Contents

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 differ 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 different meanings depending on the context, although we try to avoid this where possible.

General math notation

Symbol	Meaning
$\lfloor x \rfloor$	Floor of x, i.e., round down to nearest integer
$\lceil x \rceil$	Ceiling of x, i.e., round down to nearest integer
$\boldsymbol{x} \otimes \boldsymbol{y}$	Convolution of $\boldsymbol{x}$ and $\boldsymbol{y}$
$\boldsymbol{x} \odot \boldsymbol{y}$	Hadamard (elementwise) product of $\boldsymbol{x}$ and $\boldsymbol{y}$
$a \wedge b$	logical AND
$a \vee b$	logical OR
$\neg a$	logical NOT
$\mathbb{I}(x)$	Indicator function, $\mathbb{I}(x) = 1$ if x is true, else $\mathbb{I}(x) = 0$
∞	Infinity
$\rightarrow$	Tends towards, e.g., $n \rightarrow \infty$
$\propto$	Proportional to, so $y = ax$ can be written as $y \propto x$
$\lvert x \rvert$	Absolute value
$\lvert \mathcal{S} \rvert$	Size (cardinality) of a set
$n!$	Factorial function
∇	Vector of first derivatives
∇^2	Hessian matrix of second derivatives
$\triangleq$	Defined as
$O(\cdot)$	Big-O: roughly means order of magnitude
$\mathbb{R}$	The real numbers
$1:n$	Range (Matlab convention): $1:n = 1, 2, ..., n$
$\approx$	Approximately equal to
$\arg\max_x f(x)$	Argmax: the value x that maximizes f
$B(a,b)$	Beta function, $B(a,b) = \dfrac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$
$B(\boldsymbol{\alpha})$	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 $\boldsymbol{X}$
$\|\boldsymbol{X}\|$	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
$\text{diag}(x)$	Diagonal matrix made from vector $\boldsymbol{x}$
$\text{diag}(X)$	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)
$\|\|\boldsymbol{x}\|\| = \|\|\boldsymbol{x}\|\|_2$	Euclidean or ℓ_2 norm $\sqrt{\sum_{j=1}^{d} x_j^2}$
$\|\|\boldsymbol{x}\|\|_1$	ℓ_1 norm $\sum_{j=1}^{d} \|x_j\|$
$\boldsymbol{X}_{:,j}$	j'th column of matrix
$\boldsymbol{X}_{i,:}$	transpose of i'th 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
$\text{cov}[X]$	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 $\boldsymbol{\theta}$
λ	Precision (inverse variance) $\lambda = 1/\sigma^2$
$\boldsymbol{\Lambda}$	Precision matrix $\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$
$\text{mode}[\boldsymbol{X}]$	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
$\boldsymbol{\Sigma}$	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 $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) \| i = 1 : N\}$
$\mathcal{D}_{test}$	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 ℓ_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 $\boldsymbol{\varepsilon}$ 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

- Supervised Learning
  - Classification
  - Regression
- Unsupervised Learning
  - Discovering clusters
  - Discovering latent factors
  - Discovering graph structure
  - Matrix completion
- Reinforcement Learning

In the **predictive** or **supervised learning** approach,the goal is to learn a **mapping** from **inputs x** to **outputs** y,given a labeled set of input-output paris $D = \{(\mathbf{x_i}, y_i)\}_{i=1}N$.Here D is called the **training set**,and N is the number of training examples. In the simplest setting,each training input $\mathbf{x_i}$ is a D-dimensional vector of numbers,representing,say,the height and weight of a person,which are called **features,attributes,or covariates**.

1.2 Supervised learning

Similarly the form of the output or **response variable** can in principle be anything,but most methods assume that y_i is **categorical or nominal** variable from some finite set,$y_i \in \{1,...,C\}$.When y_i is categorical,the problem is known as **classification or pattern recognition**,and when real-valued,known as **regression**.

1.2.1 Classification

Here the goal is to learn a mapping from inputs x to outputs y,where $y \in \{1,...,C\}$,with C being the number of classes.If $C = 2$,this is called **binary classification**;if $C > 2$,this is called **multiclass classification**.if the class labels are not mutually exclusive,we call it **multi-label classification**,but this is best viewed as predicting multiple related binary class labels(a so-called **multiple output model**). One way to formalize the problem is as **function approximation**:assume $y = f(\mathbf{x})$ for some unknown function f,and the goal of learning is to estimate the function f given a labeled training set,and then to make predictions(estimate) using $\hat{y} = \hat{f}(\mathbf{x})$.Our main goal is to make predictions on novel inputs,meaning ones that we have not seen before(**generalization**.

1.2.1.1 Probabilistic predictions

We denote the probability distribution over possible labels,given the input vector $\boldsymbol{x}$ and training set $\mathcal{D}$ by $p(y|\boldsymbol{x}, \mathcal{D})$. Given a probabilistic output,we can always compute out "best guess" as to the "true label" using

$$\hat{y} = \hat{f}(\mathbf{x}) = \arg\max_{c=1}^{C} p(y = c|\mathbf{x}, D) \tag{1.1}$$

This corresponds to the most probable class label,and is called the **mode** of distribution $p(y|\boldsymbol{x}, \mathcal{D})$;it is also known as a **MAP estimate**(MAP stands for**maximum a posteriori**).

1.2.1.2 Applications

1.2.2 Regression

1.3 Unsupervised learning

Descriptive or unsupervised learning approach is sometimes called **knowledge discovery**.We will formalize out task as one of **density estimation**,that is we want to build models of the form $p(\mathbf{x_i}|\theta)$,instead of $p(y_i|\mathbf{x_i},\theta)$.

1.3.1 Discovering clusters

Let $z_i \in \{1,...,K\}$ represent the cluster to which data point i is assigned.(z_i is an exmaple of **hidden or latent** variable).

1.3.2 Discovering latent factors

Although the data may appear high dimensional,there may only be a small number of degrees of variability,corresponding to **latent factors**.The most common approach to dimensionality reduction is called **principal components analysis or PCA**.

1.3.3 Discovering graph structure

1.3.4 Matrix completion

1.3.4.1 Image inpainting

1.3.4.2 Collaborative filtering

1.3.4.3 Market basket analysis

1.4 Three elements of a machine learning model

Model = Representation + Evaluation + Optimization[1]

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

1.4.2 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.

1.4.2.1 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 $L(y_i, \widehat{y})$.

The following is some common loss functions:

1. 0-1 loss function
   $L(Y, f(X)) = \mathbb{I}(Y, f(X)) = \begin{cases} 1, & Y = f(X) \\ 0, & Y \neq f(X) \end{cases}$
2. Quadratic(squared) loss function $L(Y, f(X)) = \frac{1}{2}(Y - f(X))^2$
3. Absolute loss function $L(Y, f(X)) = |Y - f(X)|$
4. Exponential loss function $L(Y, f(X) = exp(-\hat{y}_i f(\mathbf{x_i}))$
5. Logarithmic loss function
   $L(Y, P(Y|X)) = -\log P(Y|X)$

Name	Loss	Derivative	f^*	Algorithm
Squared error	$\frac{1}{2}(y_i - f(\mathbf{x_i}))^2$	$y_i - f(\mathbf{x_i})$	$\mathbb{E}[y\|\mathbf{x_i}]$	L2Boosting
Absolute error	$\|y_i - f(\mathbf{x_i})\|$	$sgn(y_i - f(\mathbf{x_i}))$	$median(y\|\mathbf{x_i})$	Gradient boosting
Exponential loss	$exp(-\hat{y}_i f(\mathbf{x_i}))$	$-\hat{y}_i exp(-\hat{y}_i f(\mathbf{x_i}))$	$\frac{1}{2} log \frac{\pi_i}{1-\pi_i}$	AdaBoost
Logloss	$log(1 + e^{-\hat{y}_i f_i})$	$y_i - \pi_i$	$\frac{1}{2} log \frac{\pi_i}{1-\pi_i}$	LogitBoost

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

$$R_{\exp}(f) = E\left[L(Y, f(X))\right] = \int L(y, f(x)) P(x, y) \mathrm{d}x\mathrm{d}y \tag{1.2}$$

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

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

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 §**??**).

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

1.4.2.2 ERM and SRM

Definition 1.4. ERM(Empirical risk minimization)

$$\min_{f\in\mathcal{F}} R_{\text{emp}}(f) = \min_{f\in\mathcal{F}} \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) \tag{1.4}$$

Definition 1.5. Structural risk

$$R_{\text{smp}}(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda J(f) \tag{1.5}$$

Definition 1.6. SRM(Structural risk minimization)

$$\min_{f\in\mathcal{F}} R_{\text{srm}}(f) = \min_{f\in\mathcal{F}} \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda J(f) \tag{1.6}$$

1.4.3 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.

1.5 Model Selection

If data is plentiful,then on approach to avoid over-fitting is to use some of the some of the available data to train a range of models,or a given model with a range of values for its complexity parameters,and then to compare them on independent data,sometimes called a **Validation set**,and select the one having the best predictive performance.If the model design is iterated many times using a limited size data set,then some over-fitting to the validation data can occur and so it may be necessary to keep aside a third **test set** on which the performance of the selected model is finally evaluated.

1.5.0.1 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.

In many applications,however,the supply of data for training and testing will be limited.We wish to use as much of available data as possible for both training and validation set.One solution to this dilemma is to use **cross-validations**.One round of cross-validation involves partitioning a sample of data into complementary subsets, performing the analysis on one subset (called the training set), and validating the analysis on the other subset (called the

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

validation set or testing set). To reduce variability, multiple rounds of cross-validation are performed using different partitions, and the validation results are averaged over the rounds.

1.6 The Curse of Dimensionality

The common theme that arise when analyzing and organizing data in **high-dimensional spaces** is that when the **demensionality increases**,the **volume of the space** increases so fast (**grows exponentially with the dimensionality**) that the available data become sparse.

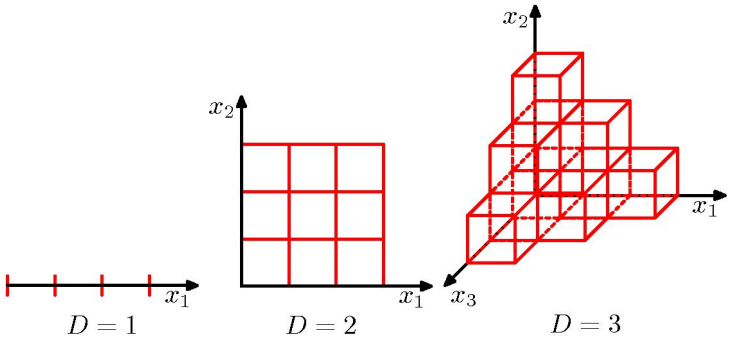


Fig. 1.1: Illustration of the curse of dimensionality:the number of regions of a regular grid grows exponentially with the dimensionality D of the space.

The volume of a sphere of radius r in D dimensions must scale as r^D,and so we write

$$V_D(r) = K_D r^D \tag{1.7}$$

The fraction of the volume of the sphere that lies between radius $r = 1-\varepsilon$ and $r = 1$

$$\frac{V_D(1) - V_D(1-\varepsilon)}{V_D(1)} = 1 - (1-\varepsilon)^D \tag{1.8}$$

for large D this fraction tends to 1 even for small values of ε.Thus most of the volume of a sphere is concentrated in a thin shell near the surface!

1.7 Decision Theory

Probability theory provides us with a consistent mathematical framework for quantifying and manipulating uncertainty.And when combined with probability theory,decision theory allows us to make optimal decisions in situations involving uncertainty such as those encountered in pattern recognition.

Suppose we have an input vector x together with a corresponding vector t of target variables, and our goal is to predict t given a new value for x. For regression problems, t will comprise continuous variables, whereas for classification problems t will represent class labels. The joint probability distribution p(x, t) provides a complete summary of the uncertainty associated with these variables. Determination of p(x, t) from a set of training data is an example of **inference** and is typically a very difficult problem.

The **decision** step, is the subject of decision theory to tell us how to make optimal decisions given the appropriate probabilities. We shall see that the decision stage is generally very simple, even trivial, once we have solved the inference problem.

Consider,for a classification problem,in which case,the input are vectors $\boldsymbol{x}$ and we denote the class i by $\mathcal{C}_k$.Using Bayes' theorem,these probabilities can be expressed in the form

$$p(\mathcal{C}_k|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\boldsymbol{x})} \tag{1.9}$$

Note that any of the quantities appearing in Bayes' theorem can be obtained from the joint distribution $p(\boldsymbol{x},\mathcal{C}_k)$ by either **marginalizing** or **conditioning** with respect to the appropriate variables.We can now interpret $p(\mathcal{C}_k)$ as the prior probability for the class $\mathcal{C}_k$,and the $p(\mathcal{C}_k|\boldsymbol{x})$ as the corresponding posterior probability,$p(\boldsymbol{x}|\mathcal{C}_k)$ is the likelihood function.If our aim it to minimize the chance of assigning $\boldsymbol{x}$ to the wrong class,then intuitively we would choose the class having the higher posterior probability.

1.7.1 Minimizing the misclassification rate

1.7.2 Minimizing the expected loss

1.7.3 The reject option

1.7.4 Inference and decision

We have broken the classification problem down into two separate stages,the **inference stage** in which we use training data to learn to **model** for $p(\mathcal{C}_k|\boldsymbol{x})$,and the subsequent **decision stage** in which we use these posterior probabilities to make optimal class assignments.An alternative possibility would be to solve both problems together and simply learn a function that maps input $\boldsymbol{x}$ directly into decisions.Such a function is called a **discriminant function**.

In fact,we can identify three distinct approaches to solving decision problems,given, in decreasing order of complexity,by

1. First solve the inference problem of determining the class-conditional densities $p(x|\mathcal{C}_k)$ for each class $\mathcal{C}_k$ individually. Also separately infer the prior class probabilities $p(\mathcal{C}_k)$. Then use *Bayes* theorem in the form

   $$p(\mathcal{C}_k|\boldsymbol{x}) = \frac{p(x|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\boldsymbol{x})} \tag{1.10}$$

   to find the posterior class probabilities $p(\mathcal{C}_k|\boldsymbol{x})$. As usual, the denominator in *Bayes* theorem can be found in terms of the quantities appearing in the numerator, because

   $$p(\boldsymbol{x}) = \sum_k p(\boldsymbol{x}|\mathcal{C}_k)p(\mathcal{C}_k) \tag{1.11}$$

   Equivalently, we can model the joint distribution $p(x,\mathcal{C}_k)$ directly and then normalize to obtain the posterior probabilities. Having found the posterior probabilities, we use decision theory to determine class membership for each new input x.Approaches that explicitly or implicitly model the **distribution of inputs as well as outputs** are known as **Generative models**,because by sampling from them it is possible to generate synthetic data points in the input space.
2. First solve the inference problem of determining the posterior class probabilities $p(\mathcal{C}_k|\boldsymbol{x})$,and then subsequently use decision theory to assign each new $\boldsymbol{x}$ to one of the classes.Approaches that model the posterior probabilities are called **discriminative models**
3. Find a function $f(\boldsymbol{x})$,called a discriminant function,which maps each input $\boldsymbol{x}$ directly into a class label.For instance,in the case of two-case problems,$f(\cdot)$ might be binary valued and such that $f = 0$ represents class $\mathcal{C}_1$ and so on.

Now consider the relative merits of there three alternatives.Approach (a) is the most demanding because it involves finding the joint distribution over both x and Ck. For many applications, x will have high dimensionality, and conse-

quently we may need a large training set in order to be able to determine the class-conditional densities to reasonable accuracy. Note that the class **priors** $p(\mathcal{C}_\parallel)$ can often be estimated simply from the **fractions of the training set data points in each of the classes**. One advantage of approach (a), however, is that it also allows the marginal density of data p(x) to be determined from (1.83). This can be useful for detecting new data points that have low probability under the model and for which the predictions may be of low accuracy, which is known as **outlier detection or novelty detection** (Bishop, 1994; Tarassenko, 1995).

However, if we only wish to make classification decisions, then it can be wasteful of computational resources, and excessively demanding of data, to find the joint distribution p(x, Ck) when in fact we only really need the posterior probabilities p(Ck—x), which can be obtained directly through approach (b). Indeed, the class conditional densities may contain a lot of structure that has little effect on the posterior probabilities, as illustrated in Figure 1.27. There has been much interest in exploring the relative merits of generative and discriminative approaches to machine learning, and in finding ways to combine them (Jebara, 2004; Lasserre et al., 2006). An even simpler approach is (c) in which we use the training data to find a discriminant function f(x) that maps each x directly onto a class label, thereby combining the inference and decision stages into a single learning problem. In the example of Figure 1.27, this would correspond to finding the value of x shown by the vertical green line, because this is the decision boundary giving the minimum probability of misclassification.

Reasons for wanting to compute the posterior probabilities include:

- **Minimize risk.**
- **Reject option.**
- **Compensating for class priors.**
- **Combining models.** For complex applications,we may wish to break the problem into a number of smaller subproblems each of which can be tackled by a separate module.As long as each of the two models gives posterior probabilities for the classes, we can combine the outputs systematically using the rules of probability. One simple way to do this is to assume that, for each class separately, the distributions of inputs are independent,so that

$$p(\boldsymbol{x_I}, \boldsymbol{x_B}|\mathcal{C}_k) = p(\boldsymbol{x_I}|\mathcal{C}_k)p(\boldsymbol{x_B}|\mathcal{C}_k) \tag{1.12}$$

  This is an example of **conditional independence** property.The posterior probability,given both data(from different modules) ,is then given by

$$p(\mathcal{C}_k|\boldsymbol{x_I}, \boldsymbol{x_B}) \propto p(\boldsymbol{x_I}, \boldsymbol{x_B}|\mathcal{C}_k)p(\mathcal{C}_k) \tag{1.13}$$

$$\propto p(\boldsymbol{x_I}|\mathcal{C}_k)p(\boldsymbol{x_B}|\mathcal{C}_k)p(\mathcal{C}_k) \tag{1.14}$$

$$\propto \frac{p(\mathcal{C}_k|\boldsymbol{x_I})p(\mathcal{C}_k|\boldsymbol{x_B})}{p(\mathcal{C}_k)} \tag{1.15}$$

1.8 Information theory

Consider a discrete random variable x and we ask how much information is received when we observe a specific value for this variable.Our measure of information content will therefore depend on the probability distribution $p(x)$,and we therefore look for a quantity $h(x)$ that is monotonic function of the probability $p(x)$ and that expresses the information content.The form of $h(\cdot)$ can be found by noting that if we have two events x and y that are unrelated,then the information gain from observing both of them should be the sum of the information gained from each of them separately,so that $h(x,y) = h(x) + h(y)$.Two unrelated events will be statistically independent and so $p(x,y) = p(x)p(y)$.From these two relationships,it is easily shown that $h(x)$ must be given by the logarithm of $p(x)$ and so we have

$$h(x) = -\log_2 p(x) \tag{1.16}$$

where the negative sign ensures that information is positive or zero.Note that low probability events x correspond to high information content.The choice of basis for the logarithm is arbitrary.

Now suppose that a sender wishes to transmit the value of a random variable to a receiver.The average amount of information that they transmit in the process is obtained by taking the expectation of 1.16 with respect to the distribution $p(x)$ and is given by

$$H[x] = -\sum_x p(x)\log_2 p(x) \tag{1.17}$$

This important quantity is called the **entropy** of the random variable x.

The **noiseless coding theorem** (Shannon,1948) states that the entropy is a lower bound on the number of bits needed to transmit the state of a random variable.

For example ,we can represente the states $x = a,b,c,d,e,f,g,h$ using, for instance, the following set of code strings: 0, 10, 110, 1110, 111100, 111101, 111110, 111111. The average length of the code that has to be transmitted is then

$$\text{average code length} = \frac{1}{2}\times 1+\frac{1}{4}\times 2+\frac{1}{8}\times 3+\frac{1}{16}\times 4+4\times\frac{1}{64}\times 6 = 2 \text{ bits} \tag{1.18}$$

Now,we switch to the use of natural logarithm in defining entropy.And entropy can be understood in an alternative view of **multiplicity**. The continuous form of the entropy is given by

$$\lim_{\Delta\to 0}\{\sum_i p(x_i)\Delta \ln p(x_i)\} = -\int p(x)\ln p(x)dx \tag{1.19}$$

where the quantity on the right-side is called the **differential entropy**.

For a density defined over multiple continuous variables,denoted collectively by the vector $\boldsymbol{x}$,the differential entropy is given by

$$H[x] = -\int p(\boldsymbol{x})lnp(\boldsymbol{x})d\boldsymbol{x} \tag{1.20}$$

In the case of discrete distributions,we see that the maximum entropy configuration correspondes to an equal distribution of probabilities across the possible states of variable.Now consider the maximum entropy configuration for a continuous variable.It will be necessary to constrain the first and second moments of $p(x)$ as well as preserving the normalization constraint.Therefore we need to maximize the differential entropy $H[x] = -\int p(\boldsymbol{x})\ln p(\boldsymbol{x})d\boldsymbol{x}$ with three constraints

$$\int_{-\infty}^{\infty} p(x)dx = 1 \tag{1.21}$$

$$\int_{-\infty}^{\infty} xp(x)dx = \mu \tag{1.22}$$

$$\int_{-\infty}^{\infty} (x-\mu)^2 p(x)dx = \sigma^2. \tag{1.23}$$

The constrained maximization can be performed using Lagrange multipliers so that we maximize the following functional with respect to $p(x)$

$$-\int_{-\infty}^{\infty} p(x)\ln p(x)dx+\lambda_1(\int_{-\infty}^{\infty} p(x)dx-1)+\lambda_2(\int_{-\infty}^{\infty} xp(x)dx-\mu)+\lambda_3(\int_{-\infty}^{\infty}(x-\mu)^2p(x)dx-\sigma^2) \tag{1.24}$$

Using the calculus of variations,we set the derivative of this functional to zero giving

$$p(x) = exp\{-1+\lambda_1+\lambda_2 x+\lambda_3(x-\mu)^2\} \tag{1.25}$$

The Lagrange multipliers can be found by back substitution of this result into the three constraint equations,leading finally to the result

$$p(x) = \frac{1}{(2\pi\sigma^2)^(1/2)}exp\{-\frac{(x-\mu)^2}{2\sigma^2}\} \tag{1.26}$$

and so the distribution that maximizes the differential entropy is the Gaussian.

If we evaluate the differential entropy of the Gaussian,we obtain

$$H[x] = \frac{1}{2}\{1+\ln(2\pi\sigma^2)\} \tag{1.27}$$

Thus we see again that the entropy increases as the distribution becomes broader,i.e.,as σ^2 increases.

Suppose we have joint distribution $p(\boldsymbol{x},\boldsymbol{y})$ from which we draw pairs of values of $\boldsymbol{x}$ and $\boldsymbol{y}$.If a value of $\boldsymbol{x}$ is already known,then the additional information needed to specify the corresponding value of $\boldsymbol{y}$ is given by $-\ln p(\boldsymbol{y}|\boldsymbol{x})$,thus the average additional information needed can be written as

$$H[\boldsymbol{y}|\boldsymbol{x}] = -\iint p(\boldsymbol{y},\boldsymbol{x}) \ln p(\boldsymbol{y}|\boldsymbol{x}) d\boldsymbol{y} d\boldsymbol{x} \tag{1.28}$$

which is called the **conditional entropy** of $\boldsymbol{y}$ given $\boldsymbol{x}$.Using the product rule,the conditional entropy satisfies the relation

$$H[\boldsymbol{x},\boldsymbol{y}] = H[\boldsymbol{y}|\boldsymbol{x}] + H[\boldsymbol{x}] \tag{1.29}$$

where $H[\boldsymbol{x},\boldsymbol{y}]$ is the differential entropy of $p(\boldsymbol{x},\boldsymbol{y})$ and $H[\boldsymbol{x}]$ is the differential entropy of the marginal distribution $p(\boldsymbol{x})$.

Proof.

$$H[\boldsymbol{x},\boldsymbol{y}] - H[\boldsymbol{y}|\boldsymbol{x}] = -\iint p(\boldsymbol{y},\boldsymbol{x}) \ln p(\boldsymbol{y},\boldsymbol{x}) d\boldsymbol{y} d\boldsymbol{x} + \iint p(\boldsymbol{y},\boldsymbol{x}) \ln p(\boldsymbol{y}|\boldsymbol{x}) d\boldsymbol{y} d\boldsymbol{x} \tag{1.30}$$

$$= \iint p(\boldsymbol{y},\boldsymbol{x}) \ln \frac{p(\boldsymbol{y}|\boldsymbol{x})}{p(\boldsymbol{y},\boldsymbol{x})} \tag{1.31}$$

$$= \iint p(\boldsymbol{y},\boldsymbol{x}) \ln \frac{1}{p(\boldsymbol{x})} d\boldsymbol{y} d\boldsymbol{x} \tag{1.32}$$

$$= -\iint p(\boldsymbol{y},\boldsymbol{x}) \ln p(\boldsymbol{x}) d\boldsymbol{y} d\boldsymbol{x} \tag{1.33}$$

$$= -\iint p(\boldsymbol{x}) \ln p(\boldsymbol{x}) d\boldsymbol{x} \tag{1.34}$$

$$= H[\boldsymbol{x}] \tag{1.35}$$

1.8.1 Relative entropy and mutual information

Consider some unknown distribution $p(\boldsymbol{x})$,and suppose that we have modelled this using an approximating distribution $q\boldsymbol{x})$.Then the average **additional** amount of information(in nats) required to specify the value of $\boldsymbol{x}$ as a result of using $q(\boldsymbol{x})$ instead of the true distribution is given by

$$KL(q \parallel p) = -\int p(\boldsymbol{x}) \ln q(\boldsymbol{x}) d\boldsymbol{x} - (-\int p(\boldsymbol{x}) \ln p(\boldsymbol{x}) d\boldsymbol{x}) \tag{1.36}$$

$$= -\int p(\boldsymbol{x}) \ln \{\frac{q\boldsymbol{x}}{p\boldsymbol{x}}\} d\boldsymbol{x} \tag{1.37}$$

This is known as the **relative entropy** or **Kullback-Leibler divergence**,or **KL divergence** (Kullback and Leibler,1951),between the distributions $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$.Note that it is not a symmetrical quantity. Now introduce the concept of **convex** functions .A function $f(x)$ is said to be convex if it has the property that every chord lies on or above the function, as shown in Figure 1.31. Any value of x in the interval from x = a to x = b can be written in the form a + (1)b where $0 \leq \leq 1$. The corresponding point on the chord is given by f(a) + (1)f(b),and the corresponding value of the function is f(a + (1)b).Convexity then implies

$$f(\lambda a + (1-\lambda)b) \leq \lambda f(a) + (1-\lambda) f(b) \tag{1.38}$$

This is equivalent to the requirement that the second derivative of the function be everywhere positive. A function is called strictly convex if the equality is satisfied only for $\lambda = 0$ and $= 1$. If a function has the opposite property, namely that every chord lies on or below the function, it is called concave, with a corresponding definition for strictly concave. If a function $f(x)$ is convex, then $f(x)$ will be concave.

Using the technique of **proof by induction**,we can show that a convex function $f(x)$ satisfies

$$f(\sum_{i=1}^{M} \lambda_i x_i) \leq \sum_{i=1}^{M} \lambda_i f(x_i) \tag{1.39}$$

where $\lambda_i \geq 0$ and $\sum_i \lambda_i = 1$,for any set of points $\{x_i\}$,this is known as **Jensen's inequality**.If we interpret the λ_i as the probability distribution over a discrete variable x taking the values $\{x_i\}$,then 1.39 can be written

$$f(\mathbb{E}[x]) \leq \mathbb{E}[f(x)] \tag{1.40}$$

where $\mathbb{E}(\cdot)$ denotes the expectation.For continuous variables,Jensen's inequality takes the form

$$f(\int \boldsymbol{x} p(\boldsymbol{x})) \leq \int f(\boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x} \tag{1.41}$$

We can apply Jensen's inequality to the KL-divergence to give

$$KL(p \parallel q) = -\int p(\boldsymbol{x}) \ln\{\frac{q(\boldsymbol{x})}{p(\boldsymbol{x})}\} d\boldsymbol{x} \geq -\ln \int q(\boldsymbol{x}) d\boldsymbol{x} = 0 \tag{1.42}$$

where we have used the fact that $-\ln x$ is a convex function,together with the normalization condition $\int q(\boldsymbol{x}) d\boldsymbol{x} = 1$.

Theorem 1.1 (log sum inequality). *Let $a_1, ..., a_n$ and $b_1, ..., b_n$ be nonnegative numbers.Denote the sum of all a_i by a and sum of all b_i by b.The log sum inequality states that*

$$\sum_{i=1}^{n} a_i \log \frac{a_i}{b_i} \geq a \log \frac{a}{b}, \tag{1.43}$$

Proof. Setting $f(x) = x \log x$,we have

$$\sum_{i=1}^{n} a_i \log \frac{a_i}{b_i} = \sum_{i=1}^{n} b_i f\left(\frac{a_i}{b_i}\right) = b \sum_{i=1}^{n} \frac{b_i}{b} f\left(\frac{a_i}{b_i}\right) \tag{1.44}$$

$$\geq b f\left(\sum_{i=1}^{n} \frac{b_i}{b} \frac{a_i}{b_i}\right) = b f\left(\frac{1}{b} \sum_{i=1}^{n} a_i\right) = b f\left(\frac{a}{b}\right) \tag{1.45}$$

$$= a \log \frac{a}{b}, \tag{1.46}$$

where the inequality follows Jensen's inequality1.39 since f is convex. So we have

$$D_{\mathrm{KL}}(P \| Q) \equiv \sum_{i=1}^{n} p_i \log_2 \frac{p_i}{q_i} \geq 1 \log \frac{1}{1} = 0. \tag{1.47}$$

Suppose we try to approximate this distribution using some parametric distribution $q(\boldsymbol{x}|\boldsymbol{\theta})$.Then the expectation of KL-divergence with respect to $p(\boldsymbol{x})$ can be approximated by a finite sum over these training points,so that

$$KL(p \parallel q) \simeq \sum_{n=1}^{N} \{-\ln q(\boldsymbol{x_n}|\boldsymbol{\theta}) + \ln p(\boldsymbol{x_n})\} \tag{1.48}$$

The second term on the right-hand side is independent of $\boldsymbol{\theta}$,and the first term is the negative log likelihood function for $\boldsymbol{\theta}$ under the distribution $q(\boldsymbol{x}|\boldsymbol{\theta})$ evaluated using the training set.Thus **minimizing this KL-divergence is equivalent to maximizing the likelihood function**.

If ,for a joint distribution,the variables $\boldsymbol{x}$ and $\boldsymbol{y}$ are not independent,we can gain some idea of whether they are 'close' to being independent by considering the KL-divergence between the joint distribution and the product of the marginals,given by

$$I[\boldsymbol{x}, \boldsymbol{y}] \equiv KL(p(\boldsymbol{x}, \boldsymbol{y}) \parallel p(\boldsymbol{x}) p(\boldsymbol{y})) \tag{1.49}$$

$$= -\iint p(\boldsymbol{x}, \boldsymbol{y}) \ln(\frac{p(\boldsymbol{x}) p(\boldsymbol{y})}{p(\boldsymbol{x}, \boldsymbol{y})}) d\boldsymbol{x} d\boldsymbol{y} \tag{1.50}$$

which is called the **mutual information** between the variables $\boldsymbol{x}$ and $\boldsymbol{y}$.From the property of the KL-divergence,we see that $I(\boldsymbol{x}, \boldsymbol{y}) \geq 0$ with equality if,and only if $\boldsymbol{x}$ and $\boldsymbol{y}$ are independent.Using the sum and product rules of probability,we

see the relationship to conditional entropy through

$$I[\boldsymbol{x},\boldsymbol{y}] = H[\boldsymbol{x}] - H[\boldsymbol{x}|\boldsymbol{y}] = H[\boldsymbol{y}] - H[\boldsymbol{y}|\boldsymbol{x}] \tag{1.51}$$

Thus we can view the mutual information as the reduction in the uncertainty about $\boldsymbol{x}$ by virtue of being told the value of $\boldsymbol{y}$ (or vice versa).

1.9 Some basic concepts

1.9.1 Parametric vs non-parametric models

1.9.2 A simple non-parametric classifier: K-nearest neighbours

1.9.2.1 Representation

$$y = f(\boldsymbol{x}) = \arg\min_{c} \sum_{\boldsymbol{x}_i \in N_k(\boldsymbol{x})} \mathbb{I}(y_i = c) \tag{1.52}$$

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.

1.9.2.2 Evaluation

No training is needed.

1.9.2.3 Optimization

No training is needed.

1.9.3 Overfitting

1.9.4 Model selection

When we have a variety of models of different complexity (e.g., linear or logistic regression models with different degree polynomials, or KNN classifiers with different values ofK), how should we pick the right one? A natural approach is to compute the **misclassification rate** on the training set for each method.

Chapter 2
Probability and Statistics

One role for the distributions is to model the probability distribution $p(\boldsymbol{x})$ of a random variable x,given a finite set of observations.This problem is known as **density estimation**.We shall assume that the data points are independent and identically distributed.

Parameter distributions are governed by a small number of adaptive parameters,such as the mean and variance in the case of a Gaussian for example.In a frequentist treatment, we choose specific values for the parameters by optimizing some criterion, such as the likelihood function. By contrast, in a Bayesian treatment we introduce prior distributions over the parameters and then use Bayes theorem to compute the corresponding posterior distribution given the observed data.

An important role is played by **conjugate priors**,that lead to posterior distributions having the same functional form as the prior,and that therefore lead to a greatly simplified Bayesian analytics.

Nonparametric density estimation methods in which the form of the distribution typically depends on the size of the data set.Parameters in such models control the model complexity rather than the form of the distribution.We cover three nonparametric methods based respectively on histograms,nearest-neighbours,and kernels.

2.1 Frequentists vs. Bayesians

There are two different interpretations of probability. One is called the **frequentist** interpretation. In this view, probabilities represent long run **frequencies** of events. For example, the above statement means that, if we flip the coin many times, we expect it to land heads about half the time.

The other interpretation is called the **Bayesian** interpretation of probability. In this view, probability is used to quantify our **uncertainty** about something; hence it is fundamentally related to information rather than repeated trials (Jaynes 2003). In the Bayesian view, the above statement means we believe the coin is equally likely to land heads or tails on the next toss

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. For example, we might want to compute the probability that the polar ice cap will melt by 2020 CE. This event will happen zero or one times, but cannot happen repeatedly. Nevertheless,we thought to be able to quantify our uncertainty about this event. To give another machine learning oriented example, we might have observed a blip on our radar screen, and want to compute the probability distribution over the location of the corresponding target (be it a bird, plane, or missile). In all these cases, the idea of repeated trials does not make sense, but the Bayesian interpretation is valid and indeed quite natural. We shall therefore adopt the Bayesian interpretation in this book. Fortunately, the basic rules of probability theory are the same, no matter which interpretation is adopted.

2.2 probability theory

2.2.1 Concepts

The expression $p(A)$ denotes the probability that event A is true.We require that $0 \leq p(A) \leq 1$,where 0 means the event definitely will not happen,and $p(A) = 1$ means the event definitely will happen.$p(\hat{A})$ denotes the probability of the event not A;this is defined to be $p(\hat{A}) = 1 - p(A)$.

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

Descrete random variable: X ,which can take on any value from a finite or countably infinite set .We denote the probability of the event that $X = x$ by $p(X = x)$,or just $p(x)$ for short.Here $p()$ is called a **probability mass function** or **pmf**.The pmfs are defined one **state space**.$\mathbb{I}$ denotes the binary **indicator funcition**.

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

probability Probability is the measure of the likeliness that an event will occur.

conditional probability A conditional probability measures the probability of an event given that (by assumption, presumption, assertion or evidence) another event has occurred.

joint probability Joint probability is a measure of two events happening at the same time, and can only be applied to situations where more than one observation can be occurred at the same time.

prior probability distribution In Bayesian statistical inference, a prior probability distribution, often called simply the prior, of an uncertain quantity p is the probability distribution that would express one's uncertainty about p **before** some evidence is taken into account.

posterior probability distribution In Bayesian statistics, the posterior probability of a random event or an uncertain proposition is the **conditional probability** that is assigned **after** the relevant evidence or background is taken into account.

likelihood function In statistics, a likelihood function (often simply the likelihood) is a function of the parameters of a statistical model.The likelihood of a set of parameter values, , given outcomes x, is equal to the **probability** or **probability desity** of those observed outcomes given those parameter values.

2.2.2 Fundamental rules

In this section,we review the basic rule of probability.

2.2.2.1 Probability of a union of two events

Given two events,A and B,we define the probability of A or B as follows:

$$p(A \cup B) = p(A) + p(B) - p(A \cap B) \tag{2.1}$$
$$= p(A) + p(B) \tag{2.2}$$

if A and B are mutually independent

2.2.2.2 Joint probabilities

We define the probability of the joint event A and B as follows:

$$p(A,B) = p(A \cap B) = p(A|B)p(B) \tag{2.3}$$

This is sometimes called the **product rule**

2.2.2.3 Conditional probability

Define the **conditional probability** of event A,given that event B is true,as follows:

$$p(A|B) = \frac{p(A,B)}{p(B)}, if p(B) > 0 \tag{2.4}$$

2.2.2.4 sum rule

$$p(X) = \sum_{Y} p(X,Y) \tag{2.5}$$

2.2.2.5 product rule

$$p(X,Y) = p(Y|X)p(X) \tag{2.6}$$
$$p(A,B|C) = p(A|C)p(B|AC) = p(B|C)p(A|BC) \tag{2.7}$$

Proof. $p(X,Y) = p(Y|X)p(X)$ is by definition.

$$p(A,B|C) = \frac{p(A,B,C)}{p(C)} \tag{2.8}$$
$$= \frac{p(A,C)p(B|A,C)}{p(C)} \tag{2.9}$$
$$= p(A|C)p(B|A,C) \tag{2.10}$$

$$p(C|A,B) = \frac{p(A,B,C)}{p(A,B)} \tag{2.11}$$
$$= \frac{\frac{p(A,B,C)}{p(B)}}{\frac{p(A,B)}{p(B)}} \tag{2.12}$$
$$= \frac{p(A,C|B)}{p(A|B)} \tag{2.13}$$

2.2.2.6 CDF

$$F(x) \triangleq P(X \leq x) = \begin{cases} \sum_{u \leq x} p(u) & \text{, discrete} \\ \int_{-\infty}^{x} f(u)\mathrm{d}u & \text{, continuous} \end{cases} \tag{2.14}$$

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

2.2.2.8 probability densities

probability densities

$$p(x \in (a,b)) = \int_a^b p(x)dx \tag{2.15}$$

The probability density function p(x) must satisfy the two conditions

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

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

$$\begin{cases} p(x) \geq 0 \\ \int_{-\infty}^{\infty} p(x)dx = 1 \end{cases} \tag{2.16}$$

Combinations of discrete and continuous variables.

$$p(x) = \int_p (x,y)dy \tag{2.17}$$

$$p(x,y) = p(y|x)p(x) \tag{2.18}$$

2.2.3 Mutivariate random variables

2.2.3.1 Joint CDF

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

$$F(x,y) \triangleq P(X \leq x, Y \leq y) = \begin{cases} \sum_{u \leq x, v \leq y} p(u,v) \\ \int_{-\infty}^{x} \int_{-\infty}^{y} f(u,v) \mathrm{d}u \mathrm{d}v \end{cases} \tag{2.19}$$

product rule:

$$p(X,Y) = P(X|Y)P(Y) \tag{2.20}$$

Chain rule:

$$p(X_{1:N}) = p(X_1)p(X_3|X_2,X_1)...p(X_N|X_{1:N-1}) \tag{2.21}$$

2.2.3.2 Marginal distribution

Marginal CDF:

$$F_X(x) \triangleq F(x,+\infty) = \begin{cases} \sum\limits_{x_i \leq x} P(X = x_i) = \sum\limits_{x_i \leq x} \sum\limits_{j=1}^{+\infty} P(X = x_i, Y = y_j) \\ \int_{-\infty}^{x} f_X(u)du = \int_{-\infty}^{x} \int_{-\infty}^{+\infty} f(u,v) \mathrm{d}u \mathrm{d}v \end{cases} \tag{2.22}$$

$$F_Y(y) \triangleq F(+\infty,y) = \begin{cases} \sum\limits_{y_j \leq y} p(Y = y_j) = \sum\limits_{i=1}^{+\infty} \sum_{y_j \leq y} P(X = x_i, Y = y_j) \\ \int_{-\infty}^{y} f_Y(v)dv = \int_{-\infty}^{+\infty} \int_{-\infty}^{y} f(u,v) \mathrm{d}u \mathrm{d}v \end{cases} \tag{2.23}$$

Marginal PMF and PDF:

$$\begin{cases} P(X = x_i) = \sum_{j=1}^{+\infty} P(X = x_i, Y = y_j) & \text{, descrete} \\ f_X(x) = \int_{-\infty}^{+\infty} f(x,y) \mathrm{d}y & \text{, continuous} \end{cases} \tag{2.24}$$

$$\begin{cases} p(Y = y_j) = \sum_{i=1}^{+\infty} P(X = x_i, Y = y_j) & \text{, descrete} \\ f_Y(y) = \int_{-\infty}^{+\infty} f(x,y) \mathrm{d}x & \text{, continuous} \end{cases} \tag{2.25}$$

2.2.3.3 Conditional distribution

Conditional PMF:

$$p(X = x_i|Y = y_j) = \frac{p(X = x_i, Y = y_j)}{p(Y = y_j)} \text{ if } p(Y) > 0 \tag{2.26}$$

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

Conditional PDF:

$$f_{X|Y}(x|y) = \frac{f(x,y)}{f_Y(y)} \tag{2.27}$$

2.2.4 Bayes rule

Bayes' theorem

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)} \tag{2.28}$$

Denominator in Bayes' theorem

$$p(X) = \sum_Y p(X|Y)p(Y) \tag{2.29}$$

Bayesian probabilities So far, we have viewed probabilities in terms of the frequencies of random,repeatable events,which we shall refer to as the classical or frequentist interpretation of probability.Now we turn to the more general Bayesian view,in which probabilities provide a quantification of uncertainty.

We can adopt a similar approach when making inferences about quantities such as the parameters **w** in the polynomial curve fitting.We capture our assumptions about **w**,before observing the data,in the form of a **prior** probability distribution $p(\mathbf{w})$.The effect of the observed data $\mathcal{D} = t_1,...,t_N$ is expressed through the conditional probability $p(D|w)$.Bayes' theorem,which takes the form

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})} \tag{2.30}$$

then allows us to evaluate the uncertainty in $\boldsymbol{w}$ **after** we have observed $\mathcal{D}$ in the form of the **posterior** probability $p(\mathbf{w}|\mathcal{D})$.

The quantity $p(D|w)$ on the right-hand side of Bayes' theorem is evaluated for the observed data set D and can be viewed as a function of the parameter vector **w**,in which case it is called the **likelihood function**.It expresses how probable the observed data set is for different settings of the parameter vector $\boldsymbol{w}$.

$$posteroir \propto likelihood \times prior \tag{2.31}$$

where all of these quantities are viewed as functions of $\boldsymbol{w}$. Summing both side with respect to **w**

$$p(\mathcal{D})p(\boldsymbol{w}|\mathcal{D}) = p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w}) \tag{2.32}$$

$$\Rightarrow \sum_{\boldsymbol{w}} p(\mathcal{D})p(\boldsymbol{w}|\mathcal{D}) = \sum_{\boldsymbol{w}} p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w}) \tag{2.33}$$

$$\Rightarrow p(\mathcal{D}) = \sum_{\boldsymbol{w}} p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w}) \tag{2.34}$$

$$\tag{2.35}$$

Integrating both side with respect to **w**

$$\int p(\mathcal{D})p(\boldsymbol{w}|\mathcal{D})d\boldsymbol{w} = \int p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w} \tag{2.36}$$

$$p(\mathcal{D}) = \int p(\mathcal{D}|\mathbf{w})p(\mathbf{w})d\mathbf{w} \tag{2.37}$$

A widely used frequentist estimator is **maximum likelihood**,in which **w** is set to the value that maximizes the likelihood function $p(\mathcal{D}|w)$.In the machine learning literature,the negative log of the likelihood function is called an **error function**.

