a Gaussian distribution.

8.3 Optimization

8.3.1 Normal Equation

When dataset is small, use Normal Equation to compute $\boldsymbol{w}$ directly.

$$y = f(x) = \boldsymbol{w}^T \boldsymbol{x} \boldsymbol{w} = (X^T X)^{-1} X^T \boldsymbol{y} \tag{8.5}$$

where

$$\begin{aligned} X &= \left[\boldsymbol{x}_1^T, \boldsymbol{x}_2^T, \cdots, \boldsymbol{x}_N^T\right]^T \\ \boldsymbol{y} &= (y_1, y_2, \cdots, y_N)^T \end{aligned}$$

Proof. We now state without proof some facts of matrix derivatives (we wont need all of these at this section).

$$tr A \triangleq \sum_{i=1}^{n} A_{ii}$$

$$\frac{\partial}{\partial A} AB = B^T \tag{8.6}$$

$$\frac{\partial}{\partial A^T} f(A) = \left[\frac{\partial}{\partial A} f(A)\right]^T \tag{8.7}$$

$$\frac{\partial}{\partial A} ABA^T C = CAB + C^T AB^T \tag{8.8}$$

$$\frac{\partial}{\partial A} |A| = |A|(A^{-1})^T \tag{8.9}$$

$$\begin{aligned} J(\boldsymbol{w}) &= \frac{1}{2N}(X\boldsymbol{w}-\boldsymbol{y})^T(X\boldsymbol{w}-\boldsymbol{y}) \\ \frac{\partial J}{\boldsymbol{w}} &= \frac{1}{2N}\frac{\partial}{\boldsymbol{w}}(\boldsymbol{w}^T X^T X \boldsymbol{w} - \boldsymbol{w}^T X^T \boldsymbol{y} - \boldsymbol{y}^T X \boldsymbol{w} + \boldsymbol{y}^T \boldsymbol{y}) \\ &= \frac{1}{2N}\frac{\partial}{\boldsymbol{w}}(\boldsymbol{w}^T X^T X \boldsymbol{w} - \boldsymbol{w}^T X^T \boldsymbol{y} - \boldsymbol{y}^T X \boldsymbol{w}) \\ &= \frac{1}{2N}\frac{\partial}{\boldsymbol{w}} tr(\boldsymbol{w}^T X^T X \boldsymbol{w} - \boldsymbol{w}^T X^T \boldsymbol{y} - \boldsymbol{y}^T X \boldsymbol{w}) \\ &= \frac{1}{2N}\frac{\partial}{\boldsymbol{w}}(tr\boldsymbol{w}^T X^T X \boldsymbol{w} - 2tr\boldsymbol{y}^T X \boldsymbol{w}) \end{aligned}$$

Combining Equations (8.7) and (8.8), we find that

$$\frac{\partial}{\partial A^T} ABA^T C = B^T A^T C^T + BA^T C$$

Let $A^T = \boldsymbol{w}, B = B^T = X^T X$, and $C = I$, Hence,

$$\begin{aligned} \frac{\partial J}{\boldsymbol{w}} &= \frac{1}{2N}(X^T X \boldsymbol{w} + X^T X \boldsymbol{w} - 2X^T \boldsymbol{y}) \\ &= \frac{1}{2N}(X^T X \boldsymbol{w} - X^T \boldsymbol{y}) \\ \frac{\partial J}{\boldsymbol{w}} &= 0 \Rightarrow X^T X \boldsymbol{w} - X^T \boldsymbol{y} = 0 \\ X^T X \boldsymbol{w} &= X^T \boldsymbol{y} \\ \boldsymbol{w} &= (X^T X)^{-1} X^T \boldsymbol{y} \end{aligned}$$

8.3.2 SGD

When dataset is large, use stochastic gradient descent(SGD).

$$\frac{\partial}{\partial \boldsymbol{w}}J(\boldsymbol{w}) = [y_i - f(\boldsymbol{x}_i)]\,\boldsymbol{x}_i \tag{8.10}$$

$$\begin{aligned}\boldsymbol{w} &= \boldsymbol{w} - \alpha\frac{\partial}{\partial \boldsymbol{w}}f(\boldsymbol{w}) \\ &= \boldsymbol{w} - \alpha\,[y_i - f(\boldsymbol{x}_i)]\,\boldsymbol{x}_i \end{aligned} \tag{8.11}$$

Chapter 9
Logistic Regression

9.1 Binomial Logistic Regression Model

9.1.1 Representation

$$P(Y=1|\boldsymbol{x}) = \frac{\exp(\boldsymbol{w}^T\boldsymbol{x})}{1+\exp(\boldsymbol{w}^T\boldsymbol{x})} \tag{9.1}$$

$$P(Y=0|\boldsymbol{x}) = \frac{1}{1+\exp(\boldsymbol{w}^T\boldsymbol{x})} \tag{9.2}$$

where $\boldsymbol{w}=(w_1,w_2,\cdots,w_n,b)$, $\boldsymbol{x}=(x_1,x_2,\cdots,1)$.

9.1.2 Evaluation

$$\max_{\boldsymbol{w}} l(\boldsymbol{w}) \tag{9.3}$$

where $l(\boldsymbol{w})$ is log likelihood function

$$\begin{aligned}
\pi(\boldsymbol{x}_i) &\triangleq P(Y=1|\boldsymbol{x}_i) \\
l(\boldsymbol{w}) &= \log\left\{\prod_{i=1}^{N}[\pi(\boldsymbol{x}_i)]^{y_i}[1-\pi(\boldsymbol{x}_i)]^{1-y_i}\right\} \\
&= \sum_{i=1}^{N}[y_i\log\pi(\boldsymbol{x}_i)+(1-y_i)\log(1-\pi(\boldsymbol{x}_i))] \\
&= \sum_{i=1}^{N}\left[y_i\log\frac{\pi(\boldsymbol{x}_i)}{1-\pi(\boldsymbol{x}_i)}+\log(1-\pi(\boldsymbol{x}_i))\right] \\
&= \sum_{i=1}^{N}[y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i)-\log(1+\exp(\boldsymbol{w}\cdot\boldsymbol{x}_i))]
\end{aligned}$$

9.1.3 Optimization

We can use stochastic gradient ascent(SGA) and quasi Newton method, etc.

