

A Brief Introduction to Machine Learning for Engineers

Other titles in Foundations and Trends® in Signal Processing

Massive MIMO Networks: Spectral, Energy, and Hardware Efficiency

Emil Bjornson, Jakob Hoydis and Luca Sanguinetti

ISBN: 978-1-68083-985-2

Using Inertial Sensors for Position and Orientation Estimation

Manon Kok, Jeroen D. Hol and Thomas B. Schon

ISBN: 978-1-68083-356-0

Computational Visual Attention Models

Milind S. Gide and Lina J. Karam

ISBN: 978-1-68083-280-8

Video Coding: Part II of Fundamentals of Source and Video Coding

Thomas Wiegand and Heiko Schwarz

ISBN: 978-1-68083-178-8

A Brief Introduction to Machine Learning for Engineers

Oswaldo Simeone

Department of Informatics

King's College London

osvaldo.simeone@kcl.ac.uk

now

the essence of knowledge

Boston — Delft



Documents

Document Sharing Hub

Foundations and Trends[®] in Signal Processing

Published, sold and distributed by:

now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:

now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

O. Simeone. *A Brief Introduction to Machine Learning for Engineers*. Foundations and Trends[®] in Signal Processing, vol. 12, no. 3-4, pp. 200–431, 2018.

ISBN: 978-1-68083-473-4

© 2018 O. Simeone

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, mechanical, photocopying, recording or otherwise, without prior written permission of the publishers.

Photocopying. In the USA: This journal is registered at the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923. Authorization to photocopy items for internal or personal use, or the internal or personal use of specific clients, is granted by now Publishers Inc for users registered with the Copyright Clearance Center (CCC). The ‘services’ for users can be found on the internet at: www.copyright.com

For those organizations that have been granted a photocopy license, a separate system of payment has been arranged. Authorization does not extend to other kinds of copying, such as that for general distribution, for advertising or promotional purposes, for creating new collective works, or for resale. In the rest of the world: Permission to photocopy must be obtained from the copyright owner. Please apply to now Publishers Inc., PO Box 1024, Hanover, MA 02339, USA; Tel. +1 781 871 0245; www.nowpublishers.com; sales@nowpublishers.com

now Publishers Inc. has an exclusive license to publish this material worldwide. Permission to use this content must be obtained from the copyright license holder. Please apply to now Publishers, PO Box 179, 2600 AD Delft, The Netherlands, www.nowpublishers.com; e-mail: sales@nowpublishers.com

Foundations and Trends[®] in Signal Processing

Volume 12, Issue 3-4, 2018

Editorial Board

Editor-in-Chief

Yonina Eldar

Technion
Israel

Editors

Pao-Chi Chang
*National Central
University*

Pamela Cosman
*University of California,
San Diego*

Michelle Effros
*California Institute of
Technology*

Yariv Ephraim
George Mason University

Alfonso Farina
Selex ES

Sadaoki Furui
*Tokyo Institute of
Technology*

Georgios Giannakis
University of Minnesota

Vivek Goyal
Boston University

Sinan Gunturk
Courant Institute

Christine Guillemot
INRIA

Robert W. Heath, Jr.
*The University of Texas at
Austin*

Sheila Hemami
Northeastern University

Lina Karam
Arizona State University

Nick Kingsbury
University of Cambridge

Alex Kot
*Nanyang Technical
University*

Jelena Kovacevic
*Carnegie Mellon
University*

Geert Leus
TU Delft

Jia Li
*Pennsylvania State
University*

Henrique Malvar
Microsoft Research

B.S. Manjunath
*University of California,
Santa Barbara*

Urbashi Mitra
*University of Southern
California*

Björn Ottersten
KTH Stockholm

Vincent Poor
Princeton University

Anna Scaglione
*University of California,
Davis*

Mihaela van der Shaar
*University of California,
Los Angeles*

Nicholas D. Sidiropoulos
*Technical University of
Crete*

Michael Unser
EPFL

P.P. Vaidyanathan
*California Institute of
Technology*

Ami Wiesel
*The Hebrew University of
Jerusalem*

Min Wu
University of Maryland

Josiane Zerubia
INRIA

Editorial Scope

Topics

Foundations and Trends® in Signal Processing publishes survey and tutorial articles in the following topics:

- Adaptive signal processing
- Audio signal processing
- Biological and biomedical signal processing
- Complexity in signal processing
- Digital signal processing
- Distributed and network signal processing
- Image and video processing
- Linear and nonlinear filtering
- Multidimensional signal processing
- Multimodal signal processing
- Multirate signal processing
- Multiresolution signal processing
- Nonlinear signal processing
- Randomized algorithms in signal processing
- Sensor and multiple source signal processing, source separation
- Signal decompositions, subband and transform methods, sparse representations
- Signal processing for communications
- Signal processing for security and forensic analysis, biometric signal processing
- Signal quantization, sampling, analog-to-digital conversion, coding and compression
- Signal reconstruction, digital-to-analog conversion, enhancement, decoding and inverse problems
- Speech/audio/image/video compression
- Speech and spoken language processing
- Statistical/machine learning
- Statistical signal processing
 - Classification and detection
 - Estimation and regression
 - Tree-structured methods

Information for Librarians

Foundations and Trends® in Signal Processing, 2018, Volume 12, 4 issues. ISSN paper version 1932-8346. ISSN online version 1932-8354. Also available as a combined paper and online subscription.

Contents

I	Basics	5
1	Introduction	6
1.1	What is Machine Learning?	6
1.2	When to Use Machine Learning?	8
1.3	Goals and Outline	11
2	A Gentle Introduction through Linear Regression	15
2.1	Supervised Learning	15
2.2	Inference	17
2.3	Frequentist Approach	19
2.4	Bayesian Approach	35
2.5	Minimum Description Length (MDL)*	42
2.6	Information-Theoretic Metrics	44
2.7	Interpretation and Causality*	46
2.8	Summary	48
3	Probabilistic Models for Learning	51
3.1	Preliminaries	52
3.2	The Exponential Family	53
3.3	Frequentist Learning	59
3.4	Bayesian Learning	62

3.5	Supervised Learning via Generalized Linear Models (GLM)	69
3.6	Maximum Entropy Property*	71
3.7	Energy-based Models*	72
3.8	Some Advanced Topics*	73
3.9	Summary	74
II	Supervised Learning	75
4	Classification	76
4.1	Preliminaries: Stochastic Gradient Descent	77
4.2	Classification as a Supervised Learning Problem	78
4.3	Discriminative Deterministic Models	80
4.4	Discriminative Probabilistic Models: Generalized Linear Models	90
4.5	Discriminative Probabilistic Models: Beyond GLM	96
4.6	Generative Probabilistic Models	101
4.7	Boosting*	105
4.8	Summary	106
5	Statistical Learning Theory*	112
5.1	A Formal Framework for Supervised Learning	113
5.2	PAC Learnability and Sample Complexity	118
5.3	PAC Learnability for Finite Hypothesis Classes	119
5.4	VC Dimension and Fundamental Theorem of PAC Learning	123
5.5	Summary	125
III	Unsupervised Learning	128
6	Unsupervised Learning	129
6.1	Unsupervised Learning	130
6.2	K -Means Clustering	133
6.3	ML, ELBO and EM	135
6.4	Directed Generative Models	146
6.5	Undirected Generative Models	154
6.6	Discriminative Models	157
6.7	Autoencoders	159

6.8	Ranking*	162
6.9	Summary	163
IV	Advanced Modelling and Inference	164
7	Probabilistic Graphical Models	165
7.1	Introduction	166
7.2	Bayesian Networks	169
7.3	Markov Random Fields	177
7.4	Bayesian Inference in Probabilistic Graphical Models	181
7.5	Summary	184
8	Approximate Inference and Learning	185
8.1	Monte Carlo Methods	186
8.2	Variational Inference	188
8.3	Monte Carlo-based Variational Inference*	196
8.4	Approximate Learning*	199
8.5	Summary	200
V	Conclusions	202
9	Concluding Remarks	203
	Appendices	206
A	Appendix A: Information Measures	207
A.1	Entropy	207
A.2	Conditional Entropy and Mutual Information	210
A.3	Divergence Measures	212
B	Appendix B: KL Divergence and Exponential Family	215
	Acknowledgements	217
	References	218

A Brief Introduction to Machine Learning for Engineers

Oswaldo Simeone¹

¹*Department of Informatics, King's College London;
osvaldo.simeone@kcl.ac.uk*

ABSTRACT

This monograph aims at providing an introduction to key concepts, algorithms, and theoretical results in machine learning. The treatment concentrates on probabilistic models for supervised and unsupervised learning problems. It introduces fundamental concepts and algorithms by building on first principles, while also exposing the reader to more advanced topics with extensive pointers to the literature, within a unified notation and mathematical framework. The material is organized according to clearly defined categories, such as discriminative and generative models, frequentist and Bayesian approaches, exact and approximate inference, as well as directed and undirected models. This monograph is meant as an entry point for researchers with an engineering background in probability and linear algebra.

