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

Machine Learning Cheat Sheet

Classical equations, diagrams and tricks in machine learning

December 4, 2013

©2013 soulmachine
Except where otherwise noted, This document is licensed under a Creative Commons Attribution-ShareAlike 3.0 Unported (CC BY-SA3.0) license
(http://creativecommons.org/licenses/by/3.0/).

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): $1 : n = 1, 2, \ldots, n$
$\approx$	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, $\dfrac{\prod_k \Gamma(\alpha_k)}{\Gamma(\sum_k \alpha_k)}$
$\binom{n}{k}$	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, $\Gamma(x)=\int_0^\infty u^{x-1}e^{-u}\mathrm{d}u$
$\Psi(x)$	Digamma function,$Psi(x)=\dfrac{d}{dx}\log\Gamma(x)$
$\mathcal{X}$	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. We use $(x_1,\cdots,x_D)$ to denote a column vector created by stacking D scalars. If we write $\boldsymbol{X}=(\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n)$, where the left hand side is a matrix, we mean to stack the $\boldsymbol{x}_i$ along the columns, creating a matrix.

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 $\sqrt{\sum_{j=1}^d x_j^2}$
$\lvert\lvert\boldsymbol{x}\rvert\rvert_1$	ℓ_1 norm $\sum_{j=1}^d \lvert x_j\rvert$
$\boldsymbol{X}_{:,j}$	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
$\text{sigm}(x)$	Sigmoid (logistic) function, $\frac{1}{1+e^{-x}}$
σ^2	Variance
Σ	Covariance matrix
$\text{var}[x]$	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
$P(y\|\boldsymbol{x})$	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:

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

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

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

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 §6).

1.2.2.2 ERM and SRM

Definition 1.4. ERM(Empirical risk minimization)

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

Definition 1.5. Structural risk

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

Definition 1.6. SRM(Structural risk minimization)

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

[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.

[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

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

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 $F(x) = \int_{-\infty}^{x} f(u)\mathrm{d}u$, 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 $\int_{-\infty}^{\infty} f(x)\mathrm{d}x = 1$.

[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 $F(x,y) \triangleq P(X \leq x \cap Y \leq y) = P(X \leq x, Y \leq y)$.

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

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:

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

Marginal PMF or PDF:

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

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

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

This is called a **generative classifier**, since it specifies how to generate the data using the class conditional density $p(\boldsymbol{x}|y=c)$ and the class prior $p(y=c)$. An alternative approach is to directly fit the class posterior, $p(y=c|\boldsymbol{x})$;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 $\mathbb{E}[X] \triangleq \int_{\mathcal{X}} xp(x)\mathrm{d}x$. 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:

$$\begin{aligned} var[X] &= \mathbb{E}[(X-\mu)^2] = \int (x-\mu)^2 p(x)\mathrm{d}x & (2.13)\\ &= \int x^2 p(x)\mathrm{d}x \int \mu^2 p(x)\mathrm{d}x - 2\mu \int xp(x)\mathrm{d}x = \mathbb{E}[X^2] - \mu^2 & (2.14)\end{aligned}$$

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 Bernoulli and binomial distributions

Definition 2.1. Now suppose we toss a coin only once. Let $X \in \{0,1\}$ be a binary random variable, with probability of success or heads of θ. We say that X has a **Bernoulli distribution**. This is written as $X \sim \text{Ber}(\theta)$, where the pmf is defined as

$$\text{Ber}(x|\theta) \triangleq \theta^{\mathbb{I}(x=1)}(1-\theta)^{\mathbb{I}(x=0)} \tag{2.17}$$

Definition 2.2. Suppose we toss a coin n times. Let $X \in \{0,1,\cdots,n\}$ be the number of heads. If the probability of heads is θ, then we say X has a **binomial distribution**, written as $X \sim \text{Bin}(n,\theta)$. The pmf is given by

$$\text{Bin}(k|n,\theta) \triangleq \binom{n}{k}\theta^k(1-\theta)^{n-k} \tag{2.18}$$

2.3.2 The multinoulli and multinomial distributions

Definition 2.3. The Bernoulli distribution can be used to model the outcome of one coin tosses. To model the outcome of tossing a K-sided dice, let $\boldsymbol{x} = (\mathbb{I}(x=1),\cdots,\mathbb{I}(x=K)) \in \{0,1\}^K$ be a random vector(this is called **dummy encoding** or **one-hot encoding**), then we say X has a **multinoulli distribution**(or **categorical distribution**), written as $X \sim \text{Cat}(\theta)$. The pmf is given by:

$$p(\boldsymbol{x}) \triangleq \prod_{k=1}^{K} \theta_k^{\mathbb{I}(x_k=1)} \tag{2.19}$$

Definition 2.4. Suppose we toss a K-sided dice n times. Let $\boldsymbol{x} = (x_1,x_2,\cdots,x_K) \in \{0,1,\cdots,n\}^K$ be a random vector, where x_j is the number of times side j of the dice occurs, then we say X has a **multinomial distribution**, written as $X \sim \text{Mu}(n,\boldsymbol{\theta})$. The pmf is given by

$$p(\boldsymbol{x}) \triangleq \binom{n}{x_1 \cdots x_k}\prod_{k=1}^{K}\theta_k^{x_k} \quad \text{where} \binom{n}{x_1 \cdots x_k} \triangleq \frac{n!}{x_1!x_2!\cdots x_K!} \tag{2.20}$$

Bernoulli distribution is just a special case of a Binomial distribution with $n=1$, and so is multinoulli distribution as to multinomial distribution. See Table 2.2 for a summary.

Table 2.1: Summary of the multinomial and related distributions.

Name	K	n	X
Bernoulli	1	1	$x \in \{0,1\}$
Binomial	1	-	$\boldsymbol{x} \in \{0,1,\cdots,n\}$
Multinoulli	-	1	$\boldsymbol{x} \in \{0,1\}^K, \sum_{k=1}^K x_k = 1$
Multinomial	-	-	$\boldsymbol{x} \in \{0,1,\cdots,n\}^K, \sum_{k=1}^K x_k = n$

2.3.3 The Poisson distribution

Definition 2.5. We say that $X \in \{0,1,2,\cdots\}$ has a **Poisson distribution** with parameter $\lambda > 0$, written $X \sim \text{Poi}(\lambda)$, if its pmf is

$$p(x|\lambda) = e^{-\lambda}\frac{\lambda^x}{x!} \tag{2.21}$$

The first term is just the normalization constant, required to ensure the distribution sums to 1.

The Poisson distribution is often used as a model for counts of rare events like radioactive decay and traffic accidents.

Table 2.2: Summary of Bernoulli, binomial multinoulli and multinomial distributions.

Name	Written as	X	$p(x)$(or $p(\boldsymbol{x})$)	$\mathbb{E}[X]$	var$[X]$
Bernoulli	$X \sim \text{Ber}(\theta)$	$x \in \{0,1\}$	$\theta^{\mathbb{I}(x=1)}(1-\theta)^{\mathbb{I}(x=0)}$	θ	$\theta(1-\theta)$
Binomial	$X \sim \text{Bin}(n,\theta)$	$x \in \{0,1,\cdots,n\}$	$\binom{n}{k}\theta^k(1-\theta)^{n-k}$	$n\theta$	$n\theta(1-\theta)$
Multinoulli	$X \sim \text{Cat}(\boldsymbol{\theta})$	$\boldsymbol{x} \in \{0,1\}^K, \sum_{k=1}^K x_k = 1$	$\prod_{k=1}^K \theta_j^{\mathbb{I}(x_j=1)}$	-	-
Multinomial	$X \sim \text{Mu}(n,\boldsymbol{\theta})$	$\boldsymbol{x} \in \{0,1,\cdots,n\}^K, \sum_{k=1}^K x_k = n$	$\binom{n}{x_1 \cdots x_k}\prod_{k=1}^K \theta_j^{x_j}$	-	-
Poisson	$X \sim \text{Poi}(\lambda)$	$x \in \{0,1,2,\cdots\}$	$e^{-\lambda}\frac{\lambda^x}{x!}$	λ	λ

2.3.4 The empirical distribution

The **empirical distribution function**[6], or **empirical cdf**, is the cumulative distribution function associated with the empirical measure of the sample. Let $\mathcal{D} = \{x_1, x_2, \cdots, x_N\}$ be a sample set, it is defined as

$$F_n(x) \triangleq \frac{1}{N}\sum_{i=1}^N \mathbb{I}(x_i \leq x) \tag{2.22}$$

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

Table 2.3: Summary of Gaussian distribution.

Name	Written as	$f(x)$	$\mathbb{E}[X]$	mode	var$[X]$
Gaussian distribution	$X \sim \mathcal{N}(\mu, \sigma^2)$	$\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$	μ	μ	σ^2

If $X \sim N(0,1)$,we say X follows a **standard normal** distribution.

The Gaussian distribution is the most widely used distribution in statistics. There are several reasons for this.

1. First, it has two parameters which are easy to interpret, and which capture some of the most basic properties of a distribution, namely its mean and variance.
2. Second,the central limit theorem (Section TODO) tells us that sums of independent random variables have an approximately Gaussian distribution, making it a good choice for modeling residual errors or noise.
3. Third, the Gaussian distribution makes the least number of assumptions (has maximum entropy), subject to the constraint of having a specified mean and variance, as we show in Section TODO; this makes it a good default choice in many cases.

6 http://en.wikipedia.org/wiki/Empirical_distribution_function

4. Finally, it has a simple mathematical form, which results in easy to implement, but often highly effective, methods, as we will see.

See (Jaynes 2003, ch 7) for a more extensive discussion of why Gaussians are so widely used.

2.4.2 Student's t-distribution

Table 2.4: Summary of Student's t-distribution.

Name	Written as	$f(x)$	$\mathbb{E}[X]$	mode	$\text{var}[X]$
Student's t-distribution	$X \sim \mathcal{T}(\mu, \sigma^2, \nu)$	$\frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})}\left[1+\frac{1}{\nu}\left(\frac{x-\mu}{\nu}\right)^2\right]$	μ	μ	$\frac{\nu\sigma^2}{\nu-2}$

where $\Gamma(x)$ is the gamma function:

$$\Gamma(x) \triangleq \int_0^\infty u^{x-1}e^{-u}\mathrm{d}x \tag{2.23}$$

μ is the mean, $\sigma^2 > 0$ is the scale parameter, and $\nu > 0$ is called the **degrees of freedom**. See Figure 2.1 for some plots.

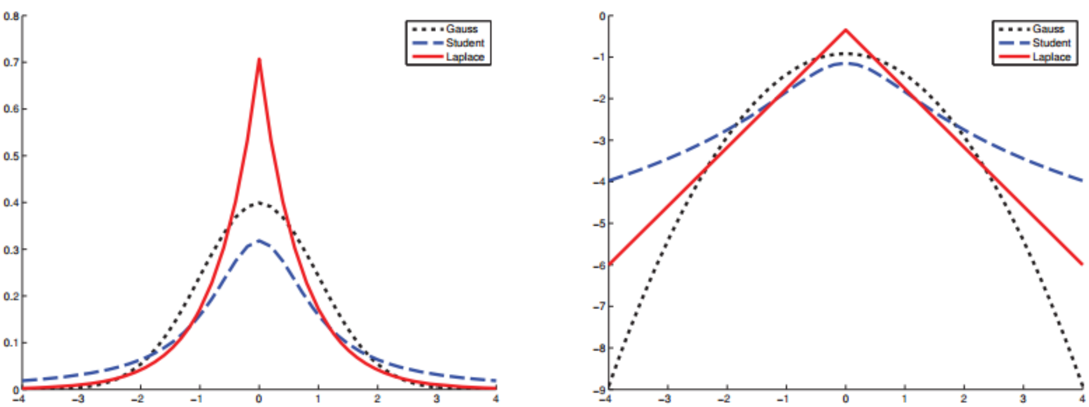


Fig. 2.1: (a) The pdfs for a $\mathcal{N}(0,1)$, $\mathcal{T}(0,1,1)$ and $Lap(0,1/\sqrt{2})$. The mean is 0 and the variance is 1 for both the Gaussian and Laplace. The mean and variance of the Student is undefined when $\nu = 1$.(b) Log of these pdfs. Note that the Student distribution is not log-concave for any parameter value, unlike the Laplace distribution, which is always log-concave (and log-convex...) Nevertheless, both are unimodal.

The variance is only defined if $\nu > 2$. The mean is only defined if $\nu > 1$.

As an illustration of the robustness of the Student distribution, consider Figure 2.2. We see that the Gaussian is affected a lot, whereas the Student distribution hardly changes. This is because the Student has heavier tails, at least for small ν(see Figure 2.1).

If $\nu = 1$, this distribution is known as the **Cauchy** or **Lorentz** distribution. This is notable for having such heavy tails that the integral that defines the mean does not converge.

To ensure finite variance, we require $\nu > 2$. It is common to use $\nu = 4$, which gives good performance in a range of problems (Lange et al. 1989). For $\nu \gg 5$, the Student distribution rapidly approaches a Gaussian distribution and loses its robustness properties.

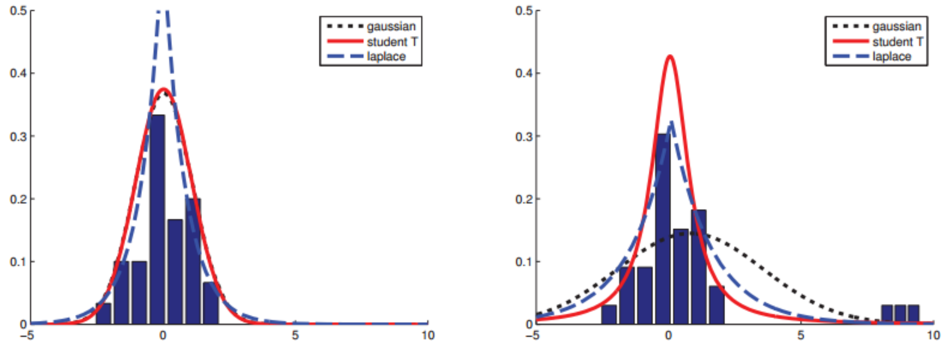


Fig. 2.2: Illustration of the effect of outliers on fitting Gaussian, Student and Laplace distributions. (a) No outliers (the Gaussian and Student curves are on top of each other). (b) With outliers. We see that the Gaussian is more affected by outliers than the Student and Laplace distributions.

2.4.3 The Laplace distribution

Table 2.5: Summary of Laplace distribution.

Name	Written as	$f(x)$	$\mathbb{E}[X]$	mode	$\text{var}[X]$
Laplace distribution	$X \sim \text{Lap}(\mu, b)$	$\frac{1}{2b}\exp\left(-\frac{\lvert x-\mu\rvert}{b}\right)$	μ	μ	$2b^2$

Here μ is a location parameter and $b > 0$ is a scale parameter. See Figure 2.1 for a plot.

Its robustness to outliers is illustrated in Figure 2.2. It also put mores probability density at 0 than the Gaussian. This property is a useful way to encourage sparsity in a model, as we will see in Section TODO.

2.4.4 The gamma distribution

Table 2.6: Summary of gamma distribution

Name	Written as	X	$f(x)$	$\mathbb{E}[X]$	mode	$\text{var}[X]$
Gamma distribution	$X \sim \text{Ga}(a, b)$	$x \in \mathbb{R}^+$	$\frac{b^a}{\Gamma(a)}x^{a-1}e^{-xb}$	$\frac{a}{b}$	$\frac{a-1}{b}$	$\frac{a}{b^2}$

Here $a > 0$ is called the shape parameter and $b > 0$ is called the rate parameter. See Figure 2.3 for some plots.

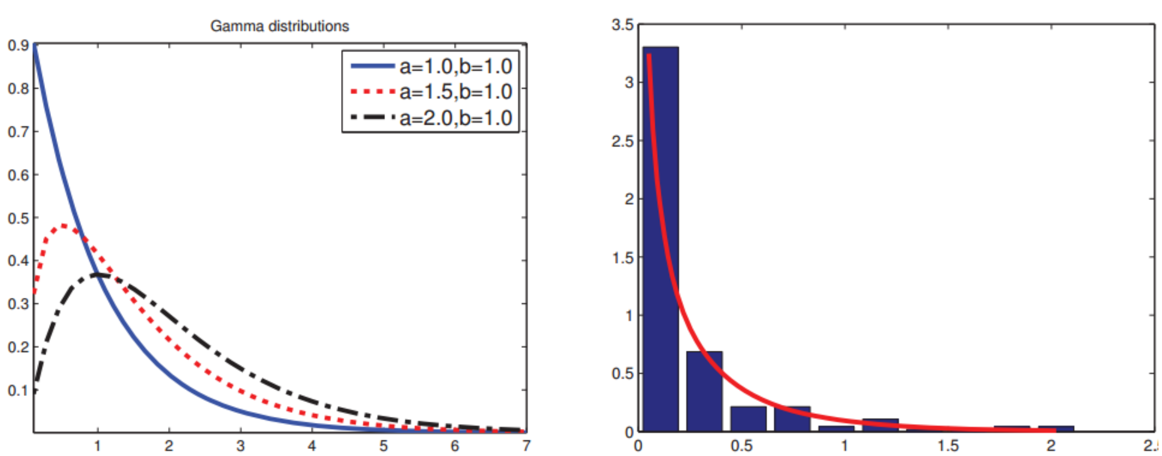


Table 2.7: Summary of Beta distribution

Name	Written as	X	$f(x)$	$\mathbb{E}[X]$	mode	var$[X]$
Beta distribution	$X \sim \text{Beta}(a,b)$	$x \in [0,1]$	$\frac{1}{B(a,b)}x^{a-1}(1-x)^{b-1}$	$\frac{a}{a+b}$	$\frac{a-1}{a+b-2}$	$\frac{ab}{(a+b)^2(a+b+1)}$

2.4.5 The beta distribution

Here $B(a,b)$is the beta function,

$$B(a,b) \triangleq \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)} \tag{2.24}$$

See Figure 2.4 for plots of some beta distributions. We require $a,b>0$ to ensure the distribution is integrable (i.e., to ensure $B(a,b)$ exists). If $a=b=1$, we get the uniform distirbution. If a and b are both less than 1, we get a bimodal distribution with spikes at 0 and 1; if a and b are both greater than 1, the distribution is unimodal.

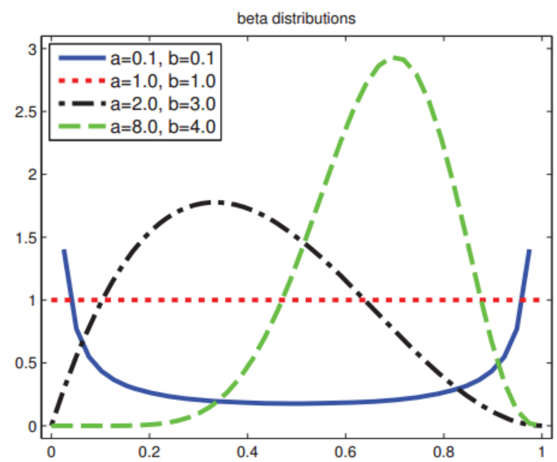


Fig. 2.4: Some beta distributions.

2.4.6 Pareto distribution

Table 2.8: Summary of Pareto distribution

Name	Written as	X	$f(x)$	$\mathbb{E}[X]$	mode	var$[X]$
Pareto distribution	$X \sim \text{Pareto}(k,m)$	$x \geq m$	$km^k x^{-(k+1)}\mathbb{I}(x \geq m)$	$\frac{km}{k-1}$ if $k>1$	m	$\frac{m^2 k}{(k-1)^2(k-2)}$ if $k>2$

The **Pareto distribution** is used to model the distribution of quantities that exhibit **long tails**, also called **heavy tails**.

As $k \rightarrow \infty$, the distribution approaches $\delta(x-m)$. See Figure 2.5(a) for some plots. If we plot the distribution on a log-log scale, it forms a straight line, of the form $\log p(x) = a\log x + c$ for some constants a and c. See Figure 2.5(b) for an illustration (this is known as a **power law**).

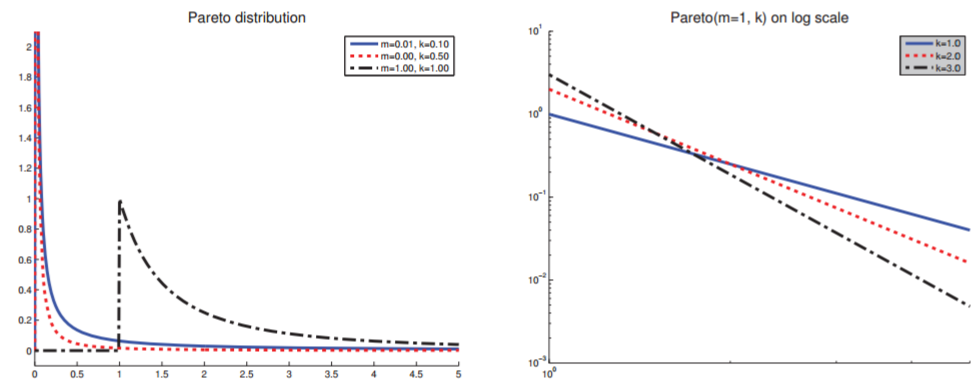


Fig. 2.5: (a) The Pareto distribution Pareto$(x|m,k)$ for $m=1$. (b) The pdf on a log-log scale.

2.5 Joint probability distributions

Given a **multivariate random variable** or **random vector** [7] $X \in \mathbb{R}^D$, the **joint probability distribution**[8] is a probability distribution that gives the probability that each of $X_1, X_2, \cdots, X_D$ falls in any particular range or discrete set of values specified for that variable. In the case of only two random variables, this is called a **bivariate distribution**, but the concept generalizes to any number of random variables, giving a **multivariate distribution**.