9.1.3.1 SGA

$$\frac{\partial}{\partial \boldsymbol{w}} l(\boldsymbol{w}) = [y_i - \pi(\boldsymbol{x}_i)]\, \boldsymbol{x}_i \tag{9.4}$$

$$\begin{aligned} \boldsymbol{w} &= \boldsymbol{w} + \alpha \frac{\partial}{\partial \boldsymbol{w}} l(\boldsymbol{w}) \\ &= \boldsymbol{w} + \alpha\, [y_i - \pi(\boldsymbol{x}_i)]\, \boldsymbol{x}_i \end{aligned} \tag{9.5}$$

Chapter 10
Support Vector Machines

10.1 Primal form

10.1.1 Representation

$$\mathcal{H}: y = f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}\boldsymbol{x} + b) \tag{10.1}$$

10.1.2 Evaluation

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{10.2}$$

$$s.t. \quad y_i(\boldsymbol{w}\boldsymbol{x}_i + b) \geqslant 1, i = 1,2,\dots,N \tag{10.3}$$

10.2 Dual form

10.2.1 Representation

$$\mathcal{H}: y = f(\boldsymbol{x}) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i y_i (\boldsymbol{x} \cdot \boldsymbol{x}_i) + b\right) \tag{10.4}$$

10.2.2 Evaluation

$$\min_{\alpha} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j) - \sum_{i=1}^{N} \alpha_i \tag{10.5}$$

$$s.t. \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{10.6}$$

$$\alpha_i \geqslant 0, i = 1,2,\dots,N \tag{10.7}$$

10.3 Primal form with regularization

10.3.1 Representation

$$\mathcal{H}: y = f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}\boldsymbol{x} + b) \tag{10.8}$$

10.3.2 Evaluation

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i \tag{10.9}$$

$$s.t. \quad y_i(\boldsymbol{w}\boldsymbol{x}_i + b) \geqslant 1 - \xi_i, i = 1, 2, \ldots, N \tag{10.10}$$

$$\xi_i \geqslant 0, i = 1, 2, \ldots, N \tag{10.11}$$

10.4 Dual form with regularization

10.4.1 Representation

$$\mathcal{H}: y = f(\boldsymbol{x}) = \text{sign}\left(\sum_{i=1}^{N}\alpha_i y_i(\boldsymbol{x} \cdot \boldsymbol{x}_i) + b\right) \tag{10.12}$$

10.4.2 Evaluation

$$\min_{\alpha} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(\boldsymbol{x}_i \cdot \boldsymbol{x}_j) - \sum_{i=1}^{N}\alpha_i \tag{10.13}$$

$$s.t. \quad \sum_{i=1}^{N}\alpha_i y_i = 0 \tag{10.14}$$

$$0 \leqslant \alpha_i \leqslant C, i = 1, 2, \ldots, N \tag{10.15}$$

$$\alpha_i = 0 \Rightarrow y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geqslant 1 \tag{10.16}$$

$$\alpha_i = C \Rightarrow y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \leqslant 1 \tag{10.17}$$

$$0 < \alpha_i < C \Rightarrow y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) = 1 \tag{10.18}$$

10.5 Hinge Loss

Linear support vector machines can also be interpreted as hinge loss minimization:

$$\min_{\boldsymbol{w},b}\sum_{i=1}^{N}[1 - y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b)]_+ + \lambda\|\boldsymbol{w}\|^2 \tag{10.19}$$

where $L(X,Y)$ is a hinge loss function

$$[z]_+ = \begin{cases} z, & z > 0 \\ 0, & z \leqslant 0 \end{cases} \tag{10.20}$$

Proof. We can write equation (10.19) as equations (10.9) $\sim$ (10.11).

Define

$$\xi_i \triangleq 1 - y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b), \xi_i \geqslant 0 \tag{10.21}$$

Then $\boldsymbol{w}, b, \xi_i$ satisfy the constraints (10.9) and (10.10). And objective function (10.11) can be written as

$$\min_{\boldsymbol{w},b} \sum_{i=1}^{N} \xi_i + \lambda \|\boldsymbol{w}\|^2$$

If $\lambda = \dfrac{1}{2C}$, then

$$\min_{\boldsymbol{w},b} \frac{1}{C} \left(\frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_{i=1}^{N} \xi_i \right) \tag{10.22}$$

It is equivalent to equation (10.9).

10.6 Kernels

10.7 Optimization

SMO, QP, SGD, etc.

Chapter 11
AdaBoost

11.1 Representation

$$y = \text{sign}(f(\boldsymbol{x})) = \text{sign}\left(\sum_{i=1}^{m} \alpha_m G_m(\boldsymbol{x})\right) \tag{11.1}$$

where $G_m(\boldsymbol{x})$ are sub classifiers.