Oner approach to determining the frequentist error bars is the **bootstrap**,in which multiple data sets are created as follows.Suppose out original data set consists of N data points.We can create a new data set X_B by drawing N points at

random from X, with **replacement**, so that some points in X may be replicated in XB, whereas other points in X may be absent from X_B. This process can be repeated L times to generate L data sets each of size N and each obtained by sampling from the original data set X. The statistical accuracy of parameter estimates can then be evaluated by looking at the variability of predictions between the different bootstrap data sets.

2.2.5 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.,

$$X \perp Y = P(X,Y) = P(X)P(Y) \tag{2.38}$$

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

$$X \perp Y|Z = P(X,Y|Z) = P(X|Z)P(Y|Z) \tag{2.39}$$

2.2.6 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**.

2.2.7 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:

$$\begin{aligned} var[X] &= \mathbb{E}[(X-\mu)^2] \qquad (2.40)\\ &= \int (x-\mu)^2 p(x)\mathrm{d}x \\ &= \int x^2 p(x)\mathrm{d}x + \mu^2 \int p(x)\mathrm{d}x - 2\mu \int xp(x)\mathrm{d}x \\ &= \mathbb{E}[X^2] - \mu^2 \qquad (2.41) \end{aligned}$$

from which we derive the useful result

$$\mathbb{E}[X^2] = \sigma^2 + \mu^2 \tag{2.42}$$

The **standard deviation** is defined as

$$std[X] \triangleq \sqrt{var[X]} \tag{2.43}$$

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

Expectations and covariances The average value of some function $f(x)$ under a probability distribution p(x) is called the expectation of f(x) and will be denoted by

$$\mathbb{E}[f] = \sum_x p(x)f(x)\mathbb{E}[f] = \int p(x)f(x)dx \tag{2.44}$$

approximation

$$\mathbb{E}[f] \simeq \frac{1}{N}\sum_{n=1}^{N} f(x_n) \tag{2.45}$$

conditional expectation with respect to a conditional distribution

$$\mathbb{E}_x[f|y] = \sum_x p(x|y)f(x) \tag{2.46}$$

variance of f(x) is defined by

$$var[f] = \mathbb{E}[(f(x) - \mathbb{E}[f(x)])^2 var[f] = \mathbb{E}[f(x)^2] - \mathbb{E}[f(x)]^2 \tag{2.47}$$

convariance

$$cov[x,y] = \mathbb{E}_{x,y}[\{x - \mathbb{E}[x]\}\{y - \mathbb{E}[y]\}] \tag{2.48}$$

In the case of two vectors of random variables **x** and **y**

$$cov[\mathbf{x},\mathbf{y}] = \mathbb{E}_{x,y}[\{\mathbf{x} - \mathbb{E}[\mathbf{x}]\}\{\mathbf{y}^T - \mathbb{E}[\mathbf{y}^T]\}]cov[\mathbf{x},\mathbf{y}] = \mathbb{E}_{x,y}[\mathbf{x}\mathbf{y}^T] - \mathbb{E}[\mathbf{x}]\mathbb{E}[\mathbf{y}^T] \tag{2.49}$$

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

$$\text{Ber}(x|\theta) \triangleq \theta^{\mathbb{I}(x=1)}(1-\theta)^{\mathbb{I}(x=0)} \tag{2.50}$$

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

$$\text{Bin}(k|n,\theta) \triangleq \binom{n}{k}\theta^k(1-\theta)^{n-k} \tag{2.51}$$

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:

$$p(\boldsymbol{x}) \triangleq \prod_{k=1}^{K} \theta_k^{\mathbb{I}(x_k=1)} \tag{2.52}$$

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

$$p(\boldsymbol{x}) \triangleq \binom{n}{x_1 \cdots x_k} \prod_{k=1}^{K} \theta_k^{x_k} \tag{2.53}$$

where $\binom{n}{x_1 \cdots x_k} \triangleq \frac{n!}{x_1! x_2! \cdots x_K!}$

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 2.1 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

Poisson distribution is a discrete probability distribution that expresses the probability of a given number of events occurring in a **fixed interval** of time and/or space if these events occur with a **known average rate** and **independently** of the time since the last event.

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

$$p(x|\lambda) = e^{-\lambda} \frac{\lambda^x}{x!} \tag{2.54}$$

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

The Poisson distribution can be derived as a **limiting case to the binomial distribution** as the number of trials goes to infinity and the expected number of successes remains fixed,known as **law of rare events**.Assume that there exists a small enough subinterval for which the probability of an event occurring twice is "negligible".Given only the information of expected number of total events in the whole interval,denoted as λ,**divide the whole interval into n subintervals** $I_1, I_2, ..., I_n$ of equal size,such that $n > \lambda$. The occurrence of an event in the whole interval can be seen

as a **Bernoulli trial**, where the ith trial corresponds to looking whether an event happens at the subinterval I_i with probability $p = \lambda/n$.Then the number of events x that occur obey **binomial distribution** $X \sim B(x;n,p)$.

$$\begin{aligned} P(x) &= B(x;n,p) \\ &= C_n^x (p)^x (1-p)^{n-x} \\ &= \frac{n!}{x!(n-x)!} p^x (1-p)^{n-x} \\ &= \frac{n(n-1)(n-2)\cdots(n-x+1)}{x!} \cdot \frac{\lambda^x}{n^x} \left(1-\frac{\lambda}{n}\right)^{n-x} \\ &= \frac{n}{n} \cdot \frac{n-1}{n} \cdots \frac{n-x+1}{n} \cdot \frac{\lambda^x}{x!} \left(1-\frac{\lambda}{n}\right)^{n-x} \\ &= \frac{n}{n} \cdot \frac{n-1}{n} \cdots \frac{n-x+1}{n} \cdot \frac{\lambda^x}{x!} \left(\left(1-\frac{\lambda}{n}\right)^{-\frac{n}{\lambda}}\right)^{-\lambda} \left(1-\frac{\lambda}{n}\right)^{-x} \end{aligned}$$

For $n \to \infty$,we have

$$(1-\frac{1}{n})(1-\frac{2}{n})\cdots(1-\frac{x-1}{n}) \qquad \to 1 \tag{2.55}$$

$$\left(\left(1-\frac{\lambda}{n}\right)^{-\frac{n}{\lambda}}\right)^{-\lambda} \qquad \to e^{-\lambda} \tag{2.56}$$

$$(1-\frac{\lambda}{n})^{-x} \qquad \to 1 \tag{2.57}$$

So we have

$$\lim_{n\to\infty} P(x) = \frac{e^{-\lambda}\lambda^x}{x!} \tag{2.58}$$

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

$$F_n(x) \triangleq \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(x_i \le x) \tag{2.59}$$

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

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.

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

Table 2.3: Summary of Gaussian distribution.

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

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.
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.More about Gaussian distribution is discussed in 2.5.

2.4.2 Student's t-distribution

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

Written as	$f(x)$	$\mathbb{E}[X]$	mode	var$[X]$
$X \sim \mathcal{T}(\mu, \sigma^2, \nu)$	$\frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})}\left[1+\frac{1}{\nu}\left(\frac{x-\mu}{\nu}\right)^2\right]$	μ	μ	$\frac{\nu\sigma^2}{\nu-2}$

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

$$\Gamma(x) \triangleq \int_0^\infty t^{x-1} e^{-t} \mathrm{d}t \tag{2.60}$$

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.

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

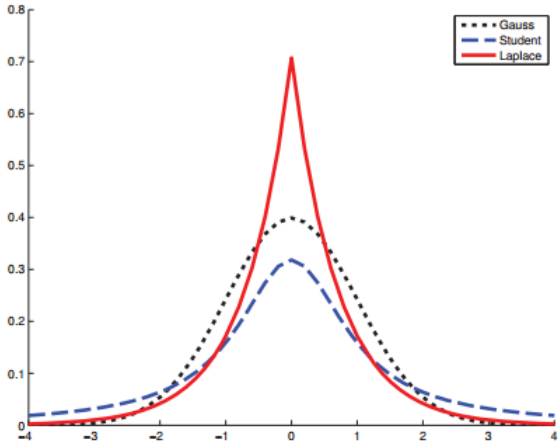
As an illustration of the robustness of the Student distribution, consider Figure 2.2. We see that the Gaussian is affected a lot, whereas the Student distribution hardly changes. This is because the Student has heavier tails, at least for small ν(see Figure 2.1).

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.

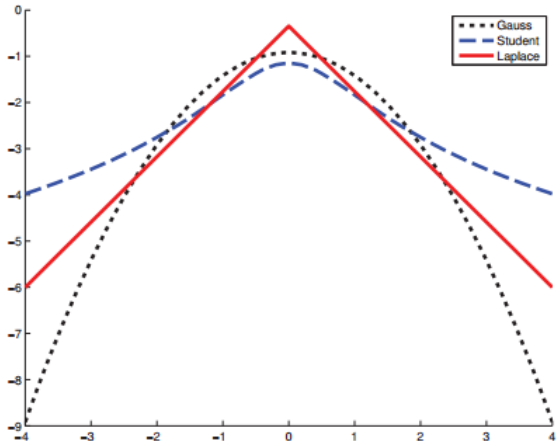
To ensure finite variance, we require $\nu > 2$. It is common to use $\nu = 4$, which gives good performance in a range of problems (Lange et al. 1989). For $\nu \gg 5$, the Student distribution rapidly approaches a Gaussian distribution and loses its robustness properties.

2.4.3 The Laplace distribution

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



(a)



(b)

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.

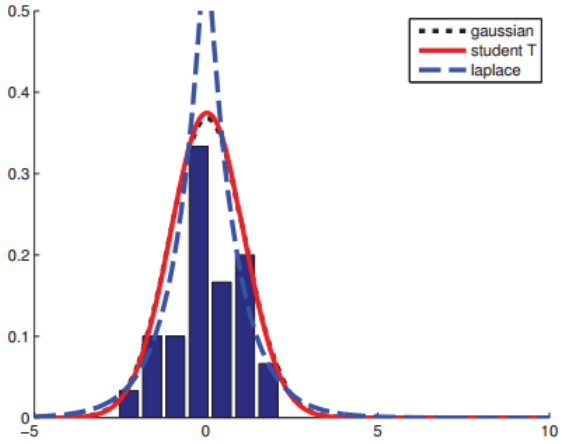
Table 2.5: Summary of Laplace distribution.

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

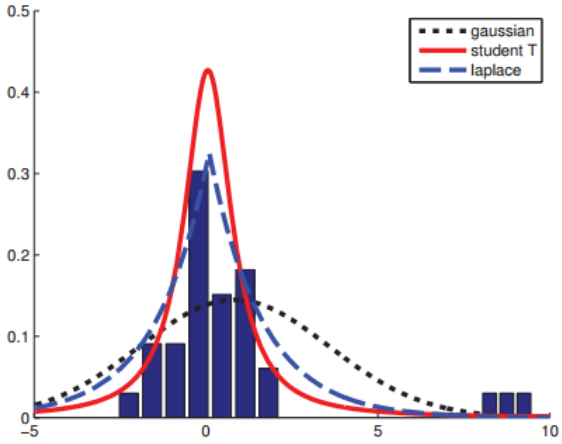
Its robustness to outliers is illustrated in Figure 2.2. It also put mores probability density at 0 than the Gaussian. This property is a useful way to encourage sparsity in a model, as we will see in Section TODO.

2.4.4 The gamma distribution

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



(a)



(b)

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.

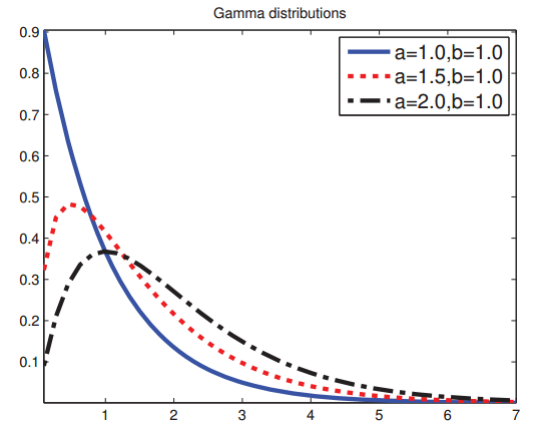
Table 2.6: Summary of gamma distribution

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

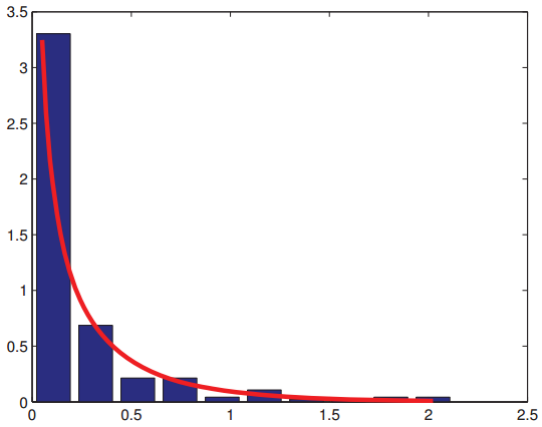
2.4.5 The beta distribution

Table 2.7: Summary of Beta distribution

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



(a)



(b)

Fig. 2.3: Some Ga$(a,b=1)$ distributions. If $a \leq 1$, the mode is at 0, otherwise it is > 0.As we increase the rate b, we reduce the horizontal scale, thus squeezing everything leftwards and upwards. (b) An empirical pdf of some rainfall data, with a fitted Gamma distribution superimposed.

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

$$B(a,b) \triangleq \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)} \tag{2.61}$$

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.

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 $k > 2$

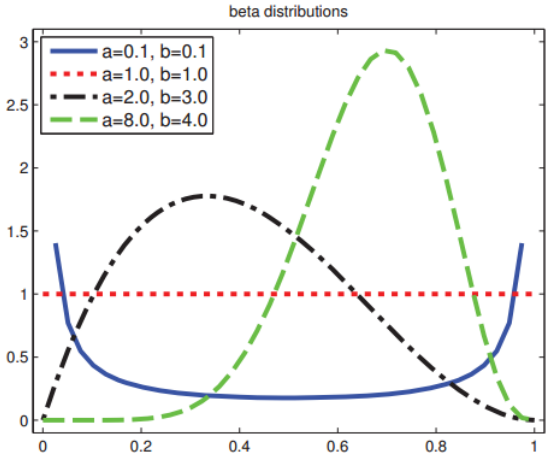


Fig. 2.4: Some beta distributions.

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

As $k \to \infty$, 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**).

2.5 The Gaussian Distribution

The Gaussian distributions,also known as the normal distributions,is a widely used model for the distribution of continuous variables.In the case of single variable x,the Gaussian is in the form

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\{-\frac{1}{(2\sigma)^2}(x-\mu)^2\} \tag{2.62}$$

where μ is the mean and σ^2 is the variance.For a D-dimensional vector $\boldsymbol{x}$,the multivariate Gaussian distribution takes the form

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\} \tag{2.63}$$

where $\boldsymbol{\mu}$ is a D-dimensional mean vector,$\boldsymbol{\Sigma}$ is a $D \times D$ covariance matrix,and $|\boldsymbol{\Sigma}|$ denotes the determinant of $\boldsymbol{\Sigma}$.

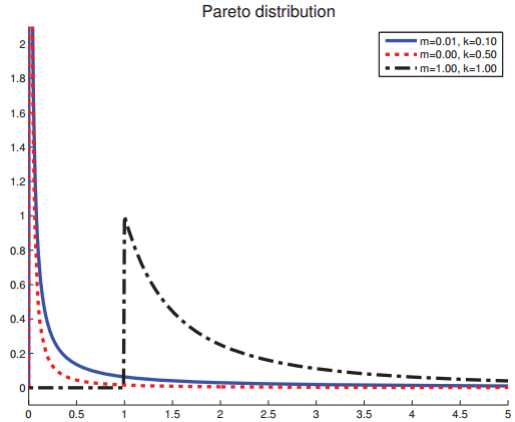
The Gaussian distribution arises in many different contexts and can be motivated from a variety of different perspectives,such as maximizing the entropy and the sum of multiple random variables.The **central limit theorem** tells us the latter.

2.5.1 Linear Transformation based on Eigenvector

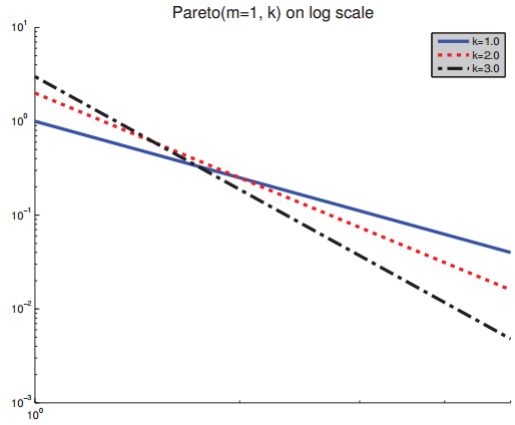
Recall from Section 2.8.2,for a $D-dimensionl$ vector $\boldsymbol{x}$,the multivariate Gaussian distribution(MVN),i.e. the **pdf** takes the form:

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\Sigma) \triangleq \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right] \tag{2.64}$$

where μ is a $D-dimensional$ mean vector,Σ is a $D \times D$ covariance matrix,and $|\Sigma|$ denotes the determinant of Σ.



(a)



(b)

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

The Gaussian distribution arises in many different contexts and can be motivated from variety of different perspectives,such as the distribution that maximizes the entropy.Another situation in which the Gaussian distribution arises is when we consider the sum of multiple random variables.The *centrallimittheorem* (due to Laplace),tells us that,subject to certain mild conditions,the sum of a set of random variables,which is of course itself a random variable,has a distribution that becomes increasingly Gaussian as the number of terms in the sum increases(Walker,1969).

The expression inside the exponent,which is the functional dependence of the Gaussian on $\boldsymbol{x}$,is through the quadratic form

$$\Delta^2 = (\boldsymbol{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{\mu}) \tag{2.65}$$

The quantity Δ is the *Mahalanobisdistance* between a data vector $\boldsymbol{x}$ and the mean vector $\boldsymbol{\mu}$ and reduces to the Euclidean distance when Σ is the identity matrix. 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

$$\boldsymbol{\Sigma}^{-1} = \boldsymbol{U}^{-T}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}^{-1} = \boldsymbol{U}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}^T = \sum_{i=1}^{D} \frac{1}{\lambda_i}\boldsymbol{u}_i\boldsymbol{u}_i^T \tag{2.66}$$

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

$$(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) = (\boldsymbol{x}-\boldsymbol{\mu})^T\left(\sum_{i=1}^{D}\frac{1}{\lambda_i}\boldsymbol{u}_i\boldsymbol{u}_i^T\right)(\boldsymbol{x}-\boldsymbol{\mu}) \tag{2.67}$$

$$= \sum_{i=1}^{D}\frac{1}{\lambda_i}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{u}_i\boldsymbol{u}_i^T(\boldsymbol{x}-\boldsymbol{\mu}) \tag{2.68}$$

$$= \sum_{i=1}^{D}\frac{y_i^2}{\lambda_i} \tag{2.69}$$

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

$$\frac{y_1^2}{\lambda_1}+\frac{y_2^2}{\lambda_2} = 1 \tag{2.70}$$

Hence we see that the contours of equal probability density of a Gaussian lie along ellipses. This is illustrated in Figure 2.6. The eigenvectors determine the orientation of the ellipse, and the eigenvalues determine how elogonated it is.

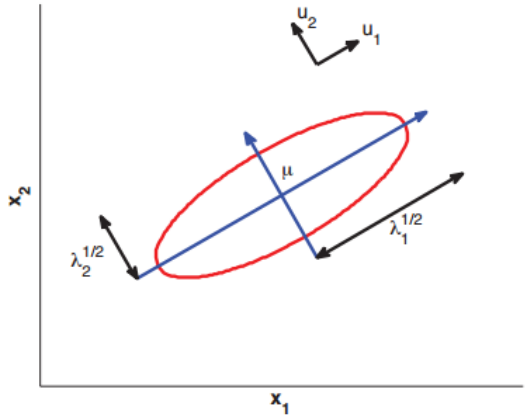


Fig. 2.6: Visualization of a 2 dimensional Gaussian density. The major and minor axes of the ellipse are defined by the first two eigenvectors of the covariance matrix, namely $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$. Based on Figure 2.7 of (Bishop 2006a)

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 $\boldsymbol{U}$.

Proof. First of all,we note that the matrix Σ can be taken to be symmetric,without loss of generality.Now we consider the eigenvector equation for the convariance matrix

$$\Sigma\boldsymbol{\mu_i} = \lambda_i\boldsymbol{\mu_i} \tag{2.71}$$

where $i = 1,...,D$.Because Σ is real,symmetric matrix its eigenvalues will be real,and its eigenvectors can be chosen to form an orthonormal set,so that

$$\boldsymbol{\mu_i}^T\boldsymbol{\mu_j} = \mathcal{I}_{ij} \tag{2.72}$$

where $\mathcal{I}_{ij}$ is the i,j element of the identity matrix and satisfies

$$\mathcal{I}_{ij} = \begin{cases} 1, \text{if i=j} \\ 0, \text{otherwise.} \end{cases} \tag{2.73}$$

The covariance matrix Σ can be expressed as an equation in terms of its eigenvectors in the form

$$\boldsymbol{\Sigma} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T \tag{2.74}$$

$$= \sum_{i=1}^{D} \lambda_i \boldsymbol{\mu_i}\boldsymbol{u_i}^{\boldsymbol{T}} \tag{2.75}$$

where $\boldsymbol{u}_i$ is the i'th column of $\boldsymbol{U}$, containing the i'th eigenvector. Similarly,the inverse covariance matrix Σ^{-1} can be expressed as

$$\boldsymbol{\Sigma}^{-1} = \sum_{i=1}^{D} \frac{1}{\lambda_i}\boldsymbol{u_i}\boldsymbol{u_i}^T \tag{2.76}$$

Substituting 2.76 into 2.65 ,the quadratic form becomes

$$\Delta^2 = \sum_{i=1}^{D} \frac{y_i^2}{\lambda_i} \tag{2.77}$$

where we have defined

$$y_i = \boldsymbol{u_i}^T(\boldsymbol{x}-\boldsymbol{u}) \tag{2.78}$$

We can interpret $\{y_i\}$ as a new coordinate system defined by the orthonormal vectors $\boldsymbol{u_i}$ that are shifted and rotated with respect to the original x_i coordinates.Forming the vector $\boldsymbol{y} = (y_1,...,y_D)^T$,we have

$$\boldsymbol{y} = \boldsymbol{U}(\boldsymbol{x}-\boldsymbol{\mu}) \tag{2.79}$$

where

$$\boldsymbol{U} = \begin{bmatrix}\boldsymbol{u_1}^T & \boldsymbol{u_2}^T & ... & \boldsymbol{u_n}^T\end{bmatrix} \tag{2.80}$$

$\boldsymbol{U}$ is an *orthogonal* matrix. A matrix whose eigenvalue are strictly positive is said to be **positive definite**, and if all the eigenvalues are nonnegative,then the covariance matrix is said to be **positive semidefinite**.

Now consider the form of Gaussian distribution in the new coordinate system defined by the y_i.In going from $\boldsymbol{x}$ to $\boldsymbol{y}$ coordinate system,we have a **Jacobian matrix** $\boldsymbol{J}$ with elements given by

$$J_{ij} = \frac{\partial x_i}{\partial y_i} = U_{ji} \tag{2.81}$$

$$\boldsymbol{J} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{y}} \tag{2.82}$$

where U_{ji} are the elements of the matrix $\boldsymbol{U}^T$.Using the orthonormality property of the matrix $\boldsymbol{U}$, we see that the square of the determinant of the Jacobian matrix is

$$|\boldsymbol{J}|^2 = |\boldsymbol{I}| = 1 \tag{2.83}$$

Also,the determinant $|\boldsymbol{\Sigma}|$ of the covariance matrix can be written as the product of its eigenvalues,and hence

$$|\boldsymbol{\Sigma}|^{1/2} = \prod_{j=1}^{D} \lambda_j^{1/2} \tag{2.84}$$

Thus in the y_i coordinate system,the Gaussian distribution takes the form

$$p(\boldsymbol{y}) = p(\boldsymbol{x})|\boldsymbol{J}| = \prod_{j=1}^{D} \frac{1}{(2\pi\lambda_j)^{1/2}} \exp\{-\frac{y_j^2}{2\lambda_j}\} \tag{2.85}$$

which is the product of D independent univariate Gaussian distributions.So we have normalized the multivariate Gaussian.

Now we can check the first and second order moments of the Gaussian.

2.5.1.1 limitations

Gaussian distribution has quadratically growing parameters with dimension.A further limitation of the Gaussian distribution is that it is intrinsically unimodal(i.e.,has a single maximum) and so is unable to provide a good approximation to multimodal distributions.We will introduce *latent* variables,also called *hidden* variables or *unobserved* variables to address both of these problems.

2.5.1.2 MLE for a MVN

Theorem 2.1. ***(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*

$$\bar{\boldsymbol{\mu}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i \triangleq \bar{\boldsymbol{x}} \tag{2.86}$$

$$\bar{\boldsymbol{\Sigma}} = \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T \tag{2.87}$$

$$= \frac{1}{N}\left(\sum_{i=1}^{N} \boldsymbol{x}_i \boldsymbol{x}_i^T\right) - \bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T \tag{2.88}$$

2.5.1.3 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

$$f(\boldsymbol{x}) = \frac{1}{Z}\exp\left(-\frac{1}{2}\boldsymbol{x}^T \boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right) \tag{2.89}$$

2.5.2 Conditional Gaussian distributions

Theorem 2.2. *If two sets of variables are jointly multivariate Gaussian, then the conditional distribution of one set conditioned on the other is again Gaussian. Similarly, the marginal distribution of either set is also Gaussian.*

The multivariate normal distribution is given by

$$\frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}}\exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right) \tag{2.90}$$

Write $\boldsymbol{x}$ as

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_a \\ \boldsymbol{x}_b \end{bmatrix}, \boldsymbol{x} = \begin{bmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{bmatrix} \tag{2.91}$$

Consider the conditional distribution $p(\boldsymbol{x}_a|\boldsymbol{x}_b)$ and marginal distribution $p(\boldsymbol{x}_a)$. Separate the components of the covariance matrix $\boldsymbol{\Sigma}$ into partitioned block matrix

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{bmatrix} \tag{2.92}$$

where $\boldsymbol{\Sigma}, \boldsymbol{\Sigma}_{aa}, \boldsymbol{\Sigma}_{bb}$ are all symmetric and $\boldsymbol{\Sigma}_{ab} = \boldsymbol{\Sigma}_{ab}^T$.

From the product rule of probability, we see that this conditional distribution can be evaluated from the joint distribution $p(\boldsymbol{x}) = p(\boldsymbol{x}_a, \boldsymbol{x}_b)$ simple by fixing $\boldsymbol{x}_b$ to the observed value and **normalizing** the resulting expression to obtain a valid probability distribution over $\boldsymbol{x}_a$.To evaluate the conditional probability from the joint probability,for any arbitrary $\boldsymbol{x}_b$,we can use **product and sum** rule of probability

$$p(\boldsymbol{x}_b) = \int p(\boldsymbol{x}_a, \boldsymbol{x}_b) d\boldsymbol{x}_a \tag{2.93}$$

$$p(\boldsymbol{x}_a \boldsymbol{x}_b) = \frac{p(\boldsymbol{x}_a, \boldsymbol{x}_b)}{p(\boldsymbol{x}_b)} \tag{2.94}$$

Because $p(\boldsymbol{x}_b)$ is observed value,so we have

$$p(\boldsymbol{x}_a \boldsymbol{x}_b) \sim \frac{p(\boldsymbol{x}_a, \boldsymbol{x}_b)}{p(\boldsymbol{x}_b)} \tag{2.95}$$

$$\sim p(\boldsymbol{x}_a, \boldsymbol{x}_b) \tag{2.96}$$

which has the **exponential quadratic form**,and hence the corresponding conditional distribution will be Gaussian.

$$-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) \tag{2.97}$$

$$= -\frac{1}{2}\left[\, (\boldsymbol{x}_a-\boldsymbol{\mu}_a)^T \;\; (\boldsymbol{x}_b-\boldsymbol{\mu}_b)^T \,\right] \begin{bmatrix} \boldsymbol{\Lambda}_{aa} & \boldsymbol{\Lambda}_{ab} \\ \boldsymbol{\Lambda}_{ba} & \boldsymbol{\Lambda}_{bb} \end{bmatrix} \begin{bmatrix} (\boldsymbol{x}_a-\boldsymbol{\mu}_a) \\ (\boldsymbol{x}_b-\boldsymbol{\mu}_b) \end{bmatrix} \tag{2.98}$$

$$= -\frac{1}{2}\left[\, (\boldsymbol{x}_a-\boldsymbol{\mu}_a)^T\boldsymbol{\Lambda}_{aa} + (\boldsymbol{x}_b-\boldsymbol{\mu}_b)^T\boldsymbol{\Lambda}_{ba} \;\; (\boldsymbol{x}_a-\boldsymbol{\mu}_a)^T\boldsymbol{\Lambda}_{ab} + (\boldsymbol{x}_b-\boldsymbol{\mu}_b)^T\boldsymbol{\Lambda}_{bb} \,\right] \begin{bmatrix} (\boldsymbol{x}_a-\boldsymbol{\mu}_a) \\ (\boldsymbol{x}_b-\boldsymbol{\mu}_b) \end{bmatrix} \tag{2.99}$$

$$= -\frac{1}{2}((\boldsymbol{x}_a-\boldsymbol{\mu}_a)^T\boldsymbol{\Lambda}_{aa}(\boldsymbol{x}_a-\boldsymbol{\mu}_a) + (\boldsymbol{x}_b-\boldsymbol{\mu}_b)^T\boldsymbol{\Lambda}_{ba}(\boldsymbol{x}_a-\boldsymbol{\mu}_a) \tag{2.100}$$

$$+ (\boldsymbol{x}_a-\boldsymbol{\mu}_a)^T\boldsymbol{\Lambda}_{ab}(\boldsymbol{x}_b-\boldsymbol{\mu}_b) + (\boldsymbol{x}_b-\boldsymbol{\mu}_b)^T\boldsymbol{\Lambda}_{bb}(\boldsymbol{x}_b-\boldsymbol{\mu}_b)) \tag{2.101}$$

With the operation called 'completing the square' of the exponent form in a General Gaussian distribution $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$

$$-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) = -\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{\Sigma}\boldsymbol{\mu} + \text{const} \tag{2.102}$$

Based on this,we can apply **method of undetermined coefficients** to determinate the super parameters.Covariance matrix is in the corresponding second-order term in $\boldsymbol{x}_a$,and mean vector is in the corresponding linear term in $\boldsymbol{x}_a$.

We need to make use of **Schur complement** for the inverse covariance matrix(**precision matrix**)

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{M} & -\boldsymbol{M}\boldsymbol{B}\boldsymbol{D}^{-1} \\ -\boldsymbol{D}^{-1}\boldsymbol{C}\boldsymbol{M} & \boldsymbol{D}^{-1}\boldsymbol{C}\boldsymbol{M}\boldsymbol{B}\boldsymbol{D}^{-1} \end{bmatrix}^{-1} \tag{2.103}$$

$$\tag{2.104}$$

where

$$\boldsymbol{M} = (\boldsymbol{A} - \boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{C})^{-1} \tag{2.105}$$

Then the precision matrix

$$\begin{bmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{\Lambda}_{aa} & \boldsymbol{\Lambda}_{ab} \\ \boldsymbol{\Lambda}_{ba} & \boldsymbol{\Lambda}_{bb} \end{bmatrix} \tag{2.106}$$

can be evaluated.

2.5.3 Marginal Gaussian distributions

Wrapping up,given joint Gaussian distribution,we can obtain partitioned Gaussians. Conditional distribution:

$$p(\boldsymbol{x}_a|\boldsymbol{x}_b) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_{a|b}, \boldsymbol{\Lambda}_{aa}^{-1}) \tag{2.107}$$

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a - \boldsymbol{\Lambda}_{aa}^{-1}\boldsymbol{\Lambda}_{ab}(\boldsymbol{x}_b - \boldsymbol{\mu}_b) \tag{2.108}$$

Marginal distribution:

$$p(\boldsymbol{x}_a) = \mathcal{N}(\boldsymbol{x}_a|\boldsymbol{\mu}_a, \boldsymbol{\Sigma}_{aa}) \tag{2.109}$$

2.5.4 Baye's theorem for Gaussian variables

2.6 The exponential family

Before defining the exponential family, we mention several reasons why it is important:

- It can be shown that, under certain regularity conditions, the exponential family is the only family of distributions with finite-sized sufficient statistics, meaning that we can compress the data into a fixed-sized summary without loss of information. This is particularly useful for online learning, as we will see later.
- The exponential family is the only family of distributions for which conjugate priors exist, which simplifies the computation of the posterior (see Section 2.6.8).
- The exponential family can be shown to be the family of distributions that makes the least set of assumptions subject to some user-chosen constraints (see Section 2.6.9).
- The exponential family is at the core of generalized linear models, as discussed in Section **??**.
- The exponential family is at the core of variational inference, as discussed in Section TODO.

2.6.1 Definition

A pdf or pmf $p(\boldsymbol{x}|\boldsymbol{\theta})$,for $\boldsymbol{x} \in \mathbb{R}^m$ and $\boldsymbol{\theta} \in \mathbb{R}^D$, is said to be in the **exponential family** if it is of the form

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = h(\boldsymbol{x})g(\boldsymbol{\theta})\exp\{\boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{x})\} \tag{2.110}$$

$$= \frac{1}{Z(\boldsymbol{\theta})}h(\boldsymbol{x})\exp[\boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{x})] \tag{2.111}$$

$$= h(\boldsymbol{x})\exp[\boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{x}) - A(\boldsymbol{\theta})] \tag{2.112}$$

where

$$Z(\boldsymbol{\theta}) = \int h(\boldsymbol{x})\exp[\boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{x})]\mathrm{d}\boldsymbol{x} \tag{2.113}$$

$$A(\boldsymbol{\theta}) = \log Z(\boldsymbol{\theta}) \tag{2.114}$$

Here $\boldsymbol{\theta}$ are called the **natural parameters** or **canonical parameters**, $\boldsymbol{\phi}(\boldsymbol{x}) \in \mathbb{R}^D$ is called a vector of **sufficient statistics**, $Z(\boldsymbol{\theta})$ is called the **partition function**, $A(\boldsymbol{\theta})$ is called the **log partition function** or **cumulant function**, and $h(\boldsymbol{x})$ is the a scaling constant, often 1. If $\boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{x}$, we say it is a **natural exponential family**.

Equation 2.112 can be generalized by writing

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = h(\boldsymbol{x})\exp[\eta(\boldsymbol{\theta})^T\boldsymbol{\phi}(\boldsymbol{x}) - A(\eta(\boldsymbol{\theta}))] \tag{2.115}$$

where η is a function that maps the parameters $\boldsymbol{\theta}$ to the canonical parameters $\boldsymbol{\eta} = \eta(\boldsymbol{\theta})$.If $\dim(\boldsymbol{\theta}) < \dim(\eta(\boldsymbol{\theta}))$, it is called a **curved exponential family**, which means we have more sufficient statistics than parameters. If $\eta(\boldsymbol{\theta}) = \boldsymbol{\theta}$, the model is said to be in **canonical form**. We will assume models are in canonical form unless we state otherwise.

2.6.2 Maximum likelihood and sufficient statistics

2.6.3 Conjugate priors

2.6.4 Noninformative prios

In many cases,however,we may have little idea of what form the distribution should take.We may then seek a form of prior distribution,called a **noninformative prior**,which is intended to have little influence on the posterior distribution as possible(Jeffries,1946;Box and Tao,1973;Bernardo and Smith,1994).

Suppose we have a distribution $p(x|\lambda)$ governed by a parameter λ,we might be tempted to propose a prior distribution $p(\lambda)$ =const as a suitable prior.Improper priors are ones in which the domain λ is unbounded and the prior distribution cannot be correctly normalized because the integral over λ diverges.A second difficulty arises from the transformation behaviour of a probability density under a nonlinear change of variables.For example

$$p_\eta(\eta) = p_\lambda(\lambda)\left|\frac{d\lambda}{d\eta}\right| = p_\lambda(\eta^2)\eta \propto \eta \tag{2.116}$$

so the density over η will not be constant.

Here are two simple examples of noninformative priors(Berger,1985). First of all,if a density takes the form

$$p(x|\mu) = f(x-\mu) \tag{2.117}$$

then the parameter μ is known as a **location parameter**.This family of densities exhibits **translation invariance** because if we shift x by a constant to give $\hat{x} = x+c$,then

$$p(\hat{x}|\hat{\mu}) = f(\hat{x}-\hat{\mu}) \tag{2.118}$$

where $\hat{\mu} = \mu + c$.

A second example,

$$p(x|\sigma) = \frac{1}{\sigma}f(\frac{x}{\sigma}) \tag{2.119}$$

where $\sigma > 0$.Note that this will be a normalized density provided f(x) is correctly normalized.The parameter σ is known as a **scale parameter**,and the density exhibits **scale invariance**.

2.6.5 Examples

2.6.5.1 Bernoulli

The Bernoulli for $x \in \{0,1\}$ can be written in exponential family form as follows:

$$\begin{aligned}\text{Ber}(x|\mu) &= \mu^x(1-\mu)^{1-x} \\ &= \exp[x\log\mu + (1-x)\log(1-\mu)]\end{aligned} \tag{2.120}$$

where $\phi(x) = (\mathbb{I}(x=0), \mathbb{I}(x=1))$ and $\boldsymbol{\theta} = (\log\mu, \log(1-\mu))$.

However, this representation is **over-complete** since $\mathbf{1}^T\phi(x) = \mathbb{I}(x=0) + \mathbb{I}(x=1) = 1$. Consequently $\boldsymbol{\theta}$ is not uniquely identifiable. It is common to require that the representation be **minimal**, which means there is a unique θ

associated with the distribution. In this case, we can just define

$$\text{Ber}(x|\mu) = (1-\mu)\exp\left(x\log\frac{\mu}{1-\mu}\right) \tag{2.121}$$

$$\text{where } \phi(x) = x, \theta = \log\frac{\mu}{1-\mu}, Z = \frac{1}{1-\mu}$$

We can recover the mean parameter μ from the canonical parameter using

$$\mu = \text{sigm}(\theta) = \frac{1}{1+e^{-\theta}} \tag{2.122}$$

2.6.5.2 Multinoulli

We can represent the multinoulli as a minimal exponential family as follows:

$$\begin{aligned}
\text{Cat}(\boldsymbol{x}|\boldsymbol{\mu}) &= \prod_{k=1}^{K} = \exp\left(\sum_{k=1}^{K} x_k \log\mu_k\right) \\
&= \exp\left[\sum_{k=1}^{K-1} x_k \log\mu_k + (1-\sum_{k=1}^{K-1} x_k)\log(1-\sum_{k=1}^{K-1}\mu_k)\right] \\
&= \exp\left[\sum_{k=1}^{K-1} x_k \log\frac{\mu_k}{1-\sum_{k=1}^{K-1}\mu_k} + \log(1-\sum_{k=1}^{K-1}\mu_k)\right] \\
&= \exp\left[\sum_{k=1}^{K-1} x_k \log\frac{\mu_k}{\mu_K} + \log\mu_K\right], \text{where } \mu_K \triangleq 1-\sum_{k=1}^{K-1}\mu_k
\end{aligned}$$

We can write this in exponential family form as follows:

$$\text{Cat}(\boldsymbol{x}|\boldsymbol{\mu}) = \exp[\boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{x}) - A(\boldsymbol{\theta})] \tag{2.123}$$

$$\boldsymbol{\theta} \triangleq (\log\frac{\mu_1}{\mu_K}, \cdots, \log\frac{\mu_{K-1}}{\mu_K}) \tag{2.124}$$

$$\boldsymbol{\phi}(\boldsymbol{x}) \triangleq (x_1, \cdots, x_{K-1}) \tag{2.125}$$

We can recover the mean parameters from the canonical parameters using

$$\mu_k = \frac{e^{\theta_k}}{1+\sum_{j=1}^{K-1} e^{\theta_j}} \tag{2.126}$$

$$\mu_K = 1 - \frac{\sum_{j=1}^{K-1} e^{\theta_j}}{1+\sum_{j=1}^{K-1} e^{\theta_j}} = \frac{1}{1+\sum_{j=1}^{K-1} e^{\theta_j}} \tag{2.127}$$

and hence

$$A(\boldsymbol{\theta}] = -\log\mu_K = \log(1+\sum_{j=1}^{K-1} e^{\theta_j}) \tag{2.128}$$

2.6.5.3 Univariate Gaussian

The univariate Gaussian can be written in exponential family form as follows:

$$\begin{aligned}\mathcal{N}(x|\mu,\sigma^2) &= \frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{1}{2\sigma^2}(x-\mu)^2\right] \\ &= \frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{1}{2\sigma^2}x^2+\frac{\mu}{\sigma^2}x-\frac{1}{2\sigma^2}\mu^2\right] \\ &= \frac{1}{Z(\boldsymbol{\theta})}\exp[\boldsymbol{\theta}^T\boldsymbol{\phi}(x)] \end{aligned} \tag{2.129}$$

where

$$\boldsymbol{\theta} = (\frac{\mu}{\sigma^2}, -\frac{1}{2\sigma^2}) \tag{2.130}$$

$$\boldsymbol{\phi}(x) = (x, x^2) \tag{2.131}$$

$$Z(\boldsymbol{\theta}) = \sqrt{2\pi}\sigma\exp(\frac{\mu^2}{2\sigma^2}) \tag{2.132}$$

2.6.5.4 Non-examples

Not all distributions of interest belong to the exponential family. For example, the uniform distribution,$X \sim U(a,b)$, does not, since the support of the distribution depends on the parameters. Also, the Student T distribution (Section TODO) does not belong, since it does not have the required form.

2.6.6 Log partition function

An important property of the exponential family is that derivatives of the log partition function can be used to generate **cumulants** of the sufficient statistics.[7] For this reason, $A(\boldsymbol{\theta})$ is sometimes called a **cumulant function**. We will prove this for a 1-parameter distribution; this can be generalized to a K-parameter distribution in a straightforward way. For the first derivative we have

For the second derivative we have

$$\begin{aligned}\frac{\mathrm{d}A}{\mathrm{d}\theta} &= \frac{\mathrm{d}}{\mathrm{d}\theta}\left\{\log\int\exp[\theta\phi(x)]\,h(x)\mathrm{d}x\right\} \\ &= \frac{\frac{\mathrm{d}}{\mathrm{d}\theta}\int\exp[\theta\phi(x)]\,h(x)\mathrm{d}x}{\int\exp[\theta\phi(x)]\,h(x)\mathrm{d}x} \\ &= \frac{\int\phi(x)exp[\theta\phi(x)]\,h(x)\mathrm{d}x}{\exp(A(\theta))} \\ &= \int\phi(x)\exp[\theta\phi(x)-A(\theta)]\,h(x)\mathrm{d}x \\ &= \int\phi(x)p(x)\mathrm{d}x = \mathbb{E}[\phi(x)] \end{aligned} \tag{2.133}$$

For the second derivative we have

[7] The first and second cumulants of a distribution are its mean $\mathbb{E}[X]$ and variance $\mathrm{var}[X]$, whereas the first and second moments are its mean $\mathbb{E}[X]$ and $\mathbb{E}[X^2]$.

$$\begin{aligned}\frac{\mathrm{d}^2 A}{\mathrm{d}\theta^2} &= \int \phi(x)\exp\left[\theta\phi(x) - A(\theta)\right] h(x)\left[\phi(x) - A'(\theta)\right]\mathrm{d}x \\ &= \int \phi(x) p(x)\left[\phi(x) - A'(\theta)\right]\mathrm{d}x \\ &= \int \phi^2(x) p(x)\mathrm{d}x - A'(\theta)\int \phi(x) p(x)\mathrm{d}x \\ &= \mathbb{E}[\phi^2(x)] - \mathbb{E}[\phi(x)]^2 = \mathrm{var}[\phi(x)] \end{aligned} \tag{2.134}$$

In the multivariate case, we have that

$$\frac{\partial^2 A}{\partial\theta_i \partial\theta_j} = \mathbb{E}[\phi_i(x)\phi_j(x)] - \mathbb{E}[\phi_i(x)]\mathbb{E}[\phi_j(x)] \tag{2.135}$$

and hence

$$\nabla^2 A(\boldsymbol{\theta}) = \mathrm{cov}[\boldsymbol{\phi}(\boldsymbol{x})] \tag{2.136}$$

Since the covariance is positive definite, we see that $A(\boldsymbol{\theta})$ is a convex function (see Section B.1).