The joint probability distribution can be expressed either in terms of a **joint cumulative distribution function** or in terms of a **joint probability density function** (in the case of continuous variables) or **joint probability mass function** (in the case of discrete variables).

2.5.1 Covariance and correlation

Definition 2.6. The **covariance** between two rvs X and Y measures the degree to which X and Y are (linearly) related. Covariance is defined as

$$Cov[X,Y] \triangleq \mathbb{E}[(X-\mathbb{E}[X])(Y-\mathbb{E}[Y])] = \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y] \tag{2.25}$$

Definition 2.7. If X is a D-dimensional random vector, its **covariance matrix** is defined to be the following symmetric, positive definite matrix:

$$Cov[X] \triangleq \mathbb{E}\left[(X-\mathbb{E}[X])(X-\mathbb{E}[X])^T\right] \tag{2.26}$$

$$= \begin{pmatrix} \mathrm{var}[X_1] & \mathrm{Cov}[X_1,X_2] & \cdots & \mathrm{Cov}[X_1,X_D] \\ \mathrm{Cov}[X_2,X_1] & \mathrm{var}[X_2] & \cdots & \mathrm{Cov}[X_2,X_D] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{Cov}[X_D,X_1] & \mathrm{Cov}[X_D,X_2] & \cdots & \mathrm{var}[X_D] \end{pmatrix} \tag{2.27}$$

Definition 2.8. The (Pearson) **correlation coefficient** between X and Y is defined as

$$\mathrm{corr}[X,Y] \triangleq \frac{\mathrm{Cov}[X,Y]}{\sqrt{\mathrm{var}[X],\mathrm{var}[Y]}} \tag{2.28}$$

A **correlation matrix** has the form

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

[8] http://en.wikipedia.org/wiki/Joint_probability_distribution

$$\mathbf{R} \triangleq \begin{pmatrix} \mathrm{corr}[X_1,X_1] & \mathrm{corr}[X_1,X_2] & \cdots & \mathrm{corr}[X_1,X_D] \\ \mathrm{corr}[X_2,X_1] & \mathrm{corr}[X_2,X_2] & \cdots & \mathrm{corr}[X_2,X_D] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{corr}[X_D,X_1] & \mathrm{corr}[X_D,X_2] & \cdots & \mathrm{corr}[X_D,X_D] \end{pmatrix} \tag{2.29}$$

The correlation coefficient can viewed as a degree of linearity between X and Y, see Figure 2.6.

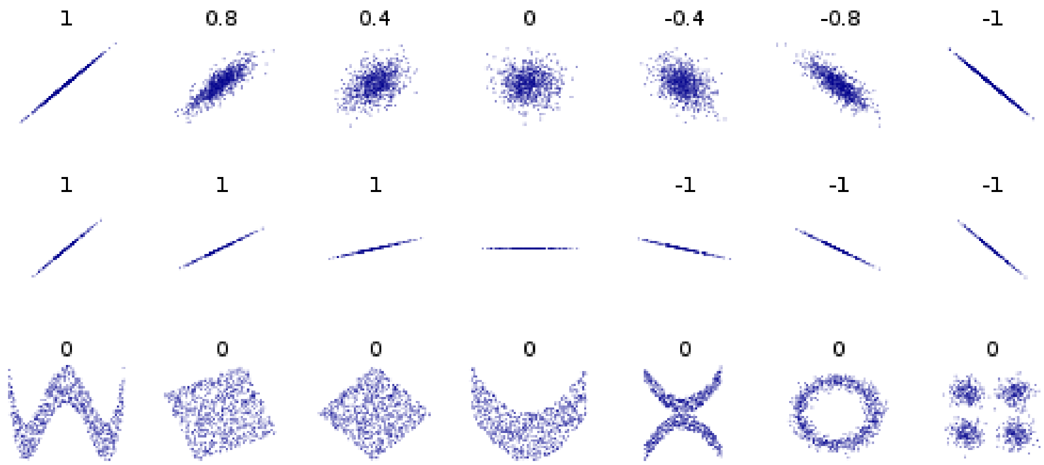


Fig. 2.6: Several sets of (x,y) points, with the Pearson correlation coefficient of x and y for each set. Note that the correlation reflects the noisiness and direction of a linear relationship (top row), but not the slope of that relationship (middle), nor many aspects of nonlinear relationships (bottom). N.B.: the figure in the center has a slope of 0 but in that case the correlation coefficient is undefined because the variance of Y is zero.Source:http://en.wikipedia.org/wiki/Correlation

Uncorrelated does not imply independent. For example, let $X \sim U(-1,1)$ and $Y = X^2$. Clearly Y is dependent on X(in fact, Y is uniquely determined by X), yet one can show that $\mathrm{corr}[X,Y] = 0$. Some striking examples of this fact are shown in Figure 2.6. This shows several data sets where there is clear dependence between X and Y, and yet the correlation coefficient is 0. A more general measure of dependence between random variables is mutual information, see Section TODO.

2.5.2 Multivariate Gaussian distribution

The **multivariate Gaussian** or **multivariate normal**(MVN) is the most widely used joint probability density function for continuous variables. We discuss MVNs in detail in Chapter 4; here we just give some definitions and plots.

The pdf of the MVN in D dimensions is defined by the following:

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\Sigma) \triangleq \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right] \tag{2.30}$$

where $\boldsymbol{\mu} = \mathbb{E}[X] \in \mathbb{R}^D$ is the mean vector, and $\Sigma = \mathrm{Cov}[X]$ is the $D \times D$ covariance matrix. The normalization constant $(2\pi)^{D/2}|\Sigma|^{1/2}$ just ensures that the pdf integrates to 1.

Figure 2.7 plots some MVN densities in 2d for three different kinds of covariance matrices. A full covariance matrix has A $D(D+1)/2$ parameters (we divide by 2 since Σ is symmetric). A diagonal covariance matrix has D parameters, and has 0s in the off-diagonal terms. A spherical or isotropic covariance,$\Sigma = \sigma^2\boldsymbol{I}_D$, has one free parameter.

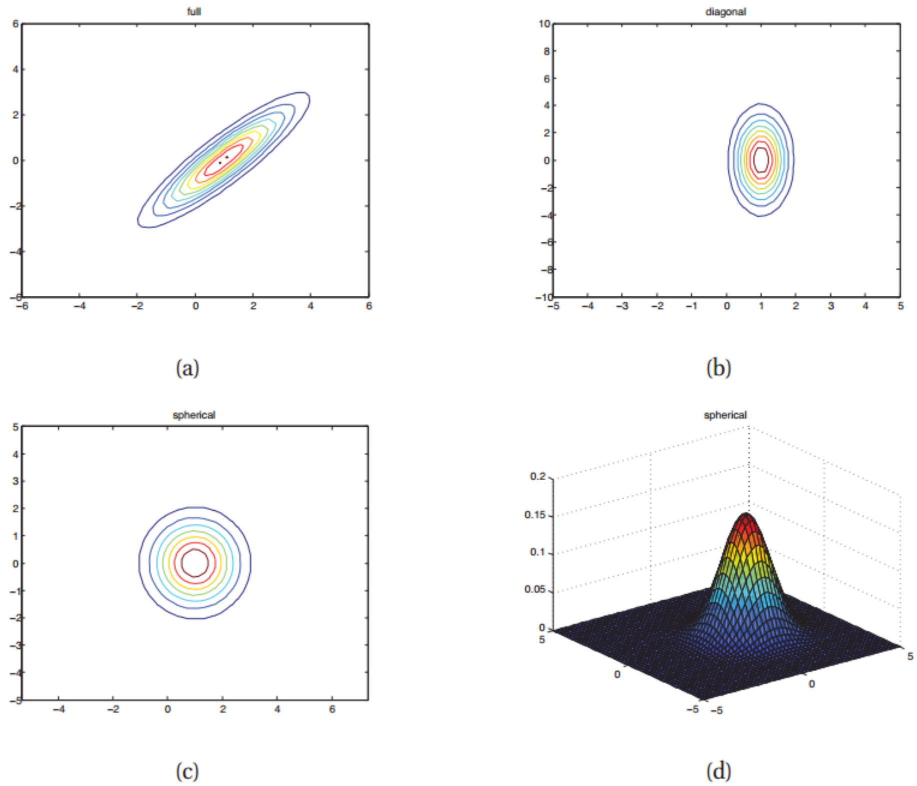


Fig. 2.7: We show the level sets for 2d Gaussians. (a) A full covariance matrix has elliptical contours.(b) A diagonal covariance matrix is an axis aligned ellipse. (c) A spherical covariance matrix has a circular shape. (d) Surface plot for the spherical Gaussian in (c).

2.5.3 Multivariate Student's t-distribution

A more robust alternative to the MVN is the multivariate Student's t-distribution, whose pdf is given by

$$\mathcal{T}(x|\boldsymbol{\mu},\Sigma,\nu) \triangleq \frac{\Gamma(\frac{\nu+D}{2})}{\Gamma(\frac{\nu}{2})}\frac{|\Sigma|^{-\frac{1}{2}}}{(\nu\pi)^{\frac{D}{2}}}\left[1+\frac{1}{\nu}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]^{-\frac{\nu+D}{2}} \tag{2.31}$$

$$= \frac{\Gamma(\frac{\nu+D}{2})}{\Gamma(\frac{\nu}{2})}\frac{|\Sigma|^{-\frac{1}{2}}}{(\nu\pi)^{\frac{D}{2}}}\left[1+(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{V}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]^{-\frac{\nu+D}{2}} \tag{2.32}$$

where Σ is called the scale matrix (since it is not exactly the covariance matrix) and $\boldsymbol{V} = \nu\Sigma$. This has fatter tails than a Gaussian. The smaller ν is, the fatter the tails. As $\nu \to \infty$, the distribution tends towards a Gaussian. The distribution has the following properties

$$\text{mean} = \boldsymbol{\mu}\ ,\ \text{mode} = \boldsymbol{\mu}\ ,\ \text{Cov} = \frac{\nu}{\nu-2}\Sigma \tag{2.33}$$

2.5.4 Dirichlet distribution

A multivariate generalization of the beta distribution is the **Dirichlet distribution**, which has support over the probability simplex, defined by

$$S_K = \left\{ \boldsymbol{x} : 0 \leq x_k \leq 1, \sum_{k=1}^{K} x_k = 1 \right\} \tag{2.34}$$

The pdf is defined as follows:

$$\mathrm{Dir}(\boldsymbol{x}|\boldsymbol{\alpha}) \triangleq \frac{1}{B(\boldsymbol{\alpha})} \prod_{k=1}^{K} x_k^{\alpha_k - 1} \mathbb{I}(\boldsymbol{x} \in S_K) \tag{2.35}$$

where $B(\alpha_1, \alpha_2, \cdots, \alpha_K)$ is the natural generalization of the beta function to K variables:

$$B(\alpha) \triangleq \frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma(\alpha_0)} \text{ where } \alpha_0 \triangleq \sum_{k=1}^{K} \alpha_k \tag{2.36}$$

Figure 2.8 shows some plots of the Dirichlet when $K = 3$, and Figure 2.9 for some sampled probability vectors. We see that α_0 controls the strength of the distribution (how peaked it is), and thekcontrol where the peak occurs. For example, $\mathrm{Dir}(1,1,1)$ is a uniform distribution, $\mathrm{Dir}(2,2,2)$ is a broad distribution centered at $(1/3,1/3,1/3)$, and $\mathrm{Dir}(20,20,20)$ is a narrow distribution centered at $(1/3,1/3,1/3)$.If $\alpha_k < 1$ for all k, we get spikes at the corner of the simplex.

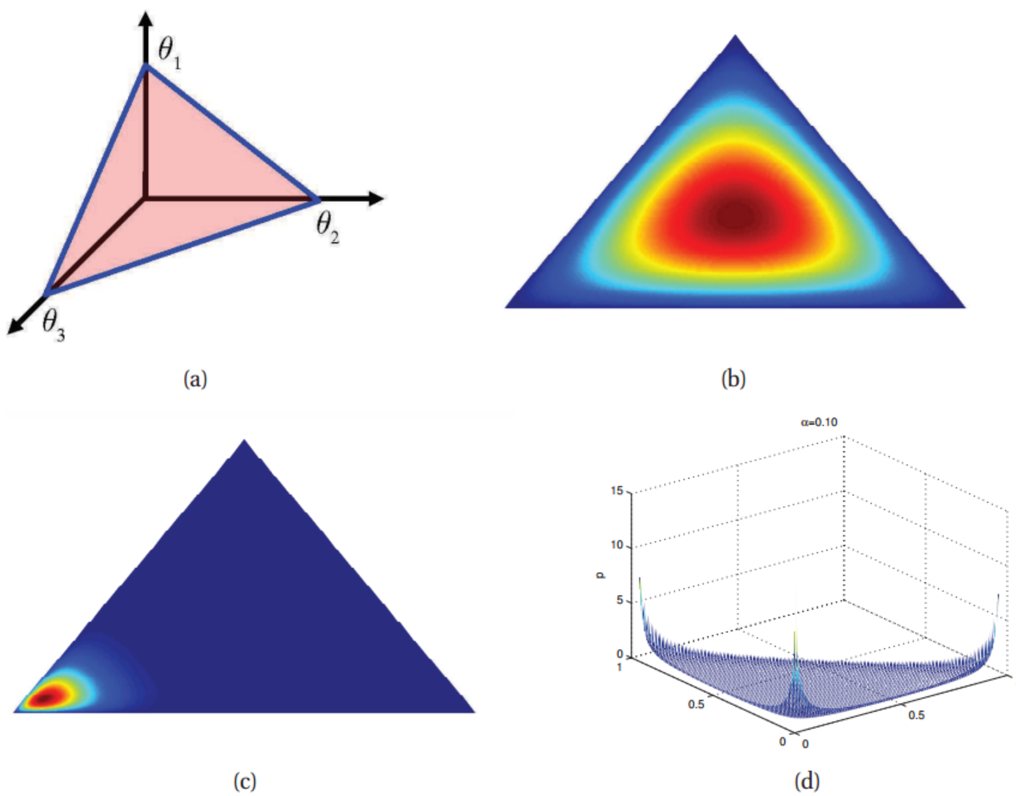


Fig. 2.8: (a) The Dirichlet distribution when $K = 3$ defines a distribution over the simplex, which can be represented by the triangular surface. Points on this surface satisfy $0 \leq \theta_k \leq 1$ and $\sum_{k=1}^{K} \theta_k = 1$. (b) Plot of the Dirichlet density when $\boldsymbol{\alpha} = (2,2,2)$. (c) $\boldsymbol{\alpha} = (20,2,2)$.

For future reference, the distribution has these properties

$$\mathbb{E}(x_k) = \frac{\alpha_k}{\alpha_0}, \ \mathrm{mode}[x_k] = \frac{\alpha_k - 1}{\alpha_0 - K}, \ \mathrm{var}[x_k] = \frac{\alpha_k(\alpha_0 - \alpha_k)}{\alpha_0^2(\alpha_0 + 1)} \tag{2.37}$$

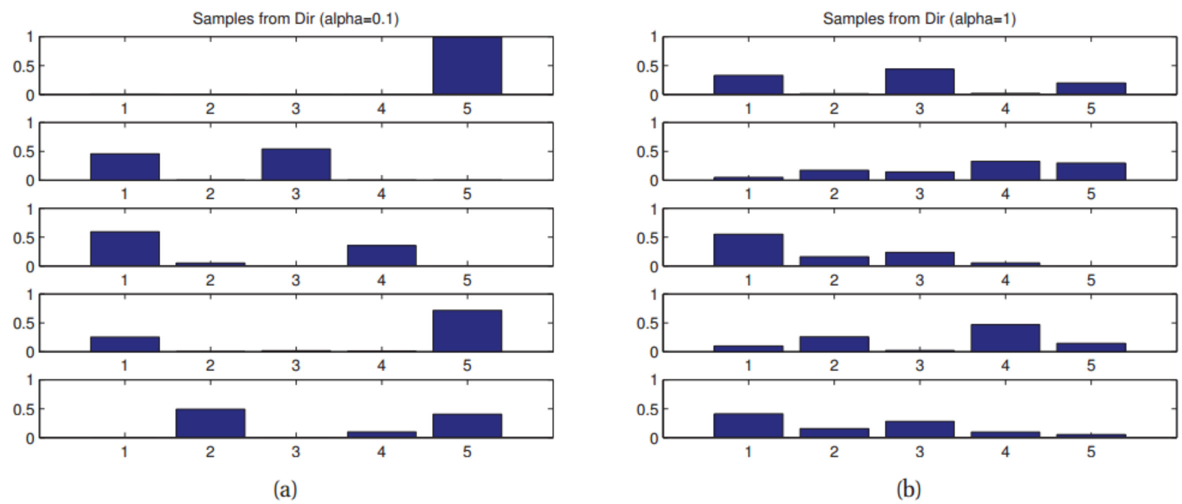


Fig. 2.9: Samples from a 5-dimensional symmetric Dirichlet distribution for different parameter values. (a) $\boldsymbol{\alpha} = (0.1, \cdots, 0.1)$. This results in very sparse distributions, with many 0s. (b) $\boldsymbol{\alpha} = (1, \cdots, 1)$. This results in more uniform (and dense) distributions.

2.6 Transformations of random variables

If $\boldsymbol{x} \sim P()$ is some random variable, and $\boldsymbol{y} = f(\boldsymbol{x})$, what is the distribution of Y? This is the question we address in this section.

2.6.1 Linear transformations

Suppose $g()$is a linear function:

$$g(\boldsymbol{x}) = A\boldsymbol{x} + b \tag{2.38}$$

First, for the mean, we have

$$\mathbb{E}[\boldsymbol{y}] = \mathbb{E}[A\boldsymbol{x} + b] = A\mathbb{E}[\boldsymbol{x}] + b \tag{2.39}$$

this is called the **linearity of expectation**.

For the covariance, we have

$$\text{Cov}[\boldsymbol{y}] = \text{Cov}[A\boldsymbol{x} + b] = A\Sigma A^T \tag{2.40}$$

2.6.2 General transformations

If X is a discrete rv, we can derive the pmf for y by simply summing up the probability mass for all the xs such that $f(x) = y$:

$$p_Y(y) = \sum_{x:g(x)=y} p_X(x) \tag{2.41}$$

If X is continuous, we cannot use Equation (2.41) since $p_X(x)$ is a density, not a pmf, and we cannot sum up densities. Instead, we work with cdfs, and write

$$F_Y(y) = P(Y \leq y) = P(g(X) \leq y) = \int_{g(X) \leq y} f_X(x) \mathrm{d}x \tag{2.42}$$

We can derive the pdf of Y by differentiating the cdf:

$$f_Y(y) = f_X(x)|\frac{dx}{dy}| \tag{2.43}$$

This is called **change of variables** formula. We leave the proof of this as an exercise.

For example, suppose $X \sim U(1,1)$, and $Y = X^2$. Then $p_Y(y) = \frac{1}{2}y^{-\frac{1}{2}}$.

2.6.3 Central limit theorem

Theorem 2.1. *Given N random variables $X_1, X_2, \cdots, X_N$, each variable is* ***independent and identically distributed***[9] ***(iid*** *for short), and each has the same mean μ and variance σ^2, then*

$$\frac{\sum\limits_{i=1}^{n} X_i - N\mu}{\sqrt{N}\sigma} \sim \mathcal{N}(0,1) \tag{2.44}$$

this can also be written as

$$\frac{\bar{X} - \mu}{\sigma/\sqrt{N}} \sim \mathcal{N}(0,1) \text{ where } \bar{X} \triangleq \frac{1}{N}\sum_{i=1}^{n} X_i \tag{2.45}$$

2.7 Monte Carlo approximation

In general, computing the distribution of a function of an rv using the change of variables formula can be difficult. One simple but powerful alternative is as follows. First we generate S samples from the distribution, call them $x_1, \cdots, x_S$. (There are many ways to generate such samples; one popular method, for high dimensional distributions, is called Markov chain Monte Carlo or MCMC; this will be explained in Chapter TODO.) Given the samples, we can approximate the distribution of $f(X)$ by using the empirical distribution of $\{f(x_s)\}_{s=1}^{S}$. This is called a **Monte Carlo approximation**[10], named after a city in Europe known for its plush gambling casinos.

We can use Monte Carlo to approximate the expected value of any function of a random variable. We simply draw samples, and then compute the arithmetic mean of the function applied to the samples. This can be written as follows:

$$\mathbb{E}[g(X)] = \int g(x)p(x)\mathrm{d}x \approx \frac{1}{S}\sum_{s=1}^{S} f(x_s) \text{ , where } x_s \sim p(X) \tag{2.46}$$

This is called **Monte Carlo integration**[11], and has the advantage over numerical integration (which is based on evaluating the function at a fixed grid of points) that the function is only evaluated in places where there is non-negligible probability.

[9] http://en.wikipedia.org/wiki/Independent_identically_distributed

[10] http://en.wikipedia.org/wiki/Monte_Carlo_method

[11] http://en.wikipedia.org/wiki/Monte_Carlo_integration

2.8 Information theory

2.8.1 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

$$\mathbb{H}(X) \triangleq -\sum_{k=1}^{K} p(X=k)\log_2 p(X=k) \tag{2.47}$$

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.