11.2 Evaluation

$$L(y, f(\boldsymbol{x})) = \exp[-yf(\boldsymbol{x})] \text{ i.e., exponential loss function}$$

$$(\alpha_m, G_m(x)) = \arg\min_{\alpha,G} \sum_{i=1}^{N} \exp\left[-y_i(f_{m-1}(\boldsymbol{x}_i) + \alpha G(\boldsymbol{x}_i))\right] \tag{11.2}$$

Define $\bar{w}_{mi} = \exp\left[-y_i(f_{m-1}(\boldsymbol{x}_i)\right]$, which is constant w.r.t. α, G

$$(\alpha_m, G_m(x)) = \arg\min_{\alpha,G} \sum_{i=1}^{N} \bar{w}_{mi} \exp\left(-y_i \alpha G(x_i)\right) \tag{11.3}$$

11.3 Optimization

11.3.1 Input

$$\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y2), \ldots, (\boldsymbol{x}_N, y_N)\} \text{ ,where } \boldsymbol{x}_i \in \mathbb{R}^D,\ y_i \in \{-1, +1\}$$
$$\text{Weak classifiers } \{G_1, G_2, \ldots, G_m\}$$

11.3.2 Output

Final classifier: $G(x)$

11.3.3 Algorithm

1. Initialize the weights' distribution of training data(when $m = 1$)

$$\mathcal{D}_1 = (w_{11}, w_{12}, \cdots, w_{1n}) = (\frac{1}{N}, \frac{1}{N}, \cdots, \frac{1}{N}),\ i = 1, 2, \cdots, N$$

2. Iterate over $m = 1, 2, \ldots, M$
   (a) Use training data with current weights' distribution $\mathcal{D}_m$ to get a classifier $G_m(\boldsymbol{x})$
   (b) Compute the error rate of $G_m(\boldsymbol{x})$ over the training data

$$e_m = P(G_m(\boldsymbol{x}_i) \neq y_i) = \sum_{i=1}^{N} w_{mi} \mathbb{I}(G_m(\boldsymbol{x}_i) \neq y_i) \tag{11.4}$$

   (c) Compute the coefficient of classifier $G_m(x)$

$$\alpha_m = \frac{1}{2} \log \frac{1 - e_m}{e_m} \tag{11.5}$$

   (d) Update the weights' distribution of training data

$$w_{m+1,i} = \frac{w_{mi}}{Z_m} \exp(-\alpha_m y_i G_m(\boldsymbol{x}_i)) \tag{11.6}$$

   where Z_m is the normalizing constant

$$Z_m = \sum_{i=1}^{N} w_{mi} \exp(-\alpha_m y_i G_m(\boldsymbol{x}_i)) \tag{11.7}$$

3. Ensemble M weak classifiers

$$G(x) = \mathrm{sign} f(\boldsymbol{x}) = \mathrm{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(\boldsymbol{x})\right] \tag{11.8}$$

11.4 The upper bound of the training error of AdaBoost

Theorem 11.1. *The upper bound of the training error of AdaBoost is*

$$\frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(G(\boldsymbol{x}_i) \neq y_i) \leq \frac{1}{N} \sum_{i=1}^{N} \exp(-y_i f(\boldsymbol{x}_i)) = \prod_{m=1}^{M} Z_m \tag{11.9}$$

Note: the following equation would help proof this theorem

$$w_{mi} \exp(-\alpha_m y_i G_m(\boldsymbol{x}_i)) = Z_m w_{m+1,i} \tag{11.10}$$

Chapter 12
EM algorithm

12.1 Jensen's inequality

12.1.1 Convex function

Definition 12.1. A real valued function $f : X \to R$ defined on a convex set X in a vector space is called **convex function** if, for any two points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ in X and any $\lambda \in [0,1]$,

$$f(\lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2) \leq \lambda f(\boldsymbol{x}_1) + (1-\lambda) f(\boldsymbol{x}_2) \tag{12.1}$$

The function f is said to be **strictly convex** if

$$f(\lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2) < \lambda f(\boldsymbol{x}_1) + (1-\lambda) f(\boldsymbol{x}_2) \tag{12.2}$$

Definition 12.2. A function f is said to be (strictly) **concave** if $-f$ is (strictly) convex.

Theorem 12.1. *If $f(x)$ is twice differentiable on $[a,b]$ and $f''(x) \geq 0$ on $[a,b]$ then $f(x)$ is convex on $[a,b]$.*

Proposition 12.1. $\log(x)$ *is strictly convex on* $(0, \infty)$.

12.1.2 Jensen's inequality

Theorem 12.2. *Let f be a convex function defined on a convex set X . If $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n \in X$ and $\lambda_1, \lambda_2, \cdots, \lambda_n \geq 0$ with $\sum\limits_{i=1}^{n} \lambda_i = 1$,*

$$f\left(\sum_{i=1}^{n} \lambda_i \boldsymbol{x}_i\right) \leq \sum_{i=1}^{n} \lambda_i f(\boldsymbol{x}_i) \tag{12.3}$$

Proposition 12.2.

$$\log\left(\sum_{i=1}^{n} \lambda_i \boldsymbol{x}_i\right) \geq \sum_{i=1}^{n} \lambda_i \log(\boldsymbol{x}_i) \tag{12.4}$$

12.2 EM algorithm

The EM algorithm is an efficient iterative procedure to compute the Maximum Likelihood (ML) estimate in the presence of missing or hidden data.

Each iteration of the EM algorithm consists of two processes: The E-step, and the M-step. In the expectation, or E-step, the missing data are estimated given the observed data and current estimate of the model parameters. This is achieved using the conditional expectation, explaining the choice of terminology. In the M-step, the likelihood function is maximized under the assumption that the missing data are known. The estimate of the missing data from the E-step are used in lieu of the actual missing data.

```
input  : observed data $\mathcal{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \cdots, \boldsymbol{x}^{(n)}\}$,joint distribution $P(\mathcal{X}, \boldsymbol{z}|\boldsymbol{\theta})$
output: model's parameters $\boldsymbol{\theta}$
// 1. identify hidden variables $\boldsymbol{z}$, write out the log likelihood function $l(\mathcal{X}, \boldsymbol{z}|\boldsymbol{\theta})$
$\boldsymbol{\theta}^{(0)}$ = ... // initialize
while (!convergency) do
    // 2. E-step: plug in $P(\mathcal{X}, \boldsymbol{z}|\boldsymbol{\theta})$, derive the formula of $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$
    $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)}) = \mathbb{E}_{\boldsymbol{z}|\mathcal{X}, \boldsymbol{\theta}^{(i)}} [\log P(\mathcal{X}, \boldsymbol{z}|\boldsymbol{\theta})]$
    // 3. M-step: find $\boldsymbol{\theta}$ that maximizes the value of $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$
    $\boldsymbol{\theta}^{(i+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$
end
```