2.6.7 MLE for the exponential family

The likelihood of an exponential family model has the form

$$p(\mathcal{D}|\boldsymbol{\theta}) = \left[\prod_{i=1}^N h(\boldsymbol{x}_i)\right] g(\boldsymbol{\theta})^N \exp\left[\boldsymbol{\theta}^T \left(\sum_{i=1}^N \boldsymbol{\phi}(\boldsymbol{x}_i)\right)\right] \tag{2.137}$$

We see that the sufficient statistics are N and

$$\boldsymbol{\phi}(\mathcal{D}) = \sum_{i=1}^N \boldsymbol{\phi}(\boldsymbol{x}_i) = (\sum_{i=1}^N \phi_1(\boldsymbol{x}_i), \cdots, \sum_{i=1}^N \phi_K(\boldsymbol{x}_i)) \tag{2.138}$$

The **Pitman-Koopman-Darmois theorem** states that, under certain regularity conditions, the exponential family is the only family of distributions with finite sufficient statistics. (Here, finite means of a size independent of the size of the data set.)

One of the conditions required in this theorem is that the support of the distribution not be dependent on the parameter.

2.6.8 Bayes for the exponential family

TODO

2.6.8.1 Likelihood

2.6.9 Maximum entropy derivation of the exponential family *

2.7 Nonparametric Methods

2.7.1 Kernel density estimators

2.7.2 Nearest-neighbour methods

2.8 Joint probability distributions

Given a **multivariate random variable** or **random vector** [8] $X \in \mathbb{R}^D$, the **joint probability distribution**[9] 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).

2.8.1 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

$$\begin{aligned} \text{cov}[X,Y] &\triangleq \mathbb{E}\left[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])\right] \\ &= \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y] \end{aligned} \tag{2.139}$$

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

$$\text{cov}[\boldsymbol{x}] \triangleq \mathbb{E}\left[(\boldsymbol{x} - \mathbb{E}[\boldsymbol{x}])(\boldsymbol{x} - \mathbb{E}[\boldsymbol{x}])^T\right] \tag{2.140}$$

$$= \begin{pmatrix} \text{var}[\boldsymbol{x_1}] & \text{Cov}[\boldsymbol{x_1}, \boldsymbol{x_2}] & \cdots & \text{Cov}[\boldsymbol{x_1}, \boldsymbol{x_D}] \\ \text{Cov}[\boldsymbol{x_2}, \boldsymbol{x_1}] & \text{var}[x_2] & \cdots & \text{Cov}[\boldsymbol{x_2}, \boldsymbol{x_D}] \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}[\boldsymbol{x_D}, \boldsymbol{x_1}] & \text{Cov}[\boldsymbol{x_D}, \boldsymbol{x_2}] & \cdots & \text{var}[\boldsymbol{x_D}] \end{pmatrix} \tag{2.141}$$

For

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

$$\text{corr}[X,Y] \triangleq \frac{\text{Cov}[X,Y]}{\sqrt{\text{var}[X], \text{var}[Y]}} \tag{2.142}$$

A **correlation matrix** has the form

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

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

$$\mathbf{R} \triangleq \begin{pmatrix} \text{corr}[X_1,X_1] & \text{corr}[X_1,X_2] & \cdots & \text{corr}[X_1,X_D] \\ \text{corr}[X_2,X_1] & \text{corr}[X_2,X_2] & \cdots & \text{corr}[X_2,X_D] \\ \vdots & \vdots & \ddots & \vdots \\ \text{corr}[X_D,X_1] & \text{corr}[X_D,X_2] & \cdots & \text{corr}[X_D,X_D] \end{pmatrix} \tag{2.143}$$

The correlation coefficient can viewed as a degree of linearity between X and Y, see Figure 2.7.

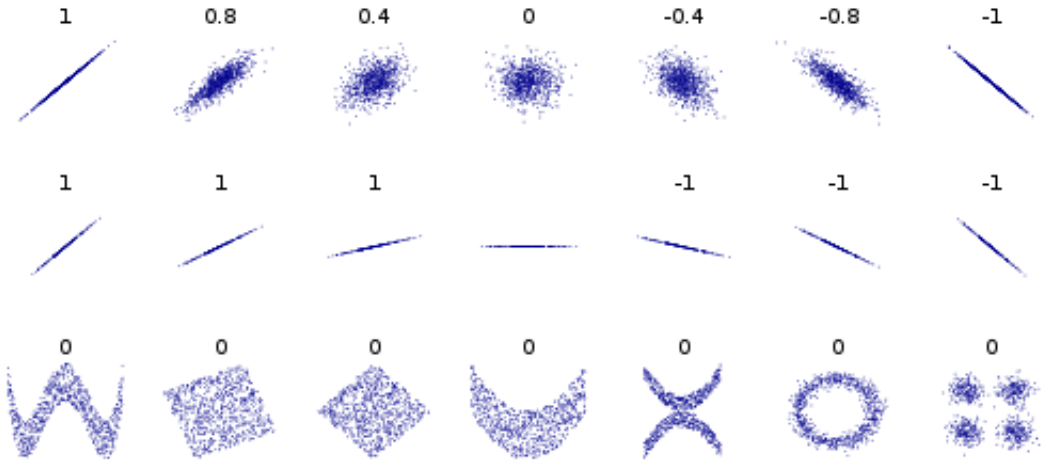


Fig. 2.7: Several sets of (x,y) points, with the Pearson correlation coefficient of x and y for each set. Note that the correlation reflects the noisiness and direction of a linear relationship (top row), but not the slope of that relationship (middle), nor many aspects of nonlinear relationships (bottom). 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 $\text{corr}[X,Y] = 0$. Some striking examples of this fact are shown in Figure 2.7. 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.

2.8.2 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:

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\Sigma) \triangleq \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right] \tag{2.144}$$

where $\boldsymbol{\mu} = \mathbb{E}[X] \in \mathbb{R}^D$ is the mean vector, and $\Sigma = \text{Cov}[X]$ 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 2.8 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.

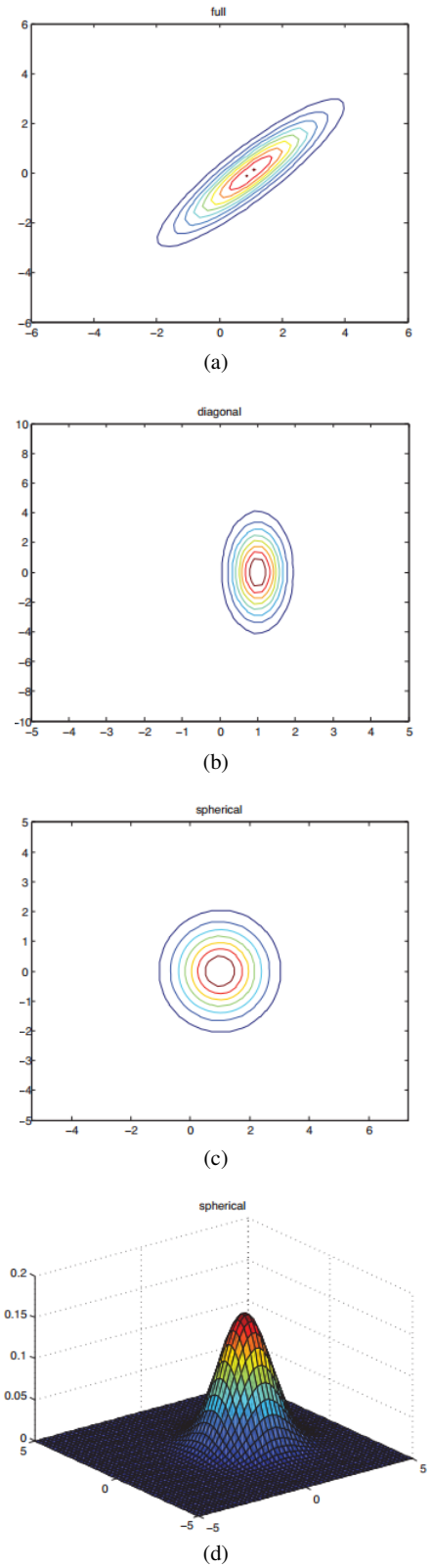


Fig. 2.8: We show the level sets for 2d Gaussians. (a) A full covariance matrix has elliptical contours.(b) A diagonal covariance matrix is an axis aligned ellipse. (c) A spherical covariance matrix has a circular shape. (d) Surface plot for the spherical Gaussian in (c).

2.8.3 Multivariate Student's t-distribution

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

$$\mathcal{T}(x|\boldsymbol{\mu},\Sigma,\nu)$$
$$\triangleq \frac{\Gamma(\frac{\nu+D}{2})}{\Gamma(\frac{\nu}{2})}\frac{|\Sigma|^{-\frac{1}{2}}}{(\nu\pi)^{\frac{D}{2}}}\left[1+\frac{1}{\nu}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]^{-\frac{\nu+D}{2}} \tag{2.145}$$
$$= \frac{\Gamma(\frac{\nu+D}{2})}{\Gamma(\frac{\nu}{2})}\frac{|\Sigma|^{-\frac{1}{2}}}{(\nu\pi)^{\frac{D}{2}}}\left[1+(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{V}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]^{-\frac{\nu+D}{2}} \tag{2.146}$$

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

$$\text{mean}=\boldsymbol{\mu}\ ,\ \text{mode}=\boldsymbol{\mu}\ ,\ \text{Cov}=\frac{\nu}{\nu-2}\Sigma \tag{2.147}$$

2.8.4 Dirichlet distribution

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

$$S_K=\left\{\boldsymbol{x}:0\le x_k\le 1,\sum_{k=1}^{K}x_k=1\right\} \tag{2.148}$$

The pdf is defined as follows:

$$\text{Dir}(\boldsymbol{x}|\boldsymbol{\alpha})\triangleq\frac{1}{B(\boldsymbol{\alpha})}\prod_{k=1}^{K}x_k^{\alpha_k-1}\mathbb{I}(\boldsymbol{x}\in S_K) \tag{2.149}$$

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

$$B(\alpha)\triangleq\frac{\prod_{k=1}^{K}\Gamma(\alpha_k)}{\Gamma(\alpha_0)}\ \text{where}\ \alpha_0\triangleq\sum_{k=1}^{K}\alpha_k \tag{2.150}$$

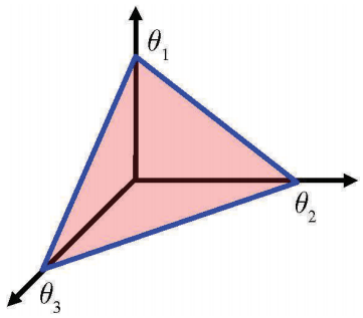
Figure 2.9 shows some plots of the Dirichlet when $K=3$, and Figure 2.10 for some sampled probability vectors. We see that α_0 controls the strength of the distribution (how peaked it is), and thekcontrol where the peak occurs. For example, $\text{Dir}(1,1,1)$ is a uniform distribution, $\text{Dir}(2,2,2)$ is a broad distribution centered at $(1/3,1/3,1/3)$, and $\text{Dir}(20,20,20)$ 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.

For future reference, the distribution has these properties

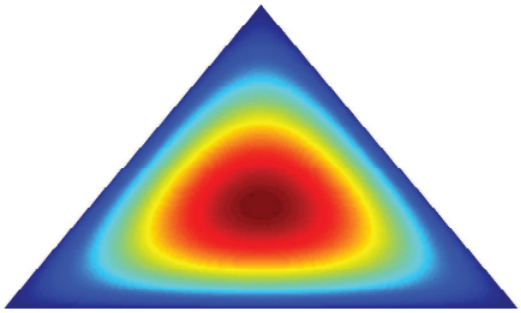
$$\mathbb{E}(x_k)=\frac{\alpha_k}{\alpha_0},\ \text{mode}[x_k]=\frac{\alpha_k-1}{\alpha_0-K},\ \text{var}[x_k]=\frac{\alpha_k(\alpha_0-\alpha_k)}{\alpha_0^2(\alpha_0+1)} \tag{2.151}$$

2.9 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.

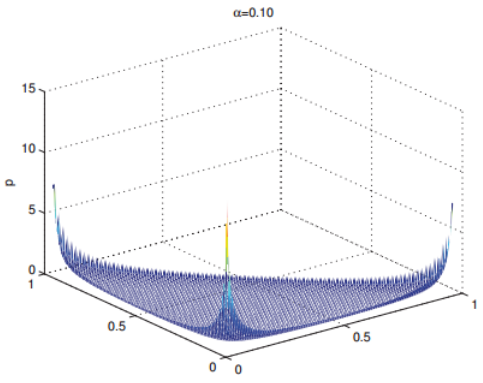


(a)



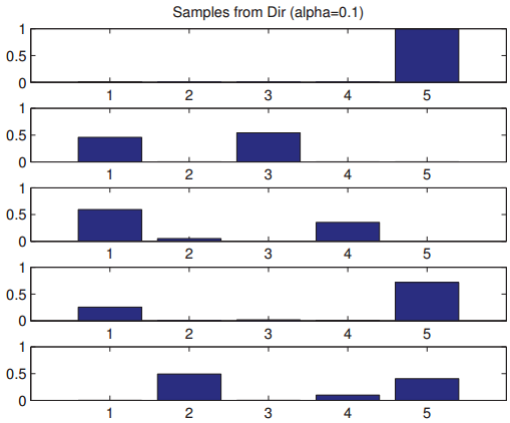
(b)

(c)

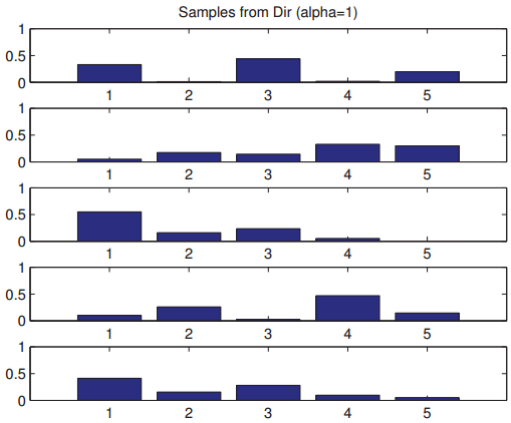


(d)

Fig. 2.9: (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) $\boldsymbol{\alpha} = (20,2,2)$.



(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.

Fig. 2.10: Samples from a 5-dimensional symmetric Dirichlet distribution for different parameter values.

2.9.1 Linear transformations

Suppose $g()$ is a linear function:

$$g(\boldsymbol{x}) = A\boldsymbol{x} + b \tag{2.152}$$

First, for the mean, we have

$$\mathbb{E}[\boldsymbol{y}] = \mathbb{E}[A\boldsymbol{x} + b] = A\mathbb{E}[\boldsymbol{x}] + b \tag{2.153}$$

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

For the covariance, we have

$$\text{Cov}[\boldsymbol{y}] = \text{Cov}[A\boldsymbol{x} + b] = A\Sigma A^T \tag{2.154}$$

2.9.2 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 $f(x) = y$:

$$p_Y(y) = \sum_{x:g(x)=y} p_X(x) \tag{2.155}$$

If X is continuous, we cannot use Equation 2.155 since $p_X(x)$ is a density, not a pmf, and we cannot sum up densities. Instead, we work with cdfs, and write

$$F_Y(y) = P(Y \leq y) = P(g(X) \leq y) = \int_{g(X) \leq y} f_X(x)\mathrm{d}x \tag{2.156}$$

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

$$f_Y(y) = f_X(x)|\frac{dx}{dy}| \tag{2.157}$$

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 $p_Y(y) = \frac{1}{2}y^{-\frac{1}{2}}$.

2.9.2.1 Multivariate change of variables *

Let f be a function $f : \mathbb{R}^n \to \mathbb{R}^n$, and let $\boldsymbol{y} = f(\boldsymbol{x})$. Then its Jacobian matrix $\boldsymbol{J}$ is given by

$$\boldsymbol{J}_{\boldsymbol{x}\to\boldsymbol{y}} \triangleq \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} \triangleq \begin{pmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \vdots & \vdots \\ \frac{\partial y_n}{\partial x_1} & \cdots & \frac{\partial y_n}{\partial x_n} \end{pmatrix} \tag{2.158}$$

$|\det(\boldsymbol{J})|$ measures how much a unit cube changes in volume when we apply f.

If f is an invertible mapping, we can define the pdf of the transformed variables using the Jacobian of the inverse mapping $\boldsymbol{y} \to \boldsymbol{x}$:

$$p_y(\boldsymbol{y}) = p_x(\boldsymbol{x})|\det(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{y}})| = p_x(\boldsymbol{x})|\det(\boldsymbol{J}_{\boldsymbol{y}\to\boldsymbol{x}})| \tag{2.159}$$

2.9.3 Central limit theorem

Given N random variables $X_1, X_2, \cdots, X_N$, each variable is **independent and identically distributed**[10](**iid** for short), and each has the same mean μ and variance σ^2, then

$$\frac{\sum_{i=1}^{n} X_i - N\mu}{\sqrt{N}\sigma} \sim \mathcal{N}(0,1) \tag{2.160}$$

this can also be written as

$$\frac{\bar{X} - \mu}{\sigma/\sqrt{N}} \sim \mathcal{N}(0,1) \quad \text{, where } \bar{X} \triangleq \frac{1}{N}\sum_{i=1}^{n} X_i \tag{2.161}$$

2.10 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 ap-

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

proximate the distribution of $f(X)$ by using the empirical distribution of $\{f(x_s)\}_{s=1}^S$. This is called a **Monte Carlo approximation**[11], 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:

$$\mathbb{E}[g(X)] = \int g(x)p(x)\mathrm{d}x \approx \frac{1}{S}\sum_{s=1}^{S} f(x_s) \tag{2.162}$$

where $x_s \sim p(X)$.

This is called **Monte Carlo integration**[12], 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.

2.11 Information theory

2.11.1 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

$$\mathbb{H}(X) \triangleq -\sum_{k=1}^{K} p(X=k)\log_2 p(X=k) \tag{2.163}$$

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.

2.11.2 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:

$$\mathbb{KL}(P||Q) \triangleq \sum_x p(x)\log_2 \frac{p(x)}{q(x)} \tag{2.164}$$

where the sum gets replaced by an integral for pdfs[13]. 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 $\lim_{x\to 0} x\ln x$. We can rewrite this as

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

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

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

$$\begin{aligned}\mathbb{KL}(p||q) &\triangleq \sum_x p(x)\log_2 p(x) - \sum_{k=1}^{K} p(x)\log_2 q(x) \\ &= \mathbb{H}(p) - \mathbb{H}(p,q)\end{aligned} \tag{2.165}$$

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

$$\mathbb{H}(p,q) = \sum_x p(x)\log_2 q(x) \tag{2.166}$$

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 §2.11.1,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 2.3. ***(Information inequality)*** $\mathbb{KL}(p||q) \geq 0$ *with equality iff* $p = q$.

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

2.11.3 Mutual information

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

$$\begin{aligned}\mathbb{I}(X;Y) &\triangleq \mathbb{KL}(P(X,Y)||P(X)P(X)) \\ &= \sum_x \sum_y p(x,y)\log\frac{p(x,y)}{p(x)p(y)}\end{aligned} \tag{2.167}$$

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:

$$\mathbb{I}(X;Y) = \mathbb{H}(X) - \mathbb{H}(X|Y) \tag{2.168}$$
$$= \mathbb{H}(Y) - \mathbb{H}(Y|X) \tag{2.169}$$
$$= \mathbb{H}(X) + \mathbb{H}(Y) - \mathbb{H}(X,Y) \tag{2.170}$$
$$= \mathbb{H}(X,Y) - \mathbb{H}(X|Y) - \mathbb{H}(Y|X) \tag{2.171}$$

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. 2.11[14].

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

$$PMI(x,y) \triangleq \log\frac{p(x,y)}{p(x)p(y)} = \log\frac{p(x|y)}{p(x)} = \log\frac{p(y|x)}{p(y)} \tag{2.172}$$

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

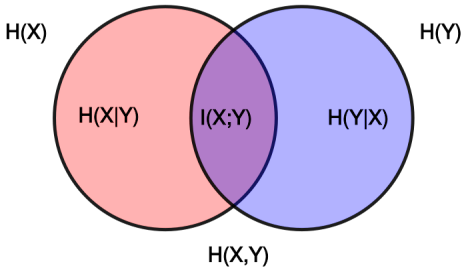


Fig. 2.11: Individual $\mathbb{H}(X), \mathbb{H}(Y)$, joint $\mathbb{H}(X,Y)$, and conditional entropies for a pair of correlated subsystems X, Y with mutual information $\mathbb{I}(X;Y)$.

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:

$$PMI(x,y) = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)} \tag{2.173}$$

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 $p(y|x)$.

Chapter 3
Generative models for discrete data

3.1 Generative classifier

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

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 $p(y=c)$. An alternative approach is to directly fit the class posterior, $p(y=c|\boldsymbol{x})$;this is known as a **discriminative classifier**.

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.

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**[15].

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.

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.

3.2.3 Posterior

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

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

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

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.

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.,

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

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

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

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**[16] 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.

$$p(\tilde{\boldsymbol{x}}|\mathcal{D}) \triangleq \mathbb{E}_{h|\mathcal{D}}[p(\tilde{\boldsymbol{x}}|h)] = \begin{cases} \sum_h p(\tilde{\boldsymbol{x}}|h)p(h|\mathcal{D}) \\ \int p(\tilde{\boldsymbol{x}}|h)p(h|\mathcal{D})\mathrm{d}h \end{cases} \tag{3.7}$$

This is just a weighted average of the predictions of each individual hypothesis and is called **Bayes model averaging**(Hoeting et al. 1999).

3.3 The beta-binomial model

3.3.1 Likelihood

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

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

3.3.2 Prior

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

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

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

3.3.3 Posterior

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

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

$$\begin{aligned} p(\theta|\mathcal{D}_a,\mathcal{D}_b) &= p(\theta,\mathcal{D}_b|\mathcal{D}_a)p(\mathcal{D}_a) \\ &\propto p(\theta,\mathcal{D}_b|\mathcal{D}_a) \\ &= p(\mathcal{D}_b,\theta|\mathcal{D}_a) \\ &= p(\mathcal{D}_b|\theta)p(\theta|\mathcal{D}_a) \\ &\text{Combine Equation 3.10 and 2.51} \\ &= \text{Bin}(N_1^b|\theta,N_1^b+N_0^b)\text{Beta}(\theta|N_1^a+a,N_0^a+b) \\ &= \text{Beta}(\theta|N_1^a+N_1^b+a,N_0^a+N_0^b+b) \end{aligned}$$

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

$$\begin{aligned} p(\tilde{x}|\mathcal{D}) &= \int_0^1 p(\tilde{x}|\theta)p(\theta|\mathcal{D})\mathrm{d}\theta \\ &= \int_0^1 \theta \mathrm{Beta}(\theta|a,b)\mathrm{d}\theta \\ &= \mathbb{E}[\theta|\mathcal{D}] = \frac{a}{a+b} \end{aligned} \tag{3.16}$$

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**

$$p(\tilde{x}|\mathcal{D}) = \frac{N_1+1}{N_0+N_1+1} \tag{3.17}$$

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

$$p(\tilde{x}|\mathcal{D}) = \int_0^1 \mathrm{Bin}(\tilde{x}|M,\theta)\mathrm{Beta}(\theta|a,b)\mathrm{d}\theta \tag{3.18}$$

$$= \binom{M}{\tilde{x}} \frac{1}{B(a,b)} \int_0^1 \theta^{\tilde{x}}(1-\theta)^{M-\tilde{x}}\theta^{a-1}(1-\theta)^{b-1}\mathrm{d}\theta \tag{3.19}$$

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

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

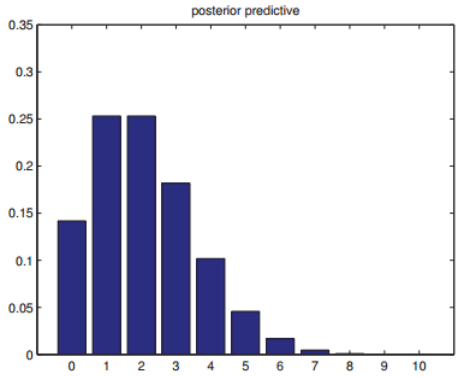
Thus we find that the posterior predictive is given by the following, known as the (compound) **beta-binomial distribution**:

$$Bb(x|a,b,M) \triangleq \binom{M}{x} \frac{B(x+a,M-x+b)}{B(a,b)} \tag{3.21}$$

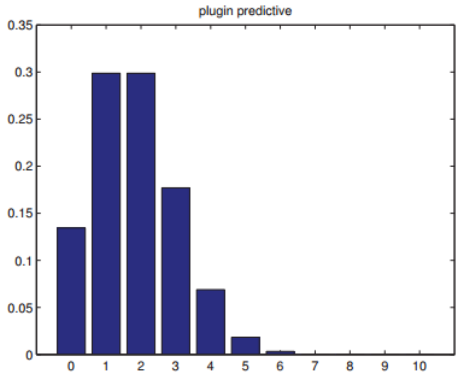
This distribution has the following mean and variance

$$\mathrm{mean} = M\frac{a}{a+b}\ ,\ \mathrm{var} = \frac{Mab}{(a+b)^2}\frac{a+b+M}{a+b+1} \tag{3.22}$$

This process is illustrated in Figure 3.1. We start with a Beta$(2,2)$ prior, and plot the posterior predictive density after seeing $N_1=3$ heads and $N_0=17$ tails. Figure 3.1(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.



(a)



(b)

Fig. 3.1: (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

$$p(\mathcal{D}|\boldsymbol{\theta}) = \binom{N}{N_1 \cdots N_k} \prod_{k=1}^{K} \theta_k^{N_k} \quad \text{where } N_k = \sum_{i=1}^{N} \mathbb{I}(y_i = k) \tag{3.23}$$

almost the same as Equation 2.53.

3.4.2 Prior

$$\text{Dir}(\boldsymbol{\theta}|\boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1} \mathbb{I}(\boldsymbol{\theta} \in S_K) \tag{3.24}$$

3.4.3 Posterior

$$p(\boldsymbol{\theta}|\mathcal{D}) \propto p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}) \tag{3.25}$$

$$\propto \prod_{k=1}^{K} \theta_k^{N_k} \theta_k^{\alpha_k - 1} = \prod_{k=1}^{K} \theta_k^{N_k + \alpha_k - 1} \tag{3.26}$$

$$= \mathrm{Dir}(\boldsymbol{\theta}|\alpha_1 + N_1, \cdots, \alpha_K + N_K) \tag{3.27}$$

From Equation 2.151, the MAP estimate is given by

$$\hat{\theta}_k = \frac{N_k + \alpha_k - 1}{N + \alpha_0 - K} \tag{3.28}$$

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

$$\hat{\theta}_k = \frac{N_k}{N} \tag{3.29}$$

3.4.4 Posterior predictive distribution

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

$$p(X = j|\mathcal{D}) = \int p(X = j|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D})\mathrm{d}\boldsymbol{\theta} \tag{3.30}$$

$$= \int p(X = j|\theta_j) \left[\int p(\boldsymbol{\theta}_{-j}, \theta_j|\mathcal{D})\mathrm{d}\boldsymbol{\theta}_{-j} \right] \mathrm{d}\theta_j \tag{3.31}$$

$$= \int \theta_j p(\theta_j|\mathcal{D})\mathrm{d}\theta_j = \mathbb{E}[\theta_j|\mathcal{D}] = \frac{\alpha_j + N_j}{\alpha_0 + N} \tag{3.32}$$

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

$$p(\boldsymbol{x}|y = c, \boldsymbol{\theta}) = \prod_{j=1}^{D} p(x_j|y = c, \boldsymbol{\theta}_{jc}) \tag{3.33}$$

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: $p(\boldsymbol{x}|y, \boldsymbol{\theta}) = \prod_{j=1}^{D} \mathrm{Ber}(x_j|\mu_{jc})$, 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: $p(\boldsymbol{x}|y, \boldsymbol{\theta}) = \prod_{j=1}^{D} \mathrm{Cat}(x_j|\boldsymbol{\mu}_{jc})$, 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

$$\begin{aligned} p(\boldsymbol{x}_i, y_i|\boldsymbol{\theta}) &= p(y_i|\boldsymbol{\pi}) \prod_j p(x_{ij}|\boldsymbol{\theta}_j) \\ &= \prod_c \pi_c^{\mathbb{I}(y_i=c)} \prod_j \prod_c p(x_{ij}|\boldsymbol{\theta}_{jc})^{\mathbb{I}(y_i=c)} \end{aligned} \tag{3.34}$$

Hence the log-likelihood is given by

$$p(\mathcal{D}|\boldsymbol{\theta}) = \sum_{c=1}^{C} N_c \log \pi_c + \sum_{j=1}^{D} \sum_{c=1}^{C} \sum_{i:y_i=c} \log p(x_{ij}|\boldsymbol{\theta}_{jc}) \tag{3.35}$$

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 $\boldsymbol{\theta}_{jc}$s. Hence we can optimize all these parameters separately.

From Equation 3.29, the MLE for the class prior is given by

$$\hat{\pi}_c = \frac{N_c}{N} \tag{3.36}$$

The MLE for $\boldsymbol{\theta}_{jc}$s depends on the type of distribution we choose to use for each feature.

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

$$\hat{\theta}_{jc} = \frac{N_{jc}}{N_c} \tag{3.37}$$

where $N_{jc} \triangleq \sum_{i:y_i=c} \mathbb{I}(y_i = c)$ 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

$$\hat{\boldsymbol{\theta}}_{jc} = (\frac{N_{j1c}}{N_c}, \frac{N_{j2c}}{N_c}, \cdots, \frac{N_{jS_j}}{N_c})^T \tag{3.38}$$

where $N_{jkc} \triangleq \sum_{i=1}^{N} \mathbb{I}(x_{ij} = a_{jk}, y_i = c)$ 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

$$\begin{aligned} y = f(\boldsymbol{x}) &= \arg\max_c P(y=c|\boldsymbol{x},\boldsymbol{\theta}) \\ &= P(y=c|\boldsymbol{\theta})\prod_{j=1}^{D} P(x_j|y=c,\boldsymbol{\theta}) \end{aligned} \tag{3.39}$$

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:

$$\log p(y=c|\boldsymbol{x},\boldsymbol{\theta}) = b_c - \log\left(\sum_{c'} e^{b_{c'}}\right) \tag{3.40}$$

where $b_c \triangleq \log p(\boldsymbol{x}|y=c,\boldsymbol{\theta}) + \log p(y=c|\boldsymbol{\theta})$.

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

$$\begin{aligned} \log(e^{-120}+e^{-121}) &= \log(e^{-120}(1+e^{-1})) \\ &= \log(1+e^{-1}) - 120 \end{aligned} \tag{3.41}$$

In general, we have

$$\sum_c e^{b_c} = \log\left[(\sum e^{b_c-B})e^B\right] = \log\left(\sum e^{b_c-B}\right) + B \tag{3.42}$$

where $B \triangleq \max\{b_c\}$.

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 2.11.3) between feature X_j and the class label Y

$$\mathbb{I}(X_j,Y) = \sum_{x_j}\sum_{y} p(x_j,y)\log\frac{p(x_j,y)}{p(x_j)p(y)} \tag{3.43}$$

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

$$\mathbb{I}_j = \sum_c \left[\theta_{jc} \pi_c \log \frac{\theta_{jc}}{\theta_j} + (1 - \theta_{jc}) \pi_c \log \frac{1 - \theta_{jc}}{1 - \theta_j} \right] \tag{3.44}$$

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.

3.5.5.1 Bernoulli product model

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:

$$\begin{aligned} p(\boldsymbol{x}_i | y_i = c, \boldsymbol{\theta}) &= \prod_{j=1}^{D} \text{Ber}(x_{ij} | \theta_{jc}) \\ &= \prod_{j=1}^{D} \theta_{jc}^{x_{ij}} (1 - \theta_{jc})^{1 - x_{ij}} \end{aligned} \tag{3.45}$$

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 $\sum_{j=1}^{D} x_{ij} = N_i$). For the class conditional densities, we can use a multinomial distribution:

$$p(\boldsymbol{x}_i | y_i = c, \boldsymbol{\theta}) = \text{Mu}(\boldsymbol{x}_i | N_i, \boldsymbol{\theta}_c) = \frac{N_i!}{\prod_{j=1}^{D} x_{ij}!} \prod_{j=1}^{D} \theta_{jc}^{x_{ij}} \tag{3.46}$$

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 3.46 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:

$$\begin{aligned} p(\boldsymbol{x}_i|y_i = c, \boldsymbol{\alpha}) &= \int \text{Mu}(\boldsymbol{x}_i|N_i, \boldsymbol{\theta}_c)\text{Dir}(\boldsymbol{\theta}_c|\boldsymbol{\alpha}_c) \\ &= \frac{N_i!}{\prod_{j=1}^{D} x_{ij}!} \prod_{j=1}^{D} \frac{B(\boldsymbol{x}_i + \boldsymbol{\alpha}_c)}{B(\boldsymbol{\alpha}_c)} \end{aligned} \tag{3.47}$$

(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

The Gaussian,also known as the **normal distribution**,is a widely used model for the **distribution of continuous variables**.In the case of a single variable x,the Gaussian distribution can be written in the form

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}} \exp\{\frac{1}{2\sigma^2(x-\mu)^2}\} \tag{4.1}$$

where the μ is the mean and σ^2 is the variance. Now we will 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.

4.1 Gaussian discriminant analysis

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

$$p(\boldsymbol{x}|y=c,\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_c,\boldsymbol{\Sigma}_c) \tag{4.2}$$

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:

$$y = \arg\max_c [\log p(y=c|\boldsymbol{\pi}) + \log p(\boldsymbol{x}|\boldsymbol{\theta})] \tag{4.3}$$

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**.

As an example, Figure 4.1 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:

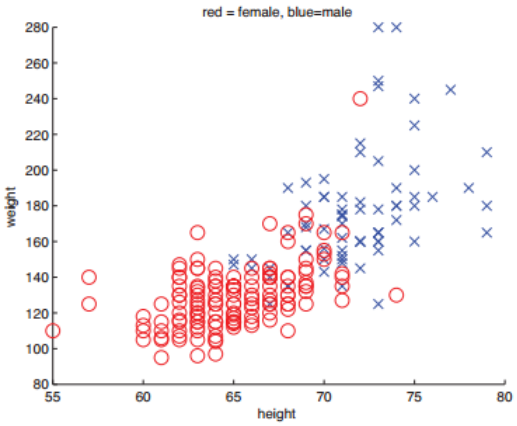
$$y = \arg\max_c (\boldsymbol{x}-\boldsymbol{\mu}_c)^T \boldsymbol{\Sigma}_c^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_c) \tag{4.4}$$

4.1.1 Quadratic discriminant analysis (QDA)

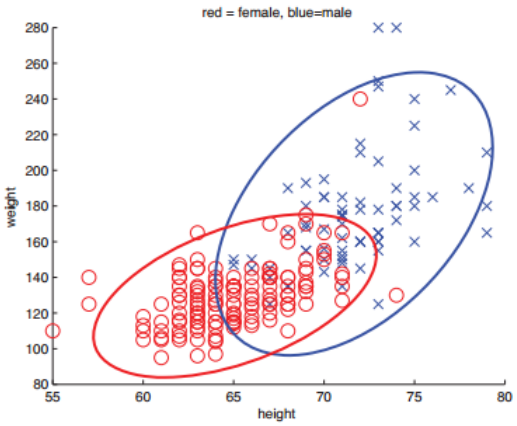
By plugging in the definition of the Gaussian density to Equation 3.1, we can get

$$p(y|\boldsymbol{x},\boldsymbol{\theta}) = \frac{\pi_c |2\pi\boldsymbol{\Sigma}_c|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]}{\sum_{c'} \pi_{c'} |2\pi\boldsymbol{\Sigma}_{c'}|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]} \tag{4.5}$$

Thresholding this results in a quadratic function of $\boldsymbol{x}$. The result is known as quadratic discriminant analysis(QDA). Figure 4.2 gives some examples of what the decision boundaries look like in 2D.



(a)



(b)

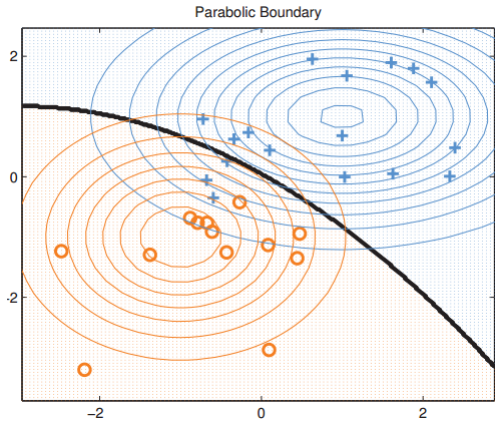
Fig. 4.1: (a) Height/weight data. (b) Visualization of 2d Gaussians fit to each class. 95% of the probability mass is inside the ellipse.

4.1.2 Linear discriminant analysis (LDA)

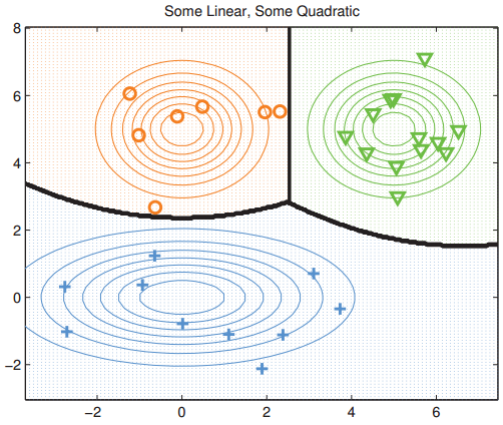
We now consider a special case in which the covariance matrices are **tied** or **shared** across classes, $\boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}$. In this case, we can simplify Equation 4.5 as follows:

$$\begin{aligned} p(y|\boldsymbol{x},\boldsymbol{\theta}) &\propto \pi_c \exp\left(\boldsymbol{\mu}_c \boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c\right) \\ &= \exp\left(\boldsymbol{\mu}_c \boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log \pi_c\right) \\ &\quad \exp\left(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right) \\ &\propto \exp\left(\boldsymbol{\mu}_c \boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log \pi_c\right) \end{aligned} \tag{4.6}$$

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



(a)



(b)

Fig. 4.2: Quadratic decision boundaries in 2D for the 2 and 3 class case.

$$\gamma_c \triangleq -\frac{1}{2}\boldsymbol{\mu}_c^T \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log \pi_c \tag{4.7}$$

$$\boldsymbol{\beta}_c \triangleq \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c \tag{4.8}$$

then we can write

$$p(y|\boldsymbol{x},\boldsymbol{\theta}) = \frac{e^{\boldsymbol{\beta}_c^T \boldsymbol{x}+\gamma_c}}{\sum_{c'} e^{\boldsymbol{\beta}_{c'}^T \boldsymbol{x}+\gamma_{c'}}} \triangleq \sigma(\boldsymbol{\eta}, c) \tag{4.9}$$

where $\boldsymbol{\eta} \triangleq (e^{\boldsymbol{\beta}_1^T \boldsymbol{x}} + \gamma_1, \cdots, e^{\boldsymbol{\beta}_C^T \boldsymbol{x}} + \gamma_C)$, $\sigma()$ is the **softmax activation function**[17], defined as follows:

$$\sigma(\boldsymbol{q}, i) \triangleq \frac{\exp(q_i)}{\sum_{j=1}^n \exp(q_j)} \tag{4.10}$$

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

$$\mathcal{S}_\alpha(\boldsymbol{x}) = \frac{\sum_{j=1}^D x_j e^{\alpha x_j}}{\sum_{j=1}^D e^{\alpha x_j}} \tag{4.11}$$

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

1. $\mathcal{S}_\alpha \to \max$ as $\alpha \to \infty$

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

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.

An interesting property of Equation 4.9 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.

4.1.3 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

$$p(y=1|\boldsymbol{x}, \boldsymbol{\theta}) = \frac{e^{\boldsymbol{\beta}_1^T \boldsymbol{x} + \gamma_1}}{e^{\boldsymbol{\beta}_0^T \boldsymbol{x} + \gamma_0} + e^{\boldsymbol{\beta}_1^T \boldsymbol{x} + \gamma_1}}) \tag{4.12}$$

$$= \frac{1}{1 + e^{(\boldsymbol{\beta}_0 - \boldsymbol{\beta}_1)^T \boldsymbol{x} + (\gamma_0 - \gamma_1)}} \tag{4.13}$$

$$= \text{sigm}((\boldsymbol{\beta}_1 - \boldsymbol{\beta}_0)^T \boldsymbol{x} + (\gamma_0 - \gamma_1)) \tag{4.14}$$

where $\text{sigm}(x)$ refers to the sigmoid function[18].

Now

$$\gamma_1 - \gamma_0 = -\frac{1}{2} \boldsymbol{\mu}_1^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_1 + \frac{1}{2} \boldsymbol{\mu}_0^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_0 + \log(\pi_1/\pi_0) \tag{4.15}$$

$$= -\frac{1}{2} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0) + \log(\pi_1/\pi_0) \tag{4.16}$$

So if we define

$$\boldsymbol{w} = \boldsymbol{\beta}_1 - \boldsymbol{\beta}_0 = \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) \tag{4.17}$$

$$\boldsymbol{x}_0 = \frac{1}{2} (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0) - (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) \frac{\log(\pi_1/\pi_0)}{(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)} \tag{4.18}$$

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

$$p(y=1|\boldsymbol{x}, \boldsymbol{\theta}) = \text{sigm}(\boldsymbol{w}^T (\boldsymbol{x} - \boldsymbol{x}_0)) \tag{4.19}$$

(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 4.3. Furthemore, if $\pi_1 = \pi_0$, then $\boldsymbol{x}_0 = \frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0)$, which is half way between the means. If we make $\pi_1 > \pi_0$, then $\boldsymbol{x}_0$ gets closer to $\boldsymbol{\mu}_0$, so more of the line belongs to class 1 a *priori*. Conversely if $\pi_1 < \pi_0$, 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.)

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

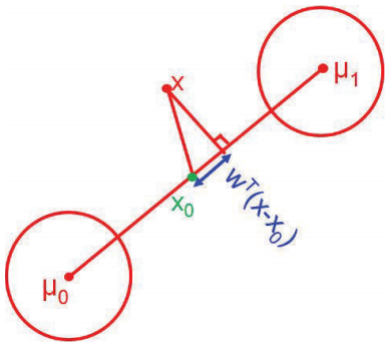


Fig. 4.3: Geometry of LDA in the 2 class case where $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{I}$.

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**:

$$d' \triangleq \frac{\mu_1 - \mu_0}{\sigma} \tag{4.20}$$

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.

4.1.4 MLE for discriminant analysis

The log-likelihood function is as follows:

$$p(\mathcal{D}|\boldsymbol{\theta}) = \sum_{c=1}^{C} \sum_{i:y_i=c} \log \pi_c + \sum_{c=1}^{C} \sum_{i:y_i=c} \log \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \tag{4.21}$$

The MLE for each parameter is as follows:

$$\bar{\boldsymbol{\mu}}_c = \frac{N_c}{N} \tag{4.22}$$

$$\bar{\boldsymbol{\mu}}_c = \frac{1}{N_c} \sum_{i:y_i=c} \boldsymbol{x}_i \tag{4.23}$$

$$\bar{\boldsymbol{\Sigma}}_c = \frac{1}{N_c} \sum_{i:y_i=c} (\boldsymbol{x}_i - \bar{\boldsymbol{\mu}}_c)(\boldsymbol{x}_i - \bar{\boldsymbol{\mu}}_c)^T \tag{4.24}$$

4.1.5 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 4.1.2).