2.8.2 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:

$$\mathbb{KL}(P||Q) \triangleq \sum_x p(x)\log_2 \frac{p(x)}{q(x)} \tag{2.48}$$

where the sum gets replaced by an integral for pdfs[12]. 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

$$\mathbb{KL}(p||q) \triangleq \sum_x p(x)\log_2 p(x) - \sum_{k=1}^{K} p(x)\log_2 q(x) = \mathbb{H}(p) - \mathbb{H}(p,q) \tag{2.49}$$

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

$$\mathbb{H}(p,q) = \sum_x p(x)\log_2 q(x) \tag{2.50}$$

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 §2.8.1,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 2.2. ***(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*.

---

[12] 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 $q=0.5p_1+0.5p_2$

2.8.3 Mutual information

Definition 2.9. Mutual information or **MI**, is defined as follows:

$$\mathbb{I}(X;Y) \triangleq \mathbb{KL}(P(X,Y)||P(X)P(X)) = \sum_x \sum_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \tag{2.51}$$

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:

$$\begin{aligned}
\mathbb{I}(X;Y) &= \mathbb{H}(X) - \mathbb{H}(X|Y) & (2.52)\\
&= \mathbb{H}(Y) - \mathbb{H}(Y|X) & (2.53)\\
&= \mathbb{H}(X) + \mathbb{H}(Y) - \mathbb{H}(X,Y) & (2.54)\\
&= \mathbb{H}(X,Y) - \mathbb{H}(X|Y) - \mathbb{H}(Y|X) & (2.55)
\end{aligned}$$

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. 2.10[13].

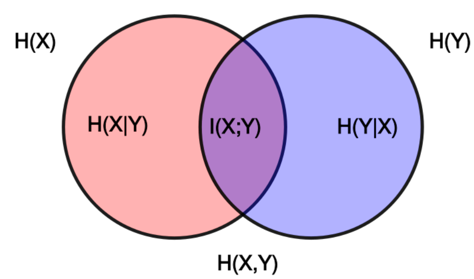


Fig. 2.10: Individual $\mathbb{H}(X), \mathbb{H}(Y)$, joint $\mathbb{H}(X,Y)$, and conditional entropies for a pair of correlated subsystems X,Y with mutual information $\mathbb{I}(X;Y)$.

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

$$PMI(x,y) \triangleq \log \frac{p(x,y)}{p(x)p(y)} = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)} \tag{2.56}$$

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:

$$PMI(x,y) = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)} \tag{2.57}$$

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 $p(y|x)$.

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

Chapter 3
Generative models for discrete data

3.1 Generative classifier

$$p(y|\boldsymbol{x},\boldsymbol{\theta})=\frac{p(y=c|\boldsymbol{\theta})p(\boldsymbol{x}|y=c,\boldsymbol{\theta})}{\sum_{c'}p(y=c'|\boldsymbol{\theta})p(\boldsymbol{x}|y=c',\boldsymbol{\theta})} \tag{3.1}$$

This is called a **generative classifier**, since it specifies how to generate the data using the **class conditional density** $p(\boldsymbol{x}|y=c)$ and the class prior $p(y=c)$.

3.2 Bayesian concept learning

Psychological research has shown that people can learn concepts from positive examples alone (Xu and Tenenbaum 2007).

We can think of learning the meaning of a word as equivalent to **concept learning**, which in turn is equivalent to binary classification. To see this, define $f(\boldsymbol{x})=1$ if xis an example of the concept C, and $f(\boldsymbol{x})=0$ otherwise. Then the goal is to learn the indicator function f, which just defines which elements are in the set C.

For pedagogical purposes, we will consider a very simple example of concept learning called the **number game**, based on part of Josh Tenenbaums PhD thesis (Tenenbaum 1999), see Figure 3.1.

3.2.1 Likelihood

$$p(\mathcal{D}|h)\triangleq\left(\frac{1}{\text{size}(h)}\right)^N=\left(\frac{1}{|h|}\right)^N \tag{3.2}$$

This crucial equation embodies what Tenenbaum calls the **size principle**, which means the model favours the simplest (smallest) hypothesis consistent with the data. This is more commonly known as **Occams razor**[14].

3.2.2 Prior

The prior is decided by human, not machines, so it is subjective. The subjectivity of the prior is controversial. For example, that a child and a math professor will reach different answers. In fact, they presumably not only have different priors, but also different hypothesis spaces. However, we can finesse that by defining the hypothesis space of the child and the math professor to be the same, and then setting the childs prior weight to be zero on certain advanced concepts. Thus there is no sharp distinction between the prior and the hypothesis space.

[14] http://en.wikipedia.org/wiki/Occam%27s_razor

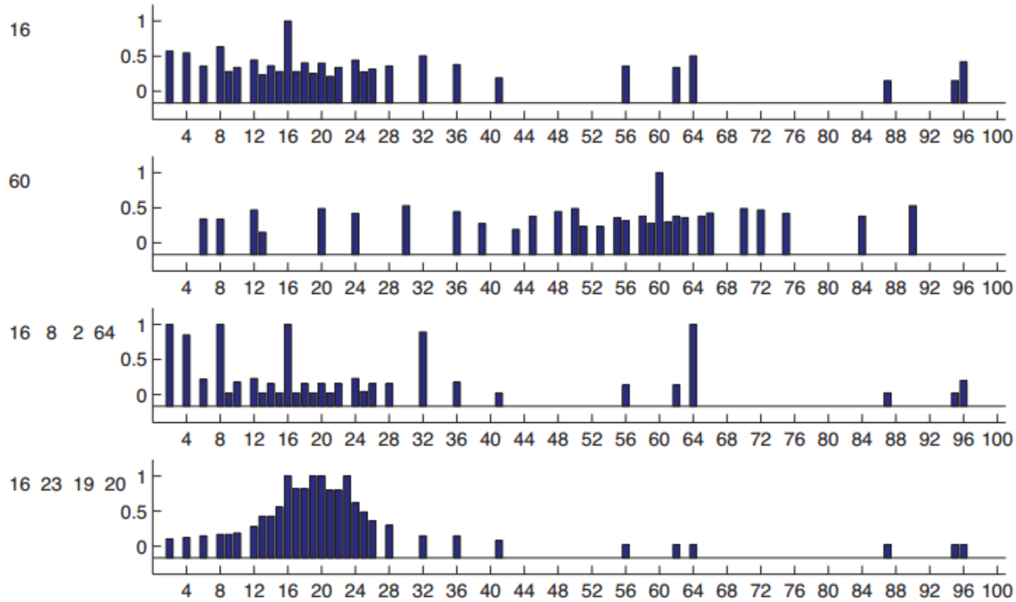


Fig. 3.1: Empirical predictive distribution averaged over 8 humans in the number game. First two rows: after seeing $\mathcal{D}=16$ and $\mathcal{D}=60$. This illustrates diffuse similarity. Third row: after seeing $\mathcal{D}=16,8,2,64$. This illustrates rule-like behaviour (powers of 2). Bottom row: after seeing $\mathcal{D}=16,23,19,20$. This illustrates focussed similarity (numbers near 20). Source: Figure 5.5 of (Tenenbaum 1999).

However, the prior is the mechanism by which background knowledge can be brought to bear on a problem. Without this, rapid learning (i.e., from small samples sizes) is impossible.

So, what prior should we use? For illustration purposes, let us use a simple prior which puts uniform probability on 30 simple arithmetical concepts, such as even numbers, odd numbers, prime numbers, numbers ending in 9, etc. We also include two unnatural concepts, namely powers of 2, plus 37 and powers of 2, except 32, but give them low prior weight. See Figure 3.2(a) for a plot of this prior.

3.2.3 Posterior

The posterior is simply the likelihood times the prior, normalized.

$$p(h|\mathcal{D}) \triangleq \frac{p(\mathcal{D}|h)p(h)}{\sum_{h'\in\mathcal{H}} p(\mathcal{D}|h')p(h')} = \frac{\mathbb{I}(\mathcal{D}\in h)p(h)}{\sum_{h'\in\mathcal{H}} \mathbb{I}(\mathcal{D}\in h')p(h')} \tag{3.3}$$

where $\mathbb{I}(\mathcal{D}\in h)p(h)$ is 1 **iff**(iff and only if) all the data are in the extension of the hypothesis h.

Figure 3.2 plots the prior, likelihood and posterior after seeing $\mathcal{D}=\{16\}$. We see that the posterior is a combination of prior and likelihood. In the case of most of the concepts, the prior is uniform, so the posterior is proportional to the likelihood. However, the unnatural concepts of powers of 2, plus 37 and powers of 2, except 32 have low posterior support, despite having high likelihood, due to the low prior. Conversely, the concept of odd numbers has low posterior support, despite having a high prior, due to the low likelihood.

Figure 3.3 plots the prior, likelihood and posterior after seeing $\mathcal{D}=\{16,8,2,64\}$. Now the likelihood is much more peaked on the powers of two concept, so this dominates the posterior.

In general, when we have enough data, the posterior $p(h|\mathcal{D})$ becomes peaked on a single concept, namely the MAP estimate, i.e.,

$$p(h|\mathcal{D}) \rightarrow \hat{h}^{MAP} \tag{3.4}$$

where $\hat{h}^{MAP}$ is the posterior mode,

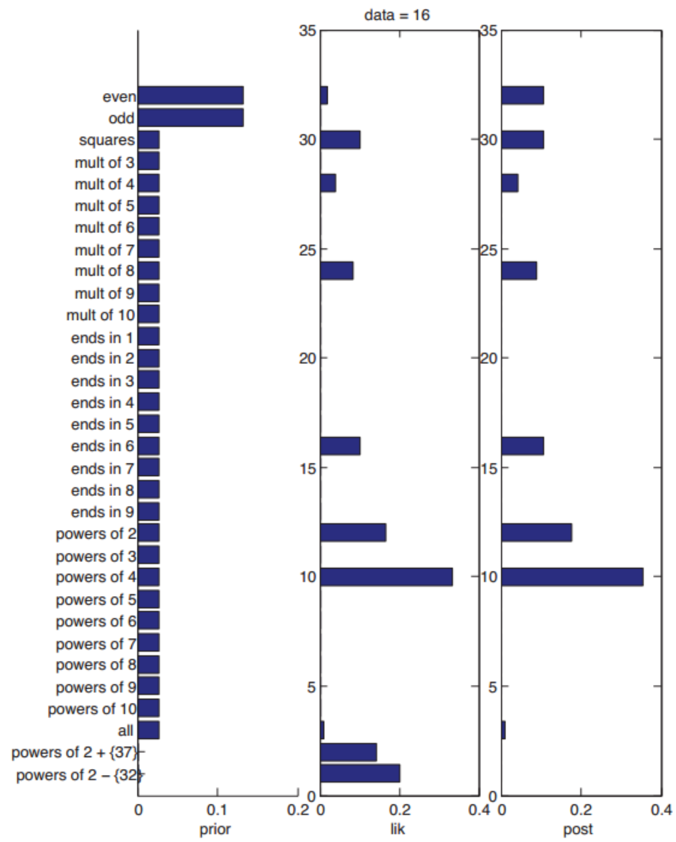


Fig. 3.2: Prior, likelihood and posterior for $\mathcal{D} = \{16\}$. Based on (Tenenbaum 1999).

$$\hat{h}^{MAP} \triangleq \arg\max_h p(h|\mathcal{D}) = \arg\max_h p(\mathcal{D}|h)p(h) = \arg\max_h [\log p(\mathcal{D}|h) + \log p(h)] \tag{3.5}$$

Since the likelihood term depends exponentially on N, and the prior stays constant, as we get more and more data, the MAP estimate converges towards the **maximum likelihood estimate** or **MLE**:

$$\hat{h}^{MLE} \triangleq \arg\max_h p(\mathcal{D}|h) = \arg\max_h \log p(\mathcal{D}|h) \tag{3.6}$$

In other words, if we have enough data, we see that the **data overwhelms the prior**.

3.2.4 Posterior predictive distribution

The concept of **posterior predictive distribution**[15] is normally used in a Bayesian context, where it makes use of the entire posterior distribution of the parameters given the observed data to yield a probability distribution over an interval rather than simply a point estimate.

$$p(\tilde{\boldsymbol{x}}|\mathcal{D}) \triangleq \mathbb{E}_{h|\mathcal{D}}[p(\tilde{\boldsymbol{x}}|h)] = \begin{cases} \sum_h p(\tilde{\boldsymbol{x}}|h)p(h|\mathcal{D}) & \text{, discrete parameters} \\ \int_h p(\tilde{\boldsymbol{x}}|h)p(h|\mathcal{D})\mathrm{d}h & \text{, continuous parameters} \end{cases} \tag{3.7}$$

This is just a weighted average of the predictions of each individual hypothesis and is called **Bayes model averaging**(Hoeting et al. 1999). This is illustrated in Figure 3.4. The dots at the bottom show the predictions from each

[15] http://en.wikipedia.org/wiki/Posterior_predictive_distribution

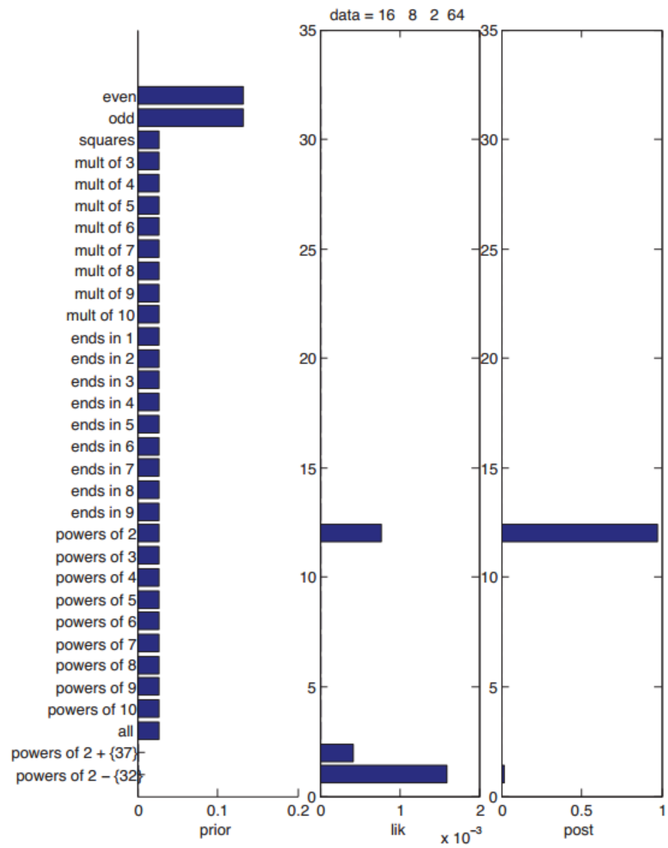


Fig. 3.3: Prior, likelihood and posterior for $\mathcal{D} = \{16, 8, 2, 64\}$. Based on (Tenenbaum 1999).

hypothesis; the vertical curve on the right shows the weight associated with each hypothesis. If we multiply each row by its weight and add up, we get the distribution at the top.

3.3 The beta-binomial model

3.3.1 Likelihood

Given $X \sim \text{Bin}(\theta)$, the likelihood of $\mathcal{D}$ is given by

$$p(\mathcal{D}|\theta) = \text{Bin}(N_1|N, \theta) \tag{3.8}$$

3.3.2 Prior

$$\text{Beta}(\theta|a, b) \propto \theta^{a-1}(1-\theta)^{b-1} \tag{3.9}$$

The parameters of the prior are called **hyper-parameters**.

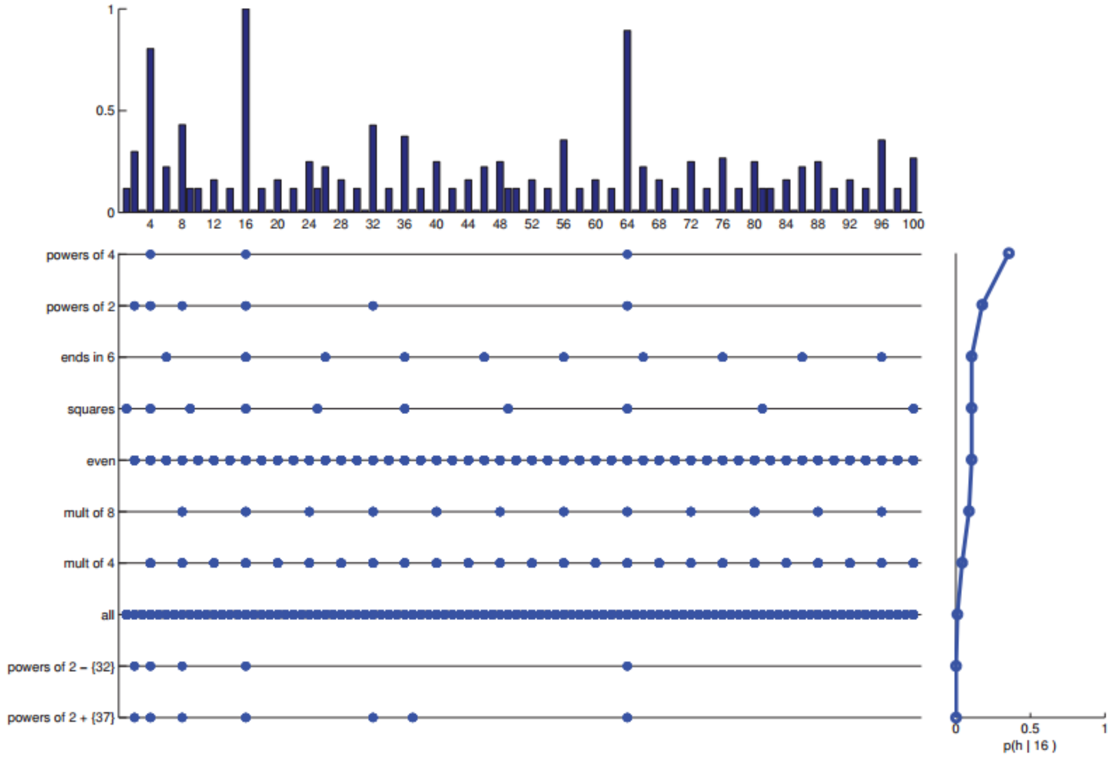


Fig. 3.4: Posterior over hypotheses and the corresponding predictive distribution after seeing one example,$\mathcal{D} = \{16\}$. A dot means this number is consistent with this hypothesis. The graph $p(h|\mathcal{D})$ on the right is the weight given to hypothesis h. By taking a weighed sum of dots, we get $p(\tilde{x}|\mathcal{D})$(top). Based on Figure 2.9 of (Tenenbaum 1999).

3.3.3 Posterior

$$p(\theta|\mathcal{D}) \propto \text{Bin}(N_1|N_1+N_0,\theta)\text{Beta}(\theta|a,b) = \text{Beta}(\theta|N_1+a, N_0 b) \tag{3.10}$$

Note that updating the posterior sequentially is equivalent to updating in a single batch. To see this, suppose we have two data sets $\mathcal{D}_a$ and $\mathcal{D}_b$ with sufficient statistics N_1^a, N_0^a and N_1^b, N_0^b. Let $N_1 = N_1^a + N_1^b$ and $N_0 = N_0^a + N_0^b$ be the sufficient statistics of the combined datasets. In batch mode we have

$$\begin{aligned}
p(\theta|\mathcal{D}_a,\mathcal{D}_b) &= p(\theta,\mathcal{D}_b|\mathcal{D}_a)p(\mathcal{D}_a) \\
&\propto p(\theta,\mathcal{D}_b|\mathcal{D}_a) \\
&= p(\mathcal{D}_b,\theta|\mathcal{D}_a) \\
&= p(\mathcal{D}_b|\theta)p(\theta|\mathcal{D}_a) \\
&\text{Combine Equation 3.10 and 2.18} \\
&= \text{Bin}(N_1^b|\theta, N_1^b+N_0^b)\text{Beta}(\theta|N_1^a+a, N_0^a+b) \\
&= \text{Beta}(\theta|N_1^a+N_1^b+a, N_0^a+N_0^b+b)
\end{aligned}$$

This makes Bayesian inference particularly well-suited to **online learning**, as we will see later.

3.3.3.1 Posterior mean and mode

From Table 2.7, the posterior mean is given by

$$\bar{\theta} = \frac{a+N_1}{a+b+N} \tag{3.11}$$

The mode is given by

$$\hat{\theta}_{MAP} = \frac{a+N_1-1}{a+b+N-2} \tag{3.12}$$

If we use a uniform prior, then the MAP estimate reduces to the MLE,

$$\hat{\theta}_{MLE} = \frac{N_1}{N} \tag{3.13}$$

We will now show that the posterior mean is convex combination of the prior mean and the MLE, which captures the notion that the posterior is a compromise between what we previously believed and what the data is telling us.

3.3.3.2 Posterior variance

The mean and mode are point estimates, but it is useful to know how much we can trust them. The variance of the posterior is one way to measure this. The variance of the Beta posterior is given by

$$\text{var}(\theta|\mathcal{D}) = \frac{(a+N_1)(b+N_0)}{(a+N_1+b+N_0)^2(a+N_1+b+N_0+1)} \tag{3.14}$$

We can simplify this formidable expression in the case that $N \gg a, b$, to get

$$\text{var}(\theta|\mathcal{D}) \approx \frac{N_1 N_0}{NNN} = \frac{\hat{\theta}_{MLE}(1-\hat{\theta}_{MLE})}{N} \tag{3.15}$$

3.3.4 Posterior predictive distribution

So far, we have been focusing on inference of the unknown parameter(s). Let us now turn our attention to prediction of future observable data.