Algorithm 3: EM algorithm

12.3 Derivation of the EM algorithm

The log likelihood function is given by

$$
\begin{aligned}
l(\boldsymbol{\theta}) &= \log P(\mathcal{X}|\boldsymbol{\theta}) \text{ ,where } \log P(\mathcal{X}|\boldsymbol{\theta}) = \sum_{i=1}^{n} \log P(\boldsymbol{x}^{(i)}|\boldsymbol{\theta}) \\
&= \log \sum_{\boldsymbol{z}} P(\mathcal{X}, \boldsymbol{z}|\boldsymbol{\theta}) \\
&= \log \sum_{\boldsymbol{z}} P(\mathcal{X}|\boldsymbol{z}, \boldsymbol{\theta}) P(\boldsymbol{z}|\boldsymbol{\theta}) \\
l(\boldsymbol{\theta}) - l(\boldsymbol{\theta}^{(i)}) &= \log \left[\sum_{\boldsymbol{z}} P(\mathcal{X}|\boldsymbol{z}, \boldsymbol{\theta}) P(\boldsymbol{z}|\boldsymbol{\theta}) \right] - \log P(\mathcal{X}|\boldsymbol{\theta}^{(i)}) \\
&= \log \left[\sum_{\boldsymbol{z}} P(\mathcal{X}|\boldsymbol{z}, \boldsymbol{\theta}) P(\boldsymbol{z}|\boldsymbol{\theta}) \frac{P(\boldsymbol{z}|\mathcal{X}, \theta^{(i)})}{P(\boldsymbol{z}|\mathcal{X}, \theta^{(i)})} \right] - \log P(\mathcal{X}|\boldsymbol{\theta}^{(i)}) \\
&= \log \left[\sum_{\boldsymbol{z}} P(\boldsymbol{z}|\mathcal{X}, \theta^{(i)}) \frac{P(\mathcal{X}|\boldsymbol{z}, \boldsymbol{\theta}) P(\boldsymbol{z}|\boldsymbol{\theta})}{P(\boldsymbol{z}|\mathcal{X}, \theta^{(i)})} \right] - \log P(\mathcal{X}|\boldsymbol{\theta}^{(i)}) \\
&\geq \sum_{\boldsymbol{z}} P(\boldsymbol{z}|\mathcal{X}, \theta^{(i)}) \log \left[\frac{P(\mathcal{X}|\boldsymbol{z}, \boldsymbol{\theta}) P(\boldsymbol{z}|\boldsymbol{\theta})}{P(\boldsymbol{z}|\mathcal{X}, \theta^{(i)})} \right] - \log P(\mathcal{X}|\boldsymbol{\theta}^{(i)}) \\
&= \sum_{\boldsymbol{z}} P(\boldsymbol{z}|\mathcal{X}, \theta^{(i)}) \log \left[\frac{P(\mathcal{X}|\boldsymbol{z}, \boldsymbol{\theta}) P(\boldsymbol{z}|\boldsymbol{\theta})}{P(\boldsymbol{z}|\mathcal{X}, \theta^{(i)}) P(\mathcal{X}|\boldsymbol{\theta}^{(i)})} \right] \\
&\triangleq B(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)}) \\
\Rightarrow &
\end{aligned}
$$

$$
\begin{aligned}
\boldsymbol{\theta}^{(i+1)} &= \arg\max_{\boldsymbol{\theta}} \left[l(\boldsymbol{\theta}^{(i)}) + B(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)}) \right] \\
&= \arg\max_{\boldsymbol{\theta}} \left\{ l(\boldsymbol{\theta}^{(i)}) + \sum_{\boldsymbol{z}} P(\boldsymbol{z}|\mathcal{X}, \boldsymbol{\theta}^{(i)}) \log \left[\frac{P(\mathcal{X}|\boldsymbol{z}, \boldsymbol{\theta})P(\boldsymbol{z}|\boldsymbol{\theta})}{P(\boldsymbol{z}|\mathcal{X}, \boldsymbol{\theta}^{(i)})P(\mathcal{X}|\boldsymbol{\theta}^{(i)})} \right] \right\} \\
&\quad \text{Now drop terms which are constant w.r.t. } \boldsymbol{\theta} \\
&= \arg\max_{\boldsymbol{\theta}} \left\{ \sum_{\boldsymbol{z}} P(\boldsymbol{z}|\mathcal{X}, \boldsymbol{\theta}^{(i)}) \log \left[P(\mathcal{X}|\boldsymbol{z}, \boldsymbol{\theta})P(\boldsymbol{z}|\boldsymbol{\theta}) \right] \right\} \\
&= \arg\max_{\boldsymbol{\theta}} \left\{ \sum_{\boldsymbol{z}} P(\boldsymbol{z}|\mathcal{X}, \boldsymbol{\theta}^{(i)}) \log \left[P(\mathcal{X}, \boldsymbol{z}|\boldsymbol{\theta}) \right] \right\} \\
&= \arg\max_{\boldsymbol{\theta}} \left\{ \mathbb{E}_{\boldsymbol{z}|\mathcal{X}, \boldsymbol{\theta}^{(i)}} \log \left[P(\mathcal{X}, \boldsymbol{z}|\boldsymbol{\theta}) \right] \right\} \qquad (12.5) \\
&\triangleq \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)}) \qquad (12.6)
\end{aligned}
$$

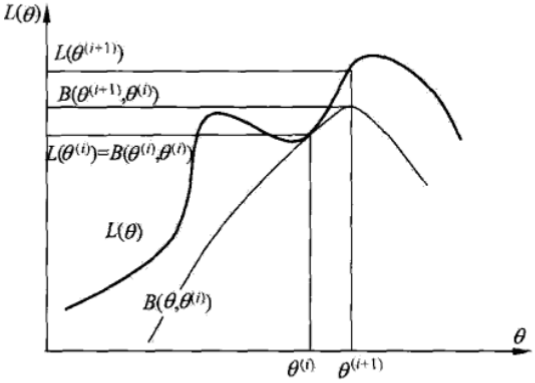

Fig. 12.1: Graphical interpretation of a single iteration of the EM algorithm: The function $B(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$ is bounded above by the log likelihood function $l(\boldsymbol{\theta})$. The functions are equal at $\boldsymbol{\theta} = \boldsymbol{\theta}^{(i)}$. The EM algorithm chooses $\boldsymbol{\theta}^{(i)}$ as the value of $\boldsymbol{\theta}$ for which $B(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$ is a maximum. Since $l(\boldsymbol{\theta}) \geq B(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$ increasing $B(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$ ensures that the value of the log likelihood function $l(\boldsymbol{\theta})$ is increased at each step.