Oswaldo Simeone (2018), “A Brief Introduction to Machine Learning for Engineers”, Foundations and Trends[®] in Signal Processing: Vol. 12, No. 3-4, pp 200–431. DOI: 10.1561/2000000102.

Notation

- Random variables or random vectors – both abbreviated as rvs – are represented using roman typeface, while their values and realizations are indicated by the corresponding standard font. For instance, the equality $x = x$ indicates that rv x takes value x .
- Matrices are indicated using uppercase fonts, with roman typeface used for random matrices.
- Vectors will be taken to be in column form.
- X^T and $X^\dagger$ are the transpose and the pseudoinverse of matrix X , respectively.
- The distribution of a rv x , either probability mass function (pmf) for a discrete rv or probability density function (pdf) for continuous rvs, is denoted as p_x , $p_x(x)$, or $p(x)$.
- The notation $x \sim p_x$ indicates that rv x is distributed according to p_x .
- For jointly distributed rvs $(x, y) \sim p_{xy}$, the conditional distribution of x given the observation $y = y$ is indicated as $p_{x|y=y}$, $p_{x|y}(x|y)$ or $p(x|y)$.
- The notation $x|y = y \sim p_{x|y=y}$ indicates that rv x is drawn according to the conditional distribution $p_{x|y=y}$.
- The notation $E_{x \sim p_x}[\cdot]$ indicates the expectation of the argument with respect to the distribution of the rv $x \sim p_x$. Accordingly, we will also write $E_{x \sim p_{x|y}}[\cdot|y]$ for the conditional expectation with respect to

the distribution $p_{x|y=y}$. When clear from the context, the distribution over which the expectation is computed may be omitted.

- The notation $\Pr_{x \sim p_x}[\cdot]$ indicates the probability of the argument event with respect to the distribution of the rv $x \sim p_x$. When clear from the context, the subscript is dropped.

- The notation $\log$ represents the logarithm in base two, while $\ln$ represents the natural logarithm.

- $x \sim \mathcal{N}(\mu, \Sigma)$ indicates that random vector x is distributed according to a multivariate Gaussian pdf with mean vector μ and covariance matrix Σ . The multivariate Gaussian pdf is denoted as $\mathcal{N}(x|\mu, \Sigma)$ as a function of x .

- $x \sim \mathcal{U}(a, b)$ indicates that rv x is distributed according to a uniform distribution in the interval $[a, b]$. The corresponding uniform pdf is denoted as $\mathcal{U}(x|a, b)$.

- $\delta(x)$ denotes the Dirac delta function or the Kronecker delta function, as clear from the context.

- $\|a\|^2 = \sum_{i=1}^N a_i^2$ is the quadratic, or l_2 , norm of a vector $a = [a_1, \dots, a_N]^T$. We similarly define the l_1 norm as $\|a\|_1 = \sum_{i=1}^N |a_i|$, and the l_0 pseudo-norm $\|a\|_0$ as the number of non-zero entries of vector a .

- I denotes the identity matrix, whose dimensions will be clear from the context. Similarly, $\mathbf{1}$ represents a vector of all ones.

- $\mathbb{R}$ is the set of real numbers; $\mathbb{R}^+$ the set of non-negative real numbers; $\mathbb{R}^-$ the set of non-positive real numbers; and $\mathbb{R}^N$ is the set of all vectors of N real numbers.

- $1(\cdot)$ is the indicator function: $1(x) = 1$ if x is true, and $1(x) = 0$ otherwise.

- $|\mathcal{S}|$ represents the cardinality of a set $\mathcal{S}$.

- $x_{\mathcal{S}}$ represents a set of rvs x_k indexed by the integers $k \in \mathcal{S}$.

Acronyms

AI: Artificial Intelligence
AMP: Approximate Message Passing
BN: Bayesian Network
DAG: Directed Acyclic Graph
ELBO: Evidence Lower BOund
EM: Expectation Maximization
ERM: Empirical Risk Minimization
GAN: Generative Adversarial Network
GLM: Generalized Linear Model
HMM: Hidden Markov Model
i.i.d.: independent identically distributed
KL: Kullback-Leibler
LASSO: Least Absolute Shrinkage and Selection Operator
LBP: Loopy Belief Propagation
LL: Log-Likelihood
LLR: Log-Likelihood Ratio
LS: Least Squares
MC: Monte Carlo
MCMC: Markov Chain Monte Carlo
MDL: Minimum Description Length
MFVI: Mean Field Variational Inference
ML: Maximum Likelihood

MRF: Markov Random Field
NLL: Negative Log-Likelihood
PAC: Probably Approximately Correct
pdf: probability density function
pmf: probability mass function
PCA: Principal Component Analysis
PPCA: Probabilistic Principal Component Analysis
QDA: Quadratic Discriminant Analysis
RBM: Restricted Boltzmann Machine
SGD: Stochastic Gradient Descent
SVM: Support Vector Machine
rv: random variable or random vector (depending on the context)
s.t.: subject to
VAE: Variational AutoEncoder
VC: Vapnik–Chervonenkis
VI: Variational Inference

Part I

Basics

1

Introduction

Having taught courses on machine learning, I am often asked by colleagues and students with a background in engineering to suggest “the best place to start” to get into this subject. I typically respond with a list of books – for a general, but slightly outdated introduction, read this book; for a detailed survey of methods based on probabilistic models, check this other reference; to learn about statistical learning, I found this text useful; and so on. This answer strikes me, and most likely also my interlocutors, as quite unsatisfactory. This is especially so since the size of many of these books may be discouraging for busy professionals and students working on other projects. This monograph is an attempt to offer a basic and compact reference that describes key ideas and principles in simple terms and within a unified treatment, encompassing also more recent developments and pointers to the literature for further study.

1.1 What is Machine Learning?

A useful way to introduce the machine learning methodology is by means of a comparison with the conventional engineering design flow. This

starts with a in-depth analysis of the problem domain, which culminates with the definition of a mathematical model. The mathematical model is meant to capture the key features of the problem under study, and is typically the result of the work of a number of experts. The mathematical model is finally leveraged to derive hand-crafted solutions to the problem that offer given optimality guarantees.

For instance, consider the problem of defining a chemical process to produce a given molecule. The conventional flow requires chemists to leverage their knowledge of models that predict the outcome of individual chemical reactions, in order to craft a sequence of suitable steps that synthesize the desired molecule. Another example is the design of speech translation or image/video compression algorithms. Both of these tasks involve the definition of models and algorithms by teams of experts, such as linguists, psychologists, and signal processing practitioners, not infrequently during the course of long standardization meetings.

The engineering design flow outlined above may be too costly and inefficient for problems in which faster or less expensive solutions are desirable. The machine learning alternative is to collect large data sets, e.g., of labelled speech, images or videos, and to use this information to train general-purpose learning machines to carry out the desired task. While the standard engineering flow relies on domain knowledge and on design optimized for the problem at hand, machine learning lets large amounts of data dictate algorithms and solutions. To this end, rather than requiring a precise model of the set-up under study, machine learning requires the specification of an objective, of a generic model to be trained, and of an optimization technique.