Consider predicting the probability of heads in a single future trial under a Beta(a,b)posterior. We have

$$p(\tilde{x}|\mathcal{D}) = \int_0^1 p(\tilde{x}|\theta)p(\theta|\mathcal{D})\mathrm{d}\theta \tag{3.16}$$

$$= \int_0^1 \theta \text{Beta}(\theta|a,b)\mathrm{d}\theta = \mathbb{E}[\theta|\mathcal{D}] = \frac{a}{a+b} \tag{3.17}$$

3.3.4.1 Overfitting and the black swan paradox

Let us now derive a simple Bayesian solution to the problem. We will use a uniform prior, so $a = b = 1$. In this case, plugging in the posterior mean gives **Laplaces rule of succession**

$$p(\tilde{x}|\mathcal{D}) = \frac{N_1+1}{N_0+N_1+1} \tag{3.18}$$

This justifies the common practice of adding 1 to the empirical counts, normalizing and then plugging them in, a technique known as **add-one smoothing**. (Note that plugging in the MAP parameters would not have this smoothing effect, since the mode becomes the MLE if $a = b = 1$, see Section 3.3.3.1.)

3.3.4.2 Predicting the outcome of multiple future trials

Suppose now we were interested in predicting the number of heads, $\tilde{x}$, in M future trials. This is given by

$$p(\tilde{x}|\mathcal{D}) = \int_0^1 \text{Bin}(\tilde{x}|M,\theta)\text{Beta}(\theta|a,b)\text{d}\theta \tag{3.19}$$

$$= \binom{M}{\tilde{x}} \frac{1}{B(a,b)} \int_0^1 \theta^{\tilde{x}}(1-\theta)^{M-\tilde{x}}\theta^{a-1}(1-\theta)^{b-1}\text{d}\theta \tag{3.20}$$

We recognize the integral as the normalization constant for a $\text{Beta}(a+\tilde{x}, M\tilde{x}+b)$ distribution. Hence

$$\int_0^1 \theta^{\tilde{x}}(1-\theta)^{M-\tilde{x}}\theta^{a-1}(1-\theta)^{b-1}\text{d}\theta = B(\tilde{x}+a, M-\tilde{x}+b) \tag{3.21}$$

Thus we find that the posterior predictive is given by the following, known as the (compound) **beta-binomial distribution**:

$$Bb(x|a,b,M) \triangleq \binom{M}{x} \frac{B(x+a, M-x+b)}{B(a,b)} \tag{3.22}$$

This distribution has the following mean and variance

$$\text{mean} = M\frac{a}{a+b}\text{ , var} = \frac{Mab}{(a+b)^2}\frac{a+b+M}{a+b+1} \tag{3.23}$$

This process is illustrated in Figure 3.5. We start with a $\text{Beta}(2,2)$ prior, and plot the posterior predictive density after seeing $N_1 = 3$ heads and $N_0 = 17$ tails. Figure 3.5(b) plots a plug-in approximation using a MAP estimate. We see that the Bayesian prediction has longer tails, spreading its probability mass more widely, and is therefore less prone to overfitting and blackswan type paradoxes.

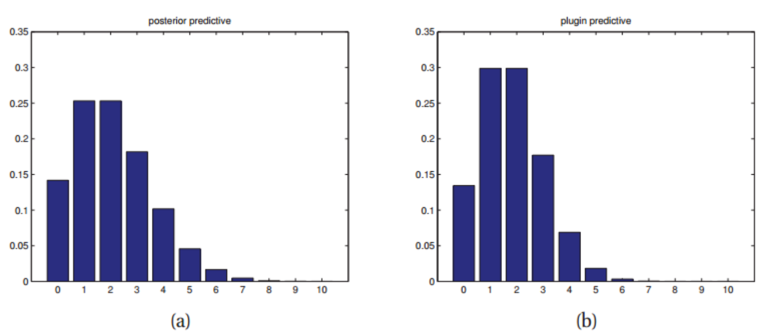


Fig. 3.5: (a) Posterior predictive distributions after seeing $N_1 = 3, N_0 = 17$. (b) MAP estimation.

3.4 The Dirichlet-multinomial model

In the previous section, we discussed how to infer the probability that a coin comes up heads. In this section, we generalize these results to infer the probability that a dice with K sides comes up as face k.

3.4.1 Likelihood

Suppose we observe N dice rolls, $\mathcal{D} = \{x_1, x_2, \cdots, x_N\}$, where $x_i \in \{1, 2, \cdots, K\}$. The likelihood has the form

$$p(\mathcal{D}|\boldsymbol{\theta}) = \binom{N}{N_1 \cdots N_k} \prod_{k=1}^{K} \theta_k^{N_k} \quad \text{where } N_k = \sum_{i=1}^{N} \mathbb{I}(y_i = k) \tag{3.24}$$

almost the same as Equation (2.20).

3.4.2 Prior

$$\mathrm{Dir}(\boldsymbol{\theta}|\boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1} \mathbb{I}(\boldsymbol{\theta} \in S_K) \tag{3.25}$$

3.4.3 Posterior

$$p(\boldsymbol{\theta}|\mathcal{D}) \propto p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}) \tag{3.26}$$

$$\propto \prod_{k=1}^{K} \theta_k^{N_k} \theta_k^{\alpha_k - 1} = \prod_{k=1}^{K} \theta_k^{N_k + \alpha_k - 1} \tag{3.27}$$

$$= \mathrm{Dir}(\boldsymbol{\theta}|\alpha_1 + N_1, \cdots, \alpha_K + N_K) \tag{3.28}$$

From Equation (2.37), the MAP estimate is given by

$$\hat{\theta}_k = \frac{N_k + \alpha_k - 1}{N + \alpha_0 - K} \tag{3.29}$$

If we use a uniform prior, $\alpha_k = 1$, we recover the MLE:

$$\hat{\theta}_k = \frac{N_k}{N} \tag{3.30}$$

3.4.4 Posterior predictive distribution

The posterior predictive distribution for a single multinoulli trial is given by the following expression:

$$p(X = j|\mathcal{D}) = \int p(X = j|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D})\mathrm{d}\boldsymbol{\theta} \tag{3.31}$$

$$= \int p(X = j|\theta_j) \left[\int p(\boldsymbol{\theta}_{-j}, \theta_j|\mathcal{D})\mathrm{d}\boldsymbol{\theta}_{-j} \right] \mathrm{d}\theta_j \tag{3.32}$$

$$= \int \theta_j p(\theta_j|\mathcal{D})\mathrm{d}\theta_j = \mathbb{E}[\theta_j|\mathcal{D}] = \frac{\alpha_j + N_j}{\alpha_0 + N} \tag{3.33}$$

where $\boldsymbol{\theta}_{-j}$ are all the components of $\boldsymbol{\theta}$ except θ_j.

The above expression avoids the zero-count problem. In fact, this form of Bayesian smoothing is even more important in the multinomial case than the binary case, since the likelihood of data sparsity increases once we start partitioning the data into many categories.

3.5 Naive Bayes classifiers

Assume the features are **conditionally independent** given the class label, then the class conditional density has the following form

$$p(\boldsymbol{x}|y,\boldsymbol{\theta}) = \prod_{j=1}^{D} p(x_j|y=c,\boldsymbol{\theta}_{jc}) \quad \text{where } \boldsymbol{x} \text{ is a } D\text{-dimensional feature vector} \tag{3.34}$$

The resulting model is called a **naive Bayes classifier**(NBC).

The form of the class-conditional density depends on the type of each feature. We give some possibilities below:

- In the case of real-valued features, we can use the Gaussian distribution: $p(\boldsymbol{x}|y,\boldsymbol{\theta}) = \prod_{j=1}^{D} \mathcal{N}(x_j|\mu_{jc},\sigma_{jc}^2)$, where μ_{jc} is the mean of feature j in objects of class c, and σ_{jc}^2 is its variance.
- In the case of binary features, $x_j \in \{0,1\}$, we can use the Bernoulli distribution: $p(\boldsymbol{x}|y,\boldsymbol{\theta}) = \prod_{j=1}^{D} \text{Ber}(x_j|\mu_{jc})$, where μ_{jc} is the probability that feature j occurs in class c. This is sometimes called the **multivariate Bernoulli naive Bayes** model. We will see an application of this below.
- In the case of categorical features, $x_j \in \{a_{j1}, a_{j2}, \cdots, a_{jS_j}\}$, we can use the multinoulli distribution: $p(\boldsymbol{x}|y,\boldsymbol{\theta}) = \prod_{j=1}^{D} \text{Cat}(x_j|\boldsymbol{\mu}_{jc})$, where $\boldsymbol{\mu}_{jc}$ is a histogram over the K possible values for x_j in class c.

Obviously we can handle other kinds of features, or use different distributional assumptions. Also, it is easy to mix and match features of different types.

3.5.1 Optimization

We now discuss how to train a naive Bayes classifier. This usually means computing the MLE or the MAP estimate for the parameters. However, we will also discuss how to compute the full posterior, $p(\boldsymbol{\theta}|\mathcal{D})$.

3.5.1.1 MLE for NBC

The probability for a single data case is given by

$$p(\boldsymbol{x}_i,y_i|\boldsymbol{\theta}) = p(y_i|\boldsymbol{\pi})\prod_j p(x_{ij}|\boldsymbol{\theta}_j) = \prod_c \pi_c^{\mathbb{I}(y_i=c)} \prod_j \prod_c p(x_{ij}|\boldsymbol{\theta}_{jc})^{\mathbb{I}(y_i=c)} \tag{3.35}$$

Hence the log-likelihood is given by

$$p(\mathcal{D}|\boldsymbol{\theta}) = \sum_{c=1}^{C} N_c \log \pi_c + \sum_{j=1}^{D}\sum_{c=1}^{C}\sum_{i:y_i=c} \log p(x_{ij}|\boldsymbol{\theta}_{jc}) \tag{3.36}$$

where $N_c \triangleq \sum_i \mathbb{I}(y_i = c)$ is the number of feature vectors in class c.

We see that this expression decomposes into a series of terms, one concerning $\boldsymbol{\pi}$, and DC terms containing the θ_{jc}s. Hence we can optimize all these parameters separately.

From Equation (3.30), the MLE for the class prior is given by

$$\hat{\pi}_c = \frac{N_c}{N} \tag{3.37}$$

The MLE for θ_{jc}s depends on the type of distribution we choose to use for each feature.

In the case of binary features, $x_j \in \{0,1\}$, $x_j|y=c \sim \text{Ber}(\theta_{jc})$, hence

$$\hat{\theta}_{jc} = \frac{N_{jc}}{N_c} \tag{3.38}$$

where $N_{jc} \triangleq \sum\limits_{i:y_i=c} \mathbb{I}(y_i = c)$ is the number that feature j occurs in class c.

In the case of categorical features, $x_j \in \{a_{j1}, a_{j2}, \cdots, a_{jS_j}\}$, $x_j|y = c \sim \text{Cat}(\boldsymbol{\theta}_{jc})$, hence

$$\hat{\boldsymbol{\theta}}_{jc} = (\frac{N_{j1c}}{N_c}, \frac{N_{j2c}}{N_c}, \cdots, \frac{N_{jS_j}}{N_c})^T \tag{3.39}$$

where $N_{jkc} \triangleq \sum\limits_{i=1}^{N} \mathbb{I}(x_{ij} = a_{jk}, y_i = c)$ is the number that feature $x_j = a_{jk}$ occurs in class c.

3.5.1.2 Bayesian naive Bayes

Use a $\text{Dir}(\boldsymbol{\alpha})$ prior for $\boldsymbol{\pi}$.

In the case of binary features, use a $\text{Beta}(\beta 0, \beta 1)$ prior for each $\boldsymbol{\theta}_{jc}$; in the case of categorical features, use a $\text{Dir}(\boldsymbol{\alpha})$ prior for each $\boldsymbol{\theta}_{jc}$. Often we just take $\boldsymbol{\alpha} = \mathbf{1}$ and $\boldsymbol{\beta} = \mathbf{1}$, corresponding to **add-one** or **Laplace smoothing**.

3.5.2 Using the model for prediction

The goal is to compute

$$y = f(\boldsymbol{x}) = \arg\max_c P(y = c|\boldsymbol{x}, \boldsymbol{\theta}) = P(y = c|\boldsymbol{\theta}) \prod_{j=1}^{D} P(x_j|y = c, \boldsymbol{\theta}) \tag{3.40}$$

We can the estimate parameters using MLE or MAP, then the posterior predictive density is obtained by simply plugging in the parameters $\bar{\boldsymbol{\theta}}$(MLE) or $\hat{\boldsymbol{\theta}}$(MAP).

Or we can use BMA, just integrate out the unknown parameters.

3.5.3 The log-sum-exp trick

when using generative classifiers of any kind, computing the posterior over class labels using Equation (3.1) can fail due to **numerical underflow**. The problem is that $p(\boldsymbol{x}|y = c)$ is often a very small number, especially if $\boldsymbol{x}$ is a high-dimensional vector. This is because we require that $\sum_{\boldsymbol{x}} p(\boldsymbol{x}|y) = 1$, so the probability of observing any particular high-dimensional vector is small. The obvious solution is to take logs when applying Bayes rule, as follows:

$$\log p(y = c|\boldsymbol{x}, \boldsymbol{\theta}) = b_c - \log\left(\sum_{c'} e^{b_{c'}}\right) \quad \text{where } b_c \triangleq \log p(\boldsymbol{x}|y = c, \boldsymbol{\theta}) + \log p(y = c|\boldsymbol{\theta}) \tag{3.41}$$

we can factor out the largest term, and just represent the remaining numbers relative to that. For example,

$$\log(e^{-120} + e^{-121}) = \log(e^{-120}(1 + e^{-1})) = \log(1 + e^{-1}) - 120 \tag{3.42}$$

In general, we have

$$\sum_c e^{b_c} = \log\left[(\sum e^{b_c - B})e^B\right] = \log\left(\sum e^{b_c - B}\right) + B \quad \text{where } B \triangleq \max\{b_c\} \tag{3.43}$$

This is called the **log-sum-exp** trick, and is widely used.

3.5.4 Feature selection using mutual information

Since an NBC is fitting a joint distribution over potentially many features, it can suffer from overfitting. In addition, the run-time cost is $O(D)$, which may be too high for some applications.

One common approach to tackling both of these problems is to perform **feature selection**, to remove irrelevant features that do not help much with the classification problem. The simplest approach to feature selection is to evaluate the relevance of each feature separately, and then take the top K,whereKis chosen based on some tradeoff between accuracy and complexity. This approach is known as **variable ranking**, **filtering**, or **screening**.

One way to measure relevance is to use mutual information (Section 2.8.3) between feature X_j and the class label Y

$$\mathbb{I}(X_j,Y) = \sum_{x_j}\sum_{y} p(x_j,y)\log\frac{p(x_j,y)}{p(x_j)p(y)} \tag{3.44}$$

If the features are binary, it is easy to show that the MI can be computed as follows

$$\mathbb{I}_j = \sum_{c}\left[\theta_{jc}\pi_c\log\frac{\theta_{jc}}{\theta_j} + (1-\theta_{jc})\pi_c\log\frac{1-\theta_{jc}}{1-\theta_j}\right] \tag{3.45}$$

where $\pi_c = p(y=c)$, $\theta_{jc} = p(x_j=1|y=c)$, and $\theta_j = p(x_j=1) = \sum_c \pi_c\theta_{jc}$.

3.5.5 Classifying documents using bag of words

Document classification is the problem of classifying text documents into different categories. One simple approach is to represent each document as a binary vector, which records whether each word is present or not, so $x_{ij}=1$ iff word j occurs in document i, otherwise $x_{ij}=0$. We can then use the following class conditional density:

$$p(\boldsymbol{x}_i|y_i=c,\boldsymbol{\theta}) = \tag{3.46}$$

3.5.5.1 Bernoulli product model

Represent each document as a binary vector, which records whether each word is present or not, so $x_{ij}=1$ iff word j occurs in document i, otherwise $x_{ij}=0$. We can then use the following class conditional density:

$$p(\boldsymbol{x}_i|y_i=c,\boldsymbol{\theta}) = \prod_{j=1}^{D}\text{Ber}(x_{ij}|\theta_{jc}) = \prod_{j=1}^{D}\theta_{jc}^{\mathbb{I}(x_{ij})}(1-\theta_{jc})^{1-\mathbb{I}(x_{ij})} \tag{3.47}$$

This is called the **Bernoulli product model**, or the **binary independence model**.

3.5.5.2 Multinomial document classifier

However, ignoring the number of times each word occurs in a document loses some information (McCallum and Nigam 1998). A more accurate representation counts the number of occurrences of each word. Specifically, let $\boldsymbol{x}_i$ be a vector of counts for document i, so $x_{ij}\in\{0,1,\cdots,N_i\}$, where N_i is the number of terms in document i(so $\sum\limits_{j=1}^{D} x_{ij}=N_i$). For the class conditional densities, we can use a multinomial distribution:

$$p(\boldsymbol{x}_i|y_i=c,\boldsymbol{\theta}) = \text{Mu}(\boldsymbol{x}_i|N_i,\boldsymbol{\theta}_c) = \frac{N_i!}{\prod_{j=1}^{D}x_{ij}!}\prod_{j=1}^{D}\theta_{jc}^{x_{ij}} \tag{3.48}$$

where we have implicitly assumed that the document length N_i is independent of the class. Here $_{jc}$ is the probability of generating word j in documents of class c; these parameters satisfy the constraint that $\sum_{j=1}^{D} \theta_{jc} = 1$ for each class c.

Although the multinomial classifier is easy to train and easy to use at test time, it does not work particularly well for document classification. One reason for this is that it does not take into account the **burstiness** of word usage. This refers to the phenomenon that most words never appear in any given document, but if they do appear once, they are likely to appear more than once, i.e., words occur in bursts.

The multinomial model cannot capture the burstiness phenomenon. To see why, note that Equation (3.48) has the form $\theta_{jc}^{x_{ij}}$, and since $\theta_{jc} \ll 1$ for rare words, it becomes increasingly unlikely to generate many of them. For more frequent words, the decay rate is not as fast. To see why intuitively, note that the most frequent words are function words which are not specific to the class, such as and, the, and but; the chance of the word and occuring is pretty much the same no matter how many time it has previously occurred (modulo document length), so the independence assumption is more reasonable for common words. However, since rare words are the ones that matter most for classification purposes, these are the ones we want to model the most carefully.

3.5.5.3 DCM model

Various ad hoc heuristics have been proposed to improve the performance of the multinomial document classifier (Rennie et al. 2003). We now present an alternative class conditional density that performs as well as these ad hoc methods, yet is probabilistically sound (Madsen et al. 2005).

Suppose we simply replace the multinomial class conditional density with the **Dirichlet Compound Multinomial** or **DCM** density, defined as follows:

$$p(\boldsymbol{x}_i|y_i = c, \boldsymbol{\alpha}) = \int \mathrm{Mu}(\boldsymbol{x}_i|N_i, \boldsymbol{\theta}_c)\mathrm{Dir}(\boldsymbol{\theta}_c|\boldsymbol{\alpha}_c) = \frac{N_i!}{\prod_{j=1}^{D} x_{ij}!} \prod_{j=1}^{D} \frac{B(\boldsymbol{x}_i + \boldsymbol{\alpha}_c)}{B(\boldsymbol{\alpha}_c)} \tag{3.49}$$

(This equation is derived in Equation TODO.) Surprisingly this simple change is all that is needed to capture the burstiness phenomenon. The intuitive reason for this is as follows: After seeing one occurence of a word, say wordj, the posterior counts on j gets updated, making another occurence of wordjmore likely. By contrast, ifj is fixed, then the occurences of each word are independent. The multinomial model corresponds to drawing a ball from an urn with Kcolors of ball, recording its color, and then replacing it. By contrast, the DCM model corresponds to drawing a ball, recording its color, and then replacing it with one additional copy; this is called the **Polya urn**.

Using the DCM as the class conditional density gives much better results than using the multinomial, and has performance comparable to state of the art methods, as described in (Madsen et al. 2005). The only disadvantage is that fitting the DCM model is more complex; see (Minka 2000e; Elkan 2006) for the details.

Chapter 4
Gaussian Models

In this chapter, we discuss the **multivariate Gaussian** or **multivariate normal(MVN)**, which is the most widely used joint probability density function for continuous variables. It will form the basis for many of the models we will encounter in later chapters.

4.1 Basics

Recall from Section 2.5.2 that the pdf for an MVN in D dimensions is defined by the following:

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\Sigma) \triangleq \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right] \tag{4.1}$$

The expression inside the exponent is the Mahalanobis distance between a data vector $\boldsymbol{x}$ and the mean vector $\boldsymbol{\mu}$, We can gain a better understanding of this quantity by performing an **eigendecomposition** of $\boldsymbol{\Sigma}$. That is, we write $\boldsymbol{\Sigma} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$, where $\boldsymbol{U}$ is an orthonormal matrix of eigenvectors satsifying $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}$, and $\boldsymbol{\Lambda}$ is a diagonal matrix of eigenvalues. Using the eigendecomposition, we have that

$$\boldsymbol{\Sigma}^{-1} = \boldsymbol{U}^{-T}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}^{-1} = \boldsymbol{U}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}^T = \sum_{i=1}^{D}\frac{1}{\lambda_i}\boldsymbol{u}_i\boldsymbol{u}_i^T \tag{4.2}$$

where $\boldsymbol{u}_i$ is the ith column of $\boldsymbol{U}$, containing the ith eigenvector. Hence we can rewrite the Mahalanobis distance as follows:

$$\begin{aligned}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) &= (\boldsymbol{x}-\boldsymbol{\mu})^T\left(\sum_{i=1}^{D}\frac{1}{\lambda_i}\boldsymbol{u}_i\boldsymbol{u}_i^T\right)(\boldsymbol{x}-\boldsymbol{\mu}) &(4.3)\\ &= \sum_{i=1}^{D}\frac{1}{\lambda_i}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{u}_i\boldsymbol{u}_i^T(\boldsymbol{x}-\boldsymbol{\mu}) = \sum_{i=1}^{D}\frac{y_i^2}{\lambda_i} &(4.4)\end{aligned}$$

where $y_i \triangleq \boldsymbol{u}_i^T(\boldsymbol{x}-\boldsymbol{\mu})$. Recall that the equation for an ellipse in 2d is

$$\frac{y_1^2}{\lambda_1} + \frac{y_2^2}{\lambda_2} = 1 \tag{4.5}$$

Hence we see that the contours of equal probability density of a Gaussian lie along ellipses. This is illustrated in Figure 4.1. The eigenvectors determine the orientation of the ellipse, and the eigenvalues determine how elogonated it is.