12.4 Examples

12.4.1 Gaussian mixture model

Definition 12.3. In Gaussian mixture model(GMM) model, each base distribution in the mixture is a multivariate Gaussian with mean μ_k and covariance matrix σ_k. Thus the model has the form

$$P(x_i|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \phi(x_i|\mu_k, \sigma_k) \qquad (12.7)$$

Figure 12.2 shows a mixture of 3 Gaussians in 2D. Each mixture component is represented by a different set of eliptical contours. Given a sufficiently large number of mixture components, a GMM can be used to approximate any density defined on $\mathbb{R}^D$.

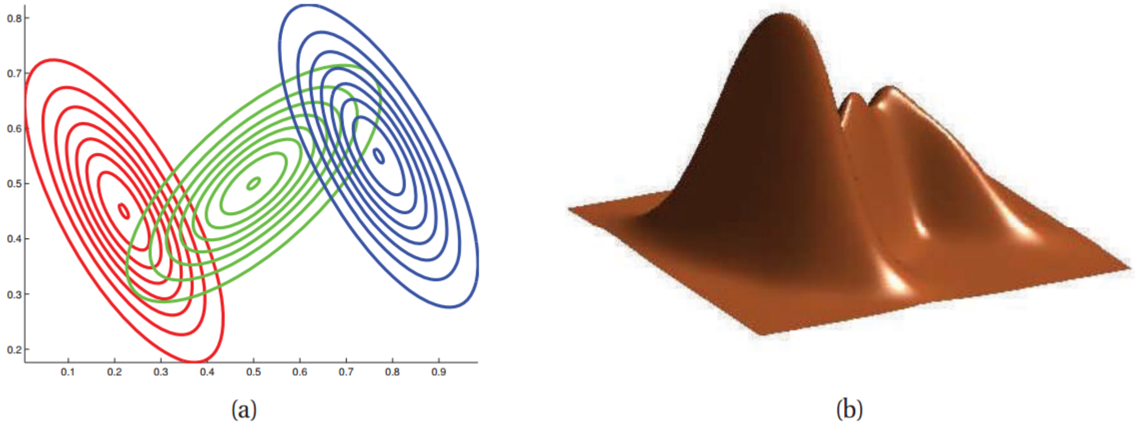

Fig. 12.2: A mixture of 3 Gaussians in 2d. (a) We show the contours of constant probability for eachcomponent in the mixture. (b) A surface plot of the overall density.

12.4.1.1 Identify hidden variable, write out the log likelihood function

Denote the hidden variable as γ_{jk}, which's meaning is:

$$\gamma_{jk} = \begin{cases} 1, & \text{the j-th sample comes from the k-th model} \\ 0, & \text{otherwise} \end{cases}$$

Given observed sample x_j and hidden variable γ_{jk}, the complete sample is $(x_j, \gamma_{j1}, \gamma_{j1}, \cdots, \gamma_{jk})$. Then the log likelihood function can be written as follows:

$$\begin{aligned}
P(\mathcal{X}, \gamma|\boldsymbol{\theta}) &= \prod_{j=1}^{N} P(x_j, \gamma_{j1}, \gamma_{j1}, \cdots, \gamma_{jk}|\boldsymbol{\theta}) \\
&= \prod_{j=1}^{N} \left\{ \prod_{k=1}^{K} [\pi_k \phi(x_i|\mu_k, \sigma_k)]^{\gamma_{jk}} \right\} \\
&= \prod_{k=1}^{K} \left\{ \prod_{j=1}^{N} [\pi_k \phi(x_i|\mu_k, \sigma_k)]^{\gamma_{jk}} \right\} \\
&= \prod_{k=1}^{K} \left\{ \pi_k^{n_k} \prod_{j=1}^{N} [\phi(x_i|\mu_k, \sigma_k)]^{\gamma_{jk}} \right\} \\
&= \prod_{k=1}^{K} \left\{ \pi_k^{n_k} \prod_{j=1}^{N} \left[\frac{1}{\sqrt{2\pi}\Sigma_k} \exp\left(-\frac{(x_j - \mu_k)^2}{2\sigma_k^2} \right) \right]^{\gamma_{jk}} \right\} \\
&\quad \text{where } n_k = \sum_{j=1}^{N} \gamma_{jk}, \sum_{k=1}^{K} n_k = N. \\
\Rightarrow \\
\log P(\mathcal{X}, \gamma|\boldsymbol{\theta}) &= \sum_{k=1}^{K} \left\{ n_k \log \pi_k + \sum_{j=1}^{N} \gamma_{jk} \left[\log(\frac{1}{\sqrt{2\pi}} - \log \sigma_k - \frac{(x_j - \mu_k)^2}{2\sigma_k^2}) \right] \right\}
\end{aligned} \tag{12.8}$$