- 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.

4.1.6 Regularized LDA *

4.1.7 Diagonal LDA

4.1.8 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).

4.2 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.

4.2.1 Statement of the result

Theorem 4.1. ***(Marginals and conditionals of an MVN).*** *Suppose $X = (\boldsymbol{x}_1, \boldsymbol{x}_2)$is jointly Gaussian with parameters*

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}, \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{pmatrix}, \boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1} = \begin{pmatrix} \boldsymbol{\Lambda}_{11} & \boldsymbol{\Lambda}_{12} \\ \boldsymbol{\Lambda}_{21} & \boldsymbol{\Lambda}_{22} \end{pmatrix}, \tag{4.25}$$

Then the marginals are given by

$$\begin{aligned} p(\boldsymbol{x}_1) &= \mathcal{N}(\boldsymbol{x}_1|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11}) \\ p(\boldsymbol{x}_2) &= \mathcal{N}(\boldsymbol{x}_2|\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22}) \end{aligned} \tag{4.26}$$

and the posterior conditional is given by

$$\boxed{\begin{aligned} p(\boldsymbol{x}_1|\boldsymbol{x}_2) &= \mathcal{N}(\boldsymbol{x}_1|\boldsymbol{\mu}_{1|2}, \boldsymbol{\Sigma}_{1|2}) \\ \boldsymbol{\mu}_{1|2} &= \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2) \\ &= \boldsymbol{\mu}_1 - \boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2) \\ &= \boldsymbol{\Sigma}_{1|2}(\boldsymbol{\Lambda}_{11}\boldsymbol{\mu}_1 - \boldsymbol{\Lambda}_{12}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2)) \\ \boldsymbol{\Sigma}_{1|2} &= \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21} = \boldsymbol{\Lambda}_{11}^{-1} \end{aligned}} \tag{4.27}$$

Equation 4.27 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.

4.2.2 Examples

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

4.2.2.1 Marginals and conditionals of a 2d Gaussian

4.2.3 information form

4.2.4 Proof

4.3 Linear Gaussian systems

Suppose we have two variables, $\boldsymbol{x}$ and $\boldsymbol{y}$.Let $\boldsymbol{x} \in \mathbb{R}^{D_x}$ be a hidden variable, and $\boldsymbol{y} \in \mathbb{R}^{D_y}$ be a noisy observation of $\boldsymbol{x}$. Let us assume we have the following prior and likelihood:

$$\boxed{\begin{aligned} p(\boldsymbol{x}) &= \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) \\ p(\boldsymbol{y}|\boldsymbol{x}) &= \mathcal{N}(\boldsymbol{y}|\boldsymbol{W}\boldsymbol{x} + \boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y) \end{aligned}} \tag{4.28}$$

where $\boldsymbol{W}$ is a matrix of size $D_y \times D_x$. This is an example of a **linear Gaussian system**. We can represent this schematically as $\boldsymbol{x} \rightarrow \boldsymbol{y}$, meaning $\boldsymbol{x}$ generates $\boldsymbol{y}$. In this section, we show how to invert the arrow, that is, how to infer $\boldsymbol{x}$ from $\boldsymbol{y}$. We state the result below, then give several examples, and finally we derive the result. We will see many more applications of these results in later chapters.

4.3.1 Statement of the result

Theorem 4.2. ***(Bayes rule for linear Gaussian systems).*** *Given a linear Gaussian system, as in Equation 4.28, the posterior $p(\boldsymbol{x}|\boldsymbol{y})$ is given by the following:*

$$\boxed{\begin{aligned} p(\boldsymbol{x}|\boldsymbol{y}) &= \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_{x|y}, \boldsymbol{\Sigma}_{x|y}) \\ \boldsymbol{\Sigma}_{x|y} &= \boldsymbol{\Sigma}_x^{-1} + \boldsymbol{W}^T \boldsymbol{\Sigma}_y^{-1} \boldsymbol{W} \\ \boldsymbol{\mu}_{x|y} &= \boldsymbol{\Sigma}_{x|y} \left[\boldsymbol{W}^T \boldsymbol{\Sigma}_y^{-1} (\boldsymbol{y} - \boldsymbol{\mu}_y) + \boldsymbol{\Sigma}_x^{-1} \boldsymbol{\mu}_x \right] \end{aligned}} \tag{4.29}$$

In addition, the normalization constant $p(\boldsymbol{y})$ is given by

$$\boxed{p(\boldsymbol{y}) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{W}\boldsymbol{\mu}_x + \boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y + \boldsymbol{W}\boldsymbol{\Sigma}_x\boldsymbol{W}^T)} \tag{4.30}$$

For the proof, see Section 4.4.3 TODO.

4.3.2 Examples

4.3.3 Proof

4.4 Digression: The Wishart distribution *

4.5 Inferring the parameters of an MVN

4.5.1 Posterior distribution of μ

4.5.2 Posterior distribution of Σ *

4.5.3 Posterior distribution of μ and Σ *

4.5.4 Sensor fusion with unknown precisions *

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

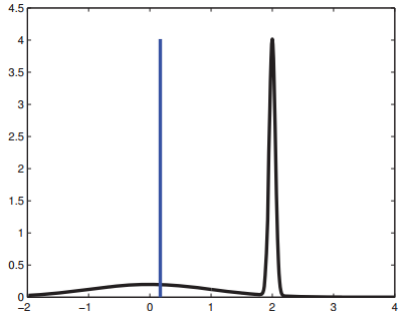
The most obvious drawback of MAP estimation, and indeed of any other *point estimate* such as the posterior mean or median, is that it does not provide any measure of uncertainty. In many applications, it is important to know how much one can trust a given estimate. We can derive such confidence measures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

5.2.1.2 Plugging in the MAP estimate can result in overfitting

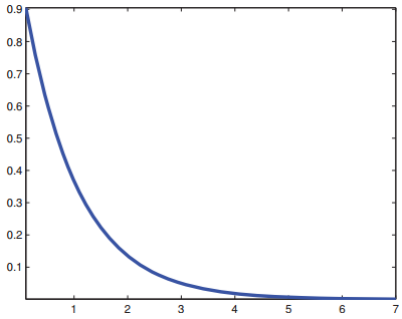
If we dont model the uncertainty in our parameters, then our predictive distribution will be overconfident. Overconfidence in predictions is particularly problematic in situations where we may be risk averse; see Section 5.7 for details.

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.



(a)



(b)

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 1.4.2.1), 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 2.157. 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 2.10). 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.

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.

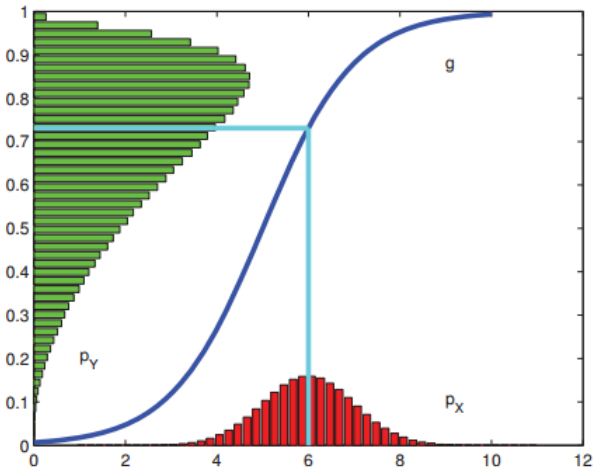


Fig. 5.2: Example of the transformation of a density under a nonlinear transform. Note how the mode of the transformed distribution is not the transform of the original mode. Based on Exercise 1.4 of (Bishop 2006b).

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.,

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

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?[19].

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.

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

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

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

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

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

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 2.20):

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

This is similar to a leave-one-out cross-validation estimate (Section 1.5.0.1) 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.

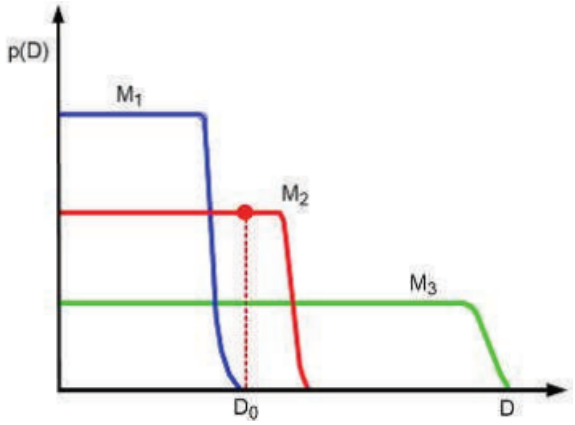


Fig. 5.3: A schematic illustration of the Bayesian Occams razor. The broad (green) curve corresponds to a complex model, the narrow (blue) curve to a simple model, and the middle (red) curve is just right. Based on Figure 3.13 of (Bishop 2006a).

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

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} \tag{5.10}$$

$$= \frac{1}{p(\mathcal{D})}\left[\frac{1}{B(a,b)}\theta^{a-1}(1-\theta)^{b-1}\right] \left[\binom{N}{N_1}\theta^{N_1}(1-\theta)^{N_0}\right] \tag{5.11}$$

$$= \binom{N}{N_1}\frac{1}{p(\mathcal{D})}\frac{1}{B(a,b)}\left[\theta^{a+N_1-1}(1-\theta)^{b+N_0-1}\right] \tag{5.12}$$

So

$$\frac{1}{B(a+N_1,b+N_0)} = \binom{N}{N_1}\frac{1}{p(\mathcal{D})}\frac{1}{B(a,b)} \tag{5.13}$$

$$p(\mathcal{D}) = \binom{N}{N_1}\frac{B(a+N_1,b+N_0)}{B(a,b)} \tag{5.14}$$

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

$$p(\mathcal{D}) = \frac{B(\boldsymbol{N}+\boldsymbol{\alpha})}{B(\boldsymbol{\alpha})} \tag{5.15}$$

$$= \frac{\Gamma(\sum_k \alpha_k)}{\Gamma(N+\sum_k \alpha_k)}\prod_k \frac{\Gamma(N_k+\alpha_k)}{\Gamma(\alpha_k)} \tag{5.16}$$

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

$$p(\mathcal{D}) = \frac{Z_N}{Z_0 Z_\ell} \tag{5.17}$$

$$= \frac{1}{(2\pi)^{ND/2}} \frac{\left(\frac{2\pi}{\kappa_N}\right)^{D/2} |\boldsymbol{S}_N|^{-\nu_N/2} 2^{(\nu_0+N)D/2} \Gamma_D(\nu_N/2)}{\left(\frac{2\pi}{\kappa_0}\right)^{D/2} |\boldsymbol{S}_0|^{-\nu_0/2} 2^{\nu_0 D/2} \Gamma_D(\nu_0/2)} \tag{5.18}$$

$$= \frac{1}{\pi^{ND/2}} \left(\frac{\kappa_0}{\kappa_N}\right)^{D/2} \frac{|\boldsymbol{S}_0|^{\nu_0/2}}{|\boldsymbol{S}_N|^{\nu_N/2}} \frac{\Gamma_D(\nu_N/2)}{\Gamma_D(\nu_0/2)} \tag{5.19}$$

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:

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

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 1.2).

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**:

$$\rho(f) = \mathbb{E}_{p(y|\boldsymbol{x})}[L(y, f(\boldsymbol{x}))] = \begin{cases} \sum_y L[y, f(\boldsymbol{x})] p(y|\boldsymbol{x}) \\ \int_y L[y, f(\boldsymbol{x})] p(y|\boldsymbol{x}) \mathrm{d}y \end{cases} \tag{5.28}$$

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 1.4.2.1), 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

$$\begin{aligned}
\rho(f) &= \int_y L[y, f(\boldsymbol{x})] p(y|\boldsymbol{x}) \mathrm{d}y \\
&= \int_y [y - f(\boldsymbol{x})]^2 p(y|\boldsymbol{x}) \mathrm{d}y \\
&= \int_y \left[y^2 - 2y f(\boldsymbol{x}) + f(\boldsymbol{x})^2\right] p(y|\boldsymbol{x}) \mathrm{d}y
\end{aligned} \tag{5.30}$$

Hence the optimal estimate is the posterior mean:

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

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.

$$\begin{aligned}
\rho(f) &= \int_y L[y, f(\boldsymbol{x})] p(y|\boldsymbol{x}) \mathrm{d}y = \int_y |y - f(\boldsymbol{x})| p(y|\boldsymbol{x}) \mathrm{d}y \\
&= \int_y [f(\boldsymbol{x}) - y] p(y < f(\boldsymbol{x})|\boldsymbol{x}) + \\
&\quad [y - f(\boldsymbol{x})] p(y \geq f(\boldsymbol{x})|\boldsymbol{x}) \mathrm{d}y \\
&\frac{\partial \rho}{\partial f} = \int_y [p(y < f(\boldsymbol{x})|\boldsymbol{x}) - p(y \geq f(\boldsymbol{x})|\boldsymbol{x})] \, \mathrm{d}y = 0 \Rightarrow \\
&p(y < f(\boldsymbol{x})|\boldsymbol{x}) = p(y \geq f(\boldsymbol{x})|\boldsymbol{x}) = 0.5 \\
&\therefore f(\boldsymbol{x}) = \text{median}
\end{aligned}$$

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
Linear Regression

6.1 Introduction

Linear regression is the work horse of statistics and (supervised) machine learning. When augmented with kernels or other forms of basis function expansion, it can model also nonlinear relationships. And when the Gaussian output is replaced with a Bernoulli or multinoulli distribution, it can be used for classification, as we will see below. So it pays to study this model in detail.

In the simplest approach,we can directly construct an appropriate function $y(\boldsymbol{x})$ whose values for new inputs $\boldsymbol{x}$ constitute the predictions for the corresponding values of t.More generally,from a probabilistic perspective,we aim to model the predictive distribution $p(t|x)$ because this expresses the uncertainty about the value of t for each value of x.

6.2 Representation

The simplest linear model for regression is **linear regression** that involves a linear combination of the input variables

$$y(\boldsymbol{x},\boldsymbol{w}) = w_0 + w_1 x_1 + ... + w_D x_D \tag{6.1}$$

where $\boldsymbol{x} = (x_1,...,x_D)^T$.

Linear regression can be made to model non-linear relationships by replacing $\boldsymbol{x}$ with some non-linear function of the inputs, $\boldsymbol{\phi}(\boldsymbol{x})$.Consider the linear combinations of fixed nonlinear functions of the input variables,of the form

$$y(\boldsymbol{x},\boldsymbol{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(x) = \sum_{j=0}^{M-1} w_j \phi_j(x) = \boldsymbol{w}^T \boldsymbol{\phi}(x) \tag{6.2}$$

where $\phi_j(x)$ are known as **basis functions**, $\boldsymbol{w} = (w_0,...,w_{M-1})^T$ and $\boldsymbol{\phi} = (\phi_0,...,\phi_{M-1})^T$. This is known as **basis function expansion**. (Note that the model is still linear in the parameters $\boldsymbol{w}$, so it is still called linear regression; the importance of this will become clear below.) A simple example are polynomial basis functions, where the model has the form

$$\boldsymbol{\phi}(x) = (1, x, \cdots, x^d) \tag{6.3}$$

As before,we assume that the target variable is given by a deterministic function $y(\boldsymbol{x},\boldsymbol{w})$ with additive Gaussian noise so that

$$t = y(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{\varepsilon} \tag{6.4}$$

where $\boldsymbol{\varepsilon}$ is a zero mean Gaussian random variable with precision(inverse variance) β.Thus

$$p(t|\boldsymbol{x},\boldsymbol{w},\beta) = \mathcal{N}(t|h(\boldsymbol{x},\boldsymbol{w}),\beta^{-1}) \tag{6.5}$$

or

$$y(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + \boldsymbol{\varepsilon} \tag{6.6}$$

$$p(y|\boldsymbol{x},\boldsymbol{\theta}) = \mathcal{N}(y|\boldsymbol{w}^T \boldsymbol{x}, \boldsymbol{\sigma}^2) \tag{6.7}$$

$$\tag{6.8}$$

where $\boldsymbol{w}$ (**weight vector**) and $\boldsymbol{x}$ are extended vectors, $\boldsymbol{x} = (1,x)$, $\boldsymbol{w} = (b,w)$ and $\boldsymbol{\varepsilon}$ has a **Gaussian** or **normal** distribution.w_0 is the intercept or **text**

6.3 Maximum likelihood estimations(least squares)

Assume the training examples are **independently and identically distributed(IID)**,we obtain the **likelihood function**,which is a function of adjustable parameters $\boldsymbol{w}$ and β,in the form

$$p(\boldsymbol{t}|\boldsymbol{X},\boldsymbol{w},\beta)=\prod_{n=1}^{N}\mathcal{N}(t_n|\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x_n}),\beta^{-1}) \tag{6.9}$$

we can write the **log-likelihood** (logarithm of the likelihood) function as follows:

$$\ell(\boldsymbol{\theta})\triangleq\log p(\mathcal{D}|\boldsymbol{\theta}) \tag{6.10}$$

A common way to estimate the parameters of a statistical model is to compute the MLE(Maximum likelihood estimations),which is defined as

$$\hat{\boldsymbol{\theta}}=\arg\max_{\boldsymbol{\theta}}\log p(\mathcal{D}|\boldsymbol{\theta}) \tag{6.11}$$

Instead of maximizing the log-likelihood, we can equivalently minimize the **negative log likelihood** or **NLL**:

$$\mathrm{NLL}(\boldsymbol{\theta})\triangleq-\ell(\boldsymbol{\theta})=-\log p(\mathcal{D}|\boldsymbol{\theta}) \tag{6.12}$$

The NLL formulation is sometimes more convenient, since many optimization software packages are designed to find the minima of functions, rather than maxima.

Now let us apply the method of MLE to the linear regression setting. Inserting the definition of the Gaussian into the above, we find that the log likelihood is given by

$$\begin{aligned}\ell(\boldsymbol{\theta})&=\log p(\boldsymbol{T}|\boldsymbol{w},\beta) &(6.13)\\ &=\sum_{n=1}^{N}\log\mathcal{N}(t_n|\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x_n}),\beta^{-1}) &(6.14)\\ &=\sum_{i=1}^{N}\log\left[\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{1}{2\sigma^2}(y_i-\boldsymbol{w}^T\boldsymbol{x}_i)^2\right)\right] &(6.15)\\ &=\frac{N}{2}\log\beta-\frac{N}{2}\log(2\pi)-\beta E_D(\boldsymbol{w}) &(6.16)\\ & &(6.17)\end{aligned}$$

where the sum-of-squares error function is defined by

$$E_D(\boldsymbol{w})=\frac{1}{2}\sum_{n=1}^{N}\{t_n-\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x_n})\}^2 \tag{6.18}$$

There multiple ways to compute the optimal solution of likelihood function: in matrix calculus, sum of terms derivatives, geometry interpretation.

6.3.1 Derivations of the MLE

Denote $\boldsymbol{t}=(t_1,t_2,\cdots,t_N)^T$, $\boldsymbol{\Phi}=\begin{pmatrix}\boldsymbol{\phi}(x_1)^T\\ \boldsymbol{\phi}(x_2)^T\\ \vdots\\ \boldsymbol{\phi}(x_N)^T\end{pmatrix}$. $\boldsymbol{\Phi}$ is an $N\times M$ matrix,called the **design matrix**, whose elements are given by $\boldsymbol{\Phi}_{nj}=\boldsymbol{\phi}_j(\boldsymbol{x_n})$ so that

$$\boldsymbol{\Phi} = \begin{bmatrix} \phi_0(\boldsymbol{x_1}) & \phi_1(\boldsymbol{x_1}) & \dots & \phi_{M-1}(\boldsymbol{x_1}) \\ \phi_0(\boldsymbol{x_2}) & \phi_1(\boldsymbol{x_2}) & \dots & \phi_{M-1}(\boldsymbol{x_2}) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_0(\boldsymbol{x_N}) & \phi_1(\boldsymbol{x_N}) & \cdot & \phi_{M-1}(\boldsymbol{x_N}) \end{bmatrix} \tag{6.19}$$

Note that the sum of terms can be expressed in matrix/vector multiplication.

$$\because \sum_{n=1}^{N} t_n \phi(\boldsymbol{x}_n)^T = \boldsymbol{t}^T \boldsymbol{\Phi} \tag{6.20}$$

$$\begin{cases} \boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\Phi_1^T} \\ \boldsymbol{\Phi_2^T} \\ \dots \\ \boldsymbol{\Phi_n^T} \end{bmatrix} \\ \boldsymbol{\Phi^T \Phi} = \begin{bmatrix} \boldsymbol{\Phi_1} & \boldsymbol{\Phi_2} & \dots & \boldsymbol{\Phi_n} \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\Phi_1^T} \\ \boldsymbol{\Phi_2^T} \\ \dots \\ \boldsymbol{\Phi_N^T} \end{bmatrix} = \sum_{i=1}^{N} \phi_i \phi_i^T \end{cases} \tag{6.21}$$

We see that maximizing the likelihood function under a conditional Gaussian noise distribution for a linear model is equivalent to $\boldsymbol{w}$ minimizing the sum-of-squares error function given by E_D, so this method is known as **least squares**.The gradient of the log likelihood function takes the form

$$\nabla \log p(\boldsymbol{t}|\boldsymbol{w}, \beta) = \sum_{n=1}^{N} \{t_n - \boldsymbol{w}^T \phi(\boldsymbol{x_n})\} \phi(\boldsymbol{x_n}) \tag{6.22}$$

Setting the gradient respect to $\boldsymbol{w}$ to zero gives

$$\nabla \log E_D = 0 \tag{6.23}$$

$$\Rightarrow \log E_D^T = 0 \tag{6.24}$$

$$\Rightarrow \sum_{n=1}^{N} \{t_n - \boldsymbol{w}^T \phi(\boldsymbol{x_n})\} \phi(\boldsymbol{x_n})^T = 0 \tag{6.25}$$

$$\Rightarrow \sum_{n=1}^{N} t_n \phi(\boldsymbol{x}_n)^T - \boldsymbol{w}^T (\sum_{n=1}^{N} \boldsymbol{\phi}(\boldsymbol{x}_n) \boldsymbol{\phi}(\boldsymbol{x}_n)^T) = 0 \tag{6.26}$$

$$\Rightarrow \boldsymbol{t}^T \boldsymbol{\Phi} - \boldsymbol{w}^T \boldsymbol{\Phi}^T \boldsymbol{\Phi} = 0 \tag{6.27}$$

Solving for $\boldsymbol{w}$ we obtain

$$\boldsymbol{w}_{ML} = \hat{\boldsymbol{w}}_{\text{OLS}} = (\boldsymbol{\Phi}^T \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T \boldsymbol{t} \tag{6.28}$$

The corresponding solution $\hat{\boldsymbol{w}}_{\text{OLS}}$ to this linear system of equations is called the **ordinary least squares** or **OLS** or **normal equation** solution. When $\mathcal{D}$ is small(for example, $N < 1000$), we can use the equation to compute $\boldsymbol{w}$ directly. Make the bias parameter explicit,the error function

$$E_D(\boldsymbol{w}) = \frac{1}{2} \sum_{n=1}^{N} \{t_n - w_0 - \sum_{j=1}^{M-1} w_j \phi_j(\boldsymbol{x}_n)\}^2 \tag{6.29}$$

Setting the derivative with respect to w_0 equal to zero,we obtain

$$w_0 = \bar{t} - \sum_{j=1}^{M-1} w_j \bar{\phi}_j \tag{6.30}$$

Then we have

$$y = \boldsymbol{w}^T \boldsymbol{\phi} \tag{6.31}$$

$$y = \sum_{j=1}^{M} w_j \phi_j + w_0 \tag{6.32}$$

$$y = \sum_{j=1}^{M} w_j \phi_j + \bar{t} - \sum_{j=1}^{M-1} w_j \bar{\phi}_j \tag{6.33}$$

$$y - \bar{y} = \sum_{j=1}^{M} w_j (\phi_j - \bar{\phi}_j) \tag{6.34}$$

which indicates that we can normalize the features by subtracting the mean of $\boldsymbol{\Phi}$ while training to reduce computation complexity but obtain the same weight parameter.

6.3.1.1 Matrix Calculus

With regard to matrix calculus, when performing **gradient** based updates, we represent matrix derivatives in **denominator layout**. Thus the **dimension** of the matrix, which we are computing derivatives with respect to, will be consistent with its gradient matrix. Denominator layout convention will make it convenient for use to compute complex derivatives with chain rule and use the result to update old parameters directly, without transposing operation.

There are multiple ways to derive matrix differential calculus:

1. From differential definition, the limit of $\frac{df}{d\boldsymbol{x}}$ when dx approaches 0. Take the difference $df = f(\boldsymbol{x} + d\boldsymbol{x}) - f(\boldsymbol{x})$, and compute $\lim_{\boldsymbol{h} \to 0} \frac{df}{d\boldsymbol{x}}$.
2. According to matrix multiplication definition, we can expand matrix cell to **sum of terms**. Then use 1 dimensional calculus, finally, rewrite to matrix form.
3. Construct **full derivative** from a sum of of **partial derivatives**. In this way, we can decompose a complex formula into several simple ones.

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

$$\frac{\partial}{\partial A} AB = B^T \tag{6.35}$$

$$\frac{\partial}{\partial A^T} f(A) = \left[\frac{\partial}{\partial A} f(A) \right]^T \tag{6.36}$$

$$\frac{\partial}{\partial A} ABA^T C = CAB + C^T AB^T \tag{6.37}$$

$$\frac{\partial}{\partial A} |A| = |A|(A^{-1})^T \tag{6.38}$$

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

To see why $\frac{\partial \boldsymbol{AB}}{\partial \boldsymbol{A}} = \boldsymbol{B}^T$, we use dimension analysis. Denote $f = f(\boldsymbol{C})$. Assume $\boldsymbol{A}$ is of dimension $m \times n$, $\boldsymbol{B}$ is of dimension $n \times p$, and of course, $\boldsymbol{C}$ is of dimension $n \times p$. Take the derivative of f with respect to $\boldsymbol{A}$:

$$\underbrace{\frac{\partial f}{\partial \boldsymbol{A}}}_{m \times n} = \underbrace{\frac{\partial f}{\partial \boldsymbol{C}}}_{m \times p} \underbrace{\frac{\partial \boldsymbol{C}}{\partial \boldsymbol{A}}}_{p \times n} \tag{6.39}$$

$$\tag{6.40}$$

And $\boldsymbol{B}^T$ is of size $p \times n$, the whole function mapping is linear. Also, we can explicitly write matrix elements in sum of terms to show that.

Another **vector** $-$ **vector** differentiation identity, if $\mathbf{A}$ is not a function of $\boldsymbol{x}$ then

$$\frac{d\boldsymbol{X}^T A \boldsymbol{X}}{d\boldsymbol{X}} \tag{6.41}$$

$$= \frac{d\boldsymbol{Y}^T A \boldsymbol{X}}{d\boldsymbol{X}} \tag{6.42}$$

$$= \frac{\partial \boldsymbol{Y}^T A \boldsymbol{X}}{\partial \boldsymbol{Y}} \frac{\partial \boldsymbol{Y}}{\partial \boldsymbol{X}} + \frac{\partial \boldsymbol{Y}^T A \boldsymbol{X}}{\partial \boldsymbol{X}} \frac{\partial \boldsymbol{X}}{\partial \boldsymbol{X}} \tag{6.43}$$

$$= \frac{\partial \boldsymbol{X}^T \boldsymbol{A}^T \boldsymbol{Y}^T}{\partial \boldsymbol{Y}} \boldsymbol{I} + \frac{\partial \boldsymbol{Y}^T A \boldsymbol{X}}{\partial \boldsymbol{X}} \boldsymbol{I} \tag{6.44}$$

$$= \boldsymbol{X}^T \boldsymbol{A}^T + \boldsymbol{Y}^T \boldsymbol{A} \tag{6.45}$$

$$= \boldsymbol{X}^T \boldsymbol{A}^T + \boldsymbol{X}^T \boldsymbol{A} \tag{6.46}$$

$$= \boldsymbol{X}^T (\boldsymbol{A} + \boldsymbol{A}^T) \tag{6.47}$$

$$= (\boldsymbol{A} + \boldsymbol{A}^T)\boldsymbol{X}, \text{ In denominator layout} \tag{6.48}$$

Making use of **partial derivative**, **chain rule** still holds.

6.3.1.2 Derivation directly from Matrix Calculus

The previous derivation is based on the sum of terms arithmetic. Now we derive it from matrix calculus.

Proof. Let's drop constants with respect to $\boldsymbol{w}$ and NLL can be written as

$$\begin{aligned}
\text{NLL}(\boldsymbol{w}) &= \frac{1}{2}\sum_{i=1}^{N}(\boldsymbol{t}_i - \boldsymbol{w}^T \phi_i)^2 \\
&= \frac{1}{2}(\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{t})^T(\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{t}) \\
&= \frac{1}{2}(\boldsymbol{w}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{w}^T\boldsymbol{\Phi}^T\boldsymbol{t} - \boldsymbol{t}^T\boldsymbol{\Phi}\boldsymbol{w} + \boldsymbol{t}^T\boldsymbol{t}) \\
&= \frac{1}{2}(\boldsymbol{w}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{w}^T\boldsymbol{\Phi}^T\boldsymbol{t} - \boldsymbol{t}^T\boldsymbol{\Phi}\boldsymbol{w}) \\
&= \frac{1}{2}\boldsymbol{w}^{\boldsymbol{T}}(\boldsymbol{\Phi}^T\boldsymbol{\Phi})\boldsymbol{w} - \boldsymbol{w}^T(\boldsymbol{\Phi}^T\boldsymbol{t}) \\
\Rightarrow \frac{\partial \text{NLL}}{\partial \boldsymbol{w}} &= \frac{1}{2}((\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \boldsymbol{\Phi}^T\boldsymbol{\Phi})\boldsymbol{w} - 2\boldsymbol{\Phi}^T\boldsymbol{t}) \\
&= \frac{1}{2}(\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{\Phi}^T\boldsymbol{t}) \\
\frac{\partial \text{NLL}}{\partial \boldsymbol{w}} &= 0 \\
\Rightarrow \boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{\Phi}^T\boldsymbol{t} &= 0 \\
\Rightarrow \boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{w} &= \boldsymbol{\Phi}^T\boldsymbol{t} \\
\Rightarrow \boldsymbol{w}_{\text{ML}} &= (\boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\boldsymbol{t}
\end{aligned}$$

Equation 6.49 is known as the **normal equation**.

6.3.2 Geometric interpretation

See Figure 6.1. Given that

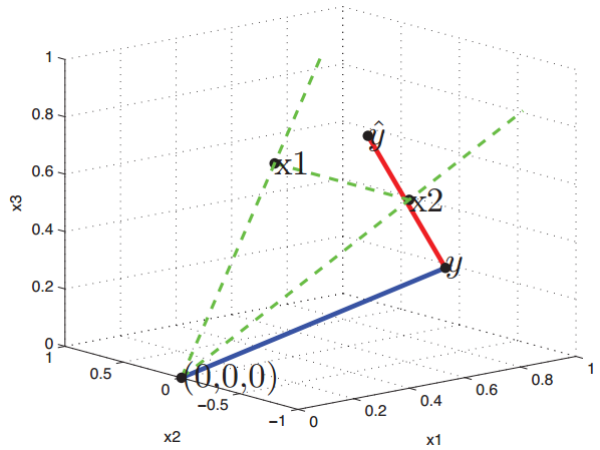


Fig. 6.1: Graphical interpretation of least squares for $N = 3$ examples and $D = 2$ features. $\tilde{\boldsymbol{x}}_1$ and $\tilde{\boldsymbol{x}}_2$ are vectors in $\mathbb{R}^3$; together they define a 2D plane. $\boldsymbol{t}$ is also a vector in $\mathbb{R}^3$ but does not lie on this 2D plane. The orthogonal projection of $\boldsymbol{t}$ onto this plane is denoted $\hat{\boldsymbol{t}}$. The red line from $\boldsymbol{t}$ to $\hat{\boldsymbol{t}}$ is the residual, whose norm we want to minimize. For visual clarity, all vectors have been converted to unit norm.

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x_1}^T \\ \boldsymbol{x_2}^T \\ \cdots \\ \boldsymbol{x_N}^T \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{x_1}} & \tilde{\boldsymbol{x_1}} & \ldots & \tilde{\boldsymbol{x_D}} \end{bmatrix} \tag{6.49}$$

$$\boldsymbol{t} = \begin{bmatrix} t_1 \\ t_2 \\ \cdots \\ t_n \end{bmatrix} \tag{6.50}$$

We seek a vector $\hat{\boldsymbol{t}} \in \mathbb{R}^N$ that lies in the column linear space of $\boldsymbol{X}$ and is as close as possible to $\boldsymbol{t}$,i.e.,we want to find

$$\hat{\boldsymbol{t}} \in span(\boldsymbol{X}) \tag{6.51}$$

$$\Rightarrow \hat{\boldsymbol{t}} = \boldsymbol{X}\boldsymbol{w} = w_1\tilde{\boldsymbol{x_1}} + \cdots + w_D\tilde{\boldsymbol{x_D}} \tag{6.52}$$

$$\hat{\boldsymbol{t}} = \arg \min_{\hat{\boldsymbol{t}} \in \text{span}(\{\tilde{\boldsymbol{x_1}}, \ldots, \tilde{\boldsymbol{x_D}}\})} \tag{6.53}$$

To minimize the norm of the residual, $\boldsymbol{t} - \hat{\boldsymbol{t}}$, we want the residual vector to be orthogonal to every column of $\boldsymbol{X}$,so $\tilde{\boldsymbol{x}}_j(\boldsymbol{t} - \hat{\boldsymbol{t}}) = 0$ for $j = 1 : D$. Hence

$$\begin{aligned} \tilde{\boldsymbol{x}}_j(\boldsymbol{t} - \hat{\boldsymbol{t}}) = 0 &\Rightarrow \boldsymbol{X}^T(\boldsymbol{t} - \boldsymbol{X}\boldsymbol{w}) = 0 \\ &\Rightarrow \boldsymbol{w} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{t} \end{aligned} \tag{6.54}$$

6.3.3 Sequential learning

Stochastic gradient descent(SGD), is also known as sequential gradient descent. Batch techniques,such as maximum likelihood, involves processing the entire training set in one go(pass), which can be computationally costly for large data sets. For sufficiently large datasets, it's worthwhile to use **sequential** algorithms,known as **on-line** algorithms.

If the error function comprises a sum over data points $E = \sum_n En$, then after presentation of pattern n, the stochastic gradient descent algorithm updates the parameter vector w using

$$E = \sum_n E_n \tag{6.55}$$

$$\boldsymbol{w}^{\tau+1} = \boldsymbol{w}^{\tau} - \eta \nabla E_n \tag{6.56}$$

where τ denotes the iteration number,the η is a learning rate parameter.For the case of sum-of-squares error function,this gives

$$\boldsymbol{w}^{\tau+1} = \boldsymbol{w}^{\tau} - \eta (t_n - \boldsymbol{w}^{(\tau)T} \phi_n)\phi_n \tag{6.57}$$

where $\phi_n = \phi(\boldsymbol{x_n})$.This is known as the **least-mean-squares** or the LMS algorithm. Notice that the gradient vector ∇E_n here, is different with the one we used to derive the closed-form solution, which was using **numerator layout** to make it convenient to solve the equation.

$$\because \frac{\partial}{\partial w_i}\text{NLL}(\boldsymbol{w}) = \sum_{i=1}^{N}(\boldsymbol{w}^T \boldsymbol{x}_i - y_i)x_{ij} \tag{6.58}$$

$$\begin{aligned} \therefore w_j =& w_j - \alpha \frac{\partial}{\partial w_j}\text{NLL}(\boldsymbol{w}) \\ =& w_j - \sum_{i=1}^{N} \alpha(\boldsymbol{w}^T \boldsymbol{x}_i - y_i)x_{ij} \end{aligned} \tag{6.59}$$

$$\therefore \boldsymbol{w} = \boldsymbol{w} - \alpha(\boldsymbol{w}^T \boldsymbol{x}_i - y_i)\boldsymbol{x} \tag{6.60}$$

6.4 Ridge regression(MAP)

One problem with ML estimation is that it can result in over-fitting. In this section, we discuss a way to ameliorate this problem by using MAP estimation with a Gaussian prior.

6.4.1 Basic idea

We can encourage the parameters to be small, thus resulting in a smoother curve, by using a zero-mean Gaussian prior:

$$p(\boldsymbol{w}) = \prod_j \mathcal{N}(w_j|0, \tau^2) \tag{6.61}$$

where $1/\tau^2$ controls the strength of the prior. The corresponding MAP estimation problem becomes

$$\arg\max_{\boldsymbol{w}} \sum_{i=1}^{N} \log \mathcal{N}(t_i|w_0 + \boldsymbol{w}^T \phi_i, \sigma^2) + \sum_{j=1}^{D} \log \mathcal{N}(w_j|0, \tau^2) \tag{6.62}$$

This is equivalent to minimizing the following

$$J(\boldsymbol{w}) = E_D(\boldsymbol{w}) + \lambda E_W(\boldsymbol{w}) \tag{6.63}$$

$$= \frac{1}{N}\sum_{i=1}^{N}(y_i - (w_0 + \boldsymbol{w}^T\boldsymbol{x}_i))^2 + \lambda\|\boldsymbol{w}\|^2, \lambda \triangleq \frac{\sigma^2}{\tau^2} \tag{6.64}$$

where the first term is the MSE/ NLL as usual, and the second term, $\lambda \geq 0$, is the regularization coefficient that controls the complexity penalty. Here $E_W(\boldsymbol{w})$ is one of the simplest forms of regularizer given by the sum-of-squares of the weight vector elements.

$$E_W(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} \tag{6.65}$$

This particular choice of regularizer is known as **weight decay**.

The corresponding solution is given by

$$\hat{\boldsymbol{w}}_{\text{ridge}} = (\lambda \boldsymbol{I}_D + \boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\boldsymbol{t} \tag{6.66}$$

This technique is known as **ridge regression**,or **penalized least squares**. In general, adding a Gaussian prior to the parameters of a model to encourage them to be small is called ℓ_2 **regularization** or **weight decay**. Note that the offset term w_0 is not regularized, since this just affects the height of the function, not its complexity.

We will consider a variety of different priors in this book. Each of these corresponds to a different form of **regularization**. This technique is very widely used to prevent overfitting.

6.4.2 Multiple outputs

Now consider the case where we wish to predict $K > 1$ target variables,which we denote collectively by the target vector $\boldsymbol{t}$.To use the same set of basis functions to model all the components of the target vector so that

$$\boldsymbol{y}(\boldsymbol{x}, \boldsymbol{w}) = \boldsymbol{W}^T\phi(x) \tag{6.67}$$

where $\boldsymbol{y}$ is a K-dimensional column vector,$\boldsymbol{W}$ is an $M \times K$ matrix of parameters.

The conditional distribution of the target vector is an isotropic Gaussian

$$p(\boldsymbol{t}|\boldsymbol{x}, \boldsymbol{W}, \beta) = \mathcal{N}(\boldsymbol{t}|\boldsymbol{W}^T\phi(x, \beta^{-1}\boldsymbol{I})) \tag{6.68}$$

The log likelihood function is then given by

$$\ln p(\boldsymbol{T}|\boldsymbol{X}, \boldsymbol{W}, \beta) = \sum_{n=1}^{N}\ln\mathcal{N}(\boldsymbol{t_n}|\boldsymbol{W}^T\phi(x_n), \beta^{-1}\boldsymbol{I}) \tag{6.69}$$

$$= \frac{NK}{2}\ln(\frac{\beta}{2\pi}) - \frac{\beta}{2}\sum_{n=1}^{N}\parallel \boldsymbol{t_n} - \boldsymbol{W}^T\phi(x_n)\parallel^2 \tag{6.70}$$

We maximize this function with respect to $\boldsymbol{W}$,giving

$$\boldsymbol{W}_{ML} = (\boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\boldsymbol{T} \tag{6.71}$$

6.4.3 Numerically stable computation *

$$\hat{\boldsymbol{w}}_{\text{ridge}} = \boldsymbol{V}(\boldsymbol{Z}^T\boldsymbol{Z} + \lambda\boldsymbol{I}_N)^{-1}\boldsymbol{Z}^T\boldsymbol{y} \tag{6.72}$$

6.4.4 Connection with PCA *

6.4.5 Regularization effects of big data

Regularization is the most common way to avoid overfitting. However, another effective approach which is not always available is to use lots of data. It should be intuitively obvious that the more training data we have, the better we will be able to learn.

In domains with lots of data, simple methods can work surprisingly well (Halevy et al. 2009). However, there are still reasons to study more sophisticated learning methods, because there will always be problems for which we have little data. For example, even in such a data-rich domain as web search, as soon as we want to start personalizing the results, the amount of data available for any given user starts to look small again (relative to the complexity of the problem).

6.5 The Bias-Variance Decomposition

6.5.0.1 representation

Data representation: N observations of x,wiritten $X \equiv (x_1, x_2, ..., x_n)^T$ together with corresponding observations of the values of t,denoted $t \equiv (t_1, t_2, ...t_N)^T$.

$$t_i = f(\boldsymbol{x_i}) + \boldsymbol{\varepsilon} \tag{6.73}$$

where the noise ε has zero mean and variance σ^2. Find a function

$$\hat{f}(x)$$

that approximates the true function

$$t = f(\boldsymbol{x})$$

as well as possible.Make "as well as possible" precise by measuring the mean squared error between y and $\hat{f}(x)$,we want $(t - \hat{f}(x))^2$ to be minimal. Hypothesis function(model representation): $y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + ... + w_M x^M = \sum_{j=0}^{M} w_j x^j$ M is the order of the polynomial,and x^j denotes x raised to the power of j.The polynomial coefficients $w_0, ...w_M$ are collectively denoted by the vector $\mathbf{w}$.