In general, we see that the Mahalanobis distance corresponds to Euclidean distance in a transformed coordinate system, where we shift by $\boldsymbol{\mu}$ and rotate by $\boldsymbol{U}$.

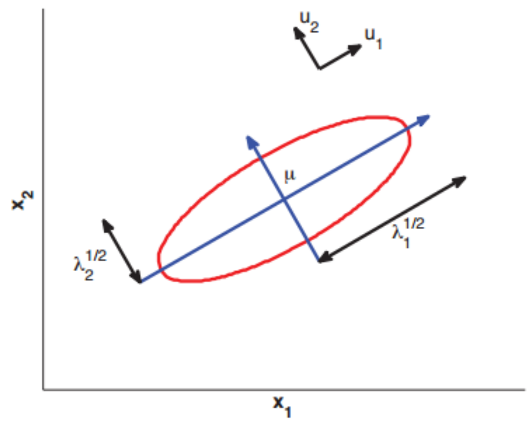


Fig. 4.1: Visualization of a 2 dimensional Gaussian density. The major and minor axes of the ellipse are defined by the first two eigenvectors of the covariance matrix, namely $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$. Based on Figure 2.7 of (Bishop 2006a)

4.1.1 MLE for a MVN

Theorem 4.1. *(**MLE for a MVN**) If we have N iid samples $\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, then the MLE for the parameters is given by*

$$\bar{\boldsymbol{\mu}} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{x}_i \triangleq \bar{\boldsymbol{x}} \tag{4.6}$$

$$\bar{\boldsymbol{\Sigma}} = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T = \frac{1}{N}\left(\sum_{i=1}^{N}\boldsymbol{x}_i\boldsymbol{x}_i^T\right) - \bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T \tag{4.7}$$

4.1.2 Maximum entropy derivation of the Gaussian *

In this section, we show that the multivariate Gaussian is the distribution with maximum entropy subject to having a specified mean and covariance (see also Section TODO). This is one reason the Gaussian is so widely used: the first two moments are usually all that we can reliably estimate from data, so we want a distribution that captures these properties, but otherwise makes as few addtional assumptions as possible.

To simplify notation, we will assume the mean is zero. The pdf has the form

$$f(\boldsymbol{x}) = \frac{1}{Z}\exp\left(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right) \tag{4.8}$$

4.2 Gaussian discriminant analysis

One important application of MVNs is to define the the class conditional densities in a generative classifier, i.e.,

$$p(\boldsymbol{x}|y = c, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \tag{4.9}$$

The resulting technique is called (Gaussian) **discriminant analysis** or **GDA** (even though it is a generative, not discriminative, classifier see Section TODO for more on this distinction). If $\boldsymbol{\Sigma}_c$ is diagonal, this is equivalent to naive Bayes.

We can classify a feature vector using the following decision rule, derived from Equation (3.1):

$$y = \arg\max_c \left[\log p(y=c|\boldsymbol{\pi}) + \log p(\boldsymbol{x}|\boldsymbol{\theta})\right] \tag{4.10}$$

When we compute the probability of $\boldsymbol{x}$ under each class conditional density, we are measuring the distance from $\boldsymbol{x}$ to the center of each class, $\boldsymbol{\mu}_c$, using Mahalanobis distance. This can be thought of as a **nearest centroids classifier**.

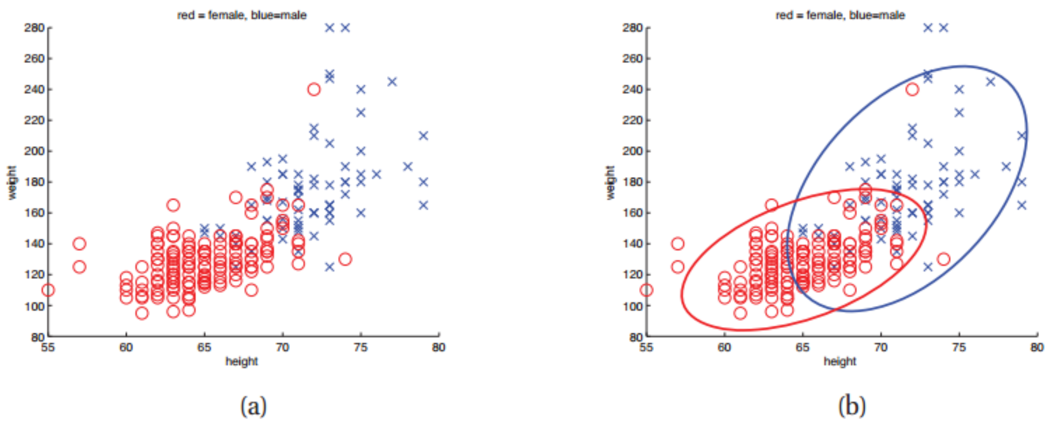


Fig. 4.2: (a) Height/weight data. (b) Visualization of 2d Gaussians fit to each class. 95% of the probability mass is inside the ellipse.

As an example, Figure 4.2 shows two Gaussian class-conditional densities in 2d, representing the height and weight of men and women. We can see that the features are correlated, as is to be expected (tall people tend to weigh more). The ellipses for each class contain 95% of the probability mass. If we have a uniform prior over classes, we can classify a new test vector as follows:

$$y = \arg\max_c (\boldsymbol{x}-\boldsymbol{\mu}_c)^T \boldsymbol{\Sigma}_c^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_c) \tag{4.11}$$

4.2.1 Quadratic discriminant analysis (QDA)

By plugging in the definition of the Gaussian density to Equation (3.1), we can get

$$p(y|\boldsymbol{x},\boldsymbol{\theta}) = \frac{\pi_c |2\pi\boldsymbol{\Sigma}_c|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]}{\sum_{c'} \pi_{c'} |2\pi\boldsymbol{\Sigma}_{c'}|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]} \tag{4.12}$$

Thresholding this results in a quadratic function ofx. The result is known as quadratic discriminant analysis(QDA). Figure 4.3 gives some examples of what the decision boundaries look like in 2D.

4.2.2 Linear discriminant analysis (LDA)

We now consider a special case in which the covariance matrices are **tied** or **shared** across classes,$\boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}$. In this case, we can simplify Equation (4.12) as follows:

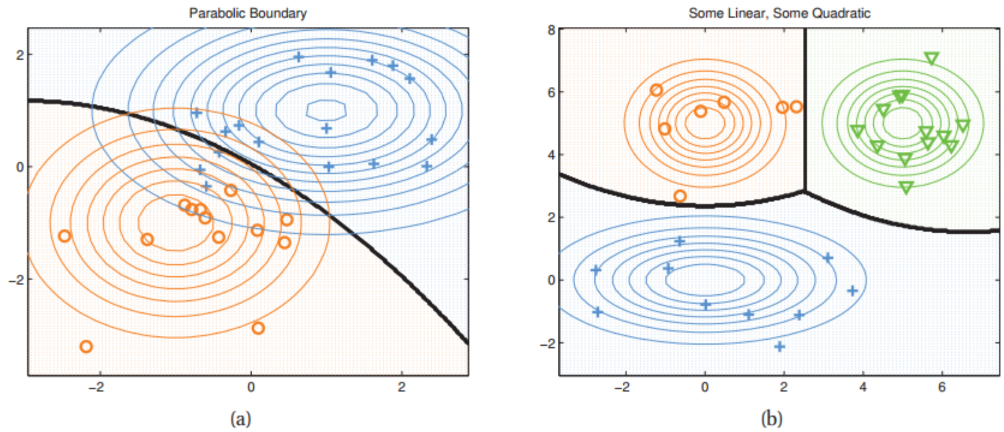


Fig. 4.3: Quadratic decision boundaries in 2D for the 2 and 3 class case.

$$p(y|\boldsymbol{x},\boldsymbol{\theta}) \propto \pi_c \exp\left(\boldsymbol{\mu}_c\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c\right) \tag{4.13}$$

$$= \exp\left(\boldsymbol{\mu}_c\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log\pi_c\right)\exp\left(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right) \tag{4.14}$$

$$\propto \exp\left(\boldsymbol{\mu}_c\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log\pi_c\right) \tag{4.15}$$

Since the quadratic term $\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}$ is independent of c, it will cancel out in the numerator and denominator. If we define

$$\gamma_c \triangleq -\frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log\pi_c \tag{4.16}$$

$$\boldsymbol{\beta}_c \triangleq \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c \tag{4.17}$$

then we can write

$$p(y|\boldsymbol{x},\boldsymbol{\theta}) = \frac{e^{\boldsymbol{\beta}_c^T\boldsymbol{x}+\gamma_c}}{\sum_{c'} e^{\boldsymbol{\beta}_{c'}^T\boldsymbol{x}+\gamma_{c'}}} \triangleq \sigma(\boldsymbol{\eta},c) \quad \text{where } \boldsymbol{\eta} \triangleq (e^{\boldsymbol{\beta}_1^T\boldsymbol{x}}+\gamma_1,\cdots,e^{\boldsymbol{\beta}_C^T\boldsymbol{x}}+\gamma_C) \tag{4.18}$$

where $\sigma()$ is the **softmax activation function**[16], defined as follows:

$$\sigma(\boldsymbol{q},i) \triangleq \frac{\exp(q_i)}{\sum_{j=1}^n \exp(q_j)} \tag{4.19}$$

When parameterized by some constant, $\alpha > 0$, the following formulation becomes a smooth, differentiable approximation of the maximum function:

$$\mathcal{S}_\alpha(\boldsymbol{x}) = \frac{\sum_{j=1}^D x_j e^{\alpha x_j}}{\sum_{j=1}^D e^{\alpha x_j}} \tag{4.20}$$

$\mathcal{S}_\alpha$ has the following properties:

1. $\mathcal{S}_\alpha \to \max$ as $\alpha \to \infty$
2. $\mathcal{S}_0$ is the average of its inputs
3. $\mathcal{S}_\alpha \to \min$ as $\alpha \to -\infty$

Note that the softmax activation function comes from the area of statistical physics, where it is common to use the **Boltzmann distribution**, which has the same form as the softmax activation function.

[16] http://en.wikipedia.org/wiki/Softmax_activation_function

An interesting property of Equation (4.18) is that, if we take logs, we end up with a linear function of $\boldsymbol{x}$. (The reason it is linear is because the $\boldsymbol{x}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{x}$ cancels from the numerator and denominator.) Thus the decision boundary between any two classes, says c and c', will be a straight line. Hence this technique is called **linear discriminant analysis** or **LDA**.

An alternative to fitting an LDA model and then deriving the class posterior is to directly fit $p(y|\boldsymbol{x}, \boldsymbol{W}) = \text{Cat}(y|\boldsymbol{W}\boldsymbol{x})$ for some $C \times D$ weight matrix $\boldsymbol{W}$. This is called **multi-class logistic regression**, or **multinomial logistic regression**. We will discuss this model in detail in Section TODO. The difference between the two approaches is explained in Section TODO.

4.2.3 Two-class LDA

To gain further insight into the meaning of these equations, let us consider the binary case. In this case, the posterior is given by

$$p(y=1|\boldsymbol{x},\boldsymbol{\theta}) = \frac{e^{\boldsymbol{\beta}_1^T \boldsymbol{x} + \gamma_1}}{e^{\boldsymbol{\beta}_0^T \boldsymbol{x} + \gamma_0} + e^{\boldsymbol{\beta}_1^T \boldsymbol{x} + \gamma_1}}) \tag{4.21}$$

$$= \frac{1}{1 + e^(\boldsymbol{\beta}_0 - \boldsymbol{\beta}_1)^T \boldsymbol{x} + (\gamma_0 - \gamma_1)} = \text{sigm}((\boldsymbol{\beta}_1 - \boldsymbol{\beta}_0)^T \boldsymbol{x} + (\gamma_0 - \gamma_1)) \tag{4.22}$$

where $\text{sigm}(x)$ refers to the sigmoid function[17].

Now

$$\gamma_1 - \gamma_0 = -\frac{1}{2}\boldsymbol{\mu}_1^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_0^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_0 + \log(\pi_1/\pi_0) \tag{4.23}$$

$$= -\frac{1}{2}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0) + \log(\pi_1/\pi_0) \tag{4.24}$$

So if we define

$$\boldsymbol{w} = \boldsymbol{\beta}_1 - \boldsymbol{\beta}_0 = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) \tag{4.25}$$

$$\boldsymbol{x}_0 = \frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0) - (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)\frac{\log(\pi_1/\pi_0)}{(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)} \tag{4.26}$$

then we have $\boldsymbol{w}^T \boldsymbol{x}_0 = -(\gamma_1 - \gamma_0)$, and hence

$$p(y=1|\boldsymbol{x},\boldsymbol{\theta}) = \text{sigm}(\boldsymbol{w}^T(\boldsymbol{x} - \boldsymbol{x}_0)) \tag{4.27}$$

(This is closely related to logistic regression, which we will discuss in Section TODO.) So the final decision rule is as follows: shift $\boldsymbol{x}$ by $\boldsymbol{x}_0$, project onto the line $\boldsymbol{w}$, and see if the result is positive or negative.

If $\boldsymbol{\Sigma} = \sigma^2 \boldsymbol{I}$, then $\boldsymbol{w}$ is in the direction of $\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0$. So we classify the point based on whether its projection is closer to $\boldsymbol{\mu}_0$ or $\boldsymbol{\mu}_1$. This is illustrated in Figure 4.4. Furthemore, if $\boldsymbol{\pi}_1 = \boldsymbol{\pi}_0$, then $\boldsymbol{x}_0 = \frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0)$, which is half way between the means. If we make $\boldsymbol{\pi}_1 > \boldsymbol{\pi}_0$, then $\boldsymbol{x}_0$ gets closer to $\boldsymbol{\mu}_0$, so more of the line belongs to class 1 *a priori*. Conversely if $\boldsymbol{\pi}_1 < \boldsymbol{\pi}_0$, the boundary shifts right. Thus we see that the class prior, c, just changes the decision threshold, and not the overall geometry, as we claimed above. (A similar argument applies in the multi-class case.)

The magnitude of $\boldsymbol{w}$ determines the steepness of the logistic function, and depends on how well-separated the means are, relative to the variance. In psychology and signal detection theory, it is common to define the **discriminability** of a signal from the background noise using a quantity called **d-prime**:

$$d' \triangleq \frac{\mu_1 - \mu_0}{\sigma} \tag{4.28}$$

[17] http://en.wikipedia.org/wiki/Sigmoid_function

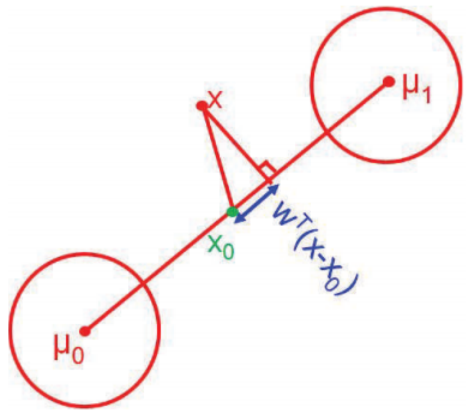


Fig. 4.4: Geometry of LDA in the 2 class case where $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{I}$.

where μ_1 is the mean of the signal and μ_0 is the mean of the noise, and σ is the standard deviation of the noise. If d' is large, the signal will be easier to discriminate from the noise.

4.2.4 MLE for discriminant analysis

The log-likelihood function is as follows:

$$p(\mathcal{D}|\boldsymbol{\theta}) = \sum_{c=1}^{C} \sum_{i:y_i=c} \log \pi_c + \sum_{c=1}^{C} \sum_{i:y_i=c} \log \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \tag{4.29}$$

The MLE for each parameter is as follows:

$$\bar{\boldsymbol{\mu}}_c = \frac{N_c}{N} \tag{4.30}$$

$$\bar{\boldsymbol{\mu}}_c = \frac{1}{N_c} \sum_{i:y_i=c} \boldsymbol{x}_i \tag{4.31}$$

$$\bar{\boldsymbol{\Sigma}}_c = \frac{1}{N_c} \sum_{i:y_i=c} (\boldsymbol{x}_i - \bar{\boldsymbol{\mu}}_c)(\boldsymbol{x}_i - \bar{\boldsymbol{\mu}}_c)^T \tag{4.32}$$

4.2.5 Strategies for preventing overfitting

The speed and simplicity of the MLE method is one of its greatest appeals. However, the MLE can badly overfit in high dimensions. In particular, the MLE for a full covariance matrix is singular if $N_c < D$. And even when $N_c > D$, the MLE can be ill-conditioned, meaning it is close to singular. There are several possible solutions to this problem:

- Use a diagonal covariance matrix for each class, which assumes the features are conditionally independent; this is equivalent to using a naive Bayes classifier (Section 3.5).
- Use a full covariance matrix, but force it to be the same for all classes,$\boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}$. This is an example of **parameter tying** or **parameter sharing**, and is equivalent to LDA (Section 4.2.2).
- Use a diagonal covariance matrix and forced it to be shared. This is called diagonal covariance LDA, and is discussed in Section TODO.

- Use a full covariance matrix, but impose a prior and then integrate it out. If we use a conjugate prior, this can be done in closed form, using the results from Section TODO; this is analogous to the Bayesian naive Bayes method in Section 3.5.1.2. See (Minka 2000f) for details.
- Fit a full or diagonal covariance matrix by MAP estimation. We discuss two different kindsof prior below.
- Project the data into a low dimensional subspace and fit the Gaussians there. See Section TODO for a way to find the best (most discriminative) linear projection.

We discuss some of these options below.

4.2.6 Regularized LDA *

4.2.7 Diagonal LDA

4.2.8 Nearest shrunken centroids classifier *

One drawback of diagonal LDA is that it depends on all of the features. In high dimensional problems, we might prefer a method that only depends on a subset of the features, for reasons of accuracy and interpretability. One approach is to use a screening method, perhaps based on mutual information, as in Section 3.5.4. We now discuss another approach to this problem known as the **nearest shrunken centroids** classifier (Hastie et al. 2009, p652).

4.3 Inference in jointly Gaussian distributions

Given a joint distribution, $p(\boldsymbol{x}_1, \boldsymbol{x}_2)$, it is useful to be able to compute marginals $p(\boldsymbol{x}_1)$ and conditionals $p(\boldsymbol{x}_1|\boldsymbol{x}_2)$. We discuss how to do this below, and then give some applications. These operations take $O(D^3)$ time in the worst case. See Section TODO for faster methods.

4.3.1 Statement of the result

Theorem 4.2. *(**Marginals and conditionals of an MVN**). Suppose $X = (\boldsymbol{x}_1, \boldsymbol{x}_2)$is jointly Gaussian with parameters*

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}, \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{pmatrix}, \boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1} = \begin{pmatrix} \boldsymbol{\Lambda}_{11} & \boldsymbol{\Lambda}_{12} \\ \boldsymbol{\Lambda}_{21} & \boldsymbol{\Lambda}_{22} \end{pmatrix}, \tag{4.33}$$

Then the marginals are given by

$$\begin{aligned} p(\boldsymbol{x}_1) &= \mathcal{N}(\boldsymbol{x}_1|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11}) \\ p(\boldsymbol{x}_2) &= \mathcal{N}(\boldsymbol{x}_2|\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22}) \end{aligned} \tag{4.34}$$

and the posterior conditional is given by

$$\boxed{\begin{aligned} p(\boldsymbol{x}_1|\boldsymbol{x}_2) &= \mathcal{N}(\boldsymbol{x}_1|\boldsymbol{\mu}_{1|2}, \boldsymbol{\Sigma}_{1|2}) \\ \boldsymbol{\mu}_{1|2} &= \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2) \\ &= \boldsymbol{\mu}_1 - \boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2) \\ &= \boldsymbol{\Sigma}_{1|2}(\boldsymbol{\Lambda}_{11}\boldsymbol{\mu}_1 - \boldsymbol{\Lambda}_{12}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2)) \\ \boldsymbol{\Sigma}_{1|2} &= \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21} = \boldsymbol{\Lambda}_{11}^{-1} \end{aligned}} \tag{4.35}$$

Equation (4.35) is of such crucial importance in this book that we have put a box around it, so you can easily find it. For the proof, see Section TODO.

We see that both the marginal and conditional distributions are themselves Gaussian. For the marginals, we just extract the rows and columns corresponding to $\boldsymbol{x}_1$ or $\boldsymbol{x}_2$. For the conditional, we have to do a bit more work. However, it is not that complicated: the conditional mean is just a linear function of $\boldsymbol{x}_2$, and the conditional covariance is just a constant matrix that is independent of $\boldsymbol{x}_2$. We give three different (but equivalent) expressions for the posterior mean, and two different (but equivalent) expressions for the posterior covariance; each one is useful in different circumstances.

4.3.2 Examples

Below we give some examples of these equations in action, which will make them seem more intuitive.

4.3.2.1 Marginals and conditionals of a 2d Gaussian

Chapter 5
Bayesian statistics

5.1 Introduction

Using the posterior distribution to summarize everything we know about a set of unknown variables is at the core of Bayesian statistics. In this chapter, we discuss this approach to statistics in more detail.