Returning to the first example above, a machine learning approach would proceed by training a general-purpose machine to predict the outcome of known chemical reactions based on a large data set, and by then using the trained algorithm to explore ways to produce more complex molecules. In a similar manner, large data sets of images or videos would be used to train a general-purpose algorithm with the aim of obtaining compressed representations from which the original input can be recovered with some distortion.

1.2 When to Use Machine Learning?

Based on the discussion above, machine learning can offer an efficient alternative to the conventional engineering flow when development cost and time are the main concerns, or when the problem appears to be too complex to afford the development of solutions with optimality guarantees. On the flip side, the approach has the key disadvantages of providing generally suboptimal performance, of producing black-box, and hence non-interpretable, solutions, and of applying only to a limited set of problems.

In order to identify tasks for which machine learning methods may be useful, reference [31] suggests the following criteria:

1. the task involves a function that maps well-defined inputs to well-defined outputs;
2. large data sets exist or can be created containing input-output pairs;
3. the task provides clear feedback with clearly definable goals and metrics;
4. the task does not involve long chains of logic or reasoning that depend on diverse background knowledge or common sense;
5. the task does not require detailed explanations for how the decision was made;
6. the task has a tolerance for error and no need for provably correct or optimal solutions;
7. the phenomenon or function being learned should not change rapidly over time; and
8. no specialized dexterity, physical skills, or mobility is required.

These criteria are useful guidelines for the decision of whether machine learning methods are suitable for a given task of interest. They also offer a convenient demarcation line between machine learning as is intended

today, with its focus on training and computational statistics tools, and more general notions of Artificial Intelligence (AI) based on knowledge and common sense [86] (see [126] for an overview on AI research).

1.2.1 Learning Tasks

We can distinguish among three different main types of machine learning problems, which are briefly introduced below. The discussion reflects the focus of this monograph on parametric probabilistic models, as further elaborated on in the next section.

1. Supervised learning: We have N labelled training examples $\mathcal{D}=\{(x_n, t_n)\}_{n=1}^N$, where x_n represents a covariate, or explanatory variable, while t_n is the corresponding target label, or response. For instance, variable x_n may represent the text of an email, while the label t_n may be a binary variable indicating whether the email is spam or not. The goal of supervised learning is to predict the value of the label t for an input x that is not in the training set. In other words, supervised learning aims at generalizing the observations in the data set $\mathcal{D}$ to new inputs. For example, an algorithm trained on a set of emails should be able to classify a new email not present in the data set $\mathcal{D}$.

We can generally distinguish between *classification* problems, in which the label t is discrete, as in the example above, and *regression* problems, in which variable t is continuous. An example of a regression task is the prediction of tomorrow's temperature t based on today's meteorological observations x .

An effective way to learn a predictor is to identify from the data set $\mathcal{D}$ a predictive distribution $p(t|x)$ from a set of parametrized distributions. The conditional distribution $p(t|x)$ defines a profile of beliefs over all possible of the label t given the input x . For instance, for temperature prediction, one could learn mean and variance of a Gaussian distribution $p(t|x)$ as a function of the input x . As a special case, the output of a supervised learning algorithm may be in the form of a deterministic predictive function $t = \hat{t}(x)$.

2. Unsupervised learning: Suppose now that we have an unlabelled set of training examples $\mathcal{D}=\{x_n\}_{n=1}^N$. Less well defined than

supervised learning, unsupervised learning generally refers to the task of learning properties of the mechanism that generates this data set. Specific tasks and applications include clustering, which is the problem of grouping similar examples x_n ; dimensionality reduction, feature extraction, and representation learning, all related to the problem of representing the data in a smaller or more convenient space; and generative modelling, which is the problem of learning a generating mechanism to produce artificial examples that are similar to available data in the data set $\mathcal{D}$.

As a generalization of both supervised and unsupervised learning, *semi-supervised learning* refers to scenarios in which not all examples are labelled, with the unlabelled examples providing information about the distribution of the covariates x .

3. Reinforcement learning: Reinforcement learning refers to the problem of inferring optimal sequential decisions based on rewards or punishments received as a result of previous actions. Under supervised learning, the “label” t refers to an action to be taken when the learner is in an informational state about the environment given by a variable x . Upon taking an action t in a state x , the learner is provided with feedback on the immediate reward accrued via this decision, and the environment moves on to a different state. As an example, an agent can be trained to navigate a given environment in the presence of obstacles by penalizing decisions that result in collisions.

Reinforcement learning is hence neither supervised, since the learner is not provided with the optimal actions t to select in a given state x ; nor is it fully unsupervised, given the availability of feedback on the quality of the chosen action. Reinforcement learning is also distinguished from supervised and unsupervised learning due to the influence of previous actions on future states and rewards.

This monograph focuses on supervised and unsupervised learning. These general tasks can be further classified along the following dimensions.

- *Passive vs. active learning:* A passive learner is given the training examples, while an active learner can affect the choice of training examples on the basis of prior observations.

- *Offline vs. online learning*: Offline learning operates over a batch of training samples, while online learning processes samples in a streaming fashion. Note that reinforcement learning operates inherently in an online manner, while supervised and unsupervised learning can be carried out by following either offline or online formulations.

This monograph considers only passive and offline learning.

1.3 Goals and Outline

This monograph aims at providing an introduction to key concepts, algorithms, and theoretical results in machine learning. The treatment concentrates on probabilistic models for supervised and unsupervised learning problems. It introduces fundamental concepts and algorithms by building on first principles, while also exposing the reader to more advanced topics with extensive pointers to the literature, within a unified notation and mathematical framework. Unlike other texts that are focused on one particular aspect of the field, an effort has been made here to provide a broad but concise overview in which the main ideas and techniques are systematically presented. Specifically, the material is organized according to clearly defined categories, such as discriminative and generative models, frequentist and Bayesian approaches, exact and approximate inference, as well as directed and undirected models. This monograph is meant as an entry point for researchers with a background in probability and linear algebra. A prior exposure to information theory is useful but not required.

Detailed discussions are provided on basic concepts and ideas, including overfitting and generalization, Maximum Likelihood and regularization, and Bayesian inference. The text also endeavors to provide intuitive explanations and pointers to advanced topics and research directions. Sections and subsections containing more advanced material that may be skipped at a first reading are marked with a star (*).

The reader will find here neither discussions on computing platform or programming frameworks, such as map-reduce, nor details on specific applications involving large data sets. These can be easily found in a vast and growing body of work. Furthermore, rather than providing

exhaustive details on the existing myriad solutions in each specific category, techniques have been selected that are useful to illustrate the most salient aspects. Historical notes have also been provided only for a few selected milestone events.

Finally, the monograph attempts to strike a balance between the algorithmic and theoretical viewpoints. In particular, all learning algorithms are introduced on the basis of theoretical arguments, often based on information-theoretic measures. Moreover, a chapter is devoted to statistical learning theory, demonstrating how to set the field of supervised learning on solid theoretical foundations. This chapter is more theoretically involved than the others, and proofs of some key results are included in order to illustrate the theoretical underpinnings of learning. This contrasts with other chapters, in which proofs of the few theoretical results are kept at a minimum in order to focus on the main ideas.

The rest of the monograph is organized into five parts. The first part covers introductory material. Specifically, Chapter 2 introduces the frequentist, Bayesian and Minimum Description Length (MDL) learning frameworks; the discriminative and generative categories of probabilistic models; as well as key concepts such as training loss, generalization, and overfitting – all in the context of a simple linear regression problem. Information-theoretic metrics are also briefly introduced, as well as the advanced topics of interpretation and causality. Chapter 3 then provides an introduction to the exponential family of probabilistic models, to Generalized Linear Models (GLMs), and to energy-based models, emphasizing main properties that will be invoked in later chapters.