Error Function(Sum of squares of errors between predictions $y(x_n, w)$ for each data point x_n and the corresponding target values t_n,so that we minimize:

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2 \tag{6.74}$$

root-mean-squre(RMS) error defined by

$$E_{RMS} = \sqrt[2]{2E(\mathbf{w}^*)/N} \tag{6.75}$$

Penalized(regularized) error function

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w} - t_n\}^2 + \frac{1}{2} \parallel \mathbf{w} \parallel^2 \tag{6.76}$$

where $\parallel \mathbf{w} \parallel^2 \equiv \mathbf{w}^T \mathbf{w} = w_0^2 + w_1^2 + ... + w_M^2$

6.5.0.2 Loss function for regression

$$\mathbb{E}[L] = \iint L(t, y(\mathbf{x})) p(\mathbf{x}, t) d\mathbf{x} dt \tag{6.77}$$

A common choice of loss function in squared loss given by

$$L(t, y(\mathbf{x})) = \{y(\mathbf{x}) - t\}^2 \tag{6.78}$$

$$\mathbb{E}[L] = \iint \{y(\mathbf{x}) - t\}^2 p(\mathbf{x}, t) d\mathbf{x} dt. \tag{6.79}$$

Minimize $\mathbb{E}[L]$ by using the calculus of variations to give

$$\frac{\delta \mathbb{E}[L]}{\delta y(\mathbf{x}))} = 2 \int \{y(\mathbf{x} - t)\} p(\mathbf{x}, t) dt = 0 \tag{6.80}$$

Solving for $y(\mathbf{x})$ and using the sum and product rules of probability,we obtain

$$y(\mathbf{x}) = \frac{\int t p(\mathbf{x}, t) dt}{p(\mathbf{x})} = \int t p(t|\mathbf{x}) dt = \mathbb{E}[t|\mathbf{x}] \tag{6.81}$$

Let's derive this result in a slightly different way.Armed with knowledge that the optimal solution is the conditional expectation,we can expand the square term as follows

$$\{y(\mathbf{x} - t)\}^2 = \{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}] + \mathbb{E}[t|\mathbf{x}] - t)\}^2 = \{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}]\}^2 + 2\{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}]\}\{\mathbb{E}[t|\mathbf{x}] - t\} + \{\mathbb{E}[t|\mathbf{x}] - t\}^2 \tag{6.82}$$

where,$\mathbb{E}[t|\mathbf{x}]$ denote $\mathbb{E}_t[t|\mathbf{x}]$.Substitute into the loss function and perform the integral over t,we see the cross-term vanishes

$$\mathbb{E}[L] = \iint \{y(\mathbf{x}) - t\}^2 p(\mathbf{x}, t) d\mathbf{x} dt \tag{6.83}$$

$$= \int \{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}]\}^2 p(\mathbf{x}) d\mathbf{x} + \int \{\mathbb{E}[t|\mathbf{x}] - t\}^2 p(\mathbf{x}) d\mathbf{x} \tag{6.84}$$

$$= \int \{y(\mathbf{x}) - h(\mathbf{x})\}^2 p(\mathbf{x}) d\mathbf{x} + \int \{h(\mathbf{x}) - t\}^2 p(\mathbf{x}) d\mathbf{x} \tag{6.85}$$

$$= \int \{y(\mathbf{x}) - h(\mathbf{x})\}^2 p(\mathbf{x}) d\mathbf{x} + \int \{h(\mathbf{x}) - t\}^2 p(\mathbf{x}, t) d\mathbf{x} dt \tag{6.86}$$

$$\tag{6.87}$$

6.5.0.3 Decomposition

For a popular choice,we use squared loss function,for which the optimal prediction is given by the conditional expectation,which we denote by h(**x**) and which is given by

$$h(\mathbf{x}) = \mathbb{E}[t|\mathbf{x}] = \int t p(t|\mathbf{x}) dt \tag{6.88}$$

Consider the integrand of the first term of 6.83,which for particular data set D takes the form

$$\{y(\mathbf{x}; D) - h(\mathbf{x})\}^2 \tag{6.89}$$

This quantity will be dependent on the particular data set D,so we take its average over the ensemble of data sets. If we add and subtract the quantity $\mathbb{E}_{\mathbb{D}}[y(\mathbf{x}; D)]$ inside the braces,and then expand,we obtain

$$\{y(\mathbf{x}; D) - h(\mathbf{x})\}^2 \tag{6.90}$$

$$= \{y(\mathbf{x}; D) - \mathbb{E}_{\mathbb{D}}[y(\mathbf{x}; D)] + \mathbb{E}_{\mathbb{D}}[y(\mathbf{x}; D)] - h(\mathbf{x})\}^2 \tag{6.91}$$

$$= \{y(\mathbf{x}; D) - \mathbb{E}_{\mathbb{D}}[y(\mathbf{x}; D)]\}^2 + \{\mathbb{E}_{\mathbb{D}}[y(\mathbf{x}; D)] - h(\mathbf{x})\}^2 + 2\{y(\mathbf{x}; D) - \mathbb{E}_{\mathbb{D}}[y(\mathbf{x}; D)]\}\{\mathbb{E}_{\mathbb{D}}[y(\mathbf{x}; D)] - h(\mathbf{x})\} \tag{6.92}$$

$$= \underbrace{\{y(\mathbf{x}; D) - \mathbb{E}_{\mathbb{D}}[y(\mathbf{x}; D)]\}^2}_{\text{variance}} + \underbrace{\{\mathbb{E}_{\mathbb{D}}[y(\mathbf{x}; D)] - h(\mathbf{x})\}^2}_{(bias)^2} + 0 \tag{6.93}$$

$$\tag{6.94}$$

The decomposition of the expected squared loss

$$\text{expected loss} = (bias)^2 + variance + noise \tag{6.95}$$

where

$$(bias)^2 = ... \tag{6.96}$$
$$variance = ... \tag{6.97}$$
$$noise = ... \tag{6.98}$$

The function $y(\mathbf{x})$ we seek to determine enters only the first term,which will be minimized when $y(\mathbf{x})$ is equal to $\mathbb{E}[t|\mathbf{x}]$,in which case this term will vanish.The second term is the variance of distribution of t,averaged over $\mathbf{x}$,representing the intrinsic variabilility of the target data and can be regarded as noise.It's the irreducible minimum value of the loss function.

More sophisticated loss function,Minkowski loss

$$\mathbb{E}[L_q] = \iint |y(\mathbf{x}) - t|^q p(\mathbf{x},t) d\mathbf{x} dt \tag{6.99}$$

6.6 Bayesian linear regression

Hold-out data can be used to determine model complexity but it will be computationally expensive and wasteful of valuable data.We therefore turn to a Bayesian treatment of linear regression,which will avoid the over-fitting problem of maximum likelihood,and which will also lead to automatic methods of determining model complexity using the training data.

6.6.1 Parameter distribution

First introduce a prior probability distribution over the model parameters.The likelihood function $p(\boldsymbol{t}|\boldsymbol{w})$ defined by 6.9 is the exponential of a quadratic function of $\boldsymbol{w}$,so the corresponding conjugate prior is Gaussian

$$p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{m_0}, \boldsymbol{S}_0) \tag{6.100}$$

The posterior distribution is proportional to the product of the likelihood and the prior.And the posterior will also be Gaussian due to the choice of conjugate Gaussian prior,which is derived in 2.5.

$$p(\boldsymbol{w}|\boldsymbol{t}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{m}_N, \boldsymbol{S}_N) \tag{6.101}$$

where

$$\boldsymbol{m}_N = \boldsymbol{S}_N(\boldsymbol{S}_0^{-1}\boldsymbol{m}_0 + \beta\boldsymbol{\Phi}^T\boldsymbol{t}) \tag{6.102}$$
$$\boldsymbol{S}_N^{-1} = \boldsymbol{S}_0^{-1} + \beta\boldsymbol{\Phi}^T\boldsymbol{\Phi} \tag{6.103}$$

Thus the maximum posterior weigh vector is simply given by $\boldsymbol{w}_{MAP} = \boldsymbol{m}_N$.If $N = 0$ then the posterior distribution reverts to the prior.Furthermore,if data points arrive sequentially,then the posterior distribution at any stage acts as the prior distribution,such that the new posterior is again given.

A zero-mean isotropic Gaussian governed by a single precision parameter α so that

$$p(\boldsymbol{w}|\alpha) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \alpha^{-1}\boldsymbol{I}) \tag{6.104}$$

for which the posterior is given by

$$\boldsymbol{m}_N = \beta \boldsymbol{S}_N \boldsymbol{\Phi}^T \boldsymbol{t} \tag{6.105}$$

$$\boldsymbol{S}_N^{-1} = \alpha \boldsymbol{I} + \beta \boldsymbol{\Phi}^T \boldsymbol{\Phi} \tag{6.106}$$

The log of posterior distribution is given by the sum of the log likelihood and the log of prior and,as a function of $\boldsymbol{w}$,takes the form

$$\log p(\boldsymbol{w}|\boldsymbol{t}) = -\frac{\beta}{2}\sum_{n=1}^{N}\{t_n - \boldsymbol{w}^T\phi(\boldsymbol{x}_n)\}^2 - \frac{\alpha}{2}\boldsymbol{w}^T\boldsymbol{w} + const \tag{6.107}$$

Maximization of this posterior w.r.t $\boldsymbol{w}$ is equivalent to minimization of the sum-of-squares error function with the addition of a quadratic regularization term,corresponding with $\lambda = \alpha/\beta$.

6.6.2 Predictive distribution

Evaluate the **predictive distribution** defined by

$$p(t|\boldsymbol{t},\alpha,\beta) = \int p(t|\boldsymbol{w},\beta)p(\boldsymbol{w}|\boldsymbol{t},\alpha,\beta)d\boldsymbol{w} \tag{6.108}$$

The right hand quantity can be interpreted as the marginalization by integration of the joint distribution of $p(t,\boldsymbol{w}|\boldsymbol{t},\alpha,\beta)$ over $\boldsymbol{w}$.And the joint distribution can be solved by the Bayesian theorem for Gaussian. The conditional distribution $p(t|\boldsymbol{x},\boldsymbol{w},\beta)$ of the target variable is given by 6.5,and the posterior weight distribution is given by 6.101.This involves the **convolution** of two Gaussian distributions.The predictive distribution take the form

$$p(t|x,\boldsymbol{t},\alpha,\beta) = \mathcal{N}(t|\boldsymbol{m}_N^T\phi(x),\sigma_N^2(\boldsymbol{x})) \tag{6.109}$$

where the variance $\sigma_N^2(\boldsymbol{x})$ is given by

$$\sigma_N^2(\boldsymbol{x}) = \frac{1}{\beta} + \phi(\boldsymbol{x})^T\boldsymbol{S}_N\phi(\boldsymbol{x}) \tag{6.110}$$

The first term represents the noise on the data whereas the second term reflects the uncertainty associated with the parameters $\boldsymbol{w}$. $\sigma_{N+1}^2(\boldsymbol{x}) \leq \sigma_N(\boldsymbol{x})$.In the limit $N \to \infty$,the second term goes to zero.

Note that if both $\boldsymbol{w}$ and β are treated as unknown,then we can introduce a conjugate prior distribution $p(\boldsymbol{w},\beta)$ given by Gaussian-gamma distribution,leading to a Student's t-distribution predictive distribution.

6.6.3 Equivalent kernel

The posterior mean solution **??** has an interpretation that will set stage for kernel methods,including Gaussian processes.The predictive mean:

$$y(\boldsymbol{x},\boldsymbol{w}) = \boldsymbol{m}_N^T\phi(\boldsymbol{x}) = \beta\phi(\boldsymbol{x})^T\boldsymbol{S}_N\boldsymbol{\Phi}^T\boldsymbol{t} = \sum_{n=1}^{N}\beta\phi(\boldsymbol{x})^T\boldsymbol{S}_N\phi(\boldsymbol{x}_n)t_n = \sum_{n=1}^{N}k(\boldsymbol{x},\boldsymbol{x}_n)t_n \tag{6.111}$$

where

$$k(\boldsymbol{x},\boldsymbol{x}') = \beta\phi(\boldsymbol{x})^T\boldsymbol{S}_N\phi(\boldsymbol{x}') \tag{6.112}$$

is known as the **smoother matrix** or **equivalent kernel**.Regression functions,such as this,which make predictions by taking linear combinations of the training set target values are known **linear smoothers**.The kernel functions are localized around x(local evidence weight more than distant evidence).

The covariance between $y(\boldsymbol{x})$ and $y(\boldsymbol{x}')$ is given by

$$cov[y(\boldsymbol{x}),y(\boldsymbol{x}')] = cov[\phi(\boldsymbol{x})^T\boldsymbol{w},\boldsymbol{w}^T\phi(\boldsymbol{x}')] \tag{6.113}$$

$$= \phi(\boldsymbol{x})^T\boldsymbol{S}_N\phi(\boldsymbol{x}') = \beta^{-1}k(\boldsymbol{x},\boldsymbol{x}') \tag{6.114}$$

We see that the predictive mean at nearby points will be highly correlated.

The formulation of linear regression in terms of kernel functions suggests an alternative approach,called **Gaussian process**:Instead of introducing a set of basis functions,which implicitly determines an equivalent kernel,we can instead define a localized kernel directly and use this to make predictions for new input vectors $\boldsymbol{x}$,given the observed training set.

The effective kernel defines the weights by which the training set target values are combined in order to make a prediction at a new value of x,and it can be shown that these weights sum to one,

$$\sum_{n=1}^{N} \|(\boldsymbol{x}, \boldsymbol{x}_n) = 1 \tag{6.115}$$

for all values of $\boldsymbol{x}$.

General kernel functions share an important property,namely that it can be expressed in the form an inner product with respect to a vector $\Psi(\boldsymbol{x})$ of nonlinear functions,so that

$$\|(\boldsymbol{x}, \boldsymbol{z}) = \Psi(\boldsymbol{x})^T \Psi(\boldsymbol{z}) \tag{6.116}$$

where $\Psi(\boldsymbol{x}) = \beta^{1/2} \boldsymbol{S}_n^{1/2} \phi(\boldsymbol{x})$

6.7 Bayesian Model Comparison

The over-fitting associated with maximum likelihood can be avoided by **marginalizing(summing or integrating)** over the model parameters instead of making point estimates of their values,no need for validation.

Suppose we wish to compare a set of L models $\{\mathcal{M}_i\}, i = 1, ...L$ over observed data $\mathcal{D}$.Evaluate the posterior distribution

$$p(\mathcal{M}_i|\mathcal{D}) \propto p(\mathcal{M}_i) p(\mathcal{D}|\mathcal{M}_i) \tag{6.117}$$

The prior allows us to express a preference for different models.$p(\mathcal{D}|\mathcal{M}_i)$ is the **model evidence**,which expresses the preference shown by the data for different models.It is also called the **marginal likelihood**.The ratio of model evidences $p(\mathcal{D}|\mathcal{M}_i)/p(\mathcal{D}|\mathcal{M}_j)$ for two models is **Bayes factor**. The predictive distribution

$$p(t|\boldsymbol{x}, \mathcal{D}) p(\mathcal{M}_i|\mathcal{D}) = p(t|\boldsymbol{x}, \mathcal{M}_i, \mathcal{D}) p(\mathcal{M}_i|\mathcal{D}) \tag{6.118}$$

$$\Rightarrow \sum_{i=1}^{L} p(t|\boldsymbol{x}, \mathcal{D}) p(\mathcal{M}_i|\mathcal{D}) = \sum_{i=1}^{L} p(t|\boldsymbol{x}, \mathcal{M}_i, \mathcal{D}) p(\mathcal{M}_i|\mathcal{D}) \tag{6.119}$$

$$\Rightarrow p(t|\boldsymbol{x}, \mathcal{D}) = \sum_{i=1}^{L} p(t|\boldsymbol{x}, \mathcal{M}_i, \mathcal{D}) p(\mathcal{M}_i|\mathcal{D}) \tag{6.120}$$

$$\tag{6.121}$$

This is an example of a **mixture distribution** in which the overall predictive distribution is obtained by averaging the predictive distributions of individual models weighted by the posterior probabilities $p(\mathcal{M}_i|\mathcal{D})$ of those models.

To use the single most probable model alone to make predictions is known as **model selection**.Model evidence

$$p(\mathcal{D}|\mathcal{M}_i) = \int p(\mathcal{D}|\boldsymbol{w}) p(\boldsymbol{w}|\mathcal{M}_i) d\boldsymbol{w} \tag{6.122}$$

because

$$p(\boldsymbol{w}|\mathcal{D}, \mathcal{M}_i) = \frac{p(\mathcal{D}|\boldsymbol{w}, \mathcal{M}_i) p(\boldsymbol{w}|\mathcal{M}_i)}{p(\mathcal{D}|\mathcal{M}_i)} \tag{6.123}$$

Assume the posterior distribution over parameters is sharply peaked around the most probable value w_{MAP},with width $\Delta w_{posterior}$,then we can approximate the integral by the value of integrand at its maximum times the width of the peak.And further assume that the prior is flat with width Δw_{prior} so that $p(w) = 1/\Delta w_{prior}$,then we have

$$p(\mathcal{D}) = \int p(\mathcal{D}|w)p(w)dw \simeq p(\mathcal{D}|w_{MAP})p(w_{MAP})\Delta w_{posterior} \simeq p(\mathcal{D}|w_{MAP})\frac{\Delta w_{posterior}}{\Delta w_{prior}} \tag{6.124}$$

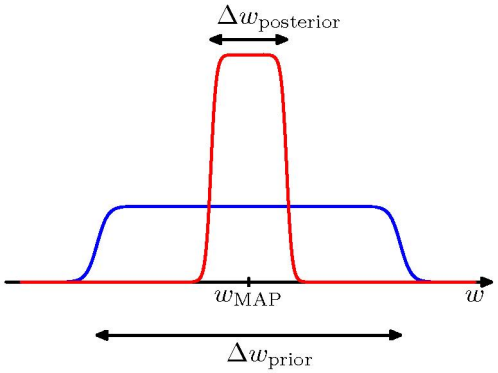


Fig. 6.2: model evidence approximation

and so taking logs

$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|w_{MAP}) + \ln(\frac{\Delta w_{posterior}}{\Delta w_{prior}}) \tag{6.125}$$

If we have a set of M parameters,assuming that all parameters have the same ratio of $\Delta w_{posterior}/\Delta w_{prior}$,we obtain

$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|w_{MAP}) + M\ln(\frac{\Delta w_{posterior}}{\Delta w_{prior}}) \tag{6.126}$$

As we increase the complexity of the model,the first term representing the fit to the data,increases(?),whereas the second term will decrease due to the dependence on M.The optimal model is given by a trade-off between these two competing terms. Average the Bayes factor over the distribution of data sets

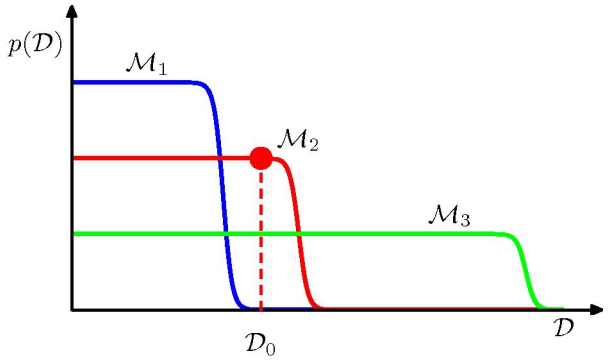


Fig. 6.3: Model complexity increases

$$\int p(\mathcal{D}|\mathcal{M}_1)\ln\frac{p(\mathcal{D}|\mathcal{M}_1)}{p(\mathcal{D}|\mathcal{M}_2)} \tag{6.127}$$

This quantity is $Kullback-Leibler$ divergence,and will be bigger than or equal to 0 if $\mathcal{M}_1$ is the correct model.

6.8 Model Evidence

Chapter 7
Linear Models for Classification

The goal in classification is to take an input vector $\boldsymbol{x}$ and to assign it to one of K discrete classes $\mathcal{C}_k$ where $K = 1, ..., K$.The input space is thereby divided into **decision regions** whose boundaries are called **decision boundaries** or **decision surfaces**.Data sets whose classes can be separated exactly by linear decision surfaces are said to be **linearly separable**.

In Chapter 1,we identified three distinct approaches to the classification problem.The simplest involves constructing a **discriminant function** that directly assigns each vector x to a specific class. A more powerful approach, however, models the **conditional probability distribution** $p(C_k|x)$ in an **inference** stage, and then subsequently uses this distribution to make optimal **decisions**. By separating inference and decision, we gain numerous benefits, as discussed in 1. There are two different approaches to determining the conditional probabilities $p(C_k|x)$. One technique is to model them **directly**, for example by representing them as parametric models and then optimizing the parameters using a training set. Alternatively, we can adopt a generative approach in which we model the **class-conditional densities** given by p(x—Ck), together with the **prior probabilities** p(Ck) for the classes, and then we compute the required posterior probabilities using Bayes theorem

$$p(\mathcal{C}_k|\boldsymbol{x}) = \frac{p(\boldsymbol{x}\mathcal{C}_k)p(\mathcal{C}_k)}{p(\boldsymbol{x})} \tag{7.1}$$

In the linear regression models considered in Chapter 3, the model prediction y(x,w) was given by a linear function of the parameters w. In the simplest case, the model is also linear in the input variables and therefore takes the form y(x) = $\boldsymbol{w}^T x + w_0$, so that y is a real number. For classification problems, however, we wish to predict discrete class labels, or more generally posterior probabilities that lie in the range (0, 1). To achieve this, we consider a generalization of this model in which we transform the linear function of w using a nonlinear function $f()$ so that

$$y(\boldsymbol{x}) = f(\boldsymbol{w}^T \boldsymbol{x} + w_0) \tag{7.2}$$

In the machine learning literature $f(\cdot)$ is known as an **activation function**,whereas its inverse is called a **link function** in the statistics literature.The decision surfaces correspond to y(x) = constant, so that $\boldsymbol{w}^T \boldsymbol{x} + w_0 = constant$ and hence the decision surfaces are linear functions of $\boldsymbol{x}$, even if the function $f()$ is nonlinear. For this reason, the class of models described by (4.3) are called **generalized linear models** (McCullagh and Nelder,1996).However,in contrast to the models used for regression,they are no longer linear in the parameters due to the presence of the nonlinear function $f(\cdot)$.

7.1 Discriminant Functions

A discriminant is a function that takes an input vector $\boldsymbol{x}$ and assigns it to be one of K classes,denoted $\mathcal{C}_k$.

7.1.1 Two classes

Linear discriminant function

$$y(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{x} + w_0 \tag{7.3}$$

where $\boldsymbol{w}$ is called a **weight vector**,and w_0 is *bias*.The negative of the bias is sometimes called **threshold**.

The value of $y(\boldsymbol{x})$ gives a signed measure of the perpendicular distance r of point $\boldsymbol{x}$ from the decision surface.Consider an arbitrary point $\boldsymbol{x}$ and let $\boldsymbol{x}_\perp$ be its orthogonal projection onto the decision surface,then

$$\boldsymbol{x} = \boldsymbol{x}_\perp + r\frac{\boldsymbol{w}}{\| \boldsymbol{w} \|} \tag{7.4}$$

$$\tag{7.5}$$

Multiplying both sides by $\boldsymbol{w}^T$ and adding $\boldsymbol{w}_0$,

$$y(\boldsymbol{x}) = y(\boldsymbol{x}_\perp) + r\frac{\boldsymbol{w}\cdot\boldsymbol{w}}{\| \boldsymbol{w} \|} \tag{7.6}$$

$$= r \| \boldsymbol{w} \| \tag{7.7}$$

$$\therefore r = \frac{y(\boldsymbol{x})}{\| \boldsymbol{w} \|} \tag{7.8}$$

7.1.2 Multiple classes

Consider the extension of linear discriminants to $K > 2$ classes,building a K-class discriminants by combining a number of two-class discriminant functions on which leads to some serious difficulties(Duda and Hart,1973).

Consider the use of $K-1$ classifier each of which solves a two-class problem of separating points int particular class $\mathcal{C}_k$ from points not in that class.This is known as a **one-versus-the-rest** classifier.An alternative is to introduce $K(K-1)/2$ binary discriminant functions,one for every possible pair classes.This is known as a **one-versus-one** classifier.Both of these two approaches run into the problem of ambiguous regions.We can avoid these difficulties by considering a single K-class discriminant comprising k linear functions of the form

$$y_k(\boldsymbol{x}) = \boldsymbol{w_k}^T\boldsymbol{x} + w_{k0} \tag{7.9}$$

and then assigning a point $\boldsymbol{x}$ to class $\mathcal{C}_k$ if $y_k(\boldsymbol{x}) > y_j(\boldsymbol{x})$ for all $j \neq k$.The decision boundary between class $\mathcal{C}_k$ and $\mathcal{C}_j$ is therefore given by $y_k(\boldsymbol{x}) = y_j(\boldsymbol{x})$ and hence corresponds to a $(D-1)$-dimensional hyperplane defined by

$$(\boldsymbol{w_k} - \boldsymbol{w_j})^T + (w_{k0} - w_{j0}) = 0 \tag{7.10}$$

The decision regions of such a discriminant are always singly connected and convex.

7.1.3 Least squares for classification

Even as a discriminant function(where we use it to make decisions directly and dispense with any probabilistic interpretation)it suffers from some severe problems.Least-squares solutions lack robustness to outliers,and this applies equally to the classification application.Additional points(outliers) produce a significant change in the location of the decision boundary.The sum-of-squares error function penalizes predictions that are 'too correct' in that they lie a long way on the correct side of the decision boundary.More than lack of robustness,least-squares solutions may gives poor results on classification problems.

The failure of least squares lies in the fact that it corresponds to maximum likelihood under the assumption of a Gaussian conditional distribution,whereas binary target vectors clearly have a distribution that is far from Gaussian(a Bernoulli).

7.1.4 Fisher's linear discriminant

7.1.4.1 representation

One way to view a linear classification model is in terms of dimensionality reduction.Consider first the case of two classes,and suppose we take the $D-dimensional$ input vector $\boldsymbol{x}$ and project it down to one dimension using

$$y = \boldsymbol{w}^T \boldsymbol{x}. \tag{7.11}$$

If we place a threshold on y and classify $y \geq -w_0$ as class $\mathcal{C}_\infty$,and otherwise class $\mathcal{C}_\in$,then we obtain our standard linear classifier discussed in the previous section.

7.1.4.2 evaluation

By adjusting the components of the weight vector $\boldsymbol{w}$,we can select a projection that maximizes the class separation.To begin with,consider a two-class problem in which there are N_1 points of class $\mathcal{C}_\infty$ and N_2 points of class $\mathcal{C}_\in$,so that the mean vectors of the two classes are given by

$$\boldsymbol{m_1} = \frac{1}{N_1} \sum_{n \in \mathcal{C}_\infty} \boldsymbol{x_n}, \boldsymbol{m_1} = \frac{1}{N_2} \sum_{n \in \mathcal{C}_\in} \boldsymbol{x_n} \tag{7.12}$$

The simplest measure of the separation of the classes,when projected onto $\boldsymbol{w}$,is the separation of the projected class means.This suggests that we might choose $\boldsymbol{w}$ so as to maximize

$$m_2 - m_1 = \boldsymbol{x}^T (\boldsymbol{m_2} - \boldsymbol{m_1}) \tag{7.13}$$

where

$$m_k = \boldsymbol{w}^T \boldsymbol{m_k} \tag{7.14}$$

is the mean of the projected data from class $\mathcal{C}_\parallel$.However this expression can be makde arbitrarily large simply by increasing the magnitude of $\boldsymbol{w}$.To solve this,we could constrain $\mathbf{2}$ to have unit length,so that $\sum_i w_i^2 = 1$.Using a Lagrange multiplier to perform the constrained maximization,we then find that $\boldsymbol{w} \propto (\boldsymbol{m_2} - \boldsymbol{m_1})$.There is still a problem with this approach that the projected data have considerable overlap for strongly nondiagonal convariances of the class distributions.

7.1.4.3 optimization

The idea proposed by Fisher is to **maximize a function that will give a large separation between the projected class means while also giving a small variance within each class,thereby minimizing the class overlap**.The projection formula 7.11 transforms the set of labelled data points in $\boldsymbol{x}$ into a labelled set in the one-dimensional space y.The within-class variance of the transformed data from class $\mathcal{C}_\parallel$ is therefore given by

$$s_k^2 = \sum_{n \in \mathcal{C}_\parallel} (y_n - m_k)^2 \tag{7.15}$$

where

$$y_n = \boldsymbol{w}^T \boldsymbol{x_n} \tag{7.16}$$

We can define the total within-class variance for the whole data set to be simply $s_1^2 + s_2^2$.The Fisher criterion is defined to be the ratio of the between-class variance to the within-class variance and is given by

$$J(\boldsymbol{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} \tag{7.17}$$

We can make the dependence on $\boldsymbol{w}$ explicit by rewrite the Fisher criterion in the form

$$J(\boldsymbol{w}) = \frac{\boldsymbol{w}^T \boldsymbol{S_B} \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S_W} \boldsymbol{w}} \tag{7.18}$$

where $\boldsymbol{S_B}$ is the $between-class$ covariance matrix,given by

$$\boldsymbol{S_B} = (\boldsymbol{m_2} - \boldsymbol{m_1})(\boldsymbol{m_2} - \boldsymbol{m_1})^T \tag{7.19}$$

so

$$\boldsymbol{w}^T \boldsymbol{S_B} \boldsymbol{w} = \boldsymbol{w}^T (\boldsymbol{m_2} - \boldsymbol{m_1})(\boldsymbol{m_2} - \boldsymbol{m_1})^T \boldsymbol{w} = (m_2 - m_1)(m_2 - m_1) \tag{7.20}$$

and $\boldsymbol{S_W}$ is the total $within-class$ covariance matrix,given by

$$\boldsymbol{S_W} = \sum_{\mathcal{C}} \sum_{n \in \mathcal{C}_1} (\boldsymbol{x_n} - \boldsymbol{m_1})(\boldsymbol{x_n} - \boldsymbol{m_1})^T \tag{7.21}$$

so

$$\boldsymbol{w}^T \boldsymbol{S_W} \boldsymbol{w} = \sum_{\mathcal{C}_k} \sum_{n \in \mathcal{C}_k} \boldsymbol{w}^T (\boldsymbol{x_n} - \boldsymbol{m_k})(\boldsymbol{x_n} - \boldsymbol{m_k})^T = \sum_{\mathcal{C}_i} \sum_{n \in \mathcal{C}_k} (y_n - m_k)^2 \tag{7.22}$$

Differentiating 7.18 with respect to $\boldsymbol{w}$,we find that $J(\boldsymbol{w})$ is maximized when

$$(\boldsymbol{w}^T \boldsymbol{S_B} \boldsymbol{w}) \boldsymbol{S_W} \boldsymbol{w} = (\boldsymbol{w}^T \boldsymbol{S_W} \boldsymbol{w}) \boldsymbol{S_B} \boldsymbol{w} \tag{7.23}$$

Rewriting this as

$$\boldsymbol{S_B} \boldsymbol{w} = \lambda \boldsymbol{S_W} \boldsymbol{w} \tag{7.24}$$

where

$$\lambda = \frac{(\boldsymbol{w}^T \boldsymbol{S_B} \boldsymbol{w})}{(\boldsymbol{w}^T \boldsymbol{S_W} \boldsymbol{w})} \tag{7.25}$$

which is called a **generalized eigenvalue** problem solution.

Proof.

$$\because \begin{cases} \dfrac{\partial}{\partial x} \dfrac{f(x)}{g(x)} = \dfrac{f'g - fg'}{g^2}, \text{where } f' = \dfrac{\partial f(x)}{\partial x} \text{ and } g' = \dfrac{\partial g(x)}{\partial x} \\ \dfrac{\partial}{\partial \boldsymbol{x}} \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} = (\boldsymbol{A} + \boldsymbol{A}^T) \boldsymbol{x} \end{cases} \tag{7.26}$$

$$\therefore \tag{7.27}$$

$$\frac{\partial}{\partial \boldsymbol{w}} J(\boldsymbol{w}) = \frac{2\boldsymbol{S_B} \boldsymbol{w} (\boldsymbol{w}^T \boldsymbol{S_W} \boldsymbol{w}) - (\boldsymbol{w}^T \boldsymbol{S_B} \boldsymbol{w}) \boldsymbol{S_W} \boldsymbol{w}}{(\boldsymbol{w}^T \boldsymbol{S_W} \boldsymbol{w})^2} \tag{7.28}$$

Setting the derivative to zero,we can get the result.

In particular,since

$$\lambda \boldsymbol{w} = \boldsymbol{S_W} (\boldsymbol{m_2} - \boldsymbol{m_1})(m_2 - m_1) \tag{7.29}$$

Multiply both sides of 7.24 by ${\boldsymbol{S_W}}^{-1}$ we then obtain

$$\boldsymbol{w} \propto {\boldsymbol{S_W}}^{-1} (\boldsymbol{m_2} - \boldsymbol{m_1}) \tag{7.30}$$

Note that if the within-class covariance is isotropic,so that $\boldsymbol{S_W}$ is proportional to the unit matrix($\boldsymbol{S_W} \propto \boldsymbol{I}$),we find that $\boldsymbol{w}$ is proportional to the difference vector of the class means.

The result 7.30 is known as $Fisher's linear discriminant$,although strictly it is not a discriminant but rather a specific choice of direction for projection of the data down to one dimension.However,the projected data can subsequently be used to construct a discriminant,by choosing a threshold y_0 so that we classify a new point as belonging to $\mathcal{C}_1$ if $y(\boldsymbol{x}) \geq y_0$ and can classify it as belonging to $\mathcal{C}_2$ otherwise.

7.1.5 Extensions to higher dimensions and multiple classes

We can extend the above idea to multiple classes,and to higher dimensional subspaces,by finding a projection $matrix \boldsymbol{W}$ which maps from D to L so as to maximize

$$J(\boldsymbol{W}) = \frac{|\boldsymbol{W}\boldsymbol{\Sigma_B}\boldsymbol{W}^T|}{|\boldsymbol{W}\Sigma_W\boldsymbol{W}^T|} \tag{7.31}$$

where

$$\Sigma_B \triangleq \sum_c \frac{N_c}{N}(\boldsymbol{m_c} - \boldsymbol{m})(\boldsymbol{m_c} - \boldsymbol{m})^T \tag{7.32}$$

$$\Sigma_W \triangleq \sum_c \frac{N_c}{N}\Sigma_c \tag{7.33}$$

$$\Sigma_c \triangleq \sum_{i:y_i=c} (\boldsymbol{x_i} - \boldsymbol{m_c})(\boldsymbol{x_i} - \boldsymbol{m_c})^T \tag{7.34}$$

The solution can be shown to be

$$\boldsymbol{W} = \Sigma_W^{-\frac{1}{2}}\boldsymbol{U} \tag{7.35}$$

where $\boldsymbol{U}$ are the $\boldsymbol{L}$ leading eigenvectors of $\Sigma_W^{-\frac{1}{2}}\Sigma_B\Sigma_W^{-1\frac{1}{2}}$,assuming Σ_W is non-singular.(If it is singular,we can first perform PCA on all the data).

7.1.6 Probabilistic interpretation of FLDA *

7.1.7 Relation to least squares

7.1.8 Fisher's discriminant for multiple classes

7.1.9 The perceptron algorithm

Another example of a linear discriminant model is the perceptron of Rosenblatt(1962),which corresponds to a two-class transformation to give a feature vector $\boldsymbol{\phi}(\boldsymbol{x})$,and this is then used to construct a generalized linear model of the form

$$y(\boldsymbol{x}) = f(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x})) \tag{7.36}$$

where the nonlinear activation function $f(\cdot)$ is given by a step function of the form

$$f(a) = \begin{cases} +1, a \geq 0 \\ -1, a \leq 0 \end{cases} \tag{7.37}$$

7.1.9.1 error function

A natural choice of error function would be the total number of misclassified patterns,which does not lead to a simple learning algorithm because the error is a piecewise constant function of $\boldsymbol{w}$,with discontinuities wherever a change in $\boldsymbol{w}$ causes the decision boundary to move across one of the data points.Methods based on gradient can't be applied.An alternative error function is known as the **perceptron criterion**,given by

$$E_P(\boldsymbol{w}) = -\sum_{n\in\mathcal{M}} \boldsymbol{w}^T\boldsymbol{\phi_n}(\boldsymbol{x})t_n \tag{7.38}$$

where we use $t \in \{-1,+1\}$ coding scheme, $\mathcal{M}$ denotes the set of all misclassified patterns.The perceptron criterion associates zero error with any pattern that is correctly classified,whereas for a misclassified pattern $\boldsymbol{x}$ it tries to minimize the quantity $-\boldsymbol{w}^T\boldsymbol{\phi_n}t_n$.The perceptron learning rule is not guaranteed to reduce the total error function at each stage.

However,the **perceptron convergence theorem** states that if there exists an exact solution(linearly separable),then the perceptron learning algorithm is guaranteed to find an exact solution if a finite number of steps.

Aside from difficulties with the learning algorithm,the perceptron does not provide probabilistic outputs,nor does it generalize readily to $K > 2$ classes.The most important limitation,however,arises from the fact that it is based on linear combinations of fixed basis functions.

7.1.9.2 optimization

We now apply the stochastic gradient descent algorithm to this error function.The change in the weight vector $\boldsymbol{w}$ is then given by

$$\boldsymbol{w}^{\tau+1} = \boldsymbol{w}^{\tau} - \eta \nabla E_P(\boldsymbol{w}) = \boldsymbol{w}^{\tau} + \eta \phi_n t_n \tag{7.39}$$

where η is the learning rate parameter and τ is an integer that indexes the steps of the algorithm.Because the perceptron function $y(\boldsymbol{x}, \boldsymbol{w})$ is unchanged if we multiply $\boldsymbol{w}$ by a constant,so we can set the learning rate η equal to 1 without of generality.

7.2 Probabilistic Generative Models

We turn next to a probabilistic view of classification and show how models with linear decision boundaries arise from simple assumptions about the distribution of the data,adopting a generative approach in which we model the class-conditional densities $p(\boldsymbol{x}|\mathcal{C}_k)$,as well as the class priors $p(\rfloor_k)$,and then use these to compute posterior probabilities $p(\mathcal{C}_k|\boldsymbol{x})$ through Bayes' theorem.

7.2.1 Two classes

Consider first of all the case two classes.The posterior probability for class $\mathcal{C}_1$ can be written as

$$p(\mathcal{C}_1|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\boldsymbol{x}|\mathcal{C}_1)p(\mathcal{C}_1) + p(\boldsymbol{x}|\mathcal{C}_2)p(\mathcal{C}_2)} \tag{7.40}$$

$$= \frac{1}{1+\exp(-a)} = \sigma(a) \tag{7.41}$$

where we have defined

$$a = \ln \frac{p(\boldsymbol{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\boldsymbol{x}|\mathcal{C}_2)p(\mathcal{C}_2)} \tag{7.42}$$

and $\sigma(a)$ is the **logistic sigmoid** function defined by

$$\sigma(a) = \frac{1}{1+\exp(-a)} \tag{7.43}$$

The term 'sigmoid' means S-shaped,sometimes called a 'squashing function' because it maps the whole real axis into a finite interval.It satisfies symmetry property

$$\sigma(-a) = 1 - \sigma(a) \tag{7.44}$$

The inverse of the logistic sigmoid is given by

$$a = \ln(\frac{\sigma}{1-\sigma}) \tag{7.45}$$

and is known as the **logit** function.It represents the log of the ratio of probabilities $ln[p(\mathcal{C}_1|\boldsymbol{x})/p(\mathcal{C}_2|\boldsymbol{x})]$ for the two classes,also known as the **log odds**.

7.2.2 Multiple classes

For the case of $K > 2$ classes,we have posterior

$$p(\mathcal{C}_k|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{\sum_j p(\boldsymbol{x}|\mathcal{C}_j)p(\mathcal{C}_j)} \tag{7.46}$$

$$= \frac{\exp(a_k)}{\sum_j \exp(a_j)} \tag{7.47}$$

which is known as the **normalized exponential** and can be regarded as a multiclass generalization of the logistic sigmoid.Here the quantities a_k are defined by

$$a_k = \ln p(\boldsymbol{x}|\mathcal{C}_k)p(\mathcal{C}_k) \tag{7.48}$$

The normalized function is also known as the **softmax function**,as it represents a smoothed version of the 'max' function because,if $a_k \gg a_j$ for all $j \neq k$,then $p(\mathcal{C}_k|\boldsymbol{x}) \simeq 1$ and,$p(\mathcal{C}_j|\boldsymbol{x}) \simeq 0$.

The following investigate the consequences of choosing specific forms for the class-conditional densities,continuous and discrete inputs.

7.2.3 Continuous inputs

Assume that the class-conditional densities are **Gaussian** sharing the **same covariance matrix** and then explore the resulting form for the posterior probabilities.Thus the density(pdf) for class $\mathcal{C}_k$ is given by

$$p(\boldsymbol{x}|\mathcal{C}_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k)\} \tag{7.49}$$

In the case of two classes,we have

$$a(\boldsymbol{x}) = \log \frac{p(\boldsymbol{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\boldsymbol{x}|\mathcal{C}_2)p(\mathcal{C}_2)} \tag{7.50}$$

$$= -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_2)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2) + \log \frac{p(\mathcal{C}_1)}{p(\mathcal{C}_2)} \tag{7.51}$$

$$= (\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T \boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu_1}^T \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_1} + \frac{1}{2}\boldsymbol{\mu_2}^T \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_2} + \log \frac{p(\mathcal{C}_1)}{\mathcal{C}_2} \tag{7.52}$$

$$= \boldsymbol{w}^T\boldsymbol{x} + w_0 \tag{7.53}$$

where

$$\boldsymbol{w} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2}) \tag{7.54}$$

$$w_0 = -\frac{1}{2}\boldsymbol{\mu_1}^T \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_1} + \frac{1}{2}\boldsymbol{\mu_2}^T \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_2} + \log \frac{p(\mathcal{C}_1)}{\mathcal{C}_2} \tag{7.55}$$

We see that the quadratic terms in $\boldsymbol{x}$ from the exponents of the Gaussian densities have cancelled (due to the common covariance matrices) leading to a linear function of $\boldsymbol{x}$ in the argument of logistic sigmoid.The prior $p(\mathcal{C}_k)$ enter only through the bias parameter w_0 so that it have effect of the parallel contours of constant posterior probability.

For the general case of K cases we have

$$a_k(\boldsymbol{x}) = \log p(\boldsymbol{x}|\mathcal{C}_k)p(\mathcal{C}_k) \tag{7.56}$$

$$= \boldsymbol{\mu_k}^T \boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu_k}^T \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_k} + \boldsymbol{x}^T \boldsymbol{\Sigma}^{-1}\boldsymbol{x} + \log p(\mathcal{C}_k) + const \tag{7.57}$$

Notice that $\boldsymbol{x}\boldsymbol{\Sigma}^{-1}\boldsymbol{x}$ is cancelled in the exponent of the softmax function.Therefore we can write

$$a_k(\boldsymbol{x}) = \boldsymbol{w_k}^T\boldsymbol{x} + w_{k0} \tag{7.58}$$

where

$$\boldsymbol{w_k} = \boldsymbol{\Sigma}^{-T}\boldsymbol{\mu_k} \tag{7.59}$$

$$w_{k0} = -\frac{1}{2}\boldsymbol{\mu_k}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_k} + \ln p(\mathcal{C}_k) \tag{7.60}$$

We see that the $a_k(\boldsymbol{x})$ are again linear functions of $\boldsymbol{x}$ due to the shared covariances,otherwise we obtain quadratic functions of $\boldsymbol{x}$,giving rise to a **quadratic discriminant**.

7.2.4 Maximum likelihood solution

Having specified a parametric functional form for the class-conditional densities $p(\boldsymbol{x}|\mathcal{C}_k)$,we can then determine the values of the parameters,together with the prior class probabilities $p(\mathcal{C}_k)$,using **maximum likelihood**.

Consider first the case of two classes,each having a Gaussian class-conditional density with a shared covariance matrix,and suppose we have a data set $\{\boldsymbol{x_n}, \boldsymbol{t_n}\}$ where $n = 1,...N$.Here $t_n = 1$ denotes class $\mathcal{C}_1$ and 0 denotes $\mathcal{C}_2$.Denote the prior class probability $p(\mathcal{C}_1) = \pi$ so that $p(\mathcal{C}_2) = 1 - \pi$.For a data point $\boldsymbol{x_n}$ from class $\mathcal{C}_1$,we have $t_n = 1$ and hence

$$p(\boldsymbol{x_n}, \mathcal{C}_1) = p(\mathcal{C}_1)p(\boldsymbol{x}|\mathcal{C}_1) = \pi\mathcal{N}(\boldsymbol{x_n}|\boldsymbol{\mu_1}, \boldsymbol{\Sigma}) \tag{7.61}$$

For class $\mathcal{C}_2$,and hence

$$p(\boldsymbol{x_n}, \mathcal{C}_2) = p(\mathcal{C}_1)p(\boldsymbol{x}|\mathcal{C}_2) = \pi\mathcal{N}(\boldsymbol{x_n}|\boldsymbol{\mu_2}, \boldsymbol{\Sigma}) \tag{7.62}$$

Thus the likelihood is given by

$$p(\boldsymbol{t}|\pi, \boldsymbol{\mu_1}, \boldsymbol{\mu_2}) = \prod_{n=1}^{N}[\pi\mathcal{N}(\boldsymbol{x_n}|\boldsymbol{\mu_1}, \boldsymbol{\Sigma})]^{t_n}[(1-\pi)\mathcal{N}(\boldsymbol{x_n}|\boldsymbol{\mu_2}, \boldsymbol{\Sigma})]^{1-t_n} \tag{7.63}$$

where $\boldsymbol{t} = (t_1, ..., t_N)^T$. Maximize the log likelihood

$$\log p(\boldsymbol{t}|\pi, \boldsymbol{\mu_1}, \boldsymbol{\mu_2}) = \sum_{n=1}^{N}\{t_n \log\pi + t_n \log\mathcal{N}(\boldsymbol{x_n}|\boldsymbol{\mu_1}, \boldsymbol{\Sigma}) + (1-t_n)\log(1-\pi) + (1-t_n)\log\mathcal{N}(\boldsymbol{x_n}|\boldsymbol{\mu_2}, \boldsymbol{\Sigma})\} \tag{7.64}$$

The terms that depend on π are

$$\sum_{n=1}^{N}\{t_n \log\pi + (1-t_n)\ln(1-\pi)\} \tag{7.65}$$

Setting the derivative with respect to π equal to zero and rearranging,we obtain

$$\sum_{n=1}^{N}\{t_n\frac{1}{\pi} + (t_n - 1)\frac{1}{1-\pi}\} = 0 \tag{7.66}$$

$$\Rightarrow\pi\sum_{n=1}^{N}(t_n - 1) - \pi\sum_{n=1}^{N}t_n + \sum_{n=1}^{N}t_n = 0 \tag{7.67}$$

$$\Rightarrow\pi = \frac{1}{N}\sum_{n=1}^{N}t_n = \frac{N_1}{N} = \frac{N_1}{N_1 + N_2} \tag{7.68}$$

where N_k denotes the total number of data points in class $\mathcal{C}_k$.Thus the maximum likelihood estimate for pi is simply the fraction of points in class as expected,which can be easily generalized to the multiclass case where ... .