5.2 Summarizing posterior distributions

The posterior $p(\boldsymbol{\theta}|\mathcal{D})$ summarizes everything we know about the unknown quantities $\boldsymbol{\theta}$. In this section, we discuss some simple quantities that can be derived from a probability distribution, such as a posterior. These summary statistics are often easier to understand and visualize than the full joint.

5.2.1 MAP estimation

We can easily compute a **point estimate** of an unknown quantity by computing the posterior mean, median or mode. In Section 5.7, we discuss how to use decision theory to choose between these methods. Typically the posterior mean or median is the most appropriate choice for a realvalued quantity, and the vector of posterior marginals is the best choice for a discrete quantity. However, the posterior mode, aka the MAP estimate, is the most popular choice because it reduces to an optimization problem, for which efficient algorithms often exist. Futhermore, MAP estimation can be interpreted in non-Bayesian terms, by thinking of the log prior as a regularizer (see Section TODO for more details).

Although this approach is computationally appealing, it is important to point out that there are various drawbacks to MAP estimation, which we briefly discuss below. This will provide motivation for the more thoroughly Bayesian approach which we will study later in this chapter(and elsewhere in this book).

5.2.1.1 No measure of uncertainty

The most obvious drawback of MAP estimation, and indeed of any other *point estimate* such as the posterior mean or median, is that it does not provide any measure of uncertainty. In many applications, it is important to know how much one can trust a given estimate. We can derive such confidence measures from the posterior, as we discuss in Section 5.2.2.

5.2.1.2 Plugging in the MAP estimate can result in overfitting

If we dont model the uncertainty in our parameters, then our predictive distribution will be overconfident. Overconfidence in predictions is particularly problematic in situations where we may be risk averse; see Section 5.7 for details.

5.2.1.3 The mode is an untypical point

Choosing the mode as a summary of a posterior distribution is often a very poor choice, since the mode is usually quite untypical of the distribution, unlike the mean or median. The basic problem is that the mode is a point of measure zero, whereas the mean and median take the volume of the space into account. See Figure 5.1.

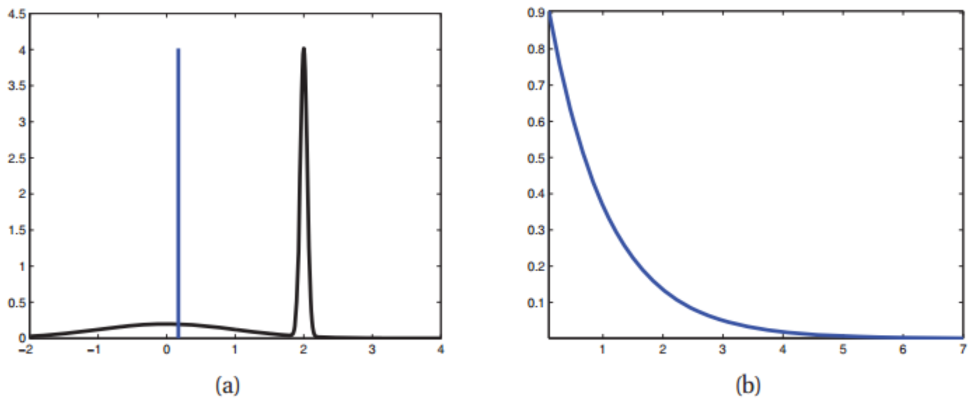


Fig. 5.1: (a) A bimodal distribution in which the mode is very untypical of the distribution. The thin blue vertical line is the mean, which is arguably a better summary of the distribution, since it is near the majority of the probability mass. (b) A skewed distribution in which the mode is quite different from the mean.

How should we summarize a posterior if the mode is not a good choice? The answer is to use decision theory, which we discuss in Section 5.7. The basic idea is to specify a loss function, where $L(\theta,\hat{\theta})$ is the loss you incur if the truth is θ and your estimate is $\hat{\theta}$. If we use 0-1 loss $L(\theta,\hat{\theta}) = \mathbb{I}(\theta \neq \hat{\theta})$(see section 1.2.2.1), then the optimal estimate is the posterior mode. 0-1 loss means you only get points if you make no errors, otherwise you get nothing: there is no partial credit under this loss function! For continuous-valued quantities, we often prefer to use squared error loss, $L(\theta,\hat{\theta}) = (\theta - \hat{\theta})^2$; the corresponding optimal estimator is then the posterior mean, as we show in Section 5.7. Or we can use a more robust loss function, $L(\theta,\hat{\theta}) = |\theta - \hat{\theta}|$, which gives rise to the posterior median.

5.2.1.4 MAP estimation is not invariant to reparameterization *

A more subtle problem with MAP estimation is that the result we get depends on how we parameterize the probability distribution. Changing from one representation to another equivalent representation changes the result, which is not very desirable, since the units of measurement are arbitrary (e.g., when measuring distance, we can use centimetres or inches).

To understand the problem, suppose we compute the posterior forx. If we define y=f(x), the distribution for yis given by Equation (2.43). The $\frac{\mathrm{dx}}{\mathrm{dy}}$ term is called the Jacobian, and it measures the change in size of a unit volume passed through f. Let $\hat{x} = \arg\max_x p_x(x)$ be the MAP estimate for x. In general it is not the case that $\hat{x} = \arg\max_x p_x(x)$ is given by $f(\hat{x})$. For example, let $X \sim \mathcal{N}(6,1)$ and $y = f(x)$, where $f(x) = 1/(1+\exp(-x+5))$.

We can derive the distribution of y using Monte Carlo simulation (see Section 2.7). The result is shown in Figure **??**. We see that the original Gaussian has become squashed by the sigmoid nonlinearity. In particular, we see that the mode of the transformed distribution is not equal to the transform of the original mode.

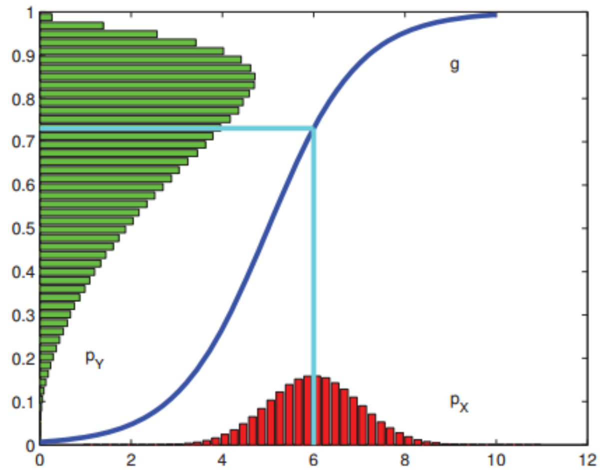


Fig. 5.2: Example of the transformation of a density under a nonlinear transform. Note how the mode of the transformed distribution is not the transform of the original mode. Based on Exercise 1.4 of (Bishop 2006b).

The MLE does not suffer from this since the likelihood is a function, not a probability density. Bayesian inference does not suffer from this problem either, since the change of measure is taken into account when integrating over the parameter space.

5.2.2 Credible intervals

In addition to point estimates, we often want a measure of confidence. A standard measure of confidence in some (scalar) quantity θ is the width of its posterior distribution. This can be measured using a $100(1\alpha)\%$ credible interval, which is a (contiguous) region $C = (\ell, u)$(standing for lower and upper) which contains 1α of the posterior probability mass, i.e.,

$$C_\alpha(\mathcal{D}) \quad \text{where } P(\ell \le \theta \le u) = 1 - \alpha \tag{5.1}$$

There may be many such intervals, so we choose one such that there is $(1\alpha)/2$ mass in each tail; this is called a **central interval**.

If the posterior has a known functional form, we can compute the posterior central interval using $\ell = F^{-1}(\alpha/2)$ and $u = F^{-1}(1-\alpha/2)$, where F is the cdf of the posterior.

If we dont know the functional form, but we can draw samples from the posterior, then we can use a Monte Carlo approximation to the posterior quantiles: we simply sort the $\mathcal{S}$ samples, and find the one that occurs at location $\alpha/\mathcal{S}$ along the sorted list. As $\mathcal{S} \to \infty$, this converges to the true quantile.

People often confuse Bayesian credible intervals with frequentist confidence intervals. However, they are not the same thing, as we discuss in Section TODO. In general, credible intervals are usually what people want to compute, but confidence intervals are usually what they actually compute, because most people are taught frequentist statistics but not Bayesian statistics. Fortunately, the mechanics of computing a credible interval is just as easy as computing a confidence interval.

5.2.3 Inference for a difference in proportions

Sometimes we have multiple parameters, and we are interested in computing the posterior distribution of some function of these parameters. For example, suppose you are about to buy something from Amazon.com, and there are two sellers offering it for the same price. Seller 1 has 90 positive reviews and 10 negative reviews. Seller 2 has 2 positive reviews and 0 negative reviews. Who should you buy from?[18].

On the face of it, you should pick seller 2, but we cannot be very confident that seller 2 is better since it has had so few reviews. In this section, we sketch a Bayesian analysis of this problem. Similar methodology can be used to compare rates or proportions across groups for a variety of other settings.

Let θ_1 and θ_2 be the unknown reliabilities of the two sellers. Since we dont know much about them, well endow them both with uniform priors, $\theta_i \sim \text{Beta}(1,1)$. The posteriors are $p(\theta_1|\mathcal{D}_1) = \text{Beta}(91,11)$ and $p(\theta_2|\mathcal{D}_2) = \text{Beta}(3,1)$.

We want to compute $p(\theta_1 > \theta_2|\mathcal{D})$. For convenience, let us define $\delta = \theta_1 - \theta_2$ as the difference in the rates. (Alternatively we might want to work in terms of the log-odds ratio.) We can compute the desired quantity using numerical integration

$$p(\delta > 0|\mathcal{D}) = \int_0^1 \int_0^1 \mathbb{I}(\theta_1 > \theta_2)\text{Beta}(\theta_1|91,11)\text{Beta}(\theta_2|3,1)\text{d}\theta_1\text{d}\theta_2 \tag{5.2}$$

We find $p(\delta > 0|\mathcal{D}) = 0.710$, which means you are better off buying from seller 1!

5.3 Bayesian model selection

In general, when faced with a set of models (i.e., families of parametric distributions) of different complexity, how should we choose the best one? This is called the **model selection** problem.

One approach is to use cross-validation to estimate the generalization error of all the candidate models, and then to pick the model that seems the best. However, this requires fitting each model K times, where K is the number of CV folds. A more efficient approach is to compute the posterior over models,

$$p(m|\mathcal{D}) = \frac{p(\mathcal{D}|m)p(m)}{\sum_{m'} p(\mathcal{D}|m')p(m')} \tag{5.3}$$

From this, we can easily compute the MAP model, $\hat{m} = \arg\max_m p(m|\mathcal{D})$. This is called **Bayesian model selection**.
If we use a uniform prior over models, this amounts to picking the model which maximizes

$$p(\mathcal{D}|m) = \int p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}|m)\text{d}\boldsymbol{\theta} \tag{5.4}$$

This quantity is called the **marginal likelihood**, the **integrated likelihood**, or the **evidence** for model m. The details on how to perform this integral will be discussed in Section 5.3.2. But first we give an intuitive interpretation of what this quantity means.

5.3.1 Bayesian Occam's razor

One might think that using $p(\mathcal{D}|m)$ to select models would always favour the model with the most parameters. This is true if we use $p(\mathcal{D}|\hat{\boldsymbol{\theta}}_m)$ to select models, where $\hat{\boldsymbol{\theta}}_m)$ is the MLE or MAP estimate of the parameters for model m, because models with more parameters will fit the data better, and hence achieve higher likelihood. However, if we integrate out the parameters, rather than maximizing them, we are automatically protected from overfitting: models with more parameters do not necessarily have higher *marginal likelihood*. This is called the **Bayesian Occams razor** effect (MacKay 1995b; Murray and Ghahramani 2005), named after the principle known as **Occams razor**, which says one should pick the simplest model that adequately explains the data.

One way to understand the Bayesian Occams razor is to notice that the marginal likelihood can be rewritten as follows, based on the chain rule of probability (Equation (2.3)):

$$p(D) = p((\boldsymbol{x}_1, y_1))p((\boldsymbol{x}_2, y_2)|(\boldsymbol{x}_1, y_1))p((\boldsymbol{x}_3, y_3)|(\boldsymbol{x}_1, y_1):(\boldsymbol{x}_2, y_2)) \cdots p((\boldsymbol{x}_N, y_N)|(\boldsymbol{x}_1, y_1):(\boldsymbol{x}_{N-1}, y_{N-1})) \tag{5.5}$$

[18] This example is from http://www.johndcook.com/blog/2011/09/27/bayesian-amazon/

This is similar to a leave-one-out cross-validation estimate (Section 1.3) of the likelihood, since we predict each future point given all the previous ones. (Of course, the order of the data does not matter in the above expression.) If a model is too complex, it will overfit the early examples and will then predict the remaining ones poorly.

Another way to understand the Bayesian Occams razor effect is to note that probabilities must sum to one. Hence $\sum_{p(\mathcal{D}')} p(m|\mathcal{D}') = 1$, where the sum is over all possible data sets. Complex models, which can predict many things, must spread their probability mass thinly, and hence will not obtain as large a probability for any given data set as simpler models. This is sometimes called the **conservation of probability mass** principle, and is illustrated in Figure 5.3.

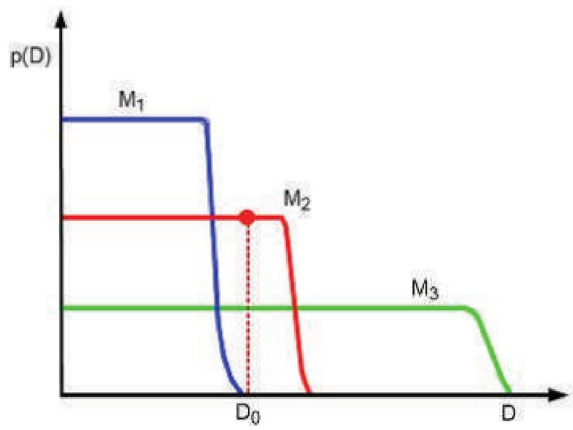


Fig. 5.3: A schematic illustration of the Bayesian Occams razor. The broad (green) curve corresponds to a complex model, the narrow (blue) curve to a simple model, and the middle (red) curve is just right. Based on Figure 3.13 of (Bishop 2006a).

When using the Bayesian approach, we are not restricted to evaluating the evidence at a finite grid of values. Instead, we can use numerical optimization to find $\lambda^* = \arg\max_\lambda p(\mathcal{D}|\lambda)$. This technique is called **empirical Bayes** or **type II maximum likelihood** (see Section 5.6 for details). An example is shown in Figure TODO(b): we see that the curve has a similar shape to the CV estimate, but it can be computed more efficiently.

5.3.2 Computing the marginal likelihood (evidence)

When discussing parameter inference for a fixed model, we often wrote

$$p(\boldsymbol{\theta}|\mathcal{D},m) \propto p(\boldsymbol{\theta}|m)p(\mathcal{D}|\boldsymbol{\theta},m) \tag{5.6}$$

thus ignoring the normalization constant $p(\mathcal{D}|m)$. This is valid since $p(\mathcal{D}|m)$is constant wrt $\boldsymbol{\theta}$. However, when comparing models, we need to know how to compute the marginal likelihood, $p(\mathcal{D}|m)$. In general, this can be quite hard, since we have to integrate over all possible parameter values, but when we have a conjugate prior, it is easy to compute, as we now show.

Let $p(\boldsymbol{\theta}) = q(\boldsymbol{\theta})/Z_0$ be our prior, where $q(\boldsymbol{\theta})$ is an unnormalized distribution, and Z_0 is the normalization constant of the prior. Let $p(\mathcal{D}|\boldsymbol{\theta}) = q(\mathcal{D}|\boldsymbol{\theta})/Z_\ell$ be the likelihood, where Z_ℓ contains any constant factors in the likelihood. Finally let $p(\boldsymbol{\theta}|\mathcal{D}) = q(\boldsymbol{\theta}|\mathcal{D})/Z_N$ be our posterior , where $q(\boldsymbol{\theta}|\mathcal{D}) = q(\mathcal{D}|\boldsymbol{\theta})q(\boldsymbol{\theta})$ is the unnormalized posterior, and Z_N is the normalization constant of the posterior. We have

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{D})} \tag{5.7}$$

$$\frac{q(\boldsymbol{\theta}|\mathcal{D})}{Z_N} = \frac{q(\mathcal{D}|\boldsymbol{\theta})q(\boldsymbol{\theta})}{Z_\ell Z_0 p(\mathcal{D})} \tag{5.8}$$

$$p(\mathcal{D}) = \frac{Z_N}{Z_0 Z_\ell} \tag{5.9}$$

So assuming the relevant normalization constants are tractable, we have an easy way to compute the marginal likelihood. We give some examples below.

5.3.2.1 Beta-binomial model

Let us apply the above result to the Beta-binomial model. Since we know $p(\boldsymbol{\theta}|\mathcal{D}) = \text{Beta}(\boldsymbol{\theta}|a', b')$, where $a' = a + N_1$, $b' = b + N_0$, we know the normalization constant of the posterior is $B(a', b')$. Hence

$$\begin{aligned}
p(\theta|\mathcal{D}) &= \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} && (5.10)\\
&= \frac{1}{p(\mathcal{D})}\left[\frac{1}{B(a,b)}\theta^{a-1}(1-\theta)^{b-1}\right]\left[\binom{N}{N_1}\theta^{N_1}(1-\theta)^{N_0}\right] && (5.11)\\
&= \binom{N}{N_1}\frac{1}{p(\mathcal{D})}\frac{1}{B(a,b)}\left[\theta^{a+N_1-1}(1-\theta)^{b+N_0-1}\right] && (5.12)
\end{aligned}$$

So

$$\begin{aligned}
\frac{1}{B(a+N_1, b+N_0)} &= \binom{N}{N_1}\frac{1}{p(\mathcal{D})}\frac{1}{B(a,b)} && (5.13)\\
p(\mathcal{D}) &= \binom{N}{N_1}\frac{B(a+N_1, b+N_0)}{B(a,b)} && (5.14)
\end{aligned}$$

The marginal likelihood for the Beta-Bernoulli model is the same as above, except it is missingthe $\binom{N}{N_1}$ term.

5.3.2.2 Dirichlet-multinoulli model

By the same reasoning as the Beta-Bernoulli case, one can show that the marginal likelihood for the Dirichlet-multinoulli model is given by

$$\begin{aligned}
p(\mathcal{D}) &= \frac{B(\boldsymbol{N}+\boldsymbol{\alpha})}{B(\boldsymbol{\alpha})} && (5.15)\\
&= \frac{\Gamma(\sum_k \alpha_k)}{\Gamma(N+\sum_k \alpha_k)}\prod_k \frac{\Gamma(N_k+\alpha_k)}{\Gamma(\alpha_k)} && (5.16)
\end{aligned}$$

5.3.2.3 Gaussian-Gaussian-Wishart model

Consider the case of an MVN with a conjugate NIW prior. Let Z_0 be the normalizer for the prior, Z_N be normalizer for the posterior, and let $Z_\ell(2\pi)^{ND/2} =$ be the normalizer for the likelihood. Then it is easy to see that

$$
\begin{aligned}
p(\mathcal{D}) &= \frac{Z_N}{Z_0 Z_\ell} && (5.17)\\
&= \frac{1}{\pi^{ND/2}} \frac{1}{2^{ND/2}} \frac{\left(\frac{2\pi}{\kappa_N}\right)^{D/2} |\boldsymbol{S}_N|^{-\nu_N/2} 2^{(\nu_0+N)D/2} \Gamma_D(\nu_N/2)}{\left(\frac{2\pi}{\kappa_0}\right)^{D/2} |\boldsymbol{S}_0|^{-\nu_0/2} 2^{\nu_0 D/2} \Gamma_D(\nu_0/2)} && (5.18)\\
& \frac{1}{\pi^{ND/2}} \left(\frac{\kappa_0}{\kappa_N}\right)^{D/2} \frac{|\boldsymbol{S}_0|^{\nu_0/2}}{|\boldsymbol{S}_N|^{\nu_N/2}} \frac{\Gamma_D(\nu_N/2)}{\Gamma_D(\nu_0/2)} && (5.19)
\end{aligned}
$$

5.3.2.4 BIC approximation to log marginal likelihood

In general, computing the integral in Equation (5.4) can be quite difficult. One simple but popular approximation is known as the **Bayesian information criterion** or **BIC**, which has the following form (Schwarz 1978):

$$
\text{BIC} \triangleq \log p(\mathcal{D}|\hat{\boldsymbol{\theta}}) - \frac{\text{dof}(\hat{\boldsymbol{\theta}})}{2} \log N \tag{5.20}
$$

where $\text{dof}(\hat{\boldsymbol{\theta}})$ is the number of **degrees of freedom** in the model, and $\hat{\boldsymbol{\theta}}$ is the MLE for the model. We see that this has the form of a **penalized log likelihood**, where the penalty term depends on the models complexity. See Section TODO for the derivation of the BIC score.