The second part concerns supervised learning. Chapter 4 covers linear and non-linear classification methods via discriminative and generative models, including Support Vector Machines (SVMs), kernel methods, logistic regression, multi-layer neural networks and boosting. Chapter 5 is a brief introduction to the statistical learning framework of the Probably Approximately Correct (PAC) theory, covering the Vapnik–Chervonenkis (VC) dimension and the fundamental theorem of PAC learning.

The third part, consisting of a single chapter, introduced unsupervised learning. In particular, in Chapter 6, unsupervised learning models are described by distinguishing among directed models, for which Expectation Maximization (EM) is derived as the iterative maximization of the Evidence Lower BOund (ELBO); undirected models, for which Restricted Boltzmann Machines (RBMs) are discussed as a representative example; discriminative models trained using the InfoMax principle; and autoencoders. Generative Adversarial Networks (GANs) are also introduced.

The fourth part covers more advanced modelling and inference approaches. Chapter 7 provides an introduction to probabilistic graphical models, namely Bayesian Networks (BNs) and Markov Random Fields (MRFs), as means to encode more complex probabilistic dependencies than the models studied in previous chapters. Approximate inference and learning methods are introduced in Chapter 8 by focusing on Monte Carlo (MC) and Variational Inference (VI) techniques. The chapter briefly introduces in a unified way techniques such as variational EM, Variational AutoEncoders (VAE), and black-box inference. Some concluding remarks are provided in the last part, consisting of Chapter 9.

We conclude this chapter by emphasizing the importance of probability as a common language for the definition of learning algorithms [34]. The centrality of the probabilistic viewpoint was not always recognized, but has deep historical roots. This is demonstrated by the following two quotes, the first from the first AI textbook published by P. H. Winston in 1977, and the second from an unfinished manuscript by J. von Neumann (see [126, 63] for more information):

“Many ancient Greeks supported Socrates opinion that deep, inexplicable thoughts came from the gods. Today’s equivalent to those gods is the erratic, even probabilistic neuron. It is more likely that increased randomness of neural behavior is the problem of the epileptic and the drunk, not the advantage of the brilliant.”

from *Artificial Intelligence*, 1977;

“All of this will lead to theories of computation which are much less rigidly of an all-or-none nature than past and present formal logic. . . There are numerous indications to make us believe that this new system of formal logic will move closer to another discipline which has been little linked in the past with logic. This is thermodynamics primarily in the form it was received from Boltzmann.”

from *The Computer and the Brain*, 1958.

2

A Gentle Introduction through Linear Regression

In this chapter, we introduce the frequentist, Bayesian and MDL learning frameworks, as well as key concepts in supervised learning, such as discriminative and generative models, training loss, generalization, and overfitting. This is done by considering a simple linear regression problem as a recurring example. We start by introducing the problem of supervised learning and by presenting some background on inference. We then present the frequentist, Bayesian and MDL learning approaches in this order. The treatment of MDL is limited to an introductory discussion, as the rest of monograph concentrates on frequentist and Bayesian viewpoints. We conclude with an introduction to the important topic of information-theoretic metrics, and with a brief introduction to the advanced topics of causal inference and interpretation.

2.1 Supervised Learning

In the standard formulation of a supervised learning problem, we are given a training set $\mathcal{D}$ containing N training points (x_n, t_n) , $n = 1, \dots, N$. The observations x_n are considered to be free variables, and known as *covariates*, *domain points*, or *explanatory variables*; while

the target variables t_n are assumed to be dependent on x_n and are referred to as *dependent variables*, *labels*, or *responses*. An example is illustrated in Fig. 2.1. We use the notation $x_{\mathcal{D}} = (x_1, \dots, x_N)^T$ for the covariates and $t_{\mathcal{D}} = (t_1, \dots, t_N)^T$ for the labels in the training set $\mathcal{D}$. Based on this data, the goal of supervised learning is to identify an algorithm to predict the label t for a new, that is, as of yet unobserved, domain point x .

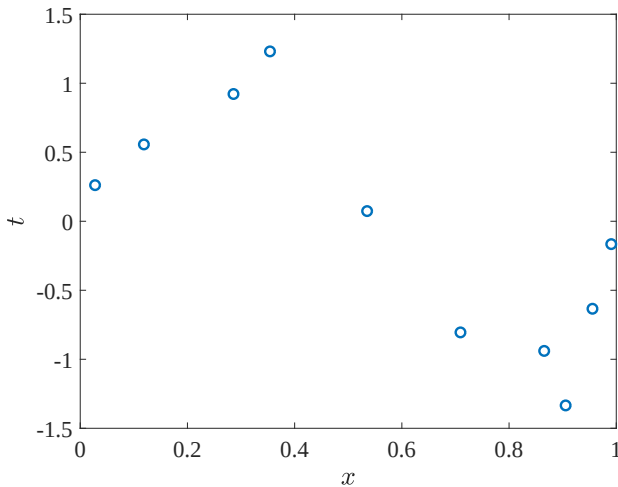


Figure 2.1: Example of a training set $\mathcal{D}$ with $N = 10$ points (x_n, t_n) , $n = 1, \dots, N$.

The outlined learning task is clearly impossible in the absence of additional information on the mechanism relating variables x and t . With reference to Fig. 2.1, unless we assume, say, that x and t are related by a function $t = f(x)$ with some properties, such as smoothness, we have no way of predicting the label t for an unobserved domain point x . This observation is formalized by the *no free lunch theorem* to be reviewed in Chapter 5: one cannot learn rules that *generalize* to unseen examples without making assumptions about the mechanism generating the data. The set of all assumptions made by the learning algorithm is known as the *inductive bias*.

This discussion points to a key difference between *memorizing* and learning. While the former amounts to mere retrieval of a value t_n

corresponding to an already observed pair $(x_n, t_n) \in \mathcal{D}$, learning entails the capability to predict the value t for an unseen domain point x . Learning, in other words, converts experience – in the form of $\mathcal{D}$ – into expertise or knowledge – in the form of a predictive algorithm. This is well captured by the following quote by Jorge Luis Borges: “*To think is to forget details, generalize, make abstractions.*” [138].

By and large, the goal of supervised learning is that of identifying a predictive algorithm that minimizes the *generalization loss*, that is, the error in the prediction of a new label t for an unobserved explanatory variable x . How exactly to formulate this problem, however, depends on one’s viewpoint on the nature of the model that is being learned. This leads to the distinction between the frequentist and the Bayesian approaches, which is central to this chapter. As it will be also discussed, the MDL philosophy deviates from the mentioned focus on prediction as the goal of learning, by targeting instead a parsimonious description of the data set $\mathcal{D}$.

2.2 Inference

Before we start our discussion of learning, it is useful to review some basic concepts concerning statistical inference, as they will be needed throughout this chapter and in the rest of this monograph. We specifically consider the inference problem of predicting a rv t given the observation of another rv x under the assumption that their joint distribution $p(x, t)$ is *known*. As a matter of terminology, it is noted that here we will use the term “inference” as it is typically intended in the literature on probabilistic graphical models (see, e.g., [80]), hence diverging from its use in other branches of the machine learning literature (see, e.g., [23]).

In order to define the problem of optimal inference, one starts by defining a non-negative *loss function* $\ell(t, \hat{t})$. This defines the cost, or loss or risk, incurred when the correct value is t while the estimate is $\hat{t}$. An important example is the ℓ_q loss

$$\ell_q(t, \hat{t}) = |t - \hat{t}|^q, \quad (2.1)$$

which includes as a special case the *quadratic loss* $\ell_2(t, \hat{t}) = (t - \hat{t})^2$, and the *0-1 loss*, or *detection error*, $\ell_0(t, \hat{t}) = |t - \hat{t}|_0$, where $|a|_0 = 1$ if $a \neq 0$ and $|a|_0 = 0$ otherwise. Once a loss function is selected, the optimal prediction $\hat{t}(x)$ for a given value of the observation x is obtained by minimizing the so-called *generalization risk or generalization loss*¹

$$L_p(\hat{t}) = E_{(x,t) \sim p_{xt}}[\ell(t, \hat{t}(x))]. \quad (2.2)$$

The notation L_p emphasizes the dependence of the generalization loss on the distribution $p(x, t)$.