Now consider the maximization with respect to $\boldsymbol{\mu_1}$.The log likelihood function those terms that depend on $\boldsymbol{\mu_1}$

$$\sum_{n=1}^{N} t_n \ln \mathcal{N}(\boldsymbol{x_n}|\boldsymbol{\mu_1}, \Sigma) = -\frac{1}{2}\sum_{n=1}^{N} t_n(\boldsymbol{x_n}-\boldsymbol{\mu_1})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x_n}-\boldsymbol{\mu_1}) + const \tag{7.69}$$

Setting the derivative with respect to $\boldsymbol{\mu_1}$ to zero and arranging ,we obtain

$$\boldsymbol{\mu_1} = \frac{1}{N_1} t_n \boldsymbol{x_n} \tag{7.70}$$

which is simply the mean of all the input vectors $\boldsymbol{x_n}$ assigned to corresponding class. Similarly,

$$\boldsymbol{\mu_2} = \frac{1}{N_2} t_n \boldsymbol{x_n} \tag{7.71}$$

Finally,consider the maximum likelihood solution for the shared covariance matrix Σ.We have

$$-\frac{1}{2}\sum_{n=1}^{N} t_n \ln|\Sigma| - \frac{1}{2}\sum_{n=1}^{N} t_n(\boldsymbol{x_n}-\boldsymbol{\mu_1})^T\Sigma^{-1}(\boldsymbol{x_n}-\boldsymbol{\mu_1}) \tag{7.72}$$

$$-\frac{1}{2}\sum_{n=1}^{N} (1-t_n) \ln|\Sigma| - \frac{1}{2}\sum_{n=1}^{N} (1-t_n)(\boldsymbol{x_n}-\boldsymbol{\mu_2})^T\Sigma^{-1}(\boldsymbol{x_n}-\boldsymbol{\mu_2}) \tag{7.73}$$

$$= -\frac{N}{2}\ln|\Sigma| - \frac{N}{2} Tr\{\Sigma^{-1}\boldsymbol{S}\} \tag{7.74}$$

where we have defined

$$\boldsymbol{S} = \frac{N_1}{N}\boldsymbol{S_1} + \frac{N_2}{N}\boldsymbol{S_2} \tag{7.75}$$

$$\boldsymbol{S_1} = \frac{1}{N_1}\sum_{n\in\mathcal{C}_1} (\boldsymbol{x_n}-\boldsymbol{\mu_1})(\boldsymbol{x_n}-\boldsymbol{\mu_1}) \tag{7.76}$$

$$\boldsymbol{S_2} = \frac{1}{N_2}\sum_{n\in\mathcal{C}_2} (\boldsymbol{x_n}-\boldsymbol{\mu_2})(\boldsymbol{x_n}-\boldsymbol{\mu_2}) \tag{7.77}$$

Using the standard result for the maximum likelihood solution for a Gaussian distribution,we see that

$$\boldsymbol{\Sigma} = \boldsymbol{S} \tag{7.78}$$

which represents a weighted average of the covariance matrices associated with each of the two classes separately.[20]

This result is easily extended to the K class problem.Note that the approach of fitting Gaussian distributions to the classes is not robust to outliers,because the maximum likelihood estimation of a Gaussian is not robust.

7.2.5 Discrete features

Assume binary feature value $x_i \in \{0,1\}$ is Bernoulli distributed,thus we have class-conditional distributions of the form

$$p(\boldsymbol{x}|\mathcal{C}_k) = \prod_{i=1}^{D} \mu_{ki}^{x_i}(1-\mu_{ki})^{1-x_i} \tag{7.79}$$

which contains D independent parameters for each class,giving

$$a_k(\boldsymbol{x}) = \sum_{i=1}^{D}\{x_i \ln\mu_{ki} + (1-x_i)\ln(1-\mu_k i)\} + \ln p(\mathcal{C}_k) \tag{7.80}$$

[20] ?

which again are linear functions of the input values x_i.Analogous results are obtained for discrete variables each of which can take $M > 2$ states.

7.2.6 Exponential family

As we have see so far,for both Gaussian distributed and discrete inputs,the posterior class probabilities are given by generalized linear models with logistic sigmoid($K = 2$ classes) or softmax($K \gg 2$ classes) activation functions.These are particular cases of a more general result obtained by assuming that the class-conditional densities p(x—Ck) are members of the exponential family of distributions. Using the form 2.6 for members of the exponential family, we see that the distribution of x can be written in the form

$$p(\boldsymbol{x}|\boldsymbol{\lambda_k}) = h(\boldsymbol{x})g(\boldsymbol{\lambda}_k)\exp\{\boldsymbol{\lambda}_k^T\boldsymbol{u}(\boldsymbol{x})\} \tag{7.81}$$

We restrict attention to the subclass for which $\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{x}$.Making use of 2.119 to introduce a scaling parameter s,then

$$p(\boldsymbol{x}|\boldsymbol{\lambda_k}, s) = \frac{1}{s}h(\boldsymbol{x}/s)g(\boldsymbol{\lambda}_k)\exp\{\frac{1}{s}\boldsymbol{\lambda}_k^T\boldsymbol{u}(\boldsymbol{x})\} \tag{7.82}$$

Note that each class have its own parameter vector $\boldsymbol{\lambda}_k$ but share the same scale parameter s.

For the two-class problem,the posterior is given by a logistic sigmoid acting on a linear function

$$a(\boldsymbol{x}) = (\boldsymbol{\lambda}_1 - \boldsymbol{\lambda}_2)^T\boldsymbol{x} + \ln g(\boldsymbol{\lambda}_1) - \ln g(\boldsymbol{\lambda}_2) + \ln p(\mathcal{C}_1) - \ln p(\mathcal{C}_2) \tag{7.83}$$

Similarly,for the K-class problem,

$$a_k(\boldsymbol{x}) = \boldsymbol{\lambda}_k^T\boldsymbol{x} + \ln g(\boldsymbol{\lambda}_k) + \ln p(\mathcal{C}_k) \tag{7.84}$$

and so again is a linear function of $\boldsymbol{x}$.

7.3 Probabilistic Discriminative Models

We have seen that the posterior probability for the two-class classification and multiclass case can be written as a logistic and softmax function respectively for a wide choice of class-conditional distributions $p(\boldsymbol{x}|\mathcal{C}_k)$.For a specific class-conditional densities $p(\boldsymbol{x}|\mathcal{C}_k)$,we use maximum likelihood to determine the parameters of the densities as well as the class priors $p(\mathcal{C}_k)$ and use Bayes' theorem to find the posterior class probabilities.However,an alternative approach is to use the functional form of the generalized linear model explicitly and to determine its parameters directly by using maximum likelihood,such as **iterative reweighted least squares** or IRLS.

The indirect approach to finding the parameters of a generalized linear model by fitting class-conditional densities and class priors separately and then applying Bayes' theorem,represents an example of **gernerative modelling**because we could take such a model and generate synthetic data by drawing values of $\boldsymbol{x}$ from the marginal distribution $p(\boldsymbol{x})$.In the direct approach,we maximize a likelihood function defined through the conditional distribution $p(\mathcal{C}_k|\boldsymbol{x})$,which represents a form of **discriminative** training.

7.3.1 Fixed basis functions

So far,we have considered classification models that work directly with the original input vector $\boldsymbol{x}$.However,all of the algorithms can equally applicable if we first make a fixed nonlinear transformation of the inputs using a vector of basis functions $\boldsymbol{\phi}(\boldsymbol{x})$.The resulting decision boundaries will be linear in the feature space $\boldsymbol{\phi}$,corresponding to nonlinear decision boundaries in the original $\boldsymbol{x}$ space.One of the basis function is typically set to a constant,say $\boldsymbol{\phi}_0(\boldsymbol{x}) = 1$,so that the corresponding parameter w_0 plays the role of a bias.

Note that nonlinear transformations cannot remove class overlap between the class-conditional densities $p(\boldsymbol{x}|\mathcal{C}_k)$.

7.3.2 Logistic regression

7.3.2.1 representation

We begin with two-class classification.The posterior probability of class $\mathcal{C}_1$ can be written as a logistic sigmoid acting on a linear function of the feature vector ϕ so that

$$p(\mathcal{C}_1|\phi) = y(\phi) = \sigma(\boldsymbol{w}^T\phi) \tag{7.85}$$

which has a linear dependence on the number of parameters.Logistic regression model can also be written as

$$p(y|\boldsymbol{x},\boldsymbol{w}) = \text{Ber}(y|\text{sigm}(\boldsymbol{w}^T\boldsymbol{\phi})) \tag{7.86}$$

where $\boldsymbol{w}$ and $\boldsymbol{\phi}$ are extended vectors, i.e., $\boldsymbol{w} = (b, w_1, w_2, \cdots, w_D)$, $\boldsymbol{\phi} = (1, \phi_1, \phi_2, \cdots, \phi_D)$.

Note that the derivative of the logistic sigmoid function is

$$\frac{d\sigma}{da} = \sigma(1-\sigma) \tag{7.87}$$

7.3.2.2 evaluation

For a data set $\{\phi_n, t_n\}$,where $t_n \in \{0,1\}$ and $\phi_n = \phi(\boldsymbol{x_n})$,with $n = 1,...,N$,the likelihood function can be written

$$p(\boldsymbol{t}|\boldsymbol{w}) = \prod_{n=1}^{N} y_n^{t_n}\{1-y_n\}^{1-t_n} \tag{7.88}$$

where $\boldsymbol{t} = (t_1,...,t_N)^T$ and $y_n = p(\mathcal{C}_1|\phi_n)$.The negative logarithm of the likelihood,which gives the **cross-entropy** error function is in the form

$$E(\boldsymbol{w}) = -\ln p(\boldsymbol{t}|\boldsymbol{w}) = -\sum_{n=1}^{N}\{t_n \ln y_n + (1-t_n)\ln(1-y_n)\} \tag{7.89}$$

Taking the gradient of the error function with respect to $\boldsymbol{w}$,we obtain

$$\nabla E(\boldsymbol{w}) = -\sum_{n=1}^{N}\{t_n\frac{y_n(1-y_n)}{y_n} - (1-t_n)\frac{y_n(q-y_n)}{1-y_n}\}\phi_n \tag{7.90}$$

$$= -\sum_{n=1}^{N}\{t_n(1-y_n) - (1-t_n)y_n\}\phi_n \tag{7.91}$$

$$= \sum_{n=1}^{N}(y_n - t_n)\phi_n \tag{7.92}$$

We see that the factor involving the derivative of the logistic sigmoid has cancelled.In particular,the contribution to the gradient from data point n is given by the 'error' $y_n - t_n$ between the target value and the prediction of the model,times the basis function vector ϕ_n,taking the same form as the gradient of the sum-of-squares error function for the linear regression model.

Note that maximum likelihood can exhibit severe **over-fitting** for data sets that are linearly separable.The singularity can be avoided by inclusion of a prior and finding a **MAP(maximum a posterior)** solution for $\boldsymbol{w}$,or equivalently by adding a **regularization** term to the error function.

7.3.2.3 optimization

See the following contents.

7.3.3 Iterative reweighted least squares

7.3.3.1 Newton-Raphson update

For logistic regression,there is no longer a closed-form solution,due to the nonlinearity of the logistic sigmoid function.The error function is concave,and can be minimized by an efficient iterative technique based on the **Newton-Raphson** iterative optimization scheme,which uses a local quadratic approximation to the log likelihood function.

$$\boldsymbol{w}^{(new)} = \boldsymbol{w}^{(old)} - \boldsymbol{H}^{-1}\nabla E(\boldsymbol{w}) \tag{7.93}$$

where $\boldsymbol{H}$ is the Hessian matrix whose elements comprise the second derivatives of $E(\boldsymbol{w})$ with respect to the components of $\boldsymbol{w}$.

Now apply the Newton-Raphson method ot the linear regression model with sum-of-squares error function.The gradient and Hessian of this error function are given by

$$\nabla E(\boldsymbol{w}) = \sum_{n=1}^{N}(\boldsymbol{w}^T\phi_n - t_n)\phi_n = \boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{\Phi}^T\boldsymbol{t} \tag{7.94}$$

$$\boldsymbol{H} = \nabla\nabla E(\boldsymbol{w}) = \sum_{n=1}^{N}\phi_n\phi_n^T = \phi^T\boldsymbol{\Phi} \tag{7.95}$$

where $\boldsymbol{\Phi}$ is the $N\times M$ design matrix,whose n^{th} row is given by ϕ_n^T.The Newton-Raphson update then takes the form

$$\boldsymbol{w}^{(new)} = \boldsymbol{w}^{(old)} - (\boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}\{\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{w}^{(old)} - \boldsymbol{\Phi}^T\boldsymbol{t}\} \tag{7.96}$$

$$= (\boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\boldsymbol{t} \tag{7.97}$$

which we recognize as the standard least-squares solution.

Now apply the Newton-Raphson update to the cross-entropy error function for logistic regression model.Gradient and Hessian of this error function are given by

$$\nabla E(\boldsymbol{w}) = \sum_{n=1}^{N}(y_n - t_n)\phi_n = \boldsymbol{\Phi}^T(\boldsymbol{y} - \boldsymbol{t}) \tag{7.98}$$

$$\boldsymbol{H} = \nabla\nabla E(\boldsymbol{w}) = \sum_{n=1}^{N}y_n(1-y_n)\phi_n\phi_n^T = \boldsymbol{\Phi}^T\boldsymbol{R}\boldsymbol{\Phi} \tag{7.99}$$

where we have introduced the $N\times N$ diagonal matrix $\boldsymbol{R}$ with elements

$$R_{nn} = y_n(1-y_n) \tag{7.100}$$

We see that the Hessian depends on $\boldsymbol{w}$ and is positive definite.

The Newton-Raphson update formula for the logistic regression model then becomes

$$\boldsymbol{w}^{(new)} = \boldsymbol{w}^{(old)} - (\boldsymbol{\Phi}^T\boldsymbol{R}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T(\boldsymbol{y} - \boldsymbol{t}) \tag{7.101}$$

$$= (\boldsymbol{\Phi}^T\boldsymbol{R}\boldsymbol{\Phi})^{-1}\{\boldsymbol{\Phi}^T\boldsymbol{R}\boldsymbol{\Phi}\boldsymbol{w}^{(old)} - \boldsymbol{\Phi}^T(\boldsymbol{y} - \boldsymbol{t})\} \tag{7.102}$$

$$= (\boldsymbol{\Phi}^T\boldsymbol{R}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\boldsymbol{R}\boldsymbol{z} \tag{7.103}$$

where $\boldsymbol{z}$ is an N-dimensional vector with elements

$$\boldsymbol{z} = \boldsymbol{\Phi}\boldsymbol{w}^{(old)} - \boldsymbol{R}^{-1}(\boldsymbol{y} - \boldsymbol{t}) \tag{7.104}$$

We see that the update formula takes the form of a set of normal equations for a weighted least-squares problem.Because the weighting matrix $\boldsymbol{R}$ depends on the parameter vector $\boldsymbol{w}$,we must apply the normal equations iteratively,each time using the new weight vector $\boldsymbol{w}$ to compute a revised weighing matrix $\boldsymbol{R}$.For this reason,the algorithm is known as **iterative reweighted least squares**,or IRLS.The elements of the diagonal weighing matrix $\boldsymbol{R}$ can be

interpreted as variances because the mean and variance of t in the logistic regression model are given by

$$\mathbb{E}[t] = \sigma(\boldsymbol{x}) = y \tag{7.105}$$

$$var[t] = \mathbb{E}[t^2] - \mathbb{E}[t]^2 = \sigma(\boldsymbol{x}) - \sigma(\boldsymbol{x})^2 = y(1-y) \tag{7.106}$$

where we have used the property $t^2 = t$ for $t \in \{0,1\}$.

7.3.3.2 Linearization

In fact,we can interpret IRLS as the solution to a linearized problem in the space of the variable $a = \boldsymbol{w}^T\phi$.Apply Taylor expansion and then we obtain

$$a_n(\boldsymbol{w}) \simeq a_n(\boldsymbol{w}^{(old)}) + \left.\frac{da_n}{dy_n}\right|_{w^{(old)}} (t_n - y_n) \tag{7.107}$$

$$= \phi_n^T \boldsymbol{w}^{(old)} - \frac{(y_n - t_n)}{y_n(1-y_n)} = z_n \tag{7.108}$$

7.3.4 Multiclass logistic regression

In our discussion of generative models for multiclass classification,the posterior probabilities are given by a **softmax** transformation of linear functions of the feature variables,so that

$$p(\mathcal{C}_k|\phi) = y_k(\phi) = \frac{\exp(a_k)}{\sum_j \exp(a_j)} \tag{7.109}$$

$$= \frac{\exp(a_k)}{sum} \tag{7.110}$$

$$\tag{7.111}$$

Softmax function is

$$\delta(\boldsymbol{z}_j) = \frac{\exp^{\boldsymbol{z}_j}}{\sum_{k=1}^K \exp^{\boldsymbol{z}_k}}, \text{for } j = 1, ..., K. \tag{7.112}$$

$$= \frac{\exp^{\boldsymbol{z}_j - \max \boldsymbol{z}}}{\sum_{k=1}^K \exp^{\boldsymbol{z}_k - \max \boldsymbol{z}}} \tag{7.113}$$

$$\log \delta(\boldsymbol{z}_j) = \boldsymbol{z}_j - \max \boldsymbol{z} - \log \sum_{k=1}^K \exp^{\boldsymbol{z}_k - \max \boldsymbol{z}} \tag{7.114}$$

where *sum* denotes $\sum_j \exp(a_j)$ the 'activations' a_k are given by

$$a_k = \boldsymbol{w}_k^T \phi \tag{7.115}$$

$\boldsymbol{w}_k$ is a $M \times 1$ column vector parameter, and ϕ is the $M \times 1$ data column vector.

Here we consider the use of maximum likelihood to determine the parameters $\{\boldsymbol{w}_k\}$ of this model directly.The derivatives of y_k with respect to all of the activations a_j are given by

$$\frac{\partial y_k}{\partial a_j} = \frac{\partial p_k}{\partial a_j} = \frac{e^{a_k} \boldsymbol{I}_{kj} sum - e^{a_k} e^{a_j}}{sum^2} \tag{7.116}$$

$$= y_k(I_{kj} - y_j) \tag{7.117}$$

where $\boldsymbol{I}_{kj}$ are the elements of the identity matrix.

Next we write down the likelihood function,using the $1-of-K$ coding scheme.

$$y = p(\boldsymbol{T}|\boldsymbol{w}_1,...,\boldsymbol{w}_K) = \prod_{n=1}^{N}\prod_{k=1}^{K} p(\mathcal{C}_k|\phi_n)^{t_{nk}} = \prod_{n=1}^{N}\prod_{k=1}^{K} y_{nk}^{t_{nk}} \tag{7.118}$$

where $y_{nk} = y_k(\phi_n)$,and $\boldsymbol{T}$ is an $N \times N$ matrix of target variables with elements t_{nk}.Taking the negative logarithm then gives

$$E(\boldsymbol{w_1},...,\boldsymbol{w_K}) = -\ln p(\boldsymbol{T}|\boldsymbol{w}_1,...,\boldsymbol{w}_K) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\ln y_{nk} \tag{7.119}$$

which is known as the $cross-entropy$ error function for the multiclass classification problem.

Take the gradient of the error function,making use of the derivatives of the softmax function,we obtain

$$\because \frac{\partial E_n}{\partial a_j} = -\sum_k t_k \frac{\partial \log p_k}{\partial a_j} \tag{7.120}$$

$$= -\sum_k t_k \frac{1}{p_k}\frac{\partial p_k}{\partial a_j} \tag{7.121}$$

$$= -t_j(1-p_j) - \sum_{k\neq j} t_k \frac{1}{p_k}(-p_k p_j) \tag{7.122}$$

$$= -t_j(1-p_j) + \sum_{k\neq j} t_k (p_j) \tag{7.123}$$

$$= -t_j + t_j p_j + \sum_{k\neq j} t_k (p_j) \tag{7.124}$$

$$= p_j\left(\sum_k t_k\right) - t_j \tag{7.125}$$

$$= p_j - t_j \tag{7.126}$$

$$= y_j - t_j \tag{7.127}$$

$$\tag{7.128}$$

$$\therefore \nabla_{w_j} E(\boldsymbol{w_1},...,\boldsymbol{w_K}) = \sum_{n=1}^{N}(y_{nj}-t_{nj})\phi_n \tag{7.129}$$

$$\sum_{n=1}^{N}\phi_n(y_{nj}-t_{nj}) \therefore \nabla_{\boldsymbol{W}} E(\boldsymbol{w}_1,...\boldsymbol{w}_k) = \boldsymbol{\Phi}^T(\boldsymbol{Y}-\boldsymbol{T}) \tag{7.130}$$

where $\phi_n = \phi(\boldsymbol{x}_n)$ is basis function of $\boldsymbol{x}_n$ of dimension $M \times 1$, $\boldsymbol{Y}$ is the $N \times K$ prediction matrix, $\boldsymbol{T}$ is the $N \times K$ target label matrix, $\boldsymbol{\Phi}$ is the $N \times M$ design matrix, $\boldsymbol{W}$ is $M \times K$ weight matrix of parameters.

$$\tag{7.131}$$

To find a batch algorithm,we evaluate the Hessian matrix

$$\nabla_{w_k}\nabla_{w_j} E(\boldsymbol{w_1},...,\boldsymbol{w_K}) = -\sum_{n=1}^{N} y_{nk}(I_{kj}-y_{nj})\phi_n\phi_n^T \tag{7.132}$$

The Hessian matrix is also positive definite and so the error function has a unique minimum.

7.3.5 Probit regression

A broad range of class-conditional distributions,described by the exponential family,the resulting posterior class probabilities are given by a logistic(or softmax) transformation acting on a linear function of the feature variables,but not all choices of class-conditional density give rise to such a simple form for the posterior probabilities.

The cumulative distribution function is given by

$$\Phi(a) = \int_{-\infty}^{a} \mathcal{N}(\theta|0,1)d\theta \tag{7.133}$$

which is known as the **probit function**.It has a sigmoidal shape.A evaluation of a closely related function is

$$erf(a) = \frac{2}{\sqrt{\pi}} \int_{0}^{a} \exp(-\theta^2/2)d\theta \tag{7.134}$$

known as the **erf function** or **error function**.The generalized linear model based on a probit function is known as **probit regression**.

One issue that occur in practical applications is that of **outliers**.Because probit activation function they decay like $\exp(-x^2)$,so probit model can be significantly more sensitive to outliers.However,the effect of mislabelling is easily incoporated into a probabilistic model by introducing a probability ε that the target value t has been flipped to the wrong value,

$$p(t|\boldsymbol{x}) = (1-\varepsilon)\sigma(\boldsymbol{x}) + \varepsilon(1-\sigma(\boldsymbol{x})) \tag{7.135}$$

$$= \varepsilon + (1-2\varepsilon)\sigma(\boldsymbol{x}) \tag{7.136}$$

where $\sigma(\boldsymbol{x})$ is the activation function with input vector $\boldsymbol{x}$.Here ε may be set in advance,or it may be treated as a **hyperparameter** whose value is inferred from data.

7.3.6 Canonical link functions

We now show that there is a general result of assuming a conditional distribution for the target variable from the exponential family,along with a corresponding choice for the activation function known as the **canonical link function**.

Making use of the restricted form of exponential family distribution 7.82 for target variable t

$$p(t|\eta,s) = \frac{1}{s}h(\frac{t}{s})g(\eta)\exp\{\frac{\eta t}{s}\} \tag{7.137}$$

The conditional mean of t,which denoted by y,is given by

$$y \equiv \mathbb{E}[t|\eta] = -s\frac{d}{d\eta}\ln g(\eta) \tag{7.138}$$

Thus y and η must be related,and we denote this relation through $\eta = \psi(y)$.

We define a **generalized linear model** to be one for which y is a nonlinear function of a linear combination of the input(or feature) variables so that

$$y = f(\boldsymbol{w}^T\boldsymbol{\phi}) \tag{7.139}$$

where $f(\cdot)$ is known as the **activation function**,and $f^{-1}(\cdot)$ is known as the **link function** in statistics.

The log likelihood function for this model,which,as a function of η ,is given by

$$\ln p(\boldsymbol{t}|\eta,s) = \sum_{n=1}^{N}\ln p(t_n|\eta,s) = \sum_{n=1}^{N}\{\ln g(\eta_n) + \frac{\eta_n t_n}{s}\} + const \tag{7.140}$$

where we assume that all observations share a common scale parameter(which corresponds to the noise variance for a Gaussian distribution for instance) and so s is independent of η.The derivative with respect to the model parameter $\boldsymbol{w}$

is then given by

$$\nabla_w \ln p(\boldsymbol{t}|\eta, s) = \sum_{n=1}^{N} \{\frac{d}{d\eta_n} \ln g(\eta_n) + \frac{t_n}{s}\} \frac{d\eta_n}{dy_n} \frac{dy_n}{da_n} \nabla a_n \tag{7.141}$$

$$= \sum_{n=1}^{N} \frac{1}{s} \{t_n - y_n\} \psi'(y_n) f'(a_n) \phi_n \tag{7.142}$$

where $a_n = \boldsymbol{w}^T \boldsymbol{\phi_n}$,and we have used $y_n = f(a_n)$ together.There is a considerable simplification if we choose a particular form of the link function $f^{-1}(y)$ given by

$$f^{-1}(y) = \psi(y) \tag{7.143}$$

In this case,the gradient of error function reduces to

$$\nabla \ln E(\boldsymbol{w}) = \frac{1}{s} \sum_{n=1}^{N} \{y_n - t_n\} \phi_n \tag{7.144}$$

For the Gaussian $s = \beta^{-1}$,whereas for logistic model $s = 1$.

7.3.7 The Laplace Approximation

7.3.8 Model comparison and BIC

7.4 Bayesian Logistic Regression

7.4.1 Laplace approximation

7.4.2 Predictive distribution

Chapter 8
Neural Networks

Applicability of models that comprised linear combinations of **fixed basis functions** is limited by the curse of dimensionality.In order the apply such models to large-scale problems,it is necessary to adapt the basis functions to the data.

Support vector machines and relevance vector machine **choose a subset from a fixed set of basis functions** and results in much sparser models.

An alternative approach is to fix the number of basis functions in advance but allow them to be **adaptive**,in other words to use parametric forms for the basis function

8.1 Feed-forward Network Functions

The linear models are based on linear combinations of fixed nonlinear basis functions $\phi_j(\boldsymbol{x})$ and take the form

$$y(\boldsymbol{x},\boldsymbol{w}) = f(\sum_{j=1}^{M} w_j \phi_j(\boldsymbol{x})) = f(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x})) \tag{8.1}$$

where $f(\cdot)$ is a nonlinear activation function in the case of classification and is the identity in the case of regression.

Neural network model can be described a series of functional transformations.First construct M linear combinations of the input variables $x_1,\ldots,x_D$ in the form

$$a_j^{(l+1)} = \sum_{i=1}^{D} w_{ji}^{(l)} x_i + w_{j0}^{(l)} \tag{8.2}$$

$$= \boldsymbol{w}^{(l)} \boldsymbol{x}^{(l)} \tag{8.3}$$

where $j = 1,\ldots,M$,and the superscript (l) indicates the $l-the$ layer of the network.We refer the parameters $w_{ji}^{(l)}$ as **weights** and $w_{j0}^{(l)}$ as **biases**.The quantity a_j are known as **activations**.Each of them is then transformed using **activation function** $h(\cdot)$ to give

$$z_j = h(a_j) \tag{8.4}$$

These quantities correspond to the outputs of the basis functions in linear model that,in the context of neural networks,are called **hidden units**.The nonlinear functions $h(\cdot)$ are generally chosen to be sigmoidal or the '$tanh$' function.

The transformation of the following layer of the network,combine these values to give **output unit activations**

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)} \tag{8.5}$$

where $k = 1,\ldots,K$,and K is the total number of outputs.

For regression,$y_k = a_k$ and binary classification,we can use $y_k = \sigma(a_k) = \dfrac{1}{1+\exp(-a)}$.Finally,for multiclass problems,a **softmax** activation function is used.**Unit activation function** is discussed later.

The process of evaluating the output of the network diagram is interpreted as a **forward propagation** of information through the network.Deterministic rather than stochastic variables of internal nodes do not represent probabilistic graphical models.Each (hidden or output) unit in such network computes a function given by

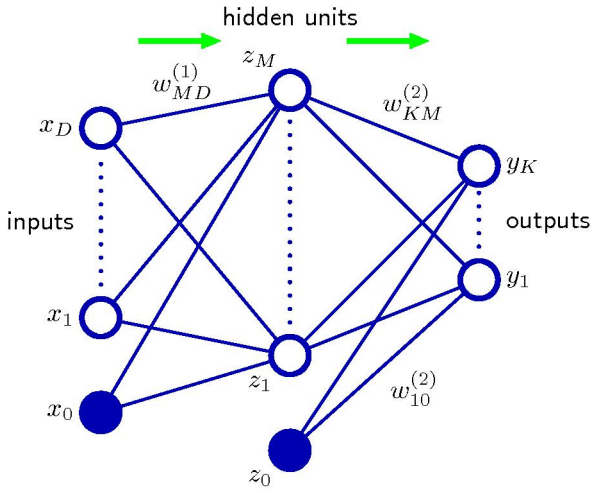


Fig. 8.1: Network diagram for two-layer neural network

$$z_k = h(\sum_j w_{kj} z_j) \tag{8.6}$$

where the sum runs over all units that send connections to unit k (including the bias parameter).

8.1.1 Weight-space symmetries

Consider a two-layer network as shown before.For M hidden units,there will be M 'sign-flip' symmetries,i.e.,2^M equivalent weight vectors.Similarly,interchanging the values of all the weights(and the bias) can give rise to the same mapping function from inputs to outputs represented by the network.There will be $M!$ equivalent weight vectors.The network have an overall weight-space symmetry factor of $M!2^M$.

8.2 Network Training

Given a training set comprising a set of input vectors $\boldsymbol{x}_n$,where $n = 1,\ldots,N$,together with a corresponding set of target vectors $\boldsymbol{t}_n$,we minimize the error function.We can provide a probabilistic interpretation to the network outputs.

For regression,and for the moment we consider a single target variable t that can take any real value.We assume that t has a Gaussian distribution with an $\boldsymbol{x}$-dependent mean,which is given by the output of the neural network,so that

$$p(t|\boldsymbol{w},\boldsymbol{w}) = \mathcal{N}(t|y(\boldsymbol{x},\boldsymbol{w}),\beta^{-1}) \tag{8.7}$$

where β is the precision (inverse variance) of the Gaussian noise.Maximize the likelihood function as in linear models.View the network as having an output activation function that id the identity,so that $y_k = a_k$.The corresponding sum-of-squares error function has the property

$$\frac{\partial E}{\partial a_k} = \frac{\partial E}{\partial y_k} = y_k - t_k \tag{8.8}$$

For binary classification in which $t \in 0,1$,consider a network having a single output whose activation function is a logistic sigmoid

$$y = \sigma(a) = \frac{1}{1+\exp(-a)} \tag{8.9}$$

We can interpret $y(\boldsymbol{x}, \boldsymbol{w})$ as the conditional probability $p(\mathcal{C}_1|\boldsymbol{x})$.the conditional distribution of targets given inputs is then a Bernoulli distribution of the form

$$p(t|\boldsymbol{x}, \boldsymbol{w}) = y(\boldsymbol{x}, \boldsymbol{w})^t \{1 - y(\boldsymbol{x}, \boldsymbol{w})^{1-t}\} \tag{8.10}$$

For K separate binary classifications,the conditional distribution of the targets is

$$p(\boldsymbol{t}|\boldsymbol{x}, \boldsymbol{w}) = \prod_{k=1}^{K} y_k(\boldsymbol{x}, \boldsymbol{w})_k^t [1 - y(\boldsymbol{x}, \boldsymbol{w})]^{1-t_k} \tag{8.11}$$

Taking the negative logarithm of the corresponding likelihood function,we get **cross-entropy** error function

$$E(\boldsymbol{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \ln y_{nk} + (1 - t_{nk}) \ln(1 - y_{nk}) \tag{8.12}$$

The derivative of the error function with respect to the **activation** for a particular output unit is

$$\frac{\partial E_n}{\partial a_k} = \frac{\partial - \sum_{k=1}^{K} t_{nk} \ln y_{nk} + (1 - t_{nk}) \ln(1 - y_{nk})}{\partial a_k} \tag{8.13}$$

$$= -t_k \frac{y_k(1 - y_k)}{y_k} + (1 - t_k \frac{-y_k(1 - y_k)}{1 - y_k}) \tag{8.14}$$

$$= y_k - t_k \tag{8.15}$$

Following the logistic regression,the **output unit activation** is given by the **softmax** function

$$y_k(\boldsymbol{x}, \boldsymbol{w}) = \frac{\exp(a_k(\boldsymbol{x}, \boldsymbol{w}))}{\sum_j \exp(a_j(\boldsymbol{x}, \boldsymbol{w}))} \tag{8.16}$$

8.2.1 Parameter optimization

Turn next to the task of finding a weight vector $\boldsymbol{w}$ which minimizes the chosen function $E(\boldsymbol{w})$. Resort to **iterative**

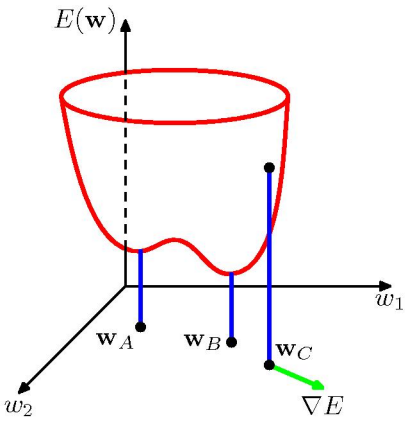


Fig. 8.2: geometrical view of the error function $E(\boldsymbol{w})$ as a surface sitting over weight space.A is a local minimum and B is the global minimum.The local gradient of the error surface is given by the vector ∇E.

numerical procedures.Most techniques involve choosing some initial value $\boldsymbol{w}^{(0)}$ for weight vector and then moving through weight space in a succession of steps of the form

$$\boldsymbol{w}^{(\eta+1)} = \boldsymbol{w}^{(\eta)} + \Delta \boldsymbol{w}^{(\eta)} \tag{8.17}$$

Many algorithms make use of gradient information,evaluating $\nabla E(\boldsymbol{w})$ at the new weight vector $\boldsymbol{w}^{(\eta+1)}$.

8.2.2 Local quadratic approximation

Consider the **Taylor expansion** of $E(\boldsymbol{w})$ around some point in weight space.

8.2.3 Use of gradient information

It is possible to evaluate the gradient of an error function efficiently by means of the backpropagation procedure.Because each evaluation of ∇E brings W items of information,we might hope to find the minimum of the function in $O(W)$ gradient evaluations.Each such evaluation takes only $O(W)$ steps and so the minimum can be now found in $O(W^2)$ steps.

8.2.4 Gradient descent optimization

Batch gradient descent

$$\boldsymbol{w}^{(\tau+1)} = \boldsymbol{w}^{(\tau)} - \eta \nabla E(\boldsymbol{w})^{(\tau)} \tag{8.18}$$

where the parameter $\eta > 0$ is known as the **learning rate**.

More efficient methods than this poor algorithm are **conjugate gradients** and **quasi-Newton** methods,which are more robust and much faster.**On-line** version of gradient has also proved useful in practice.On-line gradient descent,known as **sequential gradient descent** or **stochastic gradient descent**,makes an update to the weight vector based on one data point at a time,so that

$$\boldsymbol{w}^{(\tau+1)} = \boldsymbol{w}^{(\tau)} - \eta \nabla E(\boldsymbol{w}^{(\tau)}) \tag{8.19}$$

8.3 Error Backpropagation

The goal in this section is to find an efficient technique for evaluating the gradient of an error function $E(\boldsymbol{w})$ for a feed-forward neural network.This can be achieved using a local message passing scheme in which information is sent alternately forwards and backwards through the network,known as **error backpropagation** or **backprop**.

8.3.1 Evaluation of error-function derivatives

Many error functions of practical interest,for instance those defined by maximum likelihood for a set of i.i.d data,comprise a sum of terms,on for each data point,so that

$$E(\boldsymbol{w}) = \sum_{n=1}^{N} E_n(\boldsymbol{w}) \tag{8.20}$$

In a general feed-forward network,each unit computes a weighted sum of its inputs of the form

$$a_j = \sum_i w_{ji} z_i \tag{8.21}$$

where z_i is the activation of a unit that sends a connection to unit j and w_{ji} is the weight associated with that connection.The sum is transformed by an activation function $h(\cdot)$ to give the activation z_j of unit j in the form

$$z_j = h(a_j) \tag{8.22}$$

Evaluate the derivate of E_n with respect to a weight w_{ji},applying the **chain rule** for partial derivatives

$$\frac{\partial E_n}{\partial w_{ji}} = \frac{\partial E_n}{\partial \boldsymbol{a}_j} \frac{a_j}{\partial w_{ji}} \tag{8.23}$$

Now introduce a notation

$$\delta_j \equiv \frac{\partial E_n}{\partial a_j} \tag{8.24}$$

referred as **errors**. Using the sum form

$$\frac{\partial a_j}{\partial w_{ji}} = z_i \tag{8.25}$$

Substituting them back

$$\frac{\partial E_n}{\partial w_{ji}} = \delta_j z_i. \tag{8.26}$$

For the output units,we have

$$\delta_k = y_k - t_k \tag{8.27}$$

For hidden units,make use of the chain rule for partial derivatives

$$\delta_j \equiv \frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k} \frac{\partial a_k}{\partial a_j} \tag{8.28}$$

$$\tag{8.29}$$

where the sum runs over all units k to which unit j sends connections.

$$\because \delta_k = \frac{\partial E_n}{\partial a_k} \tag{8.30}$$

$$\therefore \delta_j \equiv \frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k} \frac{\partial a_k}{\partial a_j} \tag{8.31}$$

$$= \sum_k \delta_k \frac{\partial a_k}{\partial a_j} \tag{8.32}$$

$$= \sum_k \delta_k h'(a_j) w_{kj} \tag{8.33}$$

$$= h'(a_j) \sum_k w_{kj} \delta_k \tag{8.34}$$

8.3.1.1 back-propagation algorithm wrap-up

We'll use $\boldsymbol{w}_{ij}^l$ to denote the **weight** for the connection from the kth neuron in the $(l-1)$th layer to the jth neuron in the (l)th layer.And b_j^l for the bias of the jth neuron in the lth layer, a_j^l for the activation(weighted input) of the jth neuron in the lth layer,σ denotes the element-wise activation function,z_j^l denotes the output of jth neuron in the lth layer. Then we have

$$\boldsymbol{a}^l \equiv \boldsymbol{W}^l \boldsymbol{z}^{l-1} + \boldsymbol{b}^l \tag{8.35}$$

$$\boldsymbol{z}^l \equiv \sigma(\boldsymbol{a}^l) \tag{8.36}$$

$s \odot t$ denotes **element-wise product** of two matrices (vectors):

$$(s \odot t)_j = s_j t_j \tag{8.37}$$

Define the error σ_j^l of neuron J in the layer l by

$$\boldsymbol{\delta}_j^l \equiv \frac{\partial C}{\partial \boldsymbol{a}_j^l} \tag{8.38}$$

Using **chain rule of derivatives**,the error in the output layer is:

$$\delta_j^L = \frac{\partial C}{\partial a_j^L} \tag{8.39}$$

$$= \sum_k \frac{\partial C}{\partial z_k^L} \frac{\partial z_k^L}{\partial a_j^L} \tag{8.40}$$

$$= \frac{\partial C}{\partial z_j^L} \frac{\partial z_j^L}{\partial a_j^L}, \text{if output only depends on same neuron's input} \tag{8.41}$$

$$= \frac{\partial C}{\partial \boldsymbol{z}_j^L} \boldsymbol{\sigma}'(\boldsymbol{a}_j^L). \tag{8.42}$$

And the matrix-based form is:

$$\delta^L = \nabla_z C \odot \boldsymbol{\sigma}'(a^L). \tag{8.43}$$

$$\delta^L = \Sigma'(a^L) \nabla_z C, \tag{8.44}$$

where $\Sigma'(z^L)$ is a square matrix whose diagonal entries are the values $\boldsymbol{\sigma}'(z^L)$, and whose off-diagonal entries are zero.

Propagation (recursive relationship) in hidden layer:

$$\delta_j^l = \frac{\partial C}{\partial a_j^l} \tag{8.45}$$

$$= \sum_k \frac{\partial C}{\partial a_k^{l+1}} \frac{\partial a_k^{l+1}}{\partial a_j^l} \tag{8.46}$$

$$= \sum_k \frac{\partial a_k^{l+1}}{\partial a_j^l} \delta_k^{l+1}, \tag{8.47}$$

$$\because a_k^{l+1} = \sum_j w_{kj}^{l+1} z_j^l + b_k^{l+1} = \sum_j w_{kj}^{l+1} \sigma(a_j^l) + b_k^{l+1}. \tag{8.48}$$

$$\therefore \frac{\partial a_k^{l+1}}{\partial a_j^l} = w_{kj}^{l+1} \sigma'(a_j^l) \tag{8.49}$$

$$\therefore \delta_j^l = \sum_k w_{kj}^{l+1} \delta_k^{l+1} \sigma'(a_j^l) \tag{8.50}$$

$$\delta^l = ((\boldsymbol{W}^{l+1})^T \delta^{l+1}) \odot \sigma'(\boldsymbol{a}^l), \tag{8.51}$$

$$\delta^l = \Sigma'(a^l)(w^{l+1})^T \delta^{l+1}. \tag{8.52}$$

$$\tag{8.53}$$

Rate of change of cost with respect to bias in the network:

$$\frac{\partial C}{\partial b_j^l} = \frac{\partial C}{\partial \boldsymbol{a}_j^l} \frac{\partial \boldsymbol{a}_j^l}{\partial b_j^l} \tag{8.54}$$

$$= \frac{\partial C}{\partial \boldsymbol{a}_j^l} \cdot 1 \ = \boldsymbol{\delta}_j^l \tag{8.55}$$

Rate of change of the cost with respect to any weight matrix in the network:

$$\frac{\partial C}{\partial w_{jk}^l} = \frac{\partial C}{\partial \boldsymbol{a}_j^l} \frac{\partial \boldsymbol{a}_j^l}{\partial w_{jk}^l} \tag{8.56}$$

$$= z_k^{l-1} \boldsymbol{\delta}_j^l \tag{8.57}$$

$$\frac{\partial C}{\partial w} = \boldsymbol{\delta}_{\text{out}} \boldsymbol{z}_{\text{in}}{}^T, \tag{8.58}$$

When dealing with **batch** training instances, we sum derivatives with respect to parameters over these instances.

8.3.2 Efficiency of back propagation

8.3.3 The Jacobian matrix

Back-propagation can also be applied to the calculation of other derivatives.Consider the evaluation of the **Jacobian matrix**,whose elements are given by the derivatives of the network outputs with respects to the inputs

$$J_{ki} = \frac{\partial y_k}{\partial x_i} \tag{8.59}$$

8.4 The Hessian Matrix

8.5 Regularization in Neural Networks

8.6 Mixture Density Networks

8.7 Bayesian Neural Networks

Chapter 9
Sparse Kernel Machines

9.1 Introduction

One of significant limitations of kernel methods is that the kernel function $\|(\boldsymbol{x}_n, \boldsymbol{x}_m)$ must be evaluated for all possible pairs $\boldsymbol{x}_n$ and $\boldsymbol{x}_m$ of training points,computationally infeasible.Kernel-based algorithms that have **sparse** solutions predict for new inputs depend only on the kernel function evaluated at a subset of the training data points,such as **suport vector machine**(SVM).