As an example, consider linear regression. As we show in Section TODO, the MLE is given by $\hat{\boldsymbol{w}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$ and $\sigma^2 = \frac{1}{N}\sum_{i=1}^N (y_i - \hat{\boldsymbol{w}}^T\boldsymbol{x}_i)$. The corresponding log likelihood is given by

$$
\log p(\mathcal{D}|\hat{\boldsymbol{\theta}}) = -\frac{N}{2}\log(2\pi\hat{\sigma}^2) - \frac{N}{2} \tag{5.21}
$$

Hence the BIC score is as follows (dropping constant terms)

$$
\text{BIC} = -\frac{N}{2}\log(\hat{\sigma}^2) - \frac{D}{2}\log N \tag{5.22}
$$

where D is the number of variables in the model. In the statistics literature, it is common to use an alternative definition of BIC, which we call the BIC *cost*(since we want to minimize it):

$$
\text{BIC-cost} \triangleq -2\log p(\mathcal{D}|\hat{\boldsymbol{\theta}}) - \text{dof}(\hat{\boldsymbol{\theta}})\log N \approx -2\log p(\mathcal{D}) \tag{5.23}
$$

In the context of linear regression, this becomes

$$
\text{BIC-cost} = N\log(\hat{\sigma}^2) + D\log N \tag{5.24}
$$

The BIC method is very closely related to the **minimum description length** or **MDL** principle, which characterizes the score for a model in terms of how well it fits the data, minus how complex the model is to define. See (Hansen and Yu 2001) for details.

There is a very similar expression to BIC/ MDL called the **Akaike information criterion** or **AIC**, defined as

$$
\text{AIC}(m,\mathcal{D}) = \log p(\mathcal{D}|\hat{\boldsymbol{\theta}}_{MLE}) - \text{dof}(m) \tag{5.25}
$$

This is derived from a frequentist framework, and cannot be interpreted as an approximation to the marginal likelihood. Nevertheless, the form of this expression is very similar to BIC. We see that the penalty for AIC is less than for BIC. This causes AIC to pick more complex models. However, this can result in better predictive accuracy. See e.g., (Clarke et al. 2009, sec 10.2) for further discussion on such information criteria.

5.3.2.5 Effect of the prior

Sometimes it is not clear how to set the prior. When we are performing posterior inference, the details of the prior may not matter too much, since the likelihood often overwhelms the prior anyway. But when computing the marginal likelihood, the prior plays a much more important role, since we are averaging the likelihood over all possible parameter settings, as weighted by the prior.

If the prior is unknown, the correct Bayesian procedure is to put a prior on the prior. If the prior is unknown, the correct Bayesian procedure is to put a prior on the prior.

5.3.3 Bayes factors

Suppose our prior on models is uniform, $p(m) \propto 1$. Then model selection is equivalent to picking the model with the highest marginal likelihood. Now suppose we just have two models we are considering, call them the **null hypothesis**, M_0, and the **alternative hypothesis**, M_1. Define the **Bayes factor** as the ratio of marginal likelihoods:

$$\mathrm{BF}_{1,0} \triangleq \frac{p(\mathcal{D}|M_1)}{p(\mathcal{D}|M_0)} = \frac{p(M_1|\mathcal{D})}{p(M_2|\mathcal{D})} / \frac{p(M_1)}{p(M_0)} \tag{5.26}$$

5.4 Priors

The most controversial aspect of Bayesian statistics is its reliance on priors. Bayesians argue this is unavoidable, since nobody is a **tabula rasa** or **blank slate**: all inference must be done conditional on certain assumptions about the world. Nevertheless, one might be interested in minimizing the impact of ones prior assumptions. We briefly discuss some ways to do this below.

5.4.1 Uninformative priors

If we dont have strong beliefs about what θ should be, it is common to use an **uninformative** or **non-informative** prior, and to let the data speak for itself.

5.4.2 Robust priors

In many cases, we are not very confident in our prior, so we want to make sure it does not have an undue influence on the result. This can be done by using **robust priors**(Insua and Ruggeri 2000), which typically have heavy tails, which avoids forcing things to be too close to the prior mean.

5.4.3 Mixtures of conjugate priors

Robust priors are useful, but can be computationally expensive to use. Conjugate priors simplify the computation, but are often not robust, and not flexible enough to encode our prior knowledge. However, it turns out that a **mixture of conjugate priors** is also conjugate, and can approximate any kind of prior (Dallal and Hall 1983; Diaconis and Ylvisaker 1985). Thus such priors provide a good compromise between computational convenience and flexibility.

5.5 Hierarchical Bayes

A key requirement for computing the posterior $p(\boldsymbol{\theta}|\mathcal{D})$ is the specification of a prior $p(\boldsymbol{\theta}|\boldsymbol{\eta})$, where $\boldsymbol{\eta}$ are the hyper-parameters. What if we dont know how to set $\boldsymbol{\eta}$? In some cases, we can use uninformative priors, we we discussed above. A more Bayesian approach is to put a prior on our priors! In terms of graphical models (Chapter TODO), we can represent the situation as follows:

$$\boldsymbol{\eta} \rightarrow \boldsymbol{\theta} \rightarrow \mathcal{D} \tag{5.27}$$

This is an example of a **hierarchical Bayesian model**, also called a **multi-level** model, since there are multiple levels of unknown quantities.

5.6 Empirical Bayes

Method	**Definition**
Maximum likelihood	$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} p(\mathcal{D}\|\boldsymbol{\theta})$
MAP estimation	$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} p(\mathcal{D}\|\boldsymbol{\theta})p(\boldsymbol{\theta}\|\boldsymbol{\eta})$
ML-II (**Empirical Bayes**)	$\hat{\boldsymbol{\eta}} = \arg\max_{\boldsymbol{\eta}} \int p(\mathcal{D}\|\boldsymbol{\theta})p(\boldsymbol{\theta}\|\boldsymbol{\eta})\mathrm{d}\boldsymbol{\theta} = \arg\max_{\boldsymbol{\eta}} p(\mathcal{D}\|\boldsymbol{\eta})$
MAP-II	$\hat{\boldsymbol{\eta}} = \arg\max_{\boldsymbol{\eta}} \int p(\mathcal{D}\|\boldsymbol{\theta})p(\boldsymbol{\theta}\|\boldsymbol{\eta})p(\boldsymbol{\eta})\mathrm{d}\boldsymbol{\theta} = \arg\max_{\boldsymbol{\eta}} p(\mathcal{D}\|\boldsymbol{\eta})p(\boldsymbol{\eta})$
Full Bayes	$p(\boldsymbol{\theta},\boldsymbol{\eta}\|\mathcal{D}) \propto p(\mathcal{D}\|\boldsymbol{\theta})p(\boldsymbol{\theta}\|\boldsymbol{\eta})$

5.7 Bayesian decision theory

We have seen how probability theory can be used to represent and updates our beliefs about the state of the world. However, ultimately our goal is to convert our beliefs into actions. In this section, we discuss the optimal way to do this.

Our goal is to devise a **decision procedure** or **policy**, $f(\boldsymbol{x}) : \mathcal{X} \rightarrow \mathcal{Y}$, which minimizes the **expected loss** $R_{\exp}(f)$(see Equation (1.1)).

In the Bayesian approach to decision theory, the optimal output, having observed $\boldsymbol{x}$, is defined as the output a that minimizes the **posterior expected loss**:

$$\rho(f) = \mathbb{E}_{p(y|\boldsymbol{x})}[L(y, f(\boldsymbol{x}))] = \begin{cases} \sum\limits_{y} L[y, f(\boldsymbol{x})]p(y|\boldsymbol{x}) & , y \text{ is discrete} \\ \int\limits_{y} L[y, f(\boldsymbol{x})]p(y|\boldsymbol{x})\mathrm{d}y & , y \text{ is continuous} \end{cases} \tag{5.28}$$

Hence the **Bayes estimator**, also called the **Bayes decision rule**, is given by

$$\delta(\boldsymbol{x}) = \arg\min_{f \in \mathcal{H}} \rho(f) \tag{5.29}$$

5.7.1 Bayes estimators for common loss functions

5.7.1.1 MAP estimate minimizes 0-1 loss

When $L(y, f(x))$ is **0-1 loss**(Section 1.2.2.1), we can proof that MAP estimate minimizes 0-1 loss,

$$
\begin{aligned}
\arg\min_{f\in\mathcal{H}} \rho(f) &= \arg\min_{f\in\mathcal{H}} \sum_{i=1}^{K} L[C_k, f(\boldsymbol{x})] p(C_k|\boldsymbol{x}) \\
&= \arg\min_{f\in\mathcal{H}} \sum_{i=1}^{K} \mathbb{I}(f(\boldsymbol{x}) \neq C_k) p(C_k|\boldsymbol{x}) \\
&= \arg\min_{f\in\mathcal{H}} \sum_{i=1}^{K} p(f(\boldsymbol{x}) \neq C_k|\boldsymbol{x}) \\
&= \arg\min_{f\in\mathcal{H}} [1 - p(f(\boldsymbol{x}) = C_k|\boldsymbol{x})] \\
&= \arg\max_{f\in\mathcal{H}} p(f(\boldsymbol{x}) = C_k|\boldsymbol{x})
\end{aligned}
$$

5.7.1.2 Posterior mean minimizes ℓ_2(quadratic) loss

For continuous parameters, a more appropriate loss function is **squared error**, ℓ_2 **loss**, or **quadratic loss**, defined as $L(y, f(\boldsymbol{x})) = [y - f(\boldsymbol{x})]^2$.

The posterior expected loss is given by

$$
\rho(f) = \int\limits_y L[y, f(\boldsymbol{x})] p(y|\boldsymbol{x}) \mathrm{d}y = \int\limits_y [y - f(\boldsymbol{x})]^2 p(y|\boldsymbol{x}) \mathrm{d}y = \int\limits_y \left[y^2 - 2yf(\boldsymbol{x}) + f(\boldsymbol{x})^2\right] p(y|\boldsymbol{x}) \mathrm{d}y \tag{5.30}
$$

Hence the optimal estimate is the posterior mean:

$$
\begin{aligned}
&\frac{\partial \rho}{\partial f} = \int\limits_y [-2y + 2f(\boldsymbol{x})] p(y|\boldsymbol{x}) \mathrm{d}y = 0 \Rightarrow \\
&\int\limits_y f(\boldsymbol{x}) p(y|\boldsymbol{x}) \mathrm{d}y = \int\limits_y y p(y|\boldsymbol{x}) \mathrm{d}y \\
&f(\boldsymbol{x}) \int\limits_y p(y|\boldsymbol{x}) \mathrm{d}y = \mathbb{E}_{p(y|\boldsymbol{x})}[y] \\
&f(\boldsymbol{x}) = \mathbb{E}_{p(y|\boldsymbol{x})}[y] \qquad (5.31)
\end{aligned}
$$

This is often called the **minimum mean squared error** estimate or **MMSE** estimate.

5.7.1.3 Posterior median minimizes ℓ_1(absolute) loss

The ℓ_2 loss penalizes deviations from the truth quadratically, and thus is sensitive to outliers. A more robust alternative is the absolute or ℓ_1 loss. The optimal estimate is the posterior median, i.e., a value a such that $P(y < a|\boldsymbol{x}) = P(y \geq a|\boldsymbol{x}) = 0.5$.

Proof. TODO

5.7.1.4 Reject option

In classification problems where $p(y|\boldsymbol{x})$ is very uncertain, we may prefer to choose a reject action, in which we refuse to classify the example as any of the specified classes, and instead say dont know. Such ambiguous cases can be handled by e.g., a human expert. This is useful in **risk averse** domains such as medicine and finance.

We can formalize the reject option as follows. Let choosing $f(\boldsymbol{x}) = c_{K+1}$ correspond to picking the reject action, and choosing $f(\boldsymbol{x}) \in \{C_1, ..., C_k\}$ correspond to picking one of the classes. Suppose we define the loss function as

$$L(f(\boldsymbol{x}),y) = \begin{cases} 0 & \text{if } f(\boldsymbol{x}) = y \text{ and } f(\boldsymbol{x}), y \in \{C_1, ..., C_k\} \\ \lambda_s & \text{if } f(\boldsymbol{x}) \neq y \text{ and } f(\boldsymbol{x}), y \in \{C_1, ..., C_k\} \\ \lambda_r & \text{if } f(\boldsymbol{x}) = C_{K+1} \end{cases} \tag{5.32}$$

where λ_s is the cost of a substitution error, and λ_r is the cost of the reject action.

5.7.1.5 Supervised learning

We can define the loss incurred by $f(\boldsymbol{x})$ (i.e., using this predictor) when the unknown state of nature is $\boldsymbol{\theta}$(the parameters of the data generating mechanism) as follows:

$$L(\boldsymbol{\theta}, f) \triangleq \mathbb{E}_{p(\boldsymbol{x},y|\boldsymbol{\theta})}[\ell(y - f(\boldsymbol{x}))] \tag{5.33}$$

This is known as the **generalization error**. Our goal is to minimize the posterior expected loss, given by

$$\rho(f|\mathcal{D}) = \int p(\boldsymbol{\theta}|\mathcal{D})L(\boldsymbol{\theta}, f)\mathrm{d}\boldsymbol{\theta} \tag{5.34}$$

This should be contrasted with the frequentist risk which is defined in Equation TODO.

5.7.2 The false positive vs false negative tradeoff

In this section, we focus on binary decision problems, such as hypothesis testing, two-class classification, object/ event detection, etc. There are two types of error we can make: a **false positive**(aka **false alarm**), or a **false negative**(aka **missed detection**). The 0-1 loss treats these two kinds of errors equivalently. However, we can consider the following more general loss matrix:

TODO

Chapter 6
Perceptron

6.1 Representation

$$\mathcal{H} : y = f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}^T \boldsymbol{x} + b) \tag{6.1}$$

where $\text{sign}(x) = \begin{cases} +1, & x \geq 0 \\ -1, & x < 0 \end{cases}$, see Fig. 6.1[19].

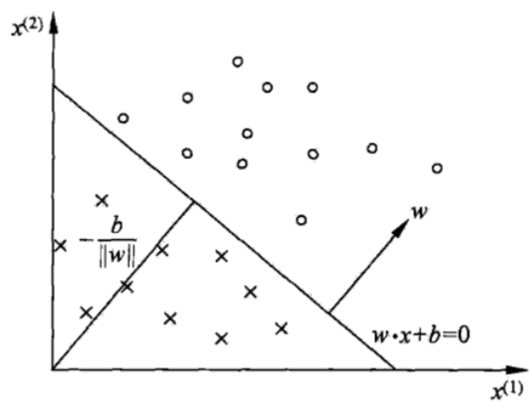


Fig. 6.1: Perceptron

6.2 Evaluation

$$L(\boldsymbol{w}, b) = -y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \tag{6.2}$$
$$R_{emp}(f) = -\sum_i y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \tag{6.3}$$
$$\tag{6.4}$$

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

6.3 Optimization

6.3.1 Primal form

Stochastic gradient descent, the pseudo code is as follows:

```
w ← 0; b ← 0; k ← 0;
while no mistakes made within the for loop do
    for i ← 1 to N do
        if y_i(w · x_i + b) ≤ 0 then
            w ← w + η y_i x_i;
            b ← b + η y_i;
            k ← k + 1;
        end
    end
end
```

Algorithm 1: Perceptron learning algorithm, primal form

6.3.1.1 Convergency

Theorem 6.1. *(**Novikoff**) If traning data set $\mathcal{D}$ is linearly separable, then*

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

$$\exists \gamma > 0,\ \forall i,\ y_i(\boldsymbol{w}_{opt} \cdot \boldsymbol{x}_i + b_{opt}) \geq \gamma \tag{6.5}$$

2.

$$k \leq \left(\frac{R}{\gamma}\right)^2,\ \textit{where}\ R = \max_{1 \leq i \leq N} ||\widehat{\boldsymbol{x}}_i|| \tag{6.6}$$

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

$$y_i(\widehat{\boldsymbol{w}}_{k-1} \cdot \widehat{\boldsymbol{x}}_i) = y_i(\boldsymbol{w}_{k-1} \cdot \boldsymbol{x}_i + b_{k-1}) \leq 0 \tag{6.7}$$

$$\widehat{\boldsymbol{w}}_k = \widehat{\boldsymbol{w}}_{k-1} + \eta y_i \widehat{\boldsymbol{x}}_i \tag{6.8}$$

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

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

6.3.2 Dual form

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i \tag{6.9}$$

$$b = \sum_{i=1}^{N} \alpha_i y_i \tag{6.10}$$

$$f(\boldsymbol{x}) = \text{sign}\left(\sum_{j=1}^{N} \alpha_j y_j \boldsymbol{x}_j \cdot \boldsymbol{x} + b\right) \tag{6.11}$$

$\boldsymbol{\alpha} \leftarrow 0$; $b \leftarrow 0$; $k \leftarrow 0$;
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 7
K-Nearest Neighbors

7.1 Representation

$$y = f(\boldsymbol{x}) = \arg\min_{c} \sum_{\boldsymbol{x}_i \in N_k(\boldsymbol{x})} \mathbb{I}(y_i = c) \tag{7.1}$$

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.

7.2 Evaluation

No training is needed.

7.3 Optimization

No training is needed.

Chapter 8
K-Means Clustering

8.1 Representation

$$y_j = k \text{ if } \|\boldsymbol{x}_j - \boldsymbol{\mu}_k\|_2^2 \text{ is minimal} \tag{8.1}$$

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

8.2 Evaluation

$$\arg\min_{\boldsymbol{\mu}} \sum_{j=1}^{N} \sum_{k=1}^{K} \gamma_{jk} \|\boldsymbol{x}_j - \boldsymbol{\mu}_k\|_2^2 \tag{8.2}$$

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

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

8.3 Optimization

E-Step:

$$\gamma_{jk}^{(i+1)} = \begin{cases} 1, & \text{if } \|\boldsymbol{x}_j - \boldsymbol{\mu}_k^{(i)}\|_2 \text{ is minimal for } \boldsymbol{\mu}_k^{(i)} \\ 0, & \text{otherwise} \end{cases} \tag{8.3}$$

M-Step:

$$\boldsymbol{\mu}_k^{(i+1)} = \frac{\sum_{j=1}^{N} \gamma_{jk}^{(i+1)} \boldsymbol{x}_j}{\sum \gamma_{jk}^{(i+1)}} \tag{8.4}$$

8.4 Tricks

8.4.1 Choosing k

8.4.2 Choosing the initial centroids(seeds)

8.4.2.1 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[20].

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.

8.5 Reference

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

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

Chapter 9
Decision Tree

Chapter 10
Linear Regression

10.1 Representation

$$y = f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b \quad ,\boldsymbol{x} \in \mathbb{R}^D, y \in \mathbb{R} \tag{10.1}$$

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

$$y = f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} \tag{10.2}$$

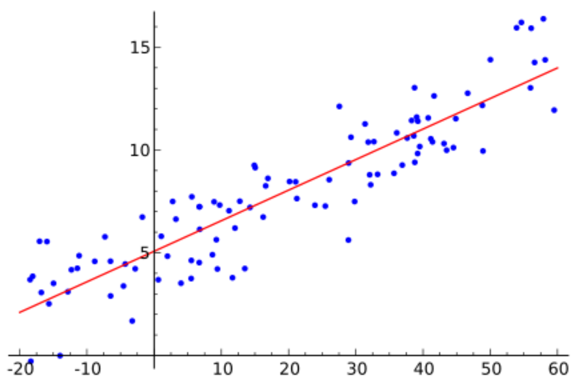


Fig. 10.1: Example of simple linear regression, which has one independent variable, from Wikipedia

10.2 Evaluation

$$R_{emp}(f) = \frac{1}{2N}\sum_{i=1}^{N}(y - f(\boldsymbol{x}))^2 = \frac{1}{2N}\sum_{i=1}^{N}\left(y - \boldsymbol{w}^T\boldsymbol{x}\right)^2 \tag{10.3}$$

$$J(\boldsymbol{w}) \triangleq R_{emp}(f) \tag{10.4}$$

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

10.3 Optimization

10.3.1 Normal Equation

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

$$\boldsymbol{w} = (X^T X)^{-1} X^T \boldsymbol{y} \tag{10.5}$$

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).