The solution of this problem is given by the optimal prediction or decision rule²

$$\hat{t}^*(x) = \arg \min_{\hat{t}} E_{t \sim p_{t|x}}[\ell(t, \hat{t})|x]. \quad (2.3)$$

This can be seen by using the law of iterated expectations $E_{(x,t) \sim p_{xt}}[\cdot] = E_{x \sim p_x}[E_{t \sim p_{t|x}}[\cdot|x]]$. Equation (2.3) shows that the optimal estimate, or prediction, $\hat{t}^*(x)$ is a function of the *posterior* distribution $p(t|x)$ of the label given the domain point x and of the loss function ℓ . Therefore, once the posterior $p(t|x)$ is known, one can evaluate the optimal prediction (2.3) for any desired loss function, without the need to know the joint distribution $p(x, t)$.

As a special case of (2.3), with the quadratic loss function ℓ_2 , the optimal prediction is the conditional mean $\hat{t}^*(x) = E_{t \sim p_{t|x}}[t|x]$; while for the 0-1 loss function ℓ_0 , the optimal decision is the mode of the posterior distribution, i.e., $\hat{t}^*(x) = \arg \max_t p(t|x)$.

For example, assume that we have

$$t|x = x \sim 0.8\delta(t - x) + 0.2\delta(t + x), \quad (2.4)$$

so that, conditioned on the event $x = x$, t equals x with probability 0.8 and $-x$ with probability 0.2. The optimal prediction is $\hat{t}^*(x) = 0.8x - 0.2x = 0.6x$ for the quadratic loss, while it is $\hat{t}^*(x) = x$ for the 0-1 loss.

¹The term generalization error or population error are also often used, but they will not be adopted in this monograph.

²The optimal estimate (2.3) is also known as Bayes' prediction or Bayes' rule, but here we will not use this terminology in order to avoid confusion with the Bayesian approach discussed below.

The goal of supervised learning methods is broadly speaking that of obtaining a predictor $\hat{t}(x)$ that performs close to the optimal predictor $\hat{t}^*(x)$, based only on the training set $\mathcal{D}$, and hence without knowledge of the joint distribution $p(x, t)$. The closeness in performance is measured by the difference between the generalization loss $L_p(\hat{t})$ achieved by the trained predictor and the *minimum* generalization loss $L_p(\hat{t}^*)$ of the optimal predictor, which depends on the true distribution $p(x, t)$. Strictly speaking, this statement applies only for the frequentist approach, which is discussed next. As it will be explained later in the chapter, in fact, while the Bayesian approach still centers around the goal of prediction, its modelling assumptions are different. Furthermore, the MDL approach concentrates on the task of data compression rather than prediction.

2.3 Frequentist Approach

According to the frequentist viewpoint, the training data points $(x_n, t_n) \in \mathcal{D}$ are independent identically distributed (i.i.d.) rvs drawn from a *true*, and unknown, distribution $p(x, t)$:

$$(x_n, t_n) \underset{\text{i.i.d.}}{\sim} p(x, t), \quad i = 1, \dots, N. \quad (2.5)$$

The new observation (x, t) is also independently generated from the same true distribution $p(x, t)$; the domain point x is observed and the label t must be predicted. Since the probabilistic model $p(x, t)$ is not known, one cannot solve directly problem (2.3) to find the optimal prediction that minimizes the generalization loss L_p in (2.2).

Before discussing the available solutions to this problem, it is worth observing that the definition of the “true” distribution $p(x, t)$ depends in practice on the way data is collected. As in the example of the “beauty AI” context, if the rankings t_n assigned to pictures x_n of faces are affected by racial biases, the distribution $p(x, t)$ will reflect these prejudices and produce skewed results [87].

Taxonomy of solutions. There are two main ways to address the problem of learning how to perform inference when not knowing the distribution $p(x, t)$:

- *Separate learning and (plug-in) inference:* Learn first an approximation, say $p_{\mathcal{D}}(t|x)$, of the conditional distribution $p(t|x)$ based

on the data $\mathcal{D}$, and then plug this approximation in (2.3) to obtain an approximation of the optimal decision as

$$\hat{t}_{\mathcal{D}}(x) = \arg \min_{\hat{t}} \mathbb{E}_{t \sim p_{\mathcal{D}}(t|x)} [\ell(t, \hat{t}) | x]. \quad (2.6)$$

- *Direct inference via Empirical Risk Minimization (ERM)*: Learn directly an approximation $\hat{t}_{\mathcal{D}}(\cdot)$ of the optimal decision rule by minimizing an empirical estimate of the generalization loss (2.2) obtained from the data set as

$$\hat{t}_{\mathcal{D}}(\cdot) = \arg \min_{\hat{t}} L_{\mathcal{D}}(\hat{t}), \quad (2.7)$$

where the empirical risk, or *empirical loss*, is

$$L_{\mathcal{D}}(\hat{t}) = \frac{1}{N} \sum_{n=1}^N \ell(t_n, \hat{t}(x_n)). \quad (2.8)$$

The notation $L_{\mathcal{D}}(\hat{t})$ highlights the dependence of the empirical loss on the predictor $\hat{t}(\cdot)$ and on the training set $\mathcal{D}$.

In practice, as we will see, both approaches optimize a set of parameters that define the probabilistic model or the predictor. Furthermore, the first approach is generally more flexible, since having an estimate $p_{\mathcal{D}}(t|x)$ of the posterior distribution $p(t|x)$ allows the prediction (2.6) to be computed for any loss function. In contrast, the ERM solution (2.7) is tied to a specific choice of the loss function ℓ . In the rest of this section, we will start by taking the first approach, and discuss later how this relates to the ERM formulation.

Linear regression example. For concreteness, in the following, we will consider the following running example inspired by [23]. In the example, data is generated according to the true distribution $p(x, t) = p(x)p(t|x)$, where $x \sim \mathcal{U}(0, 1)$ and

$$t|x = x \sim \mathcal{N}(\sin(2\pi x), 0.1). \quad (2.9)$$

The training set in Fig. 2.1 was generated from this distribution. If this true distribution were known, the optimal predictor under the ℓ_2 loss would be equal to the conditional mean

$$\hat{t}^*(x) = \sin(2\pi x). \quad (2.10)$$

Hence, the minimum generalization loss is $L_p(\hat{t}^*) = 0.1$.

It is emphasized that, while we consider this running example in order to fix the ideas, all the definitions and ideas reported in this chapter apply more generally to supervised learning problems. This will be further discussed in Chapter 4 and Chapter 5.

2.3.1 Discriminative vs. Generative Probabilistic Models

In order to learn an approximation $p_{\mathcal{D}}(t|x)$ of the predictive distribution $p(t|x)$ based on the data $\mathcal{D}$, we will proceed by first selecting a family of parametric probabilistic models, also known as a *hypothesis class*, and by then learning the parameters of the model to fit (in a sense to be made precise later) the data $\mathcal{D}$.

Consider as an example the linear regression problem introduced above. We start by modelling the label t as a polynomial function of the domain point x added to a Gaussian noise with variance β^{-1} . Parameter β is the precision, i.e., the inverse variance of the additive noise. The polynomial function with degree M can be written as

$$\mu(x, w) = \sum_{j=0}^M w_j x^j = w^T \phi(x), \quad (2.11)$$

where we have defined the weight vector $w = [w_0 \ w_1 \ \cdots \ w_M]^T$ and the *feature* vector $\phi(x) = [1 \ x \ x^2 \ \cdots \ x^M]^T$. The vector w defines the relative weight of the powers in the sum (2.11). This assumption corresponds to adopting a parametric probabilistic model $p(t|x, \theta)$ defined as

$$t|x = x \sim \mathcal{N}(\mu(x, w), \beta^{-1}), \quad (2.12)$$

with parameters $\theta = (w, \beta)$. Having fixed this hypothesis class, the parameter vector θ can be then learned from the data $\mathcal{D}$, as it will be discussed.