9.2 Maximum Margin Classifiers

Two-class classification problem using linear models

$$y(x) = \boldsymbol{w}^T \boldsymbol{\phi}(x) + b \tag{9.1}$$

where $\boldsymbol{\phi}(x)$ denotes a fixed feature-space transformation,and b is bias parameter.

For linear separable feature space,the parameters satisfies $y(\boldsymbol{x}_n) > 0$ for points having $t_n = +1$ and $y(\boldsymbol{x}_n) < 0$ for $t_n = -1$,so that $t_n y(\boldsymbol{x}_n) > 0$ for all points.

Margin is the smallest distance between the decision boundary and any of the samples.The maximum margin solution can be motivated using **computational learning theory**,also known as **statistical learning theory**.

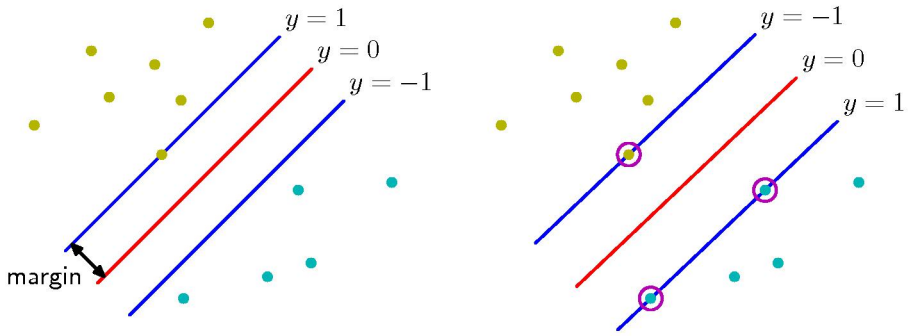


Fig. 9.1: margin

The **funtional margin** $t_n y(\boldsymbol{x}) > 0$ for data points correctly classified.The maximize it

$$\boldsymbol{w}, b = \arg\max_{\boldsymbol{w},b} \{ \frac{1}{\| \boldsymbol{w} \|} \min_n [t_n(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_n) + b)] \} \tag{9.2}$$

We can rescale parameters to set

$$t_n(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_n) + b) = 1 \tag{9.3}$$

for point closet to the surface.Then all data points satisfies the constraint

$$t_n(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_n) + b) \geq 1, n = 1, ..., N. \tag{9.4}$$

This is the **canonical representation of the decision hyperplane**.The optimization problem is equivalent to

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 \tag{9.5}$$

subject to constraints 9.4,which is a **quadratic programming** problem.

Introducing Lagrange multipliers $a_n \geq 0$

$$L(\boldsymbol{w},b,\boldsymbol{a}) = \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 - \sum_{n=1}^{N} a_n\{t_n(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_n)+b)-1\} \tag{9.6}$$

where $\boldsymbol{a} = (a_1,...,a_N)^T$.Setting the derivatives of L with respect to $\boldsymbol{w}$ and b equal to zero,we obtain conditions

$$\boldsymbol{w} = \sum_{n=1}^{N} a_n t_n \boldsymbol{\phi}(\boldsymbol{x}_n) \tag{9.7}$$

$$0 = \sum_{n=1}^{N} a_n t_n \tag{9.8}$$

Eliminating $\boldsymbol{w}$ and b gives the **dual representation** of the maximum margin problem

$$\hat{L}(\boldsymbol{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\boldsymbol{x}_n,\boldsymbol{x}_m) \tag{9.9}$$

with respect to $\boldsymbol{a}$ subject to the constraints

$$a_n \geq 0, n = 1,...,N \tag{9.10}$$

$$\sum_{n=1}^{N} a_n t_n = 0. \tag{9.11}$$

The kernel function is defined by $k(\boldsymbol{x},\boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{\phi}(\boldsymbol{x}')$.

The solution to a quadratic programming problem in M variables has computational complexity of $O(M^3)$.

$y(\boldsymbol{x})$ can be expressed by

$$y(\boldsymbol{x}) = \sum_{n=1}^{N} a_n t_n k(\boldsymbol{x},\boldsymbol{x}_n) + b. \tag{9.12}$$

The $Karush-Kuhn-Tucker(KKT)$ conditions require following the properties hold

$$a_n \geq 0 \tag{9.13}$$

$$t_n y(\boldsymbol{x}_n) - 1 \geq 0 \tag{9.14}$$

$$a_n\{t_n y(\boldsymbol{x}_n) - 1 = 0\} \tag{9.15}$$

Data points for which $a_n = 0$ will disappear and remaining ones are called **suport vectors**.They lie on the maximum margin hyperplanes in feature space.

Having solved the quadratic programming problem,the threshold parameter b

$$t_n(\sum_{m\in\mathcal{S}} a_m t_m k(\boldsymbol{x_n},\boldsymbol{x_m}) + b) = 1 \tag{9.16}$$

where $\mathcal{S}$ denotes the set of indices of the support vectors.Multiply through by t_n,making use of $t_n 2 = 1$,and then average these equations over all support vectors

$$b = \frac{1}{N_{\mathcal{S}}}\sum_{n\in\mathcal{S}}(t_n - \sum_{m\in\mathcal{S}} a_m t_m \|(\boldsymbol{x}_n,\boldsymbol{x}_m)) \tag{9.17}$$

where N_S is the total number of support vectors. Express the maximum-margin classifier in terms of the minimization of an error function with a quadratic regularizer

$$\sum_{n=1}^{N} E_\infty(y(\boldsymbol{x}_n)t_n - 1) + \lambda \parallel \boldsymbol{w} \parallel^2 \tag{9.18}$$

where $E_\infty(z)$ is zero if $z \geq 0$ and ∞ otherwise to ensure the margin constraint.

9.2.1 Overlapping class distributions

In practice,the class-conditional distributions may overlap,in which case exact separation of the training data can lead to poor generalization.Introduce **slack variables**, $\xi_n \geq 0$ where $n = 1,...N$,one for each training data points.We allow points on the 'wrong side' but with a penalty that increases of the distance from the boundary.

$$\xi_n = \mid t_n - y(\boldsymbol{x}_n) \mid \tag{9.19}$$

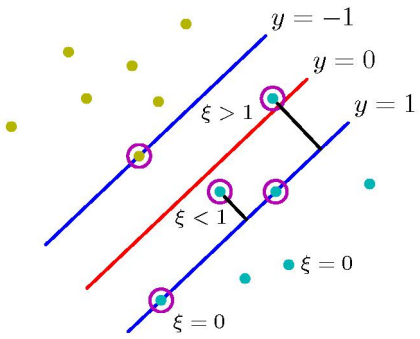


Fig. 9.2: slack variables $\varepsilon_n \geq 0$.Data points with circles around them are support vectors

Then the classification constraint are replaced by

$$t_n y(\boldsymbol{x}_n) \geq 1 - \xi_n, n = 1, ..., N \tag{9.20}$$

This is described as relaxing the hard margin constraint to give a **soft margin** and allows misclassification of training set data points.

Maximize the margin with softly penalized points on the wrong side of the margin boundary.

$$C \sum_{n=1}^{N} \xi_n + \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 \tag{9.21}$$

where the parameter C controls the trade-off between the slack variable penalty and the margin,minimizing errors and controlling model complexity.In the limit $C \longrightarrow \infty$,the model gets more complex,less data points are misclassified.

Minimization with constraint

$$L(\boldsymbol{w}, b\boldsymbol{a}) = \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 + C \sum_{n=1}^{N} \xi_n - \sum_{n=1}^{N} a_n \{t_n y(\boldsymbol{x}_n) - 1 + \xi_n\} - \sum_{n=1}^{N} \mu_n \xi_n \tag{9.22}$$

where $\{a_n \geq 0\}$ and $\{\mu_n \geq 0\}$ are Lagrange multipliers.The KKT set of conditions are given by

$$a_n \geq 0 \tag{9.23}$$
$$t_n y(\boldsymbol{x}_n) - 1 + \xi_n \geq 0 \tag{9.24}$$
$$a_n(t_n y(\boldsymbol{x}_n) - 1 + \xi_n) = 0 \tag{9.25}$$
$$\mu_n \geq 0 \tag{9.26}$$
$$\xi_n \geq 0 \tag{9.27}$$
$$\mu_n \xi_n = 0 \tag{9.28}$$

where $n = 1, ..., N$.

Optimize out $\boldsymbol{w}, b$ and $\{\xi_n\}$

$$\frac{\partial L}{\partial \boldsymbol{w}} = 0 \Rightarrow \boldsymbol{w} = \sum_{n=1}^{N} a_n t_n \tag{9.29}$$
$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N} a_n t_n = 0 \tag{9.30}$$
$$\frac{\partial L}{\partial \xi_n} = 0 \Rightarrow a_n = C - \mu_n \tag{9.31}$$

Eliminated,the dual Lagrangian is in the form

$$\hat{L}(\boldsymbol{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m \|(\boldsymbol{x}_n, \boldsymbol{x}_m) \tag{9.32}$$

which is identical to the separable case,with different constraints:

$$0 \leq a_n \leq C \tag{9.33}$$
$$\sum_{n=1}^{N} a_n t_n = 0 \tag{9.34}$$

for $n = 1, ..., N$,where the former are known as **box constraints**.

A subset of data points having $a_n = 0$ do not contribute to the predictive model.Support vectors have $a_n > 0$ and satisfy

$$t_n y(\boldsymbol{x}_n) = 1 - \xi_n \tag{9.35}$$

if $a_n < C$,then implies that $\mu_n > 0$,which requires $\xi_n = 0$ and hence such points lie on the margin.Points with $a_n = C$ can lie inside the margin and can either be correctly classified if $\xi_n \leq 1$ or misclassified if $\xi_n > 1$.

To determine b,we note that support vectors for which $0 \leq a_n \leq C$ have $\xi_n = 0$ so that $t_n y(\boldsymbol{x}_n) = 1$ and hence satisfy

$$t_n(\sum_{m \in \mathcal{S}} a_m t_m \|(\boldsymbol{x}_n, \boldsymbol{x}_m) + b) = 1 \tag{9.36}$$

A numerically stable solution is obtained by averaging:

$$b = \frac{1}{N_{\mathcal{M}}} \sum_{n \in \mathcal{M}} (t_n - \sum_{m \in \mathcal{S}} a_m t_m \|(\boldsymbol{x}_n, \boldsymbol{x}_m)) \tag{9.37}$$

where $\mathcal{M}$ denotes the set of indices of data points having $0 \leq a_n \leq C$.

9.2.2 Multiclass Support Vector Machine loss

the score for the j-th class is the j-th element: $s_j = f(\boldsymbol{x}_i, \boldsymbol{W})_j$. The Multiclass SVM loss for the i-th example is then formalized as follows:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta) \tag{9.38}$$

With linear score function $f(\boldsymbol{x}_i; \boldsymbol{W}) = \boldsymbol{W}\boldsymbol{x}_i$, rewrite the loss function in this equivalent form:

$$L_i = \sum_{j \neq y_i} \max(0, \boldsymbol{w}_j^T x_i - \boldsymbol{w}_{y_i}^T x_i + \Delta) \tag{9.39}$$

The threshold at zero $max(0,-)$ is called **hinge loss**.And there is a squared hinge loss (L2-SVM). The gradient of loss function for a single point i:

$$\nabla_{w_{y_i}} L_i = -\left(\sum_{j \neq y_i} \mathbb{1}(w_j^T x_i - w_{y_i}^T x_i + \Delta > 0) \right) x_i \tag{9.40}$$

where $\mathbb{1}$ is the indicator function that is one if the condition inside is true or zero otherwise. For the rows where $j \neq y_i$ the gradient is:

$$\nabla_{w_j} L_i = \mathbb{1}(w_j^T x_i - w_{y_i}^T x_i + \Delta > 0) x_i \tag{9.41}$$

9.2.2.1 ν-SVM

Maximize

$$\hat{L}(\boldsymbol{a}) = -\frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m \|(\boldsymbol{x}_n, \boldsymbol{x}_m) \tag{9.42}$$

subject to the constraints

$$0 \leq a_n \leq 1/N \tag{9.43}$$

$$\sum_{n=1}^{N} a_n t_n = 0 \tag{9.44}$$

$$\sum_{n=1}^{N} a_n \geq \nu \tag{9.45}$$

The parameter ν,which replaces C,can be interpreted as both an upper bound on the fraction of **margin errors** and a lower bound on the fraction of support vectors.

9.2.2.2 Optimization

chunking Lagrangian is unchanged if we remove the rows and columns of the kernel matrix corresponding to Lagrange multipliers.Implemented using **protected conjugate gradients**.

Decomposition methods solves a series of smaller quadratic programming problems but are designed so that each of these is of a fixed size.

sequantial minimal optimization or SMO Takes the concept of chunking to the extreme limit and considers just two Lagrange multipliers at a time.Heuristics are given for choosing the pair of Lagrange multipliers to be considered at each step.

Support vector machines don't manage to avoid the curse of dimensionality because there are constraints amongst the feature values that restrict the effective dimensionality of feature space.

9.2.2.3 Relation to Logistic Regression

The objective function can be written

$$\sum_{n=1}^{N} E_{SV}(y_n t_n) + \lambda \parallel \boldsymbol{w} \parallel^2 \tag{9.46}$$

where $\lambda = (2C)^{-1}$,and $E_{SV}(\cdot)$ is the **hinge** error function defined by

$$E_{SV}(y_n t_n) = [1 - y_n t_n]_+ \tag{9.47}$$

where $[\cdot]_+$ denotes the positive part.

For logistic regression,target variable $t \in \{-1, 1\}$,so

$$p(t|y) = \sigma(yt) \tag{9.48}$$

Construct an error function by taking the negative logarithm of likelihood function with a quadratic regularizer

$$\sum_{n=1}^{N} E_{LR}(y_n t_n) + \lambda \parallel \boldsymbol{w} \parallel^2 \tag{9.49}$$

where

$$E_{LR}(yt) = \ln(1 + \exp(-yt)) \tag{9.50}$$

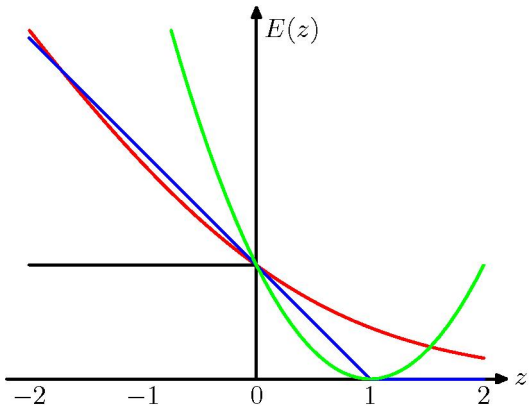


Fig. 9.3: Plot of the hinge error function used in support vector machines, shown in blue, along with the error function for logistic regression, rescaled by a factor of 1/ ln(2) so that it passes through the point (0, 1), shown in red. Also shown are the misclassification error in black and the squared error in green.

9.2.2.4 Multiclass SVMs

one-versus-the-rest approach:K separate SVMs for each class.

one-versus-one .$K(K-1)/2$ different 2-class SVMs on possible pairs of classes,which can lead to ambiguities.

single-class support vector machines,which solve an unsupervised learning problem related to probability density estimation.

9.2.2.5 SVMs for regression

In simple linear regression we minimize a regularized error function given by

$$\frac{1}{2}\sum_{n=1}^{N}\{y_n - t_n\}^2 + \frac{\lambda}{2}\parallel \boldsymbol{w} \parallel^2 . \tag{9.51}$$

To obtain **sparse solutions**,the quadratic error function is replaced by an **ε-insensitive error function**.For example

$$E_\varepsilon(y(\boldsymbol{x}) - t) = \begin{cases} 0, & \text{if } \mid y(\boldsymbol{x}) - t \mid < \varepsilon; \\ y(\boldsymbol{x}) - t \mid -\varepsilon, & \text{otherwise} \end{cases} \tag{9.52}$$

We therefore minimize

$$C\sum_{n=1}^{N} E_\varepsilon(y(\boldsymbol{x}_n) - t_n) + \frac{1}{2}\mid \boldsymbol{w} \mid^2 \tag{9.53}$$

where $y(\boldsymbol{x})$ is the prediction function.The (inverse) regularization parameter denoted C,appears in front of the error term.

Introducing two **slack variables** for each data point $\boldsymbol{x}_n$.The condition for target points to lie inside the ε-tube is that $y_n - \varepsilon \leq t_n \leq y_n + \varepsilon$,and for those outside:

$$\begin{cases} \xi_n & \geq 0 \\ \hat{\xi}_n & \geq 0, n = 1, ..., N \end{cases} \tag{9.54}$$

$$y(\boldsymbol{x}_n) + \varepsilon < t_n \leq y(\boldsymbol{x}_n) + \varepsilon + \xi_n \tag{9.55}$$

$$y(\boldsymbol{x}_n) - \varepsilon > t_n \geq y(\boldsymbol{x}_n) - \varepsilon - \hat{\xi}_n \tag{9.56}$$

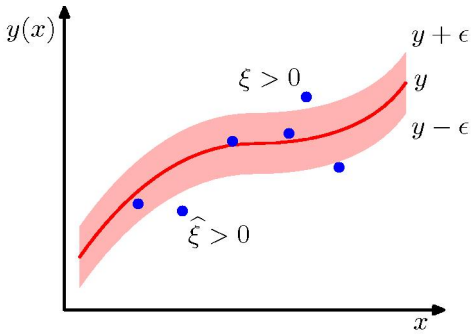


Fig. 9.4: Illustration of SVM regression, showing the regression curve together with the ε-insensitive tube.

The error function for support vector regression can then be written as

$$C\sum_{n=1}^{N}(\xi_n + \hat{\xi}_n) + \frac{1}{2}\parallel \boldsymbol{w} \parallel^2 \tag{9.57}$$

which must be minimized subject to the constraints.Introducing Lagrange multipliers

$$L = C\sum_{n=1}^{N}(\xi_n + \hat{\xi}_n) + \frac{1}{2}\parallel \boldsymbol{w} \parallel^2 - \sum_{n=1}^{N}(\mu_n\xi_n + \hat{\mu_n}\hat{\xi}_n) \tag{9.58}$$

$$- \sum_{n=1}^{N} a_n(\varepsilon + \xi_n + y_n - t_n) - \sum_{n=1}^{N}\hat{a_n}(\varepsilon + \hat{\xi}_n + y_n - t_n) \tag{9.59}$$

Substitute for $y(\boldsymbol{x})$ using 9.1 and set the derivatives of the Lagrangian with respect to $\boldsymbol{x}, b, \xi_n, \hat{\xi}_n$ to zero,giving

$$\frac{\partial L}{\partial \boldsymbol{w}} = 0 \Rightarrow \boldsymbol{w} = \sum_{n=1}^{N} (a_n - \hat{a_n})\phi(\boldsymbol{x}_n) \tag{9.60}$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N} (a_n - \hat{a_n}) = 0 \tag{9.61}$$

$$\frac{\partial L}{\partial \xi_n} = 0 \Rightarrow a_n + \mu_n = 0 \tag{9.62}$$

$$\frac{\partial L}{\partial \hat{\xi}_n} = 0 \Rightarrow \hat{a_n} + \hat{\mu}_n = 0 \tag{9.63}$$

$$\tag{9.64}$$

Using these to eliminate the corresponding variables,we see the dual problem of maximizing

$$\hat{L}(\boldsymbol{a}, \hat{\boldsymbol{a}}) = -\frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} (a_n - \hat{a_n})(a_m - \hat{a_m}) \| (\boldsymbol{x}_n, \boldsymbol{x}_m) \tag{9.65}$$

$$-\varepsilon \sum_{n=1}^{N} (a_n + \hat{a_n}) + \sum_{n=1}^{N} (a_n - \hat{a_n}) t_n \tag{9.66}$$

with respect to $\{a_n\}, \{\hat{a_n}\}$.We have the **box constraints**

$$0 \leq a_n \leq C \tag{9.67}$$

$$0 \leq \hat{a_n} \leq C \tag{9.68}$$

After substitution,the predictions can be made using

$$y(\boldsymbol{x}) = \sum_{n=1}^{N} (a_n - \hat{a_n}) \| (\boldsymbol{x}, \boldsymbol{x}_n) + b \tag{9.69}$$

The corresponding $karush - Kuhn - Tucker$(KKT) conditons for 9.58,which state that **at the solution the product of the dual variables and the constraints must vanish** are given by

$$a_n(\varepsilon + \xi_n + y_n - t_n) = 0 \tag{9.70}$$

$$\hat{a_n}(\varepsilon + \hat{\xi}_n - y_n + t_n) = 0 \tag{9.71}$$

$$(C - a_n)\xi_n = 0 \tag{9.72}$$

$$(C - \hat{a_n})\hat{\xi}_n = 0 \tag{9.73}$$

The support vectors are those data points that contribute to predictions,in other words those for which either $a_n \neq 0$ or $\hat{a_n} \neq 0$.These are points lying on the boundary of the ε-tube or outside the tube.

The parameter b satisfies

$$\varepsilon + y_n - t_n = 0 \tag{9.74}$$

for points $0 < a_n < C$.Solving for it

$$b = t_n - \varepsilon - \boldsymbol{w}^T \phi(\boldsymbol{x}_n) \tag{9.75}$$

$$= t_n - \varepsilon - \sum_{m=1}^{N} (a_m - \hat{a_m}) \| (\boldsymbol{x}_n, \boldsymbol{x}_m) \tag{9.76}$$

An alternative formulation of SVM regression is ν SVM.

$$\hat{L}(\boldsymbol{a},\hat{\boldsymbol{a}}) = -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}(a_n-\hat{a_n})(a_m-\hat{a_m})\|(\boldsymbol{x}_n,\boldsymbol{x}_m) \tag{9.77}$$

$$-0\times\varepsilon\sum_{n=1}^{N}(a_n+\hat{a_n})+\sum_{n=1}^{N}(a_n-\hat{a_n})t_n \tag{9.78}$$

subject to constraints

$$0\leq a_n\leq C/N \tag{9.79}$$

$$0\leq \hat{a_n}\leq C/N \tag{9.80}$$

$$\sum_{n=1}^{N}(a_n-\hat{a_n})=0 \tag{9.81}$$

$$\sum_{n=1}^{N}(a_n+\hat{a_n})\leq \nu C \tag{9.82}$$

$$\tag{9.83}$$

There are at most νN data points falling outside the insensitive tube,which at least νN data points are support vectors and so lie either on the tube or outside it.

9.3 Relevance Vector Machines

TODO

Chapter 10
Graphical Models

10.1 Introduction

10.2 Directed Graph

10.3 Undirected Graph

10.4 Inference in Graphical Models

10.4.1 Factor graphs

The joint distribution over a set of variables in the form of a product of factors

$$p(\boldsymbol{x}) = \prod_s f_s(\boldsymbol{x}_s) \tag{10.1}$$

where $\boldsymbol{x}_s$ denotes a set of the variables.We shall denote individual variable by x_i,which can comprise groups of variables(such as vectors and matrices).Each factor f_s is a function of a corresponding set of variable $\boldsymbol{x}_s$. Directed graphs have factors $f_s(\boldsymbol{x}_s)$ as local conditional distributions.Undirected graphs' factors are potential function over the maximal cliques.

10.4.2 The Sum-product algorithm

sum-product algorithm solves the problem of evaluating the **local marginals** over nodes or subsets of nodes.And **max-sum** algorithm find the **most probable state**,evaluating the maximal joint distribution.

The marginal is obtained by

$$p(x) = \sum_{\boldsymbol{x}\setminus x} p(\boldsymbol{x}) \tag{10.2}$$

where $\setminus x$ denotes the set of variables in $\boldsymbol{x}$ omitting x.

Join distribution

$$p(\boldsymbol{x}) = \prod_{s\in \mathrm{ne}(x)} F_s(x, \boldsymbol{X}_s) \tag{10.3}$$

Then interchanging the sums and products,we obtain

$$\text{objective} = p(x) = \sum_{\boldsymbol{x}\setminus x} p(\boldsymbol{x}) \tag{10.4}$$

$$= \prod_{s\in \mathrm{ne}(x)} \left[\sum_{X_s} F_s(x, X_s) \right] \tag{10.5}$$

$$= \prod_{s\in \mathrm{ne}(x)} \mu_{f_s\to x}(x) \tag{10.6}$$

View **messages** from the factor nodes f_s to the variable node x as

$$\mu_{f_s \to x}(x) \equiv \sum_{X_s} F_s(x, X_s) \tag{10.7}$$

$$F_s(x, X_s) = f_s(x, x_1, \ldots, x_M) G_1(x_1, X_{s1}) \ldots G_M(x_M, X_{sM}) \tag{10.8}$$

Recursive inference in the sub-graph and interchanging sums and products leads to

$$\mu_{f_s \to x}(x) = \sum_{x_1} \ldots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m \in \mathrm{ne}(f_s) \setminus x} \mu_{x_m \to f_s}(x_m) \tag{10.9}$$

where $\mathrm{ne}(x)$ denotes the set of neighbor variables. And define **messages** from variable nodes to factor nodes

$$\mu_{x_m \to f_s}(x_m) \equiv \sum_{X_{sm}} G_m(x_m, X_{sm}) \tag{10.10}$$

Again,making use of (sub)-graph factorization,we have

$$\mu_{x_m \to f_s}(x_m) = \prod_{l \in \mathrm{ne}(x_m) \setminus f_s} \left[\sum_{X_{ml}} F_l(x_m, X_{ml}) \right] \tag{10.11}$$

$$= \prod_{l \in \mathrm{ne}(x_m) \setminus f_s} \mu_{f_l \to x_m}(x_m) \tag{10.12}$$

10.4.3 Max-sum algorithm

The **sum-product** algorithm allows us to take a joint distribution $p((x))$ expressed as a factor graph and efficiently find marginal component variables.**Max-sum** find a setting of variables that has the largest probability,which is an application of **dynamic programming**. Maximize the joint distribution

$$\boldsymbol{x}^{max} = \arg\max_{\boldsymbol{x}} p(\boldsymbol{x}) \tag{10.13}$$

Joint distribution

$$p(\boldsymbol{x}) = \prod_{s \in \mathrm{ne}(x)} F_s(x, X_s) \tag{10.14}$$

Making use of **distributive law** for multiplication with max operator,similarly to add operator

$$\text{objective} = \max \ln p(\boldsymbol{x}) = \max \sum \ln F_s(x, X_s) \tag{10.15}$$

$$= \sum \max \ln F_s(x, X_s) \tag{10.16}$$

Making use of recursion,factorize $F_s(x, X_s)$ with sub-graph,we can obtain

$$\mu_{f \to x} = \max_{x_1, \ldots, x_M} \left[\ln f(x, x_1, \ldots, x_M) + \sum_{m \in \mathrm{ne}(f_s) \setminus x} \mu_{x_m \to f}(x_m) \right] \tag{10.17}$$

$$\mu_{x \to f}(x) = \sum_{l \in \mathrm{ne}(x) \setminus f} \mu_{f_l \to x}(x). \tag{10.18}$$

where $\mu_{f \to x} \equiv \max \ln F_s(x, X_s)$

Chapter 11
Mixture models and EM

11.1 Introduction

If we define a joint distribution over observed and latent variables,the corresponding distribution of the observed variables alone is marginalization.Mixture distributions can be interpreted in terms of discrete latent variables.As well as providing framework for building more complex probability distributions,mixture models can also be used to cluster data.

11.2 K-means Clusetering

11.2.1 representation

Suppose we have a data set $\{\boldsymbol{x}_1,...,\boldsymbol{x}_N\}$ consisting of N observations of a random D-dimensional Euclidean variable $\boldsymbol{x}$.Our goal is to partition the data set into some number K of clusters.First introduce a set of D-dimensional vectors $\boldsymbol{\mu}_k$,where $k=1,...,K$,in which $\boldsymbol{\mu}_k$ is a prototype associated with the k^{th} cluster,representing the centers of the clusters.

Our goal is then to find an assignment of data points to clusters,as well as a set of vectors $\{\boldsymbol{\mu}_k\}$,such that the sum of squares of the distances of each data point to its closest vector $\boldsymbol{\mu}_k$,is a minimum.

A corresponding set of binary indicator variables $r_{nk}\in\{0,1\}$,describing which of the K clusters the data point $\boldsymbol{x}_n$ is assigned to,so that if data point $\boldsymbol{x}_n$ is assigned to cluster k then $r_{nk}=1$,and $r_{nj}=0$ for $j\neq k$.This is the 1-of-K coding scheme.

11.2.2 evaluation

Define the objective function,sometimes called a **distortion measure** as

$$J=\sum_{n=1}^{N}\sum_{k=1}^{K}r_{nk}\parallel\boldsymbol{x}_n-\boldsymbol{\mu}_k\parallel^2 \tag{11.1}$$

11.2.3 optimization

Our goal is to find value for the $\{r_{nk}\}$ and the $\{\boldsymbol{\mu}_k\}$ so as to minimize J.We can do this through an iterative procedure in which each iteration involves two successive steps corresponding to successive optimizations with respect to the r_{nk} and the $\boldsymbol{\mu}_k$,which can be interpreted as parameters of each cluster's probability distribution.

Consider first the determination of the r_{nk},which can be interpreted as likelihood function of each cluster.We simply assign the n^{th} data point to the closest cluster centre.

$$r_{nk}=\begin{cases}1,\text{if } k=\arg\min_j\parallel\boldsymbol{x}_n-\boldsymbol{\mu}_j\parallel^2\\0,\text{otherwise}\end{cases} \tag{11.2}$$

Note that

$$\arg\min_j \| \boldsymbol{x}_n - \boldsymbol{\mu}_j \|^2 = \arg\min_j (\| \boldsymbol{x}_n \|^2 - 2 \| \boldsymbol{x}_n \| \| \boldsymbol{\mu}_j \| + \| \boldsymbol{\mu}_j \|^2) \tag{11.3}$$

$$= \arg\min_j (-2 \| \boldsymbol{x}_n \| \| \boldsymbol{\mu}_j \| + \| \boldsymbol{\mu}_j \|^2) \tag{11.4}$$

$$= \arg\max_j (\boldsymbol{x}_n^T \boldsymbol{\mu}_j - \frac{\boldsymbol{\mu}_j^T \boldsymbol{\mu}_j}{2}) \tag{11.5}$$

Now consider the optimization of the $\boldsymbol{\mu}_k$ with the $_{nk}$ held fixed.Setting the objective function J's derivative with respect to $\boldsymbol{\mu}_k$ to zero giving:

$$\nabla_{\boldsymbol{\mu}_k} J = 2 \sum_{n=1}^{N} r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k) = 0 \tag{11.6}$$

$$\Rightarrow \boldsymbol{\mu}_k = \frac{\sum_n r_{nk} \boldsymbol{x}_n}{\sum_n r_{nk}} \tag{11.7}$$

$$\Rightarrow \boldsymbol{\mu}_k^T = \frac{\sum_n r_{nk} \boldsymbol{x}_n^T}{\sum_n r_{nk}} \tag{11.8}$$

which can be vectorized when implementing.

Algorithm 1: K-MEANS coordinate descent

Input: A set $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$

Output: $\{r_{nk}\}$ and the $\{\boldsymbol{\mu}_k\}$ such that J is minimized.

1 initialize $\boldsymbol{\mu}_k$;

2 **repeat**

3 E(expectation).Minimize J with respect to the r_{nk}.$r_{nk} = \begin{cases} 1, \text{if } k = \arg\min_j \| \boldsymbol{x}_n - \boldsymbol{\mu}_j \|^2 \\ 0, \text{otherwise} \end{cases}$,keeping $\boldsymbol{\mu}_k$ fixed ;

4 M(maximization).Minimize J with respect to the $\boldsymbol{\mu}_k$: $\frac{\boldsymbol{\mu}_k = \sum_n r_{nk} \boldsymbol{x}_n}{\sum_n r_{nk}}$,keeping r_{nk} fixed.;

5 **until** *converged*;

The objective may converge to a local rather than global minimum(MacQueen 1967).

Derive an on-line stochastic rather than the batch version of K-means,by applying Robbin-Monro procedure :

$$\boldsymbol{\mu}_k^{new} = \boldsymbol{\mu}_k^{old} + \eta_n (\boldsymbol{x}_n - \boldsymbol{\mu}_k^{old}) \tag{11.9}$$

where η_n is the learning rate parameter,which is typically made to decrease monotonically as more data points are considered.

Generalize the K-means algorithm,introducing a general dissimilarity measure $v(\boldsymbol{x}, \boldsymbol{x}')$

$$\hat{J} = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} v(\boldsymbol{x}_n, \boldsymbol{\mu}_k) \tag{11.10}$$

which gives the ***K*-medoids** algorithm.The computation cost of E step is $O(KN)$.The M step is potentially more complex.So it is common to restrict each cluster prototype to be equal to one of the data vectors assigned to that cluster,requiring $O(N_k^2)$ evaluations of $v(\cdot, \cdot)$.

11.3 Mixtures of Gaussians

11.3.1 representation

The Gaussian mixture distribution can be written as a linear superposition of Gaussians in the form

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{11.11}$$

Introduce a K-dimensional binary random variable $\boldsymbol{z}$ having a 1-of-K representation.

$$z_k \in \{0,1\} \tag{11.12}$$

$$\sum_k z_k = 1 \tag{11.13}$$

The marginal distribution over $\boldsymbol{z}$ is specified in terms of the mixing coefficients π_k,such that

$$p(z_k = 1) = \pi_k \tag{11.14}$$

where the parameters $\{\pi_k\}$ must satisfy

$$0 \leq \pi_k \leq 1 \tag{11.15}$$

$$\sum_{k=1}^{K} \pi_k = 1 \tag{11.16}$$

Write this distribution in the form

$$p(\boldsymbol{z}) = \prod_{k=1}^{K} \pi_k^{z_k} \tag{11.17}$$

The conditional distribution of $\boldsymbol{x}$ given a particular value for $\boldsymbol{z}$ is a Gaussian:

$$p(\boldsymbol{x}|z_k = 1) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{11.18}$$

$$p(\boldsymbol{x}|\boldsymbol{z}) = \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_k} \tag{11.19}$$

The joint distribution is given by $p(\boldsymbol{z})p(\boldsymbol{x}|\boldsymbol{z})$,and the marginal distribution of $\boldsymbol{x}$ is obtained by summing the joint distribution over states of $\boldsymbol{z}$:

$$p(\boldsymbol{x}) = \sum_{\boldsymbol{z}} p(\boldsymbol{z}) p(\boldsymbol{x}|\boldsymbol{z}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{11.20}$$

which is a Gaussian mixture 11.11 .

The conditional probability of $\boldsymbol{z}$ given $\boldsymbol{x}$ can be evaluated by Bayes' theorem

$$\gamma(z_k) \equiv p(z_k = 1|\boldsymbol{x}) = \frac{p(z_k = 1)p(\boldsymbol{x}|z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(\boldsymbol{x}|z_j = 1)} \tag{11.21}$$

$$= \frac{\pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{11.22}$$

We shall view π_k as the prior probability of $z_k = 1$,and the quantity $\gamma(z_k)$ as the corresponding posterior.$\gamma(z_k)$ can also be viewed as the **responsibility** that component k takes for 'explaining' the observation $\boldsymbol{x}$.

11.3.2 Maximum likelihood

Assuming that the data points are drawn independently from the same distribution(i.i.d),the log of the likelihood function is given by

$$\ln(p(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})) = \sum_{n=1}^{N} \ln\{\sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\} \tag{11.23}$$

It is worth emphasizing that there are significant problems associated with the maximum likelihood framework applied to GMM,due to the presence of **singularities**.If one of the components of the mixture model 'collapse' onto a specific data point($\sigma_j \to 0$),the log likelihood function will go to infinity.These singularities provide another example of severe over-fitting of maximum likelihood approach.A further issue is **identifiability**(Casella and Berger,2002),which is that a K-component mixture results in $K!$ equivalent solutions.

11.3.3 EM for Gaussian mixtures

An elegant and powerful method for finding the maximum likelihood solutions for models with latent variables is called the **expectation-maximization**,or EM algorithm.

Begin by writing down the conditions that must be satisfied at a maximum of the likelihood function.Setting the derivatives of $\ln p(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})$ 11.23 with respect to the means $\boldsymbol{\mu}_k$ of the Gaussian components to zero,we obtain

$$\because 0 = -\sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)} \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}_k) \tag{11.24}$$

$$\therefore 0 = -\sum_{n=1}^{N} \gamma(z_{nk})(\boldsymbol{x}_n - \boldsymbol{\mu}_k) \tag{11.25}$$

$$\because N_k = \sum_{n=1}^{N} \gamma(z_{nk}) \tag{11.26}$$

$$\therefore \boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \boldsymbol{x}_n \tag{11.27}$$

$$\tag{11.28}$$

where we have made use of $\dfrac{de^x}{dx} = e^x$ and 2.63.We can interpret N_k as the effective number of points assigned to cluster k.

Now set the derivative of $\ln(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})$ with respect to $\boldsymbol{\Sigma}_k$ to zero:

$$\frac{\partial \ln p(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})}{\partial \boldsymbol{\Sigma}_k} = \sum_{n=1}^{N} \frac{\frac{\partial(\pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k))}{\partial \boldsymbol{\Sigma}_k}}{\sum_{j=1}^{K}\pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)} \tag{11.29}$$

$$\because \frac{\partial \ln y}{\partial x} = \frac{1}{y}\frac{\partial y}{\partial x} \tag{11.30}$$

$$\therefore \frac{\partial y}{\partial x} = y\frac{\partial \ln y}{\partial x} \tag{11.31}$$

$$\because \frac{\partial(\pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k))}{\partial \boldsymbol{\Sigma}_k} = \pi_k \frac{\partial \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\partial \boldsymbol{\Sigma}_k} \tag{11.32}$$

$$= \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\frac{\partial \ln \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\partial \boldsymbol{\Sigma}_k} \tag{11.33}$$

$$\because \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}|^{1/2}}\exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\} \tag{11.34}$$

$$\therefore \frac{\partial \ln \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\partial \boldsymbol{\Sigma}_k} = -\frac{1}{2}(\frac{\ln(|\boldsymbol{\Sigma}_k|}{\partial \boldsymbol{\Sigma}_k} + \frac{\partial(\boldsymbol{x}_n-\boldsymbol{\mu}_j)\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_n-\boldsymbol{\mu}_j)}{\partial \boldsymbol{\Sigma}_k}) \tag{11.35}$$

$$\because \frac{\partial det(\boldsymbol{X})}{\boldsymbol{X}} = det(\boldsymbol{X})(\boldsymbol{X}^{-1})^T \tag{11.36}$$

$$\frac{\partial \boldsymbol{a}^T\boldsymbol{X}^{-1}\boldsymbol{b}}{\partial \boldsymbol{X}} = -\boldsymbol{X}^{-T}\boldsymbol{a}\boldsymbol{b}^T\boldsymbol{X}^{-T} \tag{11.37}$$

$$\therefore \frac{\ln(|\boldsymbol{\Sigma}_k|}{\partial \boldsymbol{\Sigma}_k} = \frac{\frac{\partial |\boldsymbol{\Sigma}_k|}{\partial \boldsymbol{\Sigma}_k}}{|\boldsymbol{\Sigma}_k|} = \frac{|\boldsymbol{\Sigma}_k|\boldsymbol{\Sigma}_k^{-T}}{|\boldsymbol{\Sigma}_k|} = \boldsymbol{\Sigma}_k^{-1} \tag{11.38}$$

$$\therefore \frac{\partial \ln \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\partial \boldsymbol{\Sigma}_k} = -\frac{1}{2}(\boldsymbol{\Sigma}_k^{-1} - \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n-\boldsymbol{\mu}_k)(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}) \tag{11.39}$$

$$\therefore derivative = \sum_{n=1}^{N}\gamma(z_{nk})(-1/2)[\boldsymbol{\Sigma}_k^{-1} - \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n-\boldsymbol{\mu}_k)(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}] = 0 \tag{11.40}$$

$$\therefore \boldsymbol{\Sigma}_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})(\boldsymbol{x}_n-\boldsymbol{\mu}_k)(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^T \tag{11.41}$$

where we have made use of 2.63 and **The Matrix Cookbook,2012**.

Finally,we maximize $\ln p(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})$ 11.23 with respect to the mixing coefficients π_k.Achieve this using a Lagrange multiplier

$$\ln p(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) + \lambda(\sum_{k=1}^{K}\pi_k - 1) = 0 \tag{11.42}$$

$$\therefore \sum_{n=1}^{N}\frac{\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K}\pi_j\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)} + \lambda = 0 \tag{11.43}$$

where again we see the appearance of the responsibilities 11.21.If we now multiply both sides by π_k and sum over k making use of constraint $\sum_{k=1}^{K}\pi_k = 1$,we find $\lambda = -N$.Then

$$\pi_k = \frac{N_k}{N} \tag{11.44}$$

There do suggest s ample iterative scheme for finding a solution to the maximum likelihood problem.

Algorithm 2: EM for Gaussian Mixtures

repeat

1.Initialize the means $\boldsymbol{\mu}_k$,covariances $\boldsymbol{\Sigma}_k$ and mixing coefficients π_k,and evaluate the initial value of the log likelihood

2.**E step**.Evaluate the responsibilities using the current parameter values

$$\gamma(z_k) \equiv p(z_k = 1|\boldsymbol{x}) = \frac{p(z_k = 1)p(\boldsymbol{x}|z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(\boldsymbol{x}|z_j = 1)} \tag{11.45}$$

$$= \frac{\pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{11.46}$$

3. **M step**. Re-estimate the parameters using the current responsibilities

$$\boldsymbol{\mu}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \boldsymbol{x}_n \tag{11.47}$$

$$\boldsymbol{\Sigma}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T \tag{11.48}$$

$$\pi_k^{new} = \frac{N_k}{N} \tag{11.49}$$

where

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}) \tag{11.50}$$

4. Evaluate the log likelihood

$$\ln(p(\boldsymbol{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})) = \sum_{n=1}^{N} \ln\{\sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\} \tag{11.51}$$

until *convergence of either the parameters or the log likelihood*

11.4 Latent Variables View of EM

11.4.1 representation

We denote the set of all observed data by $\boldsymbol{X}$,in which the n^{th} row represents $\boldsymbol{x}_n^T$,and denote the set of all latent variables by $\boldsymbol{Z}$,with corresponding rows $\boldsymbol{z}_n^T$.The set of all model parameters is denoted by $\boldsymbol{\theta}$,so the log likelihood function is given by

$$\ln p(\boldsymbol{X}|\boldsymbol{\theta}) = \ln\{\sum_{\boldsymbol{z}} p(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta})\} \tag{11.52}$$

Replace the sum over $\boldsymbol{Z}$ with an integral for continuous latent variables.

$\{\boldsymbol{X}, \boldsymbol{Z}\}$ is called the **complete** data set,and $\{\boldsymbol{X}\}$ is **incomplete**.We consider complete-data log likelihood functions' expected value under the posterior distribution of the latent variables,corresponding to the E step.In the subse-

quent M step,maximize this expectation.The expectation of complete-data log likelihood

$$\mathcal{Q}(\boldsymbol{\theta},\boldsymbol{\theta}^{old}) = \sum_{\boldsymbol{z}} p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}^{old}) \ln p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta}) \tag{11.53}$$

Maximize this

$$\boldsymbol{\theta}^{new} = \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta},\boldsymbol{\theta}^{old}) \tag{11.54}$$

Algorithm 3: General EM algorithm

Input: $p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta})$,$\boldsymbol{Z}$
Output: output

repeat

1. Choose an initial setting for the parameters $\boldsymbol{\theta}^{old}$.
2. **E step** Evaluate $p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}^{old})$.
3. **M step** Evaluate $\boldsymbol{\theta}^{new}$ given by

$$\mathcal{Q}(\boldsymbol{\theta},\boldsymbol{\theta}^{old}) = \sum_{\boldsymbol{z}} p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}^{old}) \ln p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta}) \tag{11.55}$$

where

$$\boldsymbol{\theta}^{new} = \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta},\boldsymbol{\theta}^{old}) \tag{11.56}$$

4.

$$\boldsymbol{\theta}^{old} \leftarrow \boldsymbol{\theta}^{new} \tag{11.57}$$

until *converged*

When finding MAP(maximum posterior) solutions,maximize the quantity $\mathcal{Q}(\boldsymbol{\theta},\boldsymbol{\theta}^{old}) + \ln p(\boldsymbol{\theta})$ in M step.