12.4.1.2 E-step: derive the formula of $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$

$$
\begin{aligned}
Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(i)}) &= \mathbb{E}_{\boldsymbol{z}|\mathcal{X},\theta^{(i)}} \log\left[P(\mathcal{X},\boldsymbol{z}|\boldsymbol{\theta})\right] \\
&= \mathbb{E}_{\boldsymbol{\gamma}|\mathcal{X},\theta^{(i)}} \left\{ \sum_{k=1}^{K} \left\{ n_k \log \pi_k + \sum_{j=1}^{N} \gamma_{jk} \left[\log(\frac{1}{\sqrt{2\pi}} - \log\sigma_k - \frac{(x_j-\mu_k)^2}{2\sigma_k^2}) \right] \right\} \right\} \\
&= \sum_{k=1}^{K} \left\{ \left(\sum_{j=1}^{N} E\gamma_{jk} \right) \log \pi_k + \sum_{j=1}^{N} E\gamma_{jk} \left[\log(\frac{1}{\sqrt{2\pi}} - \log\sigma_k - \frac{(x_j-\mu_k)^2}{2\sigma_k^2}) \right] \right\} \\
&\quad \text{denote } E\gamma_{jk} \text{ as } \hat{\gamma}_{jk} \\
&= \sum_{k=1}^{K} \left\{ \left(\sum_{j=1}^{N} \hat{\gamma}_{jk} \right) \log \pi_k + \sum_{j=1}^{N} \hat{\gamma}_{jk} \left[\log(\frac{1}{\sqrt{2\pi}} - \log\sigma_k - \frac{(x_j-\mu_k)^2}{2\sigma_k^2}) \right] \right\}
\end{aligned} \tag{12.9}
$$

$$
\begin{aligned}
\hat{\gamma}_{jk} &= E\gamma_{jk} = E(\gamma_{jk}|x_j,\boldsymbol{\theta}) = E(\gamma_{jk}=1|x_j,\boldsymbol{\theta}) \\
&= \frac{P(\gamma_{jk}=1,x_j|\boldsymbol{\theta})}{\sum\limits_{k=1}^{K} P(\gamma_{jk}=1,x_j|\boldsymbol{\theta})} \\
&= \frac{P(x_j|\gamma_{jk}=1,\boldsymbol{\theta})P(\gamma_{jk}=1,\boldsymbol{\theta})}{\sum\limits_{k=1}^{K} P(x_j|\gamma_{jk}=1,\boldsymbol{\theta})P(\gamma_{jk}=1,\boldsymbol{\theta})} \\
&= \frac{\phi(x_i|\mu_k,\sigma_k)\pi_k}{\sum\limits_{k=1}^{K} \phi(x_i|\mu_k,\sigma_k)\pi_k} \\
&= \frac{\pi_k\phi(x_i|\mu_k,\sigma_k)}{\sum\limits_{k=1}^{K} \pi_k\phi(x_i|\mu_k,\sigma_k)}
\end{aligned}
$$

Use this formula to update $\hat{\gamma}_{jk}$.

12.4.1.3 M-step: find θ that maximizes the value of $Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(i)})$

Take partial derivatives of Eqn.(12.9) with respect to μ_k, σ_k^2 and let them equal to 0, we can get μ_k, σ_k^2.

$$
\frac{\partial}{\partial \mu_k} Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(i)}) = \sum_{j=1}^{N} \hat{\gamma}_{jk} \left[-\frac{1}{2\sigma_k^2} \cdot 2\,(x_j-\mu_k)\cdot(-1) \right] = 0
$$

$$
\sum_{j=1}^{N} \hat{\gamma}_{jk} \left[(x_j-\mu_k) \right] = 0
$$

$$
\hat{\mu}_k = \frac{\sum\limits_{j=1}^{N} \hat{\gamma}_{jk} x_j}{\hat{\gamma}_{jk}} \tag{12.10}
$$

$$\frac{\partial}{\partial \sigma_k^2} Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(i)}) = \sum_{j=1}^{N} \hat{\gamma}_{jk} \left[-\frac{1}{2\sigma_k^2} + \frac{1}{2\sigma_k^4} (x_j - \mu_k)^2 \right] = 0$$

$$\sum_{j=1}^{N} \hat{\gamma}_{jk} \left[-\sigma_k^2 + (x_j - \mu_k)^2 \right] = 0$$

$$\hat{\sigma}_k^2 = \frac{\sum_{j=1}^{N} \hat{\gamma}_{jk} (x_j - \mu_k)^2}{\hat{\gamma}_{jk}} \tag{12.11}$$

Grouping together only the terms that depend on π_k, we find that we need to maximize $\sum_{k=1}^{K} \left(\sum_{j=1}^{N} \hat{\gamma}_{jk} \right) \log \pi_k$. However, there is an additional constraint $\sum_{k=1}^{K} \pi_k = 1$, since they represent the probabilities $\boldsymbol{\pi}_k = P(x^{(i)} = k|\boldsymbol{\pi})$. To deal with the constraint we construct the Lagrangian

$$\mathcal{L}(\boldsymbol{\pi}) = \sum_{k=1}^{K} \left(\sum_{j=1}^{N} \hat{\gamma}_{jk} \right) \log \pi_k + \beta \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

where β is the Lagrange multiplier. Taking derivatives, we find

$$\hat{\boldsymbol{\pi}}_k = \frac{\sum_{k=1}^{K} \hat{\gamma}_{jk}}{N} \tag{12.12}$$

12.4.1.4 EM algorithm for GMM

```
input  : observed data $\mathcal{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \cdots, \boldsymbol{x}^{(n)}\}$,GMM
output: GMM's parameters $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\sigma}$
// 1. initialize
$\boldsymbol{\pi}^{(0)}$ = ...
$\boldsymbol{\mu}^{(0)}$ = ...
$\boldsymbol{\sigma}^{(0)}$ = ...
while (!convergency) do
    // 2. E-step
    $\hat{\gamma}_{jk} = \frac{\pi_k \phi(x_i|\mu_k, \sigma_k)}{\sum_{k=1}^{K} \pi_k \phi(x_i|\mu_k, \sigma_k)}$
    // 3. M-step
    $\hat{\mu}_k = \frac{\sum_{j=1}^{N} \hat{\gamma}_{jk} x_j}{\hat{\gamma}_{jk}}$
    $\hat{\sigma}_k^2 = \frac{\sum_{j=1}^{N} \hat{\gamma}_{jk} (x_j - \mu_k)^2}{\hat{\gamma}_{jk}}$
    $\hat{\boldsymbol{\pi}}_k = \frac{\sum_{k=1}^{K} \hat{\gamma}_{jk}}{N}$
end
```

Algorithm 4: EM algorithm for GMM

12.5 Generalization of EM Algorithm

EM algorithm can be interpreted as F function's maximization-maximization algorithm, based on this interpretation there are many variations and generalization, e.g., generalized EM Algorithm(GEM).