In the example above, we have parametrized the posterior distribution. Alternatively, we can parametrize, and learn, the full joint distribution $p(x, t)$. These two alternatives are introduced below.

1. Discriminative probabilistic model. With this first class of models, the posterior, or predictive, distribution $p(t|x)$ is assumed to belong to a hypothesis class $p(t|x, \theta)$ defined by a parameter vector θ . The parameter vector θ is learned from the data set $\mathcal{D}$. For a given

parameter vector θ , the conditional distribution $p(t|x, \theta)$ allows the different values of the label t to be discriminated on the basis of their posterior probability. In particular, once the model is learned, one can directly compute the predictor (2.6) for any loss function.

As an example, for the linear regression problem, once a vector of parameters $\theta_{\mathcal{D}} = (w_{\mathcal{D}}, \beta_{\mathcal{D}})$ is identified based on the data $\mathcal{D}$ during learning, the optimal prediction under the ℓ_2 loss is the conditional mean $\hat{t}_{\mathcal{D}}(x) = \mathbb{E}_{t \sim p(t|x, \theta_{\mathcal{D}})}[t|x]$, that is, $\hat{t}_{\mathcal{D}}(x) = \mu(x, w_{\mathcal{D}})$.

2. Generative probabilistic model. Instead of learning directly the posterior $p(t|x)$, one can model the joint distribution $p(x, t)$ as being part of a parametric family $p(x, t|\theta)$. Note that, as opposed to discriminative models, the joint distribution $p(x, t|\theta)$ models also the distribution of the covariates x . Accordingly, the term “generative” reflects the capacity of this type of models to generate a realization of the covariates x by using the marginal $p(x|\theta)$.

Once the joint distribution $p(x, t|\theta)$ is learned from the data, one can compute the posterior $p(t|x, \theta)$ using Bayes’ theorem, and, from it, the optimal predictor (2.6) can be evaluated for any loss function. Generative models make stronger assumptions by modeling also the distribution of the explanatory variables. As a result, an improper selection of the model may lead to more significant bias issues. However, there are potential advantages, such as the ability to deal with missing data or latent variables, such as in semi-supervised learning. We refer to Chapter 6 for further discussion (see also [23]).

In the rest of this section, for concreteness, we consider discriminative probabilistic models $p(t|x, \theta)$, although the main definitions will apply also to generative models.

2.3.2 Model Order and Model Parameters

In the linear regression example, the selection of the hypothesis class (2.12) required the definition of the polynomial degree M , while the determination of a specific model $p(t|x, \theta)$ in the class called for the selection of the parameter vector $\theta = (w, \beta)$. As we will see, these two types of variables play a significantly different role during learning and should be clearly distinguished, as discussed next.

1. Model order M (and hyperparameters): The model order defines the “capacity” of the hypothesis class, that is, the number of the degrees of freedom in the model. The larger M is, the more capable a model is to fit the available data. For instance, in the linear regression example, the model order determines the size of the weight vector w . More generally, variables that define the class of models to be learned are known as hyperparameters. As we will see, determining the model order, and more broadly the hyperparameters, requires a process known as *validation*.

2. Model parameters θ : Assigning specific values to the model parameters θ identifies a hypothesis within the given hypothesis class. This can be done by using learning criteria such as Maximum Likelihood (ML) and Maximum a Posteriori (MAP).

We postpone a discussion of validation to the next section, and we start by introducing the ML and MAP learning criteria.

2.3.3 Maximum Likelihood (ML) Learning

Assume now that the model order M is fixed, and that we are interested in learning the model parameters θ . The ML criterion selects a value of θ under which the training set $\mathcal{D}$ has the maximum probability of being observed. In other words, the value θ selected by ML is the most likely to have generated the observed training set. Note that there might be more than one such value.

To proceed, we need to write the probability (density) of the observed labels $t_{\mathcal{D}}$ in the training set $\mathcal{D}$ given the corresponding domain points x . Under the assumed discriminative model, this is given as

$$\begin{aligned} p(t_{\mathcal{D}}|x_{\mathcal{D}}, w, \beta) &= \prod_{n=1}^N p(t_n|x_n, w, \beta), \\ &= \prod_{n=1}^N \mathcal{N}(t_n|\mu(x_n, w), \beta^{-1}) \end{aligned} \quad (2.13)$$

where we have used the independence of different data points. Taking the logarithm yields the *Log-Likelihood (LL) function*

$$\begin{aligned}\ln p(t_{\mathcal{D}}|x_{\mathcal{D}}, w, \beta) &= \sum_{n=1}^N \ln p(t_n|x_n, w, \beta) \\ &= -\frac{\beta}{2} \sum_{n=1}^N (\mu(x_n, w) - t_n)^2 + \frac{N}{2} \ln \frac{\beta}{2\pi}.\end{aligned}\quad (2.14)$$

The LL function should be considered as a function of the model parameters $\theta = (w, \beta)$, since the data set $\mathcal{D}$ is fixed and given. The ML learning problem is defined by the minimization of the *Negative LL (NLL)* function as

$$\min_{w, \beta} -\frac{1}{N} \sum_{n=1}^N \ln p(t_n|x_n, w, \beta). \quad (2.15)$$

This criterion is also referred to as *cross-entropy* or *log-loss*, as further discussed in Sec. 2.6.

If one is only interested in learning only the posterior mean, as is the case when the loss function is ℓ_2 , then one can tackle problem (2.14) only over the weights w , yielding the optimization

$$\min_w L_{\mathcal{D}}(w) = \frac{1}{N} \sum_{n=1}^N (\mu(x_n, w) - t_n)^2. \quad (2.16)$$

The quantity $L_{\mathcal{D}}(w)$ is known as the *training loss*. An interesting observation is that this criterion coincides with the ERM problem (2.7) for the ℓ_2 loss if one parametrizes the predictor as $\hat{t}(x) = \mu(x, w)$.

The ERM problem (2.16) can be solved in closed form. To this end, we write the empirical loss as $L_{\mathcal{D}}(w) = N^{-1} \|t_{\mathcal{D}} - X_{\mathcal{D}}w\|^2$, with the $N \times (M + 1)$ matrix

$$X_{\mathcal{D}} = [\phi(x_1) \ \phi(x_2) \ \cdots \ \phi(x_N)]^T. \quad (2.17)$$

Its minimization hence amounts to a Least Squares (LS) problem, which yields the solution

$$w_{ML} = (X_{\mathcal{D}}^T X_{\mathcal{D}})^{-1} X_{\mathcal{D}}^T t_{\mathcal{D}}, \quad (2.18)$$

Note that, in (2.18), we have assumed the typical overdetermined case in which the inequality $N > (M + 1)$ holds. More generally, one has the ML solution $w_{ML} = X_{\mathcal{D}}^{\dagger} t_{\mathcal{D}}$. Finally, differentiating the NLL with respect to β yields instead the ML estimate

$$\frac{1}{\beta_{ML}} = L_{\mathcal{D}}(w_{ML}). \quad (2.19)$$

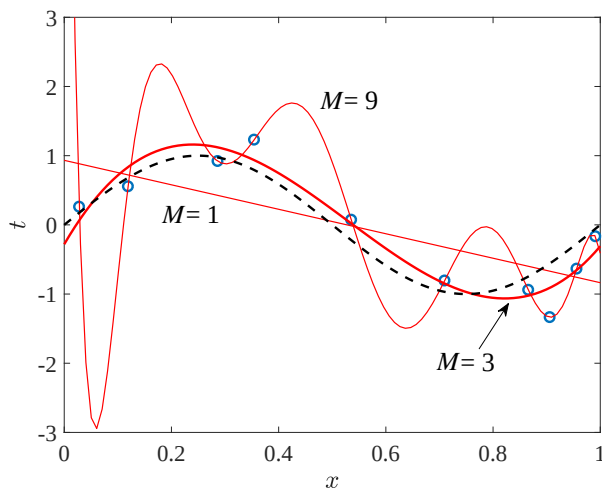


Figure 2.2: Illustration of underfitting and overfitting in ML learning: The dashed line is the optimal predictor (2.10), which depends on the unknown true distribution, while the other lines correspond to the predictor $\hat{t}_{ML}(x) = \mu(x, w_{ML})$ learned via ML with different model orders M .