11.5 The EM Algorithm in General

The **expectation maximization** algorithm,or EM,is a general technique for finding maximum likelihood solutions for probabilistic models having latent variables.

Given the same setting as previous,our goal is to maximize the likelihood function

$$p(\boldsymbol{X}|\boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta}) \tag{11.58}$$

Replace the summation by integration as appropriate to apply to continuous latent variables.

Introduce a distribution $q(\boldsymbol{Z})$ over the latent variables,and the decompose the log complete-data likelihood function

$$\because \ln p(\boldsymbol{X}|\boldsymbol{\theta}) = \ln p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta}) - \ln p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}) \tag{11.59}$$

$$\tag{11.60}$$

take the expectation over values of $\boldsymbol{Z}$ by multiplying both sides by $q(\boldsymbol{Z})$ and summing(or integrting) over $\boldsymbol{Z}$ we get

$$\ln p(\boldsymbol{X}|\boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} q(\boldsymbol{Z}) \ln p(\boldsymbol{X}|\boldsymbol{\theta}) \tag{11.61}$$

$$= \sum_{\boldsymbol{Z}} q(\boldsymbol{Z}) \ln p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta}) - \sum_{\boldsymbol{Z}} q(\boldsymbol{Z}) \ln p(\boldsymbol{Z}|\boldsymbol{X}) \tag{11.62}$$

$$= \sum_{\boldsymbol{Z}} q(\boldsymbol{Z}) (\ln \frac{p(\boldsymbol{X},\boldsymbol{Z})}{p(\boldsymbol{Z})} + \ln p(\boldsymbol{Z})) - \sum_{\boldsymbol{Z}} q(\boldsymbol{Z}) \ln p(\boldsymbol{Z}|\boldsymbol{X}) \tag{11.63}$$

$$= Q + H \tag{11.64}$$

$$= \mathcal{L}(q,\boldsymbol{\theta}) + KL(q \parallel p) \tag{11.65}$$

where

$$\mathcal{L}(q,\boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} q(\boldsymbol{Z}) \ln \{\frac{p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta})}{q(\boldsymbol{Z})}\} \tag{11.66}$$

$$KL(q \parallel p) = -\sum_{\boldsymbol{Z}} q(\boldsymbol{Z}) \ln \{\frac{p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta})}{q(\boldsymbol{Z})}\} \tag{11.67}$$

Note that $\mathcal{L}(q,\boldsymbol{\theta})$ is a functional of the distribution $p(\boldsymbol{Z})$ and KL is the Kullback-Leibler divergence between $q(\boldsymbol{Z})$ and the posterior $p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta})$.KL divergence will be nonnegative according to Jenson's inequality about convex function.

$$\because KL(q \parallel q) \geq, \text{with equality if,and only if,} q(\boldsymbol{Z}) = p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}) \tag{11.68}$$

$$\therefore \mathcal{L}(q,\boldsymbol{\theta}) \leq \ln p(\boldsymbol{X}|\boldsymbol{\theta}) \tag{11.69}$$

in other words that $\mathcal{L}(q,\boldsymbol{\theta})$ is a lower bound on $\ln p(\boldsymbol{X}|\boldsymbol{\theta})$.

The EM is a two-stage **iterative** optimization technique for finding **maximum likelihood** solutions.With the decomposition 11.66, in the E step,the **lower bound** $\mathcal{L}(q,\boldsymbol{\theta}^{old})$ is maximized with respect to $q(\boldsymbol{Z})$ holding $\boldsymbol{\theta}^{old}$ fixed.

In the subsequent M step,the distribution $q(\boldsymbol{Z})$ is held fixed and the lower bound $\mathcal{L}(q,\boldsymbol{\theta})$ is maximized with respect to $\boldsymbol{\theta}$.The nonzero KL divergence causes the log likelihood function increase more than the lower bound,as shown in figure

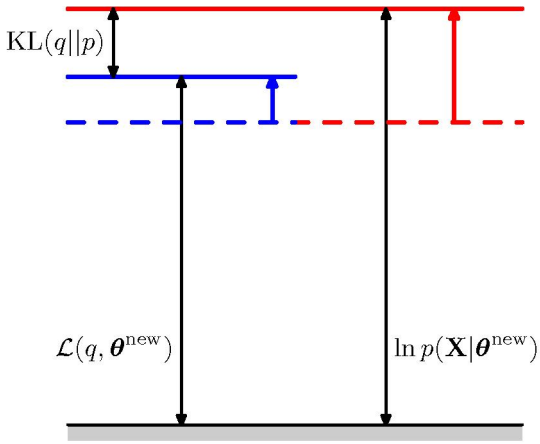


Fig. 11.1: Illustration of the M step of the EM algorithm. The distribution $q(\boldsymbol{Z})$ is held fixed and the lower bound $\mathcal{L}(q,\boldsymbol{\theta})$ is maximized with respect to the parameter vector $\boldsymbol{\theta}$ to give a revised value $\boldsymbol{\theta}^{new}$. Because the KL divergence is nonnegative, this causes the log likelihood ln p(X—) to increase by at least as much as the lower bound does

Substitute $q(\boldsymbol{Z}) = p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}^{old})$ into the lower bound,then after the E step,

$$\mathcal{L}(q,\boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}^{old}) \ln p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta}) - \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}^{old}) \ln p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}^{old}) \tag{11.70}$$

$$= \mathcal{Q}(\boldsymbol{\theta},\boldsymbol{\theta}^{old}) + const \tag{11.71}$$

where the constant is the negative entropy of the q distribution.

For the particular case of independent,identically distributed data set,$\boldsymbol{X} = (\boldsymbol{x}_1^T, \boldsymbol{x}_2^T, \cdots, \boldsymbol{x}_n^T)^T$ and $\boldsymbol{Z} = (\boldsymbol{z}_1^T, \boldsymbol{z}_2^T, \cdots, \boldsymbol{z}_n^T)^T$.Using the sum and product rules,the posterior in E step take the form

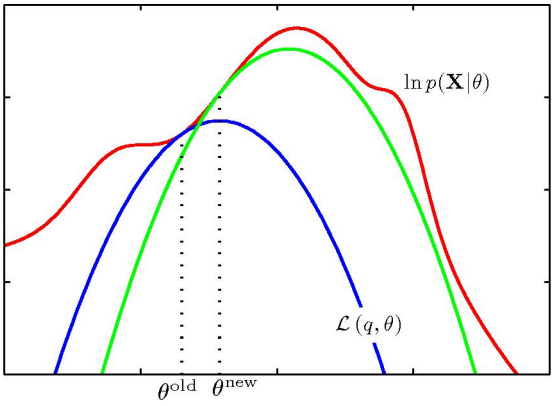


Fig. 11.2: The EM algorithm involves alternately computing a lower bound on the log likelihood for the current parameter values and then maximizing this bound to obtain the new parameter values.The red curve depicts the incomplete-data log likelihood function to maximize.curve where its value equals the log likelihood at $\boldsymbol{\theta}^{old}$ indicates the E step.In the M step,the lower bound is maximized giving $\boldsymbol{\theta}^{new}$.The shows the subsequent E step,constructing a tangential bound at $\boldsymbol{\theta}^{new}$

$$p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}) = \frac{p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta})}{\sum_{\boldsymbol{Z}} p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta})} = \frac{\prod_{n=1}^{N} p(\boldsymbol{x}_n,\boldsymbol{z}_n|\boldsymbol{\theta})}{\sum_{\boldsymbol{Z}} \prod_{n=1}^{N} p(\boldsymbol{x}_n,\boldsymbol{z}_n|\boldsymbol{\theta})} = \prod_{n=1}^{N} p(\boldsymbol{z}_n|\boldsymbol{x}_n,\boldsymbol{\theta}) \tag{11.72}$$

will also factorize with respect to n.

11.5.1 Maximum posterior

Maximize the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{X})$ with EM for models with a prior $p(\boldsymbol{\theta})$ over parameters.

$$\because p(\boldsymbol{\theta}|\boldsymbol{X}) = p(\boldsymbol{\theta},\boldsymbol{X})/p(\boldsymbol{X}) \tag{11.73}$$

$$\therefore \ln p(\boldsymbol{\theta}|\boldsymbol{X}) = \ln p(\boldsymbol{\theta},\boldsymbol{X}) - \ln p(\boldsymbol{X}) = \ln p(\boldsymbol{\theta}) + \ln p(\boldsymbol{X}|\boldsymbol{\theta}) - \ln p(\boldsymbol{X}) \tag{11.74}$$

decomposed as

$$\ln p(\boldsymbol{\theta}|\boldsymbol{X}) = \mathcal{L}(q,\boldsymbol{\theta}) + KL(q \parallel p) + \ln p(\boldsymbol{\theta}) - \ln p(\boldsymbol{X}) \tag{11.75}$$

$$\geq \mathcal{L}(q,\boldsymbol{\theta}) + \ln p(\boldsymbol{\theta}) - \ln p(\boldsymbol{X}) \tag{11.76}$$

where $\ln p(\boldsymbol{X})$ is constant.We can optimize the right-hand side alternatively with respect to q and $\boldsymbol{\theta}$ in E-step and M-step.

11.5.2 generalized EM

The **generalized EM**,or GEM,algorithm addresses the problem of an intractable M step.Instead of aiming to maximize $\mathcal{L}(q,\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$,it seeks instead to change the parameters in such a way as to increase its value.

1 Nonlinear optimization. conjugate gradients algorithm in M step
2 expectation conditional maximization(ECM). Making several constrained optimizations within each M step.

Generalize the E step by performing a **partial**,rather than complete, optimization of $\mathcal{L}(q,\boldsymbol{\theta})$ with respect to $q(\boldsymbol{Z})$.

Chapter 12
Continuous Latent Variables

12.1 Principal Component Analysis

12.1.1 Introduction

Principal Component Analysis is widely used for applications such as dimensionality reduction,lossy data compression,feature extraction,and data visualization.Also known as the Karhunen-Loeve transform.There are two definitions giving rise to the same algorithm.PCA can be defined as the orthogonal projection of the data onto a lower dimensional linear space,known as the principal subspace,such that the variance of the projected data is maximized.Equivalently,it can be defined as the linear projection the minimizes the average projection cost, defined as the linear projection that minimizes the average projection cost,defined as the mean squared distance between the data points and their projections.

12.1.2 Maximum variance formulation

Consider a data set of observations $\{x_n\}$ where $n = 1,...,N$,and x_n is a Euclidean variable with dimensionality D. Our goal is to project the data onto a space having dimensionality $M < D$ while maximizing the variance of the projected data.We define the direction of this space using a D-dimensional unit vector $\mathbf{u_1^T u_1} = 1$.Each data point $\mathbf{x_n}$ is then projected onto a scalar value $\mathbf{u_1^T x_n}$.The mean of the projected data is $\mathbf{u_1^T}\bar{\mathbf{x}}$ where the $\bar{\mathbf{x}}$ is the sample set mean given by

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x_n} \tag{12.1}$$

and the variance of the projected data is given by

$$\frac{1}{N}\sum_{n=1}^{N}\{\mathbf{u_1^T x_n} - \mathbf{u_1^T}\bar{\mathbf{x}}\}^2 = \frac{1}{N}\sum_{n=1}^{N}\{\mathbf{u_1^T}(\mathbf{x_n} - \bar{\mathbf{x}})\}^2 \tag{12.2}$$

$$= \frac{1}{N}\sum_{n=1}^{N}\{\mathbf{u_1^T}(\mathbf{x_n} - \bar{\mathbf{x}})(\mathbf{x_n} - \bar{\mathbf{x}})^{\mathbf{T}}\mathbf{u_1^T}\} \tag{12.3}$$

$$= \mathbf{u_1^T S u_1} \tag{12.4}$$

where $\mathbf{S}$ is the data covariance matrix defined by

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x_n} - \bar{\mathbf{x}})(\mathbf{x_n} - \bar{\mathbf{x}})^T \tag{12.5}$$

We now maximize the projected variance $\mathbf{u_1^T S u_1}$ with respect to $\mathbf{u_1}$,which is a constrained maximization to prevent $\| \mathbf{u_1} \| \to \infty$ .The appropriate constraint comes from the normalization condition $\mathbf{u_1^T u_1} = 1$.To enforce this constraint,we introduce a Lagrange multiplier that we shall denote by λ_1,and then make an unconstrained maximization of

$$\mathbf{u_1^T S u_1} + \lambda_1(1 - \mathbf{u_1^T u_1}) \tag{12.6}$$

By setting the derivative with respect to $\mathbf{u_1}$ equal to zero,we see that this quantity will have a stationary point when

$$\mathbf{S u_1} = \lambda_1 \mathbf{u_1} \tag{12.7}$$

which says that $\mathbf{u_1}$ must be an eigenvector of $\mathbf{S}$.If we left-multiply by $\mathbf{u_1^T}$ and make use of $\mathbf{u_1^T}u_1 = 1$,we see that the variance is given by

$$\mathbf{u_1^T S u_1} = \lambda_1 \tag{12.8}$$

and so the variance will be a maximum when we set $\mathbf{u_1}$ equal to the eigenvector having the largest eigenvalue λ_1.This eigenvector is known as the first principal component.

12.1.3 Minimum-error formulation

12.1.4 Applications of PCA

12.1.5 PCA for high-dimensional data

12.2 Probabilistic PCA

12.3 Kernel PCA

12.4 Nonlinear Latent Variable Models

Chapter 13
Sequential Data

13.1 Hidden Markov Models

13.1.1 representation

Directed Probabilistic Graphical Models with **Latent Variables** to represent **Hidden Markov Models**.Hidden Markov Models are **State Space Models**.

Introduce latent variable $\boldsymbol{z}_n$ with 1-of-K coding scheme. **Transition probabilities** or **transition matrix** $\boldsymbol{A}_{jk} = p(z_{nk} = 1|z_{n-1,j} = 1)$ and probabilities satisfy $0 \leq \boldsymbol{A}_{jk} \leq 1$ with $\sum_k A_{jk} = 1$.So that matrix $\boldsymbol{A}$ has $K(K-1)$ independent parameters.Write the conditional probability explicitly

$$p(\boldsymbol{z}_n|\boldsymbol{z}_{n-1}, \boldsymbol{A}) = \prod_{k=1}^{K}\prod_{j=1}^{K} \boldsymbol{A}_{jk}^{z_{n-1,j}z_{nk}} \tag{13.1}$$

The initial latent node does not have a parent node,so the marginal prior distribution is represented by a vector of probabilities $\boldsymbol{\pi}$ with $\pi_k \equiv p(z_{1k} = 1)$,so that

$$p(\boldsymbol{z}_1|\boldsymbol{\pi}) = \prod_{k=1}^{K} \boldsymbol{\pi}_k^{z_{1k}} \tag{13.2}$$

where $\sum_k^{\pi_k} = 1$. The **emission probabilities** can be represented in the form

$$p(\boldsymbol{x}_n|\boldsymbol{z}_n, \boldsymbol{\phi}) = \prod_{k=1}^{K} p(\boldsymbol{x}_n|\boldsymbol{\phi}_k)^{z_{nk}} \tag{13.3}$$

For **homogeneous** models,the joint distribution over both latent and observed variables is given by

$$p(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta}) = p(\boldsymbol{z}_1|\boldsymbol{\pi}) \left[\prod_{n=2}^{N} p(\boldsymbol{z}_n|\boldsymbol{z}_{n-1}, \boldsymbol{A})\right] \prod_{m=1}^{N} p(\boldsymbol{x}_m|\boldsymbol{z}_m, \boldsymbol{\phi}) \tag{13.4}$$

where $\boldsymbol{\theta} = \{\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{\phi}\}$ denotes the set of parameters governing the model.

13.1.2 evaluation

The likelihood function is obtained from the joint distribution by marginalizing over the latent variables

$$p(\boldsymbol{X}|\boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} p(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta}) \tag{13.5}$$

13.1.3 optimization

13.1.3.1 EM algorithm

Direct maximization of the likelihood function lead to complex expressions with no close-form solutions(not i.i.d),so we turn the **expectation maximization** (simplified **variational bayesian**).In the **E step**,take these parameters to find the posterior distribution of the latent variables $p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}^{\text{old}})$.Then use this posterior distribution to evaluate the expectation of the logarithm of complete-data likelihood function,as a function of parameters $\boldsymbol{\theta}$

$$\mathcal{Q}(\boldsymbol{\theta},\boldsymbol{\theta}^{\text{old}}) = \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}^{\text{old}}) \ln p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta}) \tag{13.6}$$

Introduce some notation

$$\gamma(\boldsymbol{z}_n) = p(\boldsymbol{z}_n|\boldsymbol{X},\theta^{\text{old}}) = \mathbb{E}[z_{nk}] = \sum_{\boldsymbol{z}_n} \gamma(\boldsymbol{z}_n) \tag{13.7}$$

$$\xi(\boldsymbol{z}_{n-1},\boldsymbol{z}_n) = p(\boldsymbol{z}_{n-1},\boldsymbol{z}_n|\boldsymbol{X},\boldsymbol{\theta}^{\text{old}}) = \mathbb{E}[z_{n-1,j}z_{nk}] = \sum_{\boldsymbol{z}_{n-1},\boldsymbol{z}_n} \xi(\boldsymbol{z}_{n-1},\boldsymbol{z}_n) z_{n-1,j}z_{nk} \tag{13.8}$$

Substitute the joint distribution $p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta})$13.4,we obtain

$$\mathcal{Q}(\boldsymbol{\theta},\boldsymbol{\theta}^{\text{old}}) = \sum k = K\gamma(z_1 k)\ln\pi_k + \sum_{n=2}^{N}\sum_{j=1}^{K}\sum_{k=1}^{K} \xi(\boldsymbol{z}_{n-1},\boldsymbol{z}_n)\ln \boldsymbol{A}_{jk} \tag{13.9}$$

$$+ \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma(z_{nk})\ln p(\boldsymbol{x}_n|\boldsymbol{\phi}_k) \tag{13.10}$$

where we have marginalized over the parameters' joint distribution.

In **M step** we maximize $\mathcal{Q}$ with respect to the parameters,in which we treat γ and ξ as constant.

In the case of discrete multinomial observed variables,the conditional distribution of the observations takes the form

$$p(\boldsymbol{x}|\boldsymbol{z}) = \prod_{i=1}^{D}\prod_{k=1}^{K} \mu_{ik}^{x_i z_k} \tag{13.11}$$

subject to $\sum_{i=1}^{D}\mu_{ik} = 1$. Introduce **Lagrangian multiplier** λ,the objective is to maximize

$$\mathcal{L}(\boldsymbol{\mu},\lambda) = \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma(z_{nk})\sum_{i=1}^{D} x_{ni}\ln\mu_{ik} + \lambda\left(\sum_{i=1}^{D}\mu_{ik} - 1\right) \tag{13.12}$$

Set the derivatives to zero

$$\because \begin{cases} \dfrac{\partial\mathcal{L}}{\partial\mu_{ik}} & = \sum_{n=1}^{N}\gamma(z_{nk})x_{ni}\dfrac{1}{\mu_{ik}} + \lambda = 0 \\ \dfrac{\partial\mathcal{L}}{\partial\lambda} & = \sum_{i=1}^{D}\mu_{ik} - 1 = 0 \end{cases} \tag{13.13}$$

$$\therefore \begin{cases} \mu_{ik} & = -\dfrac{\sum_{n=1}^{N}\gamma(z_{nk})x_{ni}}{\lambda} \\ \lambda & = \sum_{i=1}^{D}\gamma(z_{nk}) \end{cases} \tag{13.14}$$

$$\Longrightarrow \tag{13.15}$$

$$\mu_{ik} = \frac{\sum_{n=1}^{N}\gamma(z_{nk})x_{ni}}{\sum_{n=1}^{N}\gamma(z_{nk})} \tag{13.16}$$

13.1.3.2 forward-backward algorithm

The quantities γ and ξ are posterior distribution of latent variables,and can be obtained using a two-stage message passing algorithm,known as Baum-Welch,a.k.a alpha-beta algorithm.

13.1.3.3 scaling factors

In the recursion relationship,these probabilities are significantly less than unity,as we work our way forward along the chain,the values of α can go to zero exponentially quickly.

13.1.4 predict

13.1.5 title

13.2 Linear Dynamic Systems

TODO

Chapter 14
Combining Models

14.1 Introduction

committees Train L different models and then make predictions using the average of the predictions made by each model.

boosting Train multiple models in sequence in which the error function used to train a particular model depends on the performance of the previous models.

decision trees Different models are responsible for making predictions in different regions of input space.

mixtures of experts Models are viewed as mixture distributions in which the component densities,as well as the mixing coefficients,are conditioned on the input variables.

$$p(t|\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k(\boldsymbol{x}) p(t|\boldsymbol{x},k) \tag{14.1}$$

in which $\pi_k(\boldsymbol{x}) = p(k|\boldsymbol{x})$ represents the input-dependent mixing coefficients,and k indexes the model.

14.2 Bayesian Model Averaging

In Bayesian model averaging the whole data set is generated by a single model.By contrast,when we combine multiple models,we see that different data points within the data set can potentially be generated from different values of the latent variable $\boldsymbol{z}$ and hence by different components.

14.3 Committees

The simplest way to construct a committee is to average the predictions of a set of individual models,to cancel the contribution arising from variance and bias.

Bootstrap data to introduce variability between the different models within the committee.Suppose we generate M bootstrap data sets

$$y_{COM}(\boldsymbol{x}) = \frac{1}{M} \sum_{m=1}^{M} y_m(\boldsymbol{x}) \tag{14.2}$$

where $m = 1,...,M$.This procedure is known as **bootstrap aggregation** or **bagging**.

14.4 Boosting

Here we describe the most widely used form of boosting algorithm:**AdaBoost**,short for 'adaptive boosting'.The base classifiers are known as **weak learners** and are trained in **sequence** using a **weighted form of the data set** in which the weighting coefficient associated with each data point depends on the performance of the previous classifiers.

Consider a two-class classification problem,in which the training data comprises input vectors $\boldsymbol{x}_1,...,\boldsymbol{x}_N$ along with corresponding binary target variables $t_1,...,t_N$ where $t_n \in \{-1,1\}$.Each data point is given an associated weighting parameter w_n,initially set $1/N$ for all.A base classifier function $y(\boldsymbol{x}) \in \{-1,1\}$

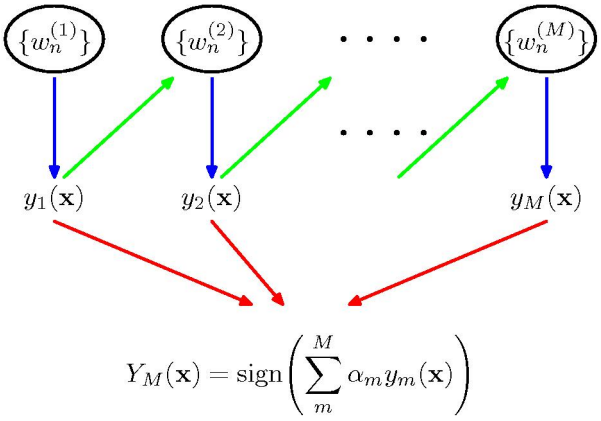


Fig. 14.1: Shematic illustration of boosting framework

Algorithm 4: AdaBoost

Input: A set $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$,$\{t_1, ..., t_N\}$

Output: $y(\boldsymbol{x})$,.

1 1. Initialize the data weighting coefficients $\{w_n\}$ by setting $w_n^1 = 1/N$ for $n = 1, ..., N$.

2 2. **for** $m = 1, ..., N$ **do**

3 (a) Fit a classifier $y_m(\boldsymbol{x})$ to the training data by minimizing the weighted error function

$$J_m = \sum_{n=1}^{N} w_n^{(m)} I(y_m(\boldsymbol{x}_n) \neq t_n) \tag{14.3}$$

where $I(y_m(\boldsymbol{x}_n) \neq t_n)$ is the indicator function and equals 1 when $y_m(\boldsymbol{x}_n) = t_n$ and 0 otherwise.

4 (b) Evaluate the quantities

$$\varepsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} I(y_m(\boldsymbol{x}_n) \neq t_n)}{\sum_{n=1}^{N} w_n^{(m)}} \tag{14.4}$$

and then use these to evaluate

$$\alpha_m = \ln\{\frac{1-\varepsilon_m}{\varepsilon_m}\}. \tag{14.5}$$

5 (c) Update the data weighting coefficients

$$w_n^{(m+1)} = w_n^{(m)} \exp\{\alpha_m I(y_m(\boldsymbol{x}_m) \neq t_n)\} \tag{14.6}$$

6 3. Make predictions using the final model,which is given by

$$Y_M(\boldsymbol{x}) = sign(\sum_{m=1}^{M} \alpha_m y_m(\boldsymbol{x})) \tag{14.7}$$

14.4.1 Minimizing exponential error

Consider the exponential error function defined by

$$E = \sum_{n=1}^{N} \exp\{-t_n f_m(\boldsymbol{x}_n)\} \tag{14.8}$$

where $f_m(\boldsymbol{x})$ is a classifier defined in terms of a linear combination of base classifiers $y_l(\boldsymbol{x})$ of the form

$$f_m(\boldsymbol{x}) = \frac{1}{2}\sum_{l=1}^{m} \alpha_l y_l(\boldsymbol{x}) \tag{14.9}$$

and $t_n \in \{-1, 1\}$ are the training set target values.Our goal is to minimize E with respect to the weighting coefficients α_l and parameters of the base classifiers $y_l(\boldsymbol{x})$.

Separating off the contribution from base classifier $y_m(\boldsymbol{x})$,

$$E = \sum_{n=1}^{N} \exp\{-t_n f_m(\boldsymbol{x}_n)\} \tag{14.10}$$

$$= \sum_{n=1}^{N} \exp\{-t_n \frac{1}{2}\sum_{l=1}^{m} \alpha_l y_l(\boldsymbol{x})\} \tag{14.11}$$

$$= \sum_{n=1}^{N} \exp\{-t_n f_{m-1}(\boldsymbol{x}_n) - \frac{1}{2} t_n \alpha_m y_m(\boldsymbol{x}_n)\} \tag{14.12}$$

$$= \sum_{n=1}^{N} w_n^{(m)} \exp\{-\frac{1}{2} t_n \alpha_m y_m(\boldsymbol{x}_n)\} \tag{14.13}$$

where the coefficients $w_n^{(m)} = \exp\{-t_n f_{m-1}(\boldsymbol{x}_n)\}$ can be viewed as constants because we are optimizing only α_m and $y_m(\boldsymbol{x})$.Denote by $\mathcal{T}_m$ the set of data points correctly classified by $y_m(\boldsymbol{x})$ and misclassified points by $\mathcal{M}_m$,then we in turn rewrite the error function

$$E = e^{-\alpha_m/2} \sum_{n\in\mathcal{T}_m} w_n^{(m)} + e^{\alpha_m/2} \sum_{n\in\mathcal{M}_m} w_n^{(m)} \tag{14.14}$$

$$= (e^{\alpha_m/2} - e^{-\alpha_m/2}) \sum_{n=1}^{N} w_n^{(m)} I(y_m(\boldsymbol{x}_n) \neq t_n) + e^{-\alpha_m/2} \sum_{n=1}^{N} w_n^{(m)} \tag{14.15}$$

Then we can minimize this with respect to $y_m(\boldsymbol{x}_n)$ and α_m.

$$\because w_n^{(m)} = \exp\{-t_n f_{m-1}(\boldsymbol{x}_n)\} \tag{14.16}$$

$$\therefore w_n^{(m+1)} = \exp\{-t_n f_m(\boldsymbol{x}_n)\} \tag{14.17}$$

$$\therefore w_n^{(m+1)} = w_n^{(m)} \exp\{-\frac{1}{2} t_n \alpha_m y_m(\boldsymbol{x}_n)\} \tag{14.18}$$

Making use of the fact that

$$t_n y_m(\boldsymbol{x}) = 1 - 2I(y_m(\boldsymbol{x}_n) \neq t_n) \tag{14.19}$$

we see updates at the next iteration

$$w_n^{(m+1)} = w_n^{(m)} \exp(-\alpha_m/2) \exp\{\alpha_m I(y_m(\boldsymbol{x}_n) \neq t_n)\} \tag{14.20}$$

Because the term $\exp(-\alpha_m/2)$ is independent of n,so can be discarded.

14.4.2 Error functions for boosting

14.5 Tree-based Models

14.6 Conditional Mixture Models

Appendix A
Matrix Decomposition

A.1 LU Decomposition

A.2 QR Decomposition

A.3 Cholesky Decomposition

A.4 Eigen value Decomposition

Definition A.1. An **eigenvector** of an $n \times n$ matrix A is a nonzero vector $\boldsymbol{x}$ such that $A\boldsymbol{x} = \lambda\boldsymbol{x}$ for some scalar λ.A scalar λ is called **eigenvalue** of A if there is a nontrivial solution $\boldsymbol{x}$ of $A\boldsymbol{x} = \lambda\boldsymbol{x}$;such an $\boldsymbol{x}$ is called an eigenvector corresponding to λ[1].

A.5 Singular Value Decomposition

A.5.1 Definition

Definition A.2. Any matrix can be decomposed as follows

$$\underbrace{\boldsymbol{X}}_{N\times D} = \underbrace{\boldsymbol{U}}_{N\times N}\underbrace{\boldsymbol{\Sigma}}_{N\times D}\underbrace{\boldsymbol{V}^T}_{D\times D} \tag{A.1}$$

where $\boldsymbol{U}$ is an $N \times N$ matrix whose columns are orthornormal(so $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}$), $\boldsymbol{V}$ is $D \times D$ matrix whose rows and columns are orthonormal (so $\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{V}\boldsymbol{V}^T = \boldsymbol{I}_D$), and $\boldsymbol{\Sigma}$ is a $N \times D$ matrix containing the $r = \min(N,D)$ singular values $\sigma_i \geq 0$ on the main diagonal, with 0s filling the rest of the matrix.

A.5.2 Proof

Let A be an $m \times n$ matrix.Then A^TA is symmetric and can be orthogonally diagonalized with eigenvectors.The **singular values** of A are the square root of the eigenvalues of A^TA,denoted by $\sigma_1, \sigma_2, ..., \sigma_n$.That is $\sigma_i = \sqrt{\lambda_i}$ for $1 \leq i \leq n$. The eigenvalues are usually arranged so that

$$\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n \geq 0 \tag{A.2}$$

Theorem A.1. *Suppose $\boldsymbol{v}_1, \boldsymbol{v_2}..., \boldsymbol{v_n}$ is an orthogonal basis of $\mathbb{R}^n$ consisting of eigenvector of A^TA,arranged so that the corresponding eigenvalues of A^TA satisfy $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n \geq 0$ and suppose A has r nonzero singular values.Then $A\boldsymbol{v_1}, ..., A\boldsymbol{v_r}$ is an orthogonal basis for ColA,and rankA = r.*

Proof. Because $\mathbf{v_i}$ and $\boldsymbol{v_j}$ are orthogonal for $i \neq j$,

$$(A\boldsymbol{v_i})^T(A\boldsymbol{v_j}) = \boldsymbol{v_i}^TA^TA\boldsymbol{v_j} = \boldsymbol{v_i}^T(\lambda_j\boldsymbol{v_j}) = 0 \tag{A.3}$$

We therefore have

[1] An eigenvalue may be zero

$$A\boldsymbol{v_i} = \sigma_i \boldsymbol{u_i} \tag{A.4}$$

For a general vector $\boldsymbol{x}$,since eigenvectors are orthogonal unit vectors,we have

$$\boldsymbol{x} = (\boldsymbol{v_1} \cdot \boldsymbol{x})\boldsymbol{v_1} + (\boldsymbol{v_2} \cdot \boldsymbol{x})\boldsymbol{v_2} + ... + (\boldsymbol{v_n} \cdot \boldsymbol{x})\boldsymbol{v_n} \tag{A.5}$$

This means that

$$M\boldsymbol{x} = (\boldsymbol{v_1} \cdot \boldsymbol{x})M\boldsymbol{v_1} + (\boldsymbol{v_2} \cdot \boldsymbol{x})M\boldsymbol{v_2} + ... + (\boldsymbol{v_n} \cdot \boldsymbol{x})M\boldsymbol{v_n} \tag{A.6}$$

$$M\boldsymbol{x} = (\boldsymbol{v_1} \cdot \boldsymbol{x})\sigma_1\boldsymbol{u_1} + (\boldsymbol{v_2} \cdot \boldsymbol{x})\sigma_2\boldsymbol{u_2} + ... + (\boldsymbol{v_n} \cdot \boldsymbol{x})\sigma_n\boldsymbol{u_n} \tag{A.7}$$

Remember that dot product can be computed using the vector transpose

$$\boldsymbol{v} \cdot \boldsymbol{u} = \boldsymbol{v^T u} \tag{A.8}$$

which leads to

$$M\boldsymbol{x} = \boldsymbol{u_1}\sigma_1\boldsymbol{v_1^T x} + \boldsymbol{u_2}\sigma_2\boldsymbol{v_2^T x} + ... + \boldsymbol{u_n}\sigma_n\boldsymbol{v_n^T x} \tag{A.9}$$

$$M = \boldsymbol{u_1}\sigma_1\boldsymbol{v_1^T} + \boldsymbol{u_2}\sigma_2\boldsymbol{v_2^T} + ... + \boldsymbol{u_n}\sigma_n\boldsymbol{v_n^T} \tag{A.10}$$

And this is usually expressed by writing

$$M = U\Sigma V^T \tag{A.11}$$

As for $\boldsymbol{u_i}$,we have

$$\begin{cases} (A^TA)\boldsymbol{v_i} = \lambda_i\boldsymbol{v_i} \\ A\boldsymbol{v_i} = \sigma_i\boldsymbol{u_i} \end{cases}$$
$$\Rightarrow A^T\sigma_i\boldsymbol{u_i} = \lambda_i\boldsymbol{v_i}$$
$$\Rightarrow \sigma_i A^T\boldsymbol{u_i} = \lambda_i\boldsymbol{v_i}$$
$$\Rightarrow (AA^T)\boldsymbol{u_i} = \sigma_i A\boldsymbol{v_i} = \lambda_i\boldsymbol{u_i}$$

So we can see that $\boldsymbol{u_i}$ is the eigenvector of symmetric matrix AA^T,and $\boldsymbol{v_i}$ is the eigenvector of symmetric matrix A^TA.In summary,$\boldsymbol{u_i}$ and $\boldsymbol{v_i}$ are the **left-eigenvector** and **right-eigenvectors** of matrix A.

A.5.3 Application

A.5.3.1 Principal Component Analysis

The projection vectors for principal component projection are the left-eigenvectors

Appendix B
Optimization methods

B.1 Convexity

Definition B.1. (**Convex set**) We say aset $\mathcal{S}$ is convex if for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{S}$, we have

$$\lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2 \in \mathcal{S}, \forall \lambda \in [0,1] \tag{B.1}$$

Definition B.2. (**Convex function**) A function $f(\boldsymbol{x})$ is called convex if its **epigraph**(the set of points above the function) defines a convex set. Equivalently, a function $f(\boldsymbol{x})$ is called convex if it is defined on a convex set and if, for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{S}$, and any $\lambda \in [0,1]$, we have

$$f(\lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2) \leq \lambda f(\boldsymbol{x}_1) + (1-\lambda) f(\boldsymbol{x}_2) \tag{B.2}$$

Definition B.3. A function $f(\boldsymbol{x})$ is said to be **strictly convex** if the inequality is strict

$$f(\lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2) < \lambda f(\boldsymbol{x}_1) + (1-\lambda) f(\boldsymbol{x}_2) \tag{B.3}$$

Definition B.4. A function $f(\boldsymbol{x})$ is said to be (strictly) **concave** if $-f(\boldsymbol{x})$ is (strictly) convex.

Theorem B.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 B.1. $\log(x)$ *is strictly convex on* $(0,\infty)$.

Intuitively, a (strictly) convex function has a bowl shape, and hence has a unique global minimum x^* corresponding to the bottom of the bowl. Hence its second derivative must be positive everywhere, $\frac{\mathrm{d}^2}{\mathrm{d}x^2} f(x) > 0$. A twice-continuously differentiable, multivariate function f is convex iff its Hessian is positive definite for all $\boldsymbol{x}$. In the machine learning context, the function f often corresponds to the NLL.

Models where the NLL is convex are desirable, since this means we can always find the globally optimal MLE. We will see many examples of this later in the book. However, many models of interest will not have concave likelihoods. In such cases, we will discuss ways to derive locally optimal parameter estimates.

B.2 Gradient descent

B.2.1 Stochastic gradient descent

Algorithm 5: 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**
      $\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta y_i \boldsymbol{x}_i$;
      $b \leftarrow b + \eta y_i$;
      $k \leftarrow k + 1$;

B.2.2 Batch gradient descent

B.2.3 Line search

The **line search**[2] approach first finds a descent direction along which the objective function f will be reduced and then computes a step size that determines how far $\boldsymbol{x}$ should move along that direction. The descent direction can be computed by various methods, such as gradient descent(Section B.2), Newton's method(Section B.4) and Quasi-Newton method(Section B.5). The step size can be determined either exactly or inexactly.

B.2.4 Momentum term

B.3 Lagrange duality

B.3.1 Primal form

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

$$xyz \tag{B.4}$$

B.3.2 Dual form

B.4 Newton's method

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}_k) + \boldsymbol{g}_k^T(\boldsymbol{x}-\boldsymbol{x}_k) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_k)^T \boldsymbol{H}_k(\boldsymbol{x}-\boldsymbol{x}_k)$$

$$\text{where } \boldsymbol{g}_k \triangleq \boldsymbol{g}(\boldsymbol{x}_k) = f'(\boldsymbol{x}_k), \boldsymbol{H}_k \triangleq \boldsymbol{H}(\boldsymbol{x}_k),$$

$$\boldsymbol{H}(\boldsymbol{x}) \triangleq \left[\frac{\partial^2 f}{\partial x_i \partial x_j}\right]_{D\times D} \quad \text{(Hessian matrix)}$$

$$f'(\boldsymbol{x}) = \boldsymbol{g}_k + \boldsymbol{H}_k(\boldsymbol{x}-\boldsymbol{x}_k) = 0 \Rightarrow \tag{B.5}$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \boldsymbol{H}_k^{-1}\boldsymbol{g}_k \tag{B.6}$$

Algorithm 6: Newtons method for minimizing a strictly convex function

```
Initialize x_0
while (!convergency) do
    Evaluate g_k = ∇f(x_k)
    Evaluate H_k = ∇²f(x_k)
    d_k = −H_k^{-1} g_k
    Use line search to find step size η_k along d_k
    x_{k+1} = x_k + η_k d_k
```

[2] http://en.wikipedia.org/wiki/Line_search

B.5 Quasi-Newton method

From Equation B.5 we can infer out the **quasi-Newton condition** as follows:

$$\begin{aligned} f'(\boldsymbol{x}) - \boldsymbol{g}_k &= \boldsymbol{H}_k(\boldsymbol{x} - \boldsymbol{x}_k) \\ \boldsymbol{g}_{k-1} - \boldsymbol{g}_k &= \boldsymbol{H}_k(\boldsymbol{x}_{k-1} - \boldsymbol{x}_k) \Rightarrow \\ \boldsymbol{g}_k - \boldsymbol{g}_{k-1} &= \boldsymbol{H}_k(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) \\ \boldsymbol{g}_{k+1} - \boldsymbol{g}_k &= \boldsymbol{H}_{k+1}(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k) \quad \text{(quasi-Newton condition)} \end{aligned} \tag{B.7}$$

The idea is to replace $\boldsymbol{H}_k^{-1}$ with a approximation $\boldsymbol{B}_k$, which satisfies the following properties:

1. $\boldsymbol{B}_k$ must be symmetric
2. $\boldsymbol{B}_k$ must satisfies the quasi-Newton condition, i.e., $\boldsymbol{g}_{k+1} - \boldsymbol{g}_k = \boldsymbol{B}_{k+1}(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k)$.
   Let $\boldsymbol{y}_k = \boldsymbol{g}_{k+1} - \boldsymbol{g}_k$, $\boldsymbol{\delta}_k = \boldsymbol{x}_{k+1} - \boldsymbol{x}_k$, then
   $$\boldsymbol{B}_{k+1}\boldsymbol{y}_k = \boldsymbol{\delta}_k \tag{B.8}$$
3. Subject to the above, $\boldsymbol{B}_k$ should be as close as possible to $\boldsymbol{B}_{k-1}$.

Note that we did not require that $\boldsymbol{B}_k$ be positive definite. That is because we can show that it must be positive definite if $\boldsymbol{B}_{k-1}$ is. Therefore, as long as the initial Hessian approximation $\boldsymbol{B}_0$ is positive definite, all $\boldsymbol{B}_k$ are, by induction.

B.5.1 DFP

Updating rule:

$$\boldsymbol{B}_{k+1} = \boldsymbol{B}_k + \boldsymbol{P}_k + \boldsymbol{Q}_k \tag{B.9}$$

From Equation B.8 we can get

$$\boldsymbol{B}_{k+1}\boldsymbol{y}_k = \boldsymbol{B}_k\boldsymbol{y}_k + \boldsymbol{P}_k\boldsymbol{y}_k + \boldsymbol{Q}_k\boldsymbol{y}_k = \boldsymbol{\delta}_k$$

To make the equation above establish, just let

$$\begin{aligned} \boldsymbol{P}_k\boldsymbol{y}_k &= \boldsymbol{\delta}_k \\ \boldsymbol{Q}_k\boldsymbol{y}_k &= -\boldsymbol{B}_k\boldsymbol{y}_k \end{aligned}$$

In DFP algorithm, $\boldsymbol{P}_k$ and $\boldsymbol{Q}_k$ are

$$\boldsymbol{P}_k = \frac{\boldsymbol{\delta}_k\boldsymbol{\delta}_k^T}{\boldsymbol{\delta}_k^T\boldsymbol{y}_k} \tag{B.10}$$

$$\boldsymbol{Q}_k = -\frac{\boldsymbol{B}_k\boldsymbol{y}_k\boldsymbol{y}_k^T\boldsymbol{B}_k}{\boldsymbol{y}_k^T\boldsymbol{B}_k\boldsymbol{y}_k} \tag{B.11}$$

B.5.2 BFGS

Use $\boldsymbol{B}_k$ as a approximation to $\boldsymbol{H}_k$, then the quasi-Newton condition becomes

$$\boldsymbol{B}_{k+1}\boldsymbol{\delta}_k = \boldsymbol{y}_k \tag{B.12}$$

The updating rule is similar to DFP, but $\boldsymbol{P}_k$ and $\boldsymbol{Q}_k$ are different. Let

$$\boldsymbol{P}_k \boldsymbol{\delta}_k = \boldsymbol{y}_k$$
$$\boldsymbol{Q}_k \boldsymbol{\delta}_k = -\boldsymbol{B}_k \boldsymbol{\delta}_k$$

Then

$$\boldsymbol{P}_k = \frac{\boldsymbol{y}_k \boldsymbol{y}_k^T}{\boldsymbol{y}_k^T \boldsymbol{\delta}_k} \tag{B.13}$$

$$\boldsymbol{Q}_k = -\frac{\boldsymbol{B}_k \boldsymbol{\delta}_k \boldsymbol{\delta}_k^T \boldsymbol{B}_k}{\boldsymbol{\delta}_k^T \boldsymbol{B}_k \boldsymbol{\delta}_k} \tag{B.14}$$

B.5.3 Broyden

Broyden's algorithm is a linear combination of DFP and BFGS.

Glossary

Use the template *glossary.tex* together with the Springer document class SVMono (monograph-type books) or SVMult (edited books) to style your glossary in the Springer layout.

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.

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.

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.

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.

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.