12.5.1 F function's maximization-maximization algorithm

Definition 12.4. Given the probability distribution of the hidden variable Z is $\tilde{P}(Z)$, define **F function** as the following:

$$F(\tilde{P},\boldsymbol{\theta}) = \mathbb{E}_{\tilde{P}}[\log P(X,Z|\boldsymbol{\theta})] + H(\tilde{P}) \tag{12.13}$$

Where $H(\tilde{P}) = -\mathbb{E}_{\tilde{P}} \log \tilde{P}(Z)$, which is $\tilde{P}(Z)$'s entropy. Usually we assume that $P(X,Z|\boldsymbol{\theta})$ is continuous w.r.t. $\boldsymbol{\theta}$, therefore $F(\tilde{P},\boldsymbol{\theta})$ is continuous w.r.t. $\tilde{P}$ and $\boldsymbol{\theta}$.

Lemma 12.1. *For a fixed $\boldsymbol{\theta}$, there is only one distribution $\tilde{P}_\theta$ which maximizes $F(\tilde{P},\boldsymbol{\theta})$*

$$\tilde{P}_\theta(Z) = P(Z|X,\boldsymbol{\theta}) \tag{12.14}$$

and $\tilde{P}_\theta$ is continuous w.r.t. $\boldsymbol{\theta}$.

Proof. Given a fixed $\boldsymbol{\theta}$, we can get $\tilde{P}_\theta$ which maximizes $F(\tilde{P},\boldsymbol{\theta})$. we construct the Lagrangian

$$\mathcal{L}(\tilde{P},\boldsymbol{\theta}) = \mathbb{E}_{\tilde{P}}[\log P(X,Z|\boldsymbol{\theta})] - \mathbb{E}_{\tilde{P}} \log \tilde{P}_\theta(Z) + \lambda \left[1 - \sum_Z \tilde{P}(Z)\right] \tag{12.15}$$

Take partial derivative with respect to $\tilde{P}_\theta(Z)$ then we get

$$\frac{\partial \mathcal{L}}{\partial \tilde{P}_\theta(Z)} = \log P(X,Z|\theta) - \log \tilde{P}_\theta(Z) - 1 - \lambda$$

Let it equal to 0, we can get

$$\lambda = \log P(X,Z|\theta) - \log \tilde{P}_\theta(Z) - 1$$

Then we can derive that $\tilde{P}_\theta(Z)$ is proportional to $P(X,Z|\theta)$

$$\begin{aligned}
\frac{P(X,Z|\theta)}{\tilde{P}_\theta(Z)} &= e^{1+\lambda} \\
\Rightarrow \tilde{P}_\theta(Z) &= \frac{P(X,Z|\boldsymbol{\theta})}{e^{1+\lambda}} \\
\sum_Z \tilde{P}_\theta(Z) = 1 \Rightarrow \sum_Z \frac{P(X,Z|\boldsymbol{\theta})}{e^{1+\lambda}} &= 1 \Rightarrow P(X|\boldsymbol{\theta}) = e^{1+\lambda} \\
\tilde{P}_\theta(Z) &= \frac{P(X,Z|\boldsymbol{\theta})}{e^{1+\lambda}} = \frac{P(X,Z|\boldsymbol{\theta})}{P(X|\boldsymbol{\theta})} = P(Z|X,\boldsymbol{\theta})
\end{aligned}$$

Lemma 12.2. *If $\tilde{P}_\theta(Z) = P(Z|X,\boldsymbol{\theta})$, then*

$$F(\tilde{P},\boldsymbol{\theta}) = \log P(X|\boldsymbol{\theta}) \tag{12.16}$$

Theorem 12.3. *One iteration of EM algorithm can be implemented as F function's maximization-maximization.*

Assume $\boldsymbol{\theta}^{(i)}$ is the estimation of $\boldsymbol{\theta}$ in the i-th iteration, $\tilde{P}^{(i)}$ is the estimation of $\tilde{P}$ in the i-th iteration. Then in the (i+1)-th iteration two steps are:

1. for fixed $\boldsymbol{\theta}^{(i)}$, find $\tilde{P}^{(i+1)}$ that maximizes $F(\tilde{P},\boldsymbol{\theta}^{(i)})$;
2. for fixed $\tilde{P}^{(i+1)}$, find $\boldsymbol{\theta}^{(i+1)}$ that maximizes $F(\tilde{P}^{(i+1)},\boldsymbol{\theta})$.