Overfitting and Underfitting. Adopting the ℓ_2 loss, let us now compare the predictor $\hat{t}_{ML}(x) = \mu(x, w_{ML})$ learned via ML with the optimal, but unknown, predictor $\hat{t}^*(x)$ in (2.10). To this end, Fig. 2.2 shows the optimal predictor $\hat{t}^*(x)$ as a dashed line and the ML-based predictor $\hat{t}_{ML}(x)$ obtained with different values of the model order M for the training set $\mathcal{D}$ in Fig. 2.1 (also shown in Fig. 2.2 for reference).

We begin by observing that, with $M = 1$, the ML predictor *underfits* the data: the model is not rich enough to capture the variations present in the data. As a result, the training loss $L_{\mathcal{D}}(w_{ML})$ in (2.16) is large.

In contrast, with $M = 9$, the ML predictor *overfits* the data: the model is too rich and, in order to account for the observations in the training set, it yields inaccurate predictions outside it. In this case, the training loss $L_{\mathcal{D}}(w)$ in (2.16) is small, but the generalization loss

$$L_p(w_{ML}) = \mathbb{E}_{(x,t) \sim p_{xt}} [\ell(t, \mu(x, w_{ML}))] \quad (2.20)$$

is large. With overfitting, the model is memorizing the training set, rather than learning how to generalize to unseen examples.

The choice $M = 3$ appears to be the best by comparison with the optimal predictor. Note that this observation is in practice precluded given the impossibility to determine $\hat{t}^*(x)$ and hence the generalization loss. We will discuss below how to estimate the generalization loss using validation.

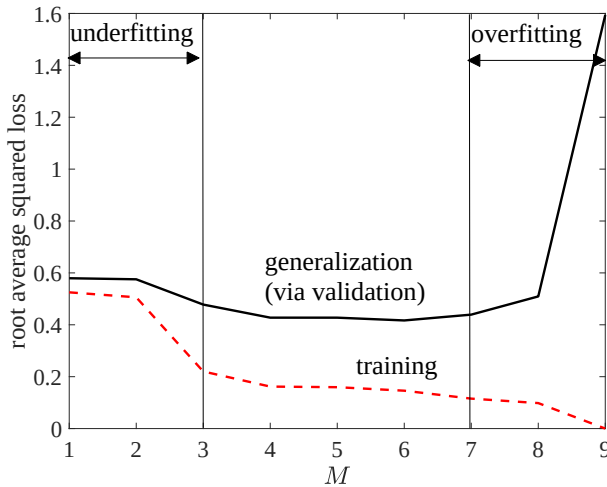


Figure 2.3: Square root of the generalization loss $L_p(w_{ML})$ and of the training loss $L_{\mathcal{D}}(w_{ML})$ as a function of the model order M for the training data set in Fig. 2.1.

The impact of the model order M on training and generalization losses is further elaborated on in Fig. 2.3, which shows the squared root of the generalization loss $L_p(w_{ML})$ and of the training loss $L_{\mathcal{D}}(w_{ML})$ as a function of M for the same training data set. A first remark is that, as expected, the training loss is smaller than the generalization loss,

since the latter accounts for all pairs $(x, t) \sim p(x, t)$, while the former includes only the training points used for learning. More importantly, the key observation here is that increasing M allows one to better fit – and possibly overfit – the training set, hence reducing $L_{\mathcal{D}}(w_{ML})$. The generalization loss $L_p(w_{ML})$ instead tends to decrease at first, as we move away from the underfitting regime, but it eventually increases for sufficiently large M . The widening of the gap between training and generalization provides evidence that overfitting is occurring. From Fig. 2.3, we can hence conclude that, in this example, model orders larger than $M = 7$ should be avoided since they lead to overfitting, while model order less than $M = 3$ should also not be considered in order to avoid underfitting.

What if we had more data? Extrapolating from the behavior observed in Fig. 2.2, we can surmise that, as the number N of data points increases, overfitting is avoided even for large values of M . In fact, when the training set is big as compared to the number of parameters in θ , we expect the training loss $L_{\mathcal{D}}(w)$ to provide an accurate measure of the generalization loss $L_p(w)$ for all possible values of w . Informally, we have the approximation $L_{\mathcal{D}}(w) \simeq L_p(w)$ simultaneously for all values of w as long as N is large enough. Therefore, the weight vector w_{ML} that minimizes the training loss $L_{\mathcal{D}}(w)$ also (approximately) minimizes the generalization loss $L_p(w)$. It follows that, for large N , the ML parameter vector w_{ML} tends to the optimal value w^* (assuming for simplicity of argument that it is unique) that minimizes the generalization loss among all predictors in the model, i.e.,

$$w^* = \underset{w}{\operatorname{argmin}} L_p(w). \quad (2.21)$$

This discussion will be made precise in Chapter 5.

To offer numerical evidence for the point just made, Fig. 2.4 plots the (square root of the) generalization and training losses versus N , where the training sets were generated at random from the true distribution. From the figure, we can make the following important observations. First, overfitting – as measured by the gap between training and generalization losses – vanishes as N increases. This is a consequence of the discussed approximate equalities $L_{\mathcal{D}}(w) \simeq L_p(w)$ and $w_{ML} \simeq w^*$,

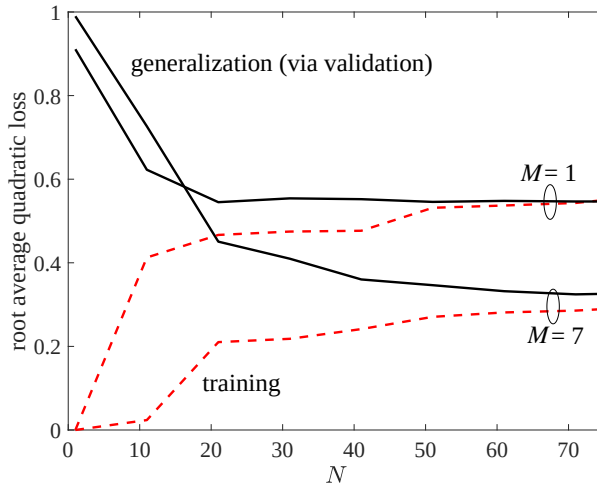


Figure 2.4: Square root of the generalization loss $L_p(w_{ML})$ and of the training loss $L_{\mathcal{D}}(w_{ML})$ as a function of the training set size N . The asymptote of the generalization and training losses is given by the minimum generalization loss $L_p(w^*)$ (cf. (2.21)) achievable for the given model order (see Fig. 2.5).

which are valid as N grows large, which imply the approximate equalities $L_{\mathcal{D}}(w_{ML}) \simeq L_p(w_{ML}) \simeq L_p(w^*)$.

Second, it is noted that the training loss $L_{\mathcal{D}}(w_{ML})$ tends to the minimum generalization loss $L_p(w^*)$ for the given M from below, while the generalization loss $L_p(w_{ML})$ tends to it from above. This is because, as N increases, it becomes more difficult to fit the data set $\mathcal{D}$, and hence $L_{\mathcal{D}}(w_{ML})$ increases. Conversely, as N grows large, the ML estimate becomes more accurate, because of the increasingly accurate approximation $w_{ML} \simeq w^*$, and thus the generalization loss $L_p(w_{ML})$ decreases.

Third, selecting a smaller model order M yields an improved generalization loss when the training set is small, while a larger value of M is desirable when the data set is bigger. In fact, as further discussed below, when N is small, the *estimation error* caused by overfitting dominates the *bias* caused by the choice of a small hypothesis class. In contrast, for sufficiently large training sets, the estimation error vanishes

and the performance is dominated by the bias induced by the selection of the model.