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

$$\frac{\partial}{\partial A} AB = B^T \tag{10.6}$$

$$\frac{\partial}{\partial A^T} f(A) = \left[\frac{\partial}{\partial A} f(A)\right]^T \tag{10.7}$$

$$\frac{\partial}{\partial A} ABA^T C = CAB + C^T AB^T \tag{10.8}$$

$$\frac{\partial}{\partial A} |A| = |A|(A^{-1})^T \tag{10.9}$$

Then,

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

Combining Equations (10.7) and (10.8), 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,

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

10.3.2 SGD

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

$$
\because \frac{\partial}{\partial \boldsymbol{w}}J(\boldsymbol{w}) = [y_i - f(\boldsymbol{x}_i)]\,\boldsymbol{x}_i \tag{10.10}
$$

$$
\begin{aligned}
\therefore \boldsymbol{w} =& \boldsymbol{w} - \alpha\frac{\partial}{\partial \boldsymbol{w}}f(\boldsymbol{w}) \\
=& \boldsymbol{w} - \alpha\,[y_i - f(\boldsymbol{x}_i)]\,\boldsymbol{x}_i
\end{aligned} \tag{10.11}
$$

Chapter 11
Logistic Regression

11.1 Binomial Logistic Regression Model

11.1.1 Representation

$$
\begin{aligned}
P(Y=1|\boldsymbol{x}) &= \frac{\exp(\boldsymbol{w}^T\boldsymbol{x})}{1+\exp(\boldsymbol{w}^T\boldsymbol{x})} && (11.1)\\
P(Y=0|\boldsymbol{x}) &= \frac{1}{1+\exp(\boldsymbol{w}^T\boldsymbol{x})} && (11.2)
\end{aligned}
$$

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

11.1.2 Evaluation

$$\max_{\boldsymbol{w}} \ell(\boldsymbol{w}) \tag{11.3}$$

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

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

11.1.3 Optimization

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

11.1.3.1 SGA

$$\frac{\partial}{\partial \boldsymbol{w}}\ell(\boldsymbol{w}) = [y_i - \boldsymbol{\pi}(\boldsymbol{x}_i)]\,\boldsymbol{x}_i \tag{11.4}$$

$$\begin{aligned}\boldsymbol{w} &= \boldsymbol{w} + \alpha\frac{\partial}{\partial \boldsymbol{w}}\ell(\boldsymbol{w}) \\ &= \boldsymbol{w} + \alpha\,[y_i - \boldsymbol{\pi}(\boldsymbol{x}_i)]\,\boldsymbol{x}_i \end{aligned} \tag{11.5}$$

Chapter 12
Support Vector Machines

12.1 Primal form

12.1.1 Representation

$$\mathcal{H}: y = f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}\boldsymbol{x} + b) \tag{12.1}$$

12.1.2 Evaluation

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{12.2}$$

$$s.t. \quad y_i(\boldsymbol{w}\boldsymbol{x}_i + b) \geqslant 1, i = 1, 2, \ldots, N \tag{12.3}$$

12.2 Dual form

12.2.1 Representation

$$\mathcal{H}: y = f(\boldsymbol{x}) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i y_i (\boldsymbol{x} \cdot \boldsymbol{x}_i) + b\right) \tag{12.4}$$

12.2.2 Evaluation

$$\min_{\alpha} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j) - \sum_{i=1}^{N} \alpha_i \tag{12.5}$$

$$s.t. \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{12.6}$$

$$\alpha_i \geqslant 0, i = 1, 2, \ldots, N \tag{12.7}$$

12.3 Primal form with regularization

12.3.1 Representation

$$\mathcal{H} : y = f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}\boldsymbol{x} + b) \tag{12.8}$$

12.3.2 Evaluation

$$\begin{aligned}
\min_{\boldsymbol{w},b} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \xi_i && (12.9)\\
s.t. \quad & y_i(\boldsymbol{w}\boldsymbol{x}_i + b) \geqslant 1 - \xi_i, i = 1, 2, \dots, N && (12.10)\\
& \xi_i \geqslant 0, i = 1, 2, \dots, N && (12.11)
\end{aligned}$$

12.4 Dual form with regularization

12.4.1 Representation

$$\mathcal{H} : y = f(\boldsymbol{x}) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i y_i (\boldsymbol{x} \cdot \boldsymbol{x}_i) + b\right) \tag{12.12}$$

12.4.2 Evaluation

$$\begin{aligned}
\min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j) - \sum_{i=1}^{N} \alpha_i && (12.13)\\
s.t. \quad & \sum_{i=1}^{N} \alpha_i y_i = 0 && (12.14)\\
& 0 \leqslant \alpha_i \leqslant C, i = 1, 2, \dots, N && (12.15)
\end{aligned}$$

$$\begin{aligned}
\alpha_i = 0 &\Rightarrow y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geqslant 1 && (12.16)\\
\alpha_i = C &\Rightarrow y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \leqslant 1 && (12.17)\\
0 < \alpha_i < C &\Rightarrow y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) = 1 && (12.18)
\end{aligned}$$

12.5 Hinge Loss

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

$$\min_{\boldsymbol{w},b} \sum_{i=1}^{N} [1 - y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b)]_+ + \lambda \|\boldsymbol{w}\|^2 \tag{12.19}$$

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

$$[z]_+ = \begin{cases} z, & z > 0 \\ 0, & z \leqslant 0 \end{cases} \tag{12.20}$$

Proof. We can write equation (12.19) as equations (12.9) $\sim$ (12.11).

Define

$$\xi_i \triangleq 1 - y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b), \xi_i \geqslant 0 \tag{12.21}$$

Then $\boldsymbol{w}, b, \xi_i$ satisfy the constraints (12.9) and (12.10). And objective function (12.11) can be written as

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

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

$$\min_{\boldsymbol{w},b} \frac{1}{C} \left(\frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_{i=1}^{N} \xi_i \right) \tag{12.22}$$

It is equivalent to equation (12.9).

12.6 Kernels

12.7 Optimization

SMO, QP, SGD, etc.

Chapter 13
AdaBoost

13.1 Representation

$$y = \text{sign}(f(\boldsymbol{x})) = \text{sign}\left(\sum_{i=1}^{m} \alpha_m G_m(\boldsymbol{x})\right) \tag{13.1}$$

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

13.2 Evaluation

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

$$(\alpha_m, G_m(x)) = \arg\min_{\alpha,G} \sum_{i=1}^{N} \exp\left[-y_i(f_{m-1}(\boldsymbol{x}_i) + \alpha G(\boldsymbol{x}_i))\right] \tag{13.2}$$

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

$$(\alpha_m, G_m(x)) = \arg\min_{\alpha,G} \sum_{i=1}^{N} \bar{w}_{mi} \exp\left(-y_i \alpha G(x_i)\right) \tag{13.3}$$

13.3 Optimization

13.3.1 Input

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

13.3.2 Output

Final classifier: $G(x)$

13.3.3 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

$$e_m = P(G_m(\boldsymbol{x}_i) \neq y_i) = \sum_{i=1}^{N} w_{mi} \mathbb{I}(G_m(\boldsymbol{x}_i) \neq y_i) \tag{13.4}$$

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

$$\alpha_m = \frac{1}{2} \log \frac{1 - e_m}{e_m} \tag{13.5}$$

   (d) Update the weights' distribution of training data

$$w_{m+1,i} = \frac{w_{mi}}{Z_m} \exp(-\alpha_m y_i G_m(\boldsymbol{x}_i)) \tag{13.6}$$

   where Z_m is the normalizing constant

$$Z_m = \sum_{i=1}^{N} w_{mi} \exp(-\alpha_m y_i G_m(\boldsymbol{x}_i)) \tag{13.7}$$

3. Ensemble M weak classifiers

$$G(x) = \operatorname{sign} f(\boldsymbol{x}) = \operatorname{sign} \left[\sum_{m=1}^{M} \alpha_m G_m(\boldsymbol{x}) \right] \tag{13.8}$$

13.4 The upper bound of the training error of AdaBoost

Theorem 13.1. *The upper bound of the training error of AdaBoost is*

$$\frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(G(\boldsymbol{x}_i) \neq y_i) \leq \frac{1}{N} \sum_{i=1}^{N} \exp(-y_i f(\boldsymbol{x}_i)) = \prod_{m=1}^{M} Z_m \tag{13.9}$$

Note: the following equation would help proof this theorem

$$w_{mi} \exp(-\alpha_m y_i G_m(\boldsymbol{x}_i)) = Z_m w_{m+1,i} \tag{13.10}$$

Chapter 14
EM algorithm

14.1 Jensen's inequality

14.1.1 Convex function

Definition 14.1. 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 $\lambda \in [0,1]$,

$$f(\lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2) \leq \lambda f(\boldsymbol{x}_1) + (1-\lambda) f(\boldsymbol{x}_2) \tag{14.1}$$

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

$$f(\lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2) < \lambda f(\boldsymbol{x}_1) + (1-\lambda) f(\boldsymbol{x}_2) \tag{14.2}$$

Definition 14.2. A function f is said to be (strictly) **concave** if $-f$ is (strictly) convex.

Theorem 14.1. *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 14.1. $\log(x)$ *is strictly convex on* $(0,\infty)$.

14.1.2 Jensen's inequality

Theorem 14.2. *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 $\sum\limits_{i=1}^{n} \lambda_i = 1$,*

$$f\left(\sum_{i=1}^{n} \lambda_i \boldsymbol{x}_i\right) \leq \sum_{i=1}^{n} \lambda_i f(\boldsymbol{x}_i) \tag{14.3}$$

Proposition 14.2.

$$\log\left(\sum_{i=1}^{n} \lambda_i \boldsymbol{x}_i\right) \geq \sum_{i=1}^{n} \lambda_i \log(\boldsymbol{x}_i) \tag{14.4}$$

14.2 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 $\ell(\mathcal{X}, \boldsymbol{z}|\boldsymbol{\theta})$
$\boldsymbol{\theta}^{(0)}$ = ... // initialize
while *(!convergency)* **do**
    // 2. E-step: plug in $P(\mathcal{X}, \boldsymbol{z}|\boldsymbol{\theta})$, derive the formula of $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$
    $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)}) = \mathbb{E}_{\boldsymbol{z}|\mathcal{X}, \boldsymbol{\theta}^{(i)}}\left[\log P(\mathcal{X}, \boldsymbol{z}|\boldsymbol{\theta})\right]$
    // 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

14.3 Derivation of the EM algorithm

The log likelihood function is given by

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

$$
\begin{aligned}
\boldsymbol{\theta}^{(i+1)} &= \arg\max_{\boldsymbol{\theta}} \left[\ell(\boldsymbol{\theta}^{(i)}) + B(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})\right] \\
&= \arg\max_{\boldsymbol{\theta}} \left\{\ell(\boldsymbol{\theta}^{(i)}) + \sum_{\boldsymbol{z}} P(\boldsymbol{z}|\mathcal{X}, \boldsymbol{\theta}^{(i)}) \log\left[\frac{P(\mathcal{X}|\boldsymbol{z}, \boldsymbol{\theta})P(\boldsymbol{z}|\boldsymbol{\theta})}{P(\boldsymbol{z}|\mathcal{X}, \boldsymbol{\theta}^{(i)})P(\mathcal{X}|\boldsymbol{\theta}^{(i)})}\right]\right\} \\
&\quad \text{Now drop terms which are constant w.r.t. } \boldsymbol{\theta} \\
&= \arg\max_{\boldsymbol{\theta}} \left\{\sum_{\boldsymbol{z}} P(\boldsymbol{z}|\mathcal{X}, \boldsymbol{\theta}^{(i)}) \log\left[P(\mathcal{X}|\boldsymbol{z}, \boldsymbol{\theta})P(\boldsymbol{z}|\boldsymbol{\theta})\right]\right\} \\
&= \arg\max_{\boldsymbol{\theta}} \left\{\sum_{\boldsymbol{z}} P(\boldsymbol{z}|\mathcal{X}, \boldsymbol{\theta}^{(i)}) \log\left[P(\mathcal{X}, \boldsymbol{z}|\boldsymbol{\theta})\right]\right\} \\
&= \arg\max_{\boldsymbol{\theta}} \left\{\mathbb{E}_{\boldsymbol{z}|\mathcal{X}, \boldsymbol{\theta}^{(i)}} \log\left[P(\mathcal{X}, \boldsymbol{z}|\boldsymbol{\theta})\right]\right\} \qquad (14.5) \\
&\triangleq \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)}) \qquad (14.6)
\end{aligned}
$$

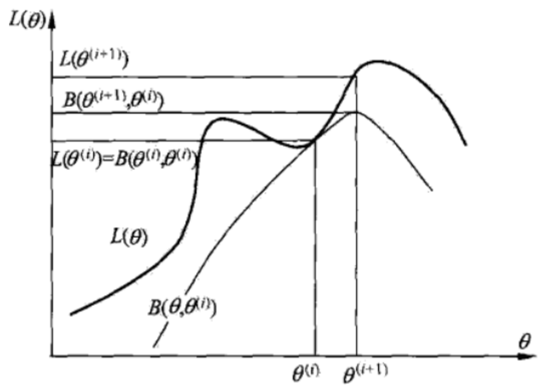


Fig. 14.1: Graphical interpretation of a single iteration of the EM algorithm: The function $B(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$ is bounded above by the log likelihood function $\ell(\boldsymbol{\theta})$. The functions are equal at $\boldsymbol{\theta} = \boldsymbol{\theta}^{(i)}$. The EM algorithm chooses $\boldsymbol{\theta}^{(i)}$ as the value of $\boldsymbol{\theta}$ for which $B(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$ is a maximum. Since $\ell(\boldsymbol{\theta}) \geq B(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$ increasing $B(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$ ensures that the value of the log likelihood function $\ell(\boldsymbol{\theta})$ is increased at each step.

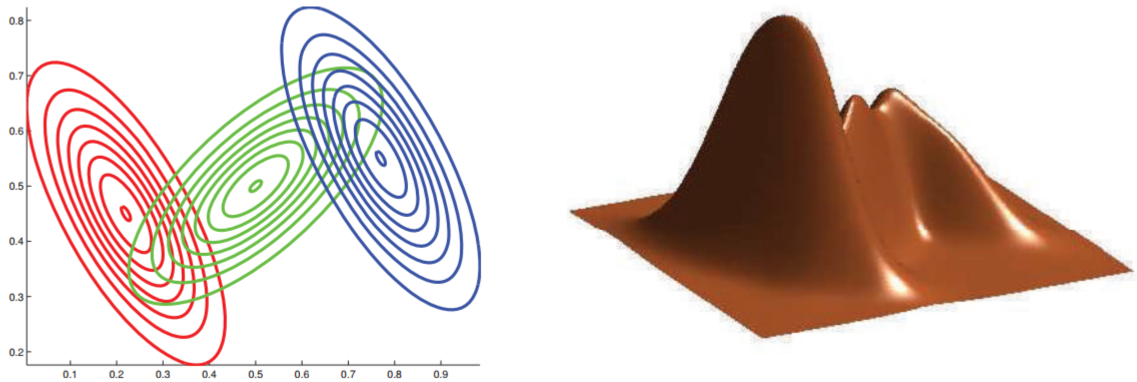
14.4 Examples

14.4.1 Gaussian mixture model

Definition 14.3. 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

$$
P(x_i|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \phi(x_i|\mu_k, \sigma_k) \qquad (14.7)
$$

Figure 14.2 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 $\mathbb{R}^D$.



(a) (b)

Fig. 14.2: 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.

14.4.1.1 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:

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

$$\Rightarrow$$

$$\log P(\mathcal{X},\gamma|\boldsymbol{\theta}) = \sum_{k=1}^{K}\left\{n_k\log\pi_k+\sum_{j=1}^{N}\gamma_{jk}\left[\log(\frac{1}{\sqrt{2\pi}}-\log\sigma_k-\frac{(x_j-\mu_k)^2}{2\sigma_k^2})\right]\right\} \tag{14.8}$$

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

$$
\begin{aligned}
Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(i)}) &= \mathbb{E}_{\boldsymbol{z}|\mathcal{X},\theta^{(i)}} \log\left[P(\mathcal{X},\boldsymbol{z}|\boldsymbol{\theta})\right] \\
&= \mathbb{E}_{\boldsymbol{\gamma}|\mathcal{X},\theta^{(i)}} \left\{ \sum_{k=1}^{K} \left\{ n_k \log \pi_k + \sum_{j=1}^{N} \gamma_{jk} \left[\log(\frac{1}{\sqrt{2\pi}} - \log\sigma_k - \frac{(x_j-\mu_k)^2}{2\sigma_k^2}) \right] \right\} \right\} \\
&= \sum_{k=1}^{K} \left\{ \left(\sum_{j=1}^{N} E\gamma_{jk} \right) \log \pi_k + \sum_{j=1}^{N} E\gamma_{jk} \left[\log(\frac{1}{\sqrt{2\pi}} - \log\sigma_k - \frac{(x_j-\mu_k)^2}{2\sigma_k^2}) \right] \right\} \\
&\quad \text{denote } E\gamma_{jk} \text{ as } \hat{\gamma}_{jk} \\
&= \sum_{k=1}^{K} \left\{ \left(\sum_{j=1}^{N} \hat{\gamma}_{jk} \right) \log \pi_k + \sum_{j=1}^{N} \hat{\gamma}_{jk} \left[\log(\frac{1}{\sqrt{2\pi}} - \log\sigma_k - \frac{(x_j-\mu_k)^2}{2\sigma_k^2}) \right] \right\}
\end{aligned} \tag{14.9}
$$

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

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

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

Take partial derivatives of Eqn.(14.9) with respect to μ_k, σ_k^2 and let them equal to 0, we can get μ_k, σ_k^2.

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

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

$$
\hat{\mu}_k = \frac{\sum_{j=1}^{N} \hat{\gamma}_{jk} x_j}{\hat{\gamma}_{jk}} \tag{14.10}
$$

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

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

$$\hat{\sigma}_k^2 = \frac{\sum\limits_{j=1}^{N} \hat{\gamma}_{jk} (x_j - \mu_k)^2}{\hat{\gamma}_{jk}} \tag{14.11}$$

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

$$\hat{\boldsymbol{\pi}}_k = \frac{\sum\limits_{k=1}^{K} \hat{\gamma}_{jk}}{N} \tag{14.12}$$

14.4.1.4 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
  $\hat{\gamma}_{jk} = \frac{\pi_k \phi(x_i|\mu_k, \sigma_k)}{\sum\limits_{k=1}^{K} \pi_k \phi(x_i|\mu_k, \sigma_k)}$
  // 3. M-step
  $\hat{\mu}_k = \frac{\sum\limits_{j=1}^{N} \hat{\gamma}_{jk} x_j}{\hat{\gamma}_{jk}}$
  $\hat{\sigma}_k^2 = \frac{\sum\limits_{j=1}^{N} \hat{\gamma}_{jk} (x_j - \mu_k)^2}{\hat{\gamma}_{jk}}$
  $\hat{\boldsymbol{\pi}}_k = \frac{\sum\limits_{k=1}^{K} \hat{\gamma}_{jk}}{N}$
end

Algorithm 4: EM algorithm for GMM

14.5 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).

14.5.1 F function's maximization-maximization algorithm

Definition 14.4. Given the probability distribution of the hidden variable Z is $\tilde{P}(Z)$, define **F function** as the following:

$$F(\tilde{P},\boldsymbol{\theta}) = \mathbb{E}_{\tilde{P}}[\log P(X,Z|\boldsymbol{\theta})] + H(\tilde{P}) \tag{14.13}$$

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

$$\tilde{P}_\theta(Z) = P(Z|X,\boldsymbol{\theta}) \tag{14.14}$$

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

$$\mathcal{L}(\tilde{P},\boldsymbol{\theta}) = \mathbb{E}_{\tilde{P}}[\log P(X,Z|\boldsymbol{\theta})] - \mathbb{E}_{\tilde{P}} \log \tilde{P}_\theta(Z) + \lambda \left[1 - \sum_Z \tilde{P}(Z)\right] \tag{14.15}$$

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

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

Let it equal to 0, we can get

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

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

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

Lemma 14.2. *If $\tilde{P}_\theta(Z) = P(Z|X,\boldsymbol{\theta})$, then*

$$F(\tilde{P},\boldsymbol{\theta}) = \log P(X|\boldsymbol{\theta}) \tag{14.16}$$

Theorem 14.3. *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 14.1, 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

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

14.5.2 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 $\ell(\boldsymbol{\theta})$, 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.

14.6 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 15

Hidden markov Model

Chapter 16
BayesNets

16.1 Chain rule

$$p(x) = p(x_1) \prod_{v=2}^{V} p(x_v | x_{1:v-1}) \tag{16.1}$$

16.2 Markov chain

$$p(x) = p(x_1) \prod_{v=2}^{V} p(x_v | x_{v-1}) \tag{16.2}$$

16.3 DGM

$$p(x|G) = \prod_{v=1}^{V} p(x_v | x_{pa(v)}) \tag{16.3}$$

16.4 Inference

$$p(x_h | x_v, \theta) = \frac{p(x_h, x_v | \theta)}{\sum_{x'_h} p(x'_h, x_v | \theta)} \tag{16.4}$$

$$p(x_q | x_v, \theta) = \sum_{x_n} p(x_q, x_n | x_v, \theta) \tag{16.5}$$

16.5 Learning

16.5.1 MAP

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, p(\theta) \prod_{n=1}^{N} p(x_n | \theta) \tag{16.6}$$

16.5.2 Learning from complete data

$$p(D|\theta) = \prod_{n=1}^{N} p(x_n|\theta) = \prod_{n=1}^{N}\prod_{v=1}^{V} p(x_{nv}|x_{n,pa(v)}, \theta_v) = \prod_{v=1}^{V} p(D_v|\theta_v) \tag{16.7}$$

16.5.3 Multinoulli Learning

Multinoulli Distribution

$$Cat(x|\mu) = \prod_{k=1}^{K} \mu_k^{x_k} \tag{16.8}$$

then from 16.3 and 16.8:

$$p(x|G,\theta) = \prod_{v=1}^{V}\prod_{c=1}^{C_v}\prod_{k=1}^{K} \theta_{vck}^{y_{vck}} \tag{16.9}$$

Likelihood

$$p(D|G,\theta) = \prod_{n=1}^{N} p(x_n|G,\theta) = \prod_{n=1}^{N}\prod_{v=1}^{V}\prod_{c=1}^{C_{nv}}\prod_{k=1}^{K} \theta_{vck}^{y_{nvck}} \tag{16.10}$$

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.

16.6 d-separation

1. P contains a chain

$$p(x,z|y) = \frac{p(x,y,z)}{p(y)} = \frac{p(x)p(y|x)p(z|y)}{p(y)} = \frac{p(x,y)p(z|y)}{p(y)} = p(x|y)p(z|y) \tag{16.11}$$

2. P contains a fork

$$p(x,z|y) = \frac{p(x,y,z)}{p(y)} = \frac{p(y)p(x|y)p(z|y)}{p(y)} = p(x|y)p(z|y) \tag{16.12}$$

3. P contains v-structure

$$p(x,z|y) = \frac{p(x,y,z)}{p(y)} = \frac{p(x)p(z)p(y|x,z)}{p(y)} \neq p(x|y)p(z|y) \tag{16.13}$$

16.7 Markov blanket

$$mb(t) = ch(t) \cup pa(t) \cup copa(t) \tag{16.14}$$

16.8 Reference

Mlapp chapter 10 Bayes nets

Chapter 17
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.