Proof. (1) According to Lemma 12.1, we can get

$$\tilde{P}^{(i+1)}(Z) = P(Z|X, \boldsymbol{\theta}^{(i)})$$

(2) According above, we can get

$$\begin{aligned} F(\tilde{P}^{(i+1)}, \boldsymbol{\theta}) &= \mathbb{E}_{\tilde{P}^{(i+1)}} [\log P(X, Z|\boldsymbol{\theta})] + H(\tilde{P}^{(i+1)}) \\ &= \sum_Z P(Z|X, \boldsymbol{\theta}^{(i)}) \log P(X, Z|\boldsymbol{\theta}) + H(\tilde{P}^{(i+1)}) \\ &= Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)}) + H(\tilde{P}^{(i+1)}) \end{aligned}$$

Then

$$\boldsymbol{\theta}^{(i+1)} = \arg\max_{\theta} F(\tilde{P}^{(i+1)}, \boldsymbol{\theta}) = \arg\max_{\theta} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$$

12.5.2 The Generalized EM Algorithm(GEM)

In the formulation of the EM algorithm described above, $\boldsymbol{\theta}^{(i+1)}$ was chosen as the value of $\boldsymbol{\theta}$ for which $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$ was maximized. While this ensures the greatest increase in $l(\boldsymbol{\theta})$, it is however possible to relax the requirement of maximization to one of simply increasing $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$ so that $Q(\boldsymbol{\theta}^{(i+1)}, \boldsymbol{\theta}^{(i)}) \geq Q(\boldsymbol{\theta}^{(i)}, \boldsymbol{\theta}^{(i)})$. This approach, to simply increase and not necessarily maximize $Q(\boldsymbol{\theta}^{(i+1)}, \boldsymbol{\theta}^{(i)})$ is known as the Generalized Expectation Maximization (GEM) algorithm and is often useful in cases where the maximization is difficult. The convergence of the GEM algorithm is similar to the EM algorithm.

12.6 Reference

[1] Dempster AP, Laird NM, Rubin DB. Maximum-likelihood from incomplete data via the EM algorithm. J.Royal Statist. Soc. Ser.B., 1977, 39

[2] Sean Borman. The Expectation Maximization Algorithm A short tutorial. 2009. http://www.seanborman.com/publications/EM_algorithm.pdf

Chapter 13
Hidden markov Model

Chapter 14
BayesNets

14.1 Chain rule

$$p(x) = p(x_1)\prod_{v=2}^{V} p(x_v|x_{1:v-1}) \tag{14.1}$$

14.2 Markov chain

$$p(x) = p(x_1)\prod_{v=2}^{V} p(x_v|x_{v-1}) \tag{14.2}$$

14.3 DGM

$$p(x|G) = \prod_{v=1}^{V} p(x_v|x_{pa(v)}) \tag{14.3}$$

14.4 Inference

$$p(x_h|x_v,\theta) = \frac{p(x_h,x_v|\theta)}{\sum_{x'_h} p(x'_h,x_v|\theta)} \tag{14.4}$$

$$p(x_q|x_v,\theta) = \sum_{x_n} p(x_q,x_n|x_v,\theta) \tag{14.5}$$

14.5 Learning

14.5.1 MAP

$$\hat{\theta} = \underset{\theta}{\mathrm{argmax}}\, p(\theta)\prod_{n=1}^{N} p(x_n|\theta) \tag{14.6}$$

14.5.2 Learning from complete data

$$p(D|\theta) = \prod_{n=1}^{N} p(x_n|\theta) = \prod_{n=1}^{N}\prod_{v=1}^{V} p(x_{nv}|x_{n,pa(v)},\theta_v) = \prod_{v=1}^{V} p(D_v|\theta_v) \tag{14.7}$$

14.5.3 Multinoulli Learning

Multinoulli Distribution

$$Cat(x|\mu) = \prod_{k=1}^{K} \mu_k^{x_k} \tag{14.8}$$

then from 14.3 and 14.8:

$$p(x|G,\theta) = \prod_{v=1}^{V}\prod_{c=1}^{C_v}\prod_{k=1}^{K} \theta_{vck}^{y_{vck}} \tag{14.9}$$

Likelihood

$$p(D|G,\theta) = \prod_{n=1}^{N} p(x_n|G,\theta) = \prod_{n=1}^{N}\prod_{v=1}^{V}\prod_{c=1}^{C_{nv}}\prod_{k=1}^{K} \theta_{vck}^{y_{nvck}} \tag{14.10}$$

where $y_{nv} = f(pa(x_{nv}))$, f(x) is a map from x to a vector, there is only one element in the vector is 1.

14.6 d-separation

1. P contains a chain

$$p(x,z|y) = \frac{p(x,y,z)}{p(y)} = \frac{p(x)p(y|x)p(z|y)}{p(y)} = \frac{p(x,y)p(z|y)}{p(y)} = p(x|y)p(z|y) \tag{14.11}$$

2. P contains a fork

$$p(x,z|y) = \frac{p(x,y,z)}{p(y)} = \frac{p(y)p(x|y)p(z|y)}{p(y)} = p(x|y)p(z|y) \tag{14.12}$$

3. P contains v-structure

$$p(x,z|y) = \frac{p(x,y,z)}{p(y)} = \frac{p(x)p(z)p(y|x,z)}{p(y)} \neq p(x|y)p(z|y) \tag{14.13}$$

14.7 Markov blanket

$$mb(t) = ch(t) \cup pa(t) \cup copa(t) \tag{14.14}$$

14.8 Reference

Mlapp chapter 10 Bayes nets

Chapter 15

Conditional Random Field

Appendix A
Optimization methods

A.1 Gradient descent

A.1.1 Stochastic gradient descent

```
input  : Training data $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) | i = 1 : N\}$
output: A linear model: $y_i = \boldsymbol{\theta}^T \boldsymbol{x}$
$\boldsymbol{w} \leftarrow 0$; $b \leftarrow 0$; $k \leftarrow 0$;
while no mistakes made within the for loop do
    for $i \leftarrow 1$ to $N$ do
        if $y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \leq 0$ then
            $\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta y_i \boldsymbol{x}_i$;
            $b \leftarrow b + \eta y_i$;
            $k \leftarrow k + 1$;
        end
    end
end
```

Algorithm 5: Stochastic gradient descent

A.1.2 Batch gradient descent

A.2 Lagrange duality

A.2.1 Primal form

Consider the following, which we'll call the **primal** optimization problem:

$$xyz \tag{A.1}$$

A.2.2 Dual form

Glossary

feture vector A feture vector to represent one data.

loss function a function that maps an event onto a real number intuitively representing some ”cost” associated with the event.

glossary term Write here the description of the glossary term. Write here the description of the glossary term. Write here the description of the glossary term.