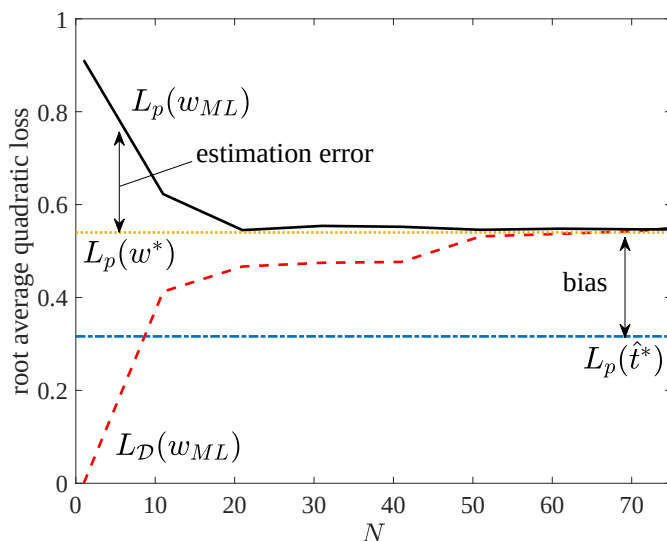


Figure 2.5: Illustration of the bias and training error based on the decomposition (2.22).

Bias and generalization gap. The previous paragraph introduced the notions of *estimation error* and *bias* associated with the selection of a given model order M . While Chapter 5 will provide a more extensive discussion on these concepts, it is useful to briefly review them here in the context of ML learning. Estimation error and bias refer to the following decomposition of the generalization loss achieved by the given solution w_{ML}

$$L_p(w_{ML}) = L_p(\hat{t}^*) + (L_p(w^*) - L_p(\hat{t}^*)) + (L_p(w_{ML}) - L_p(w^*)). \quad (2.22)$$

This decomposition is illustrated in Fig. 2.5 for $M = 1$. In (2.22), the term $L_p(\hat{t}^*) = 0.1$ (the figure shows the square root) is, as seen, the minimum achievable generalization loss without any constraint on the hypothesis class. The term $(L_p(w^*) - L_p(\hat{t}^*))$ represents the *bias*, or approximation error, caused by the choice of the given hypothesis class, and hence also by the choice of M . This is because, by (2.21), $L_p(w^*)$

is the best generalization loss for the given model. Recall that the loss $L_p(w^*)$ can be achieved when N is large enough. Finally, the term $(L_p(w_{ML}) - L_p(w^*))$ is the *estimation error* or *generalization gap*³ that is incurred due to the fact that N is not large enough and hence we have $w_{ML} \neq w^*$.

From the decomposition (2.22), a large N allows us to reduce the estimation error, but it has no effect on the bias. This is seen in Fig. 2.4, where the asymptote achieved by the generalization loss as N increases equals the minimum generalization loss $L_p(w^*)$ for the given model order. Choosing a small value of M in the regime of large data imposes a floor on the achievable generalization loss that no amount of additional data can overcome.

Validation and testing. In the discussion above, it was assumed that the generalization loss $L_p(w)$ can somehow be evaluated. Since this depends on the true unknown distribution $p(x, t)$, this evaluation is, strictly speaking, not possible. How then to estimate the generalization loss in order to enable model order selection using a plot as in Fig. 2.3? The standard solution is to use validation.

The most basic form of validation prescribes the division of the available data into two sets: a *hold-out, or validation, set* and the *training set*. The validation set is used to evaluate an approximation of the generalization loss $L_p(w)$ via the empirical average

$$L_p(w) \simeq \frac{1}{N_v} \sum_{n=1}^{N_v} \ell(t_n, \mu(x_n, w)), \quad (2.23)$$

where the sum is done over the N_v elements of the validation set.

The just described hold-out approach to validation has a clear drawback, as part of the available data needs to be set aside and not used for training. This means that the number of examples that can be used for training is smaller than the number of overall available data points. To partially obviate this problem, a more sophisticated, and commonly used, approach to validation is *k-fold cross-validation*. With this method, the available data points are partitioned, typically at random, into k equally sized subsets. The generalization loss is then

³This is also defined as generalization error in some references.

estimated by averaging k different estimates. Each estimate is obtained by retaining one of the k subsets for validation and the remaining $k - 1$ subsets for training. When $k = N$, this approach is also known as *leave-one-out* cross-validation.

Test set. Once a model order M and model parameters θ have been obtained via learning and validation, one typically needs to produce an estimate of the generalization loss obtained with this choice (M, θ) . The generalization loss estimated via validation cannot be used for this purpose. In fact, the validation loss tends to be smaller than the actual value of the generalization loss. After all, we have selected the model order so as to yield the smallest possible error on the validation set. The upshot is that the final estimate of the generalization loss should be done on a separate set of data points, referred to as the *test set*, that are set aside for this purpose and not used at any stage of learning. As an example, in competitions among different machine learning algorithms, the test set is kept by a judge to evaluate the submissions and is never shared with the contestants.

2.3.4 Maximum A Posteriori (MAP) Criterion

We have seen that the decision regarding the model order M in ML learning concerns the tension between bias, whose reduction calls for a larger M , and estimation error, whose decrease requires a smaller M . ML provides a single integer parameter, M , as a gauge to trade off bias and estimation error. As we will discuss here, the MAP approach and, more generally, regularization, enable a finer control of these two terms. The key idea is to leverage prior information available on the behavior of the parameters in the absence, or presence, of overfitting.

To elaborate, consider the following experiment. Evaluate the ML solution w_{ML} in (2.18) for different values of M and observe how it changes as we move towards the overfitting regime by increasing M (see also [23, Table 1.1]). For the experiment reported in Fig. 2.2, we obtain the following values: for $M = 1$, $w_{ML} = [0.93, -1.76]^T$; for $M = 3$, $w_{ML} = [-0.28, 13.32, -35.92, 22.56]^T$; and for $M = 9$, $w_{ML} = [13.86, -780.33, 12.99 \times 10^3, -99.27 \times 10^3, 416.49 \times 10^3, -1.03 \times 10^6, 1.56 \times 10^6, 1.40 \times 10^6, 0.69 \times 10^6, -1.44 \times 10^6]$. These results suggest that a

manifestation of overfitting is the large value of norm $\|w\|$ of the vector of weights. We can use this observation as prior information, that is as part of the inductive bias, in designing a learning algorithm.

To this end, we can impose a *prior distribution* on the vector of weights that gives lower probability to large values. A possible, but not the only, way to do this is to assume a Gaussian prior as

$$w \sim \mathcal{N}(0, \alpha^{-1}I), \quad (2.24)$$

so that all weights are a priori i.i.d. zero-mean Gaussian variables with variance α^{-1} . Increasing α forces the weights to be smaller as it reduces the probability associated with large weights. The precision variable α is an example of a hyperparameter. In a Bayesian framework, hyperparameters control the distribution of the model parameters. As anticipated, hyperparameters are determined via validation.

Rather than maximizing the LL, that is, probability density

$$p(t_{\mathcal{D}}|x_{\mathcal{D}}, w, \beta)$$

of the labels in the training set, as for ML, the MAP criterion prescribes the maximization of the joint probability distribution of weights and of labels given the prior $p(w) = \mathcal{N}(w|0, \alpha^{-1}I)$, that is,

$$p(t_{\mathcal{D}}, w|x_{\mathcal{D}}, \beta) = p(w) \prod_{n=1}^N p(t_n|x_n, w, \beta). \quad (2.25)$$

Note that a prior probability can also be assumed for the parameter β , but in this example we leave β as a deterministic parameter. The MAP learning criterion can hence be formulated as

$$\min_{w, \beta} - \sum_{n=1}^N \ln p(t_n|x_n, w, \beta) - \ln p(w). \quad (2.26)$$

The name “Maximum a Posteriori” is justified by the fact that problem (2.26) is equivalent to maximizing the *posterior distribution* of the parameters w given the available data, as we will further discuss in the next section. This yields the following problem for the weight vector

$$\min_w L_{\mathcal{D}}(w) + \frac{\lambda}{N} \|w\|^2, \quad (2.27)$$

where we have defined $\lambda = \alpha/\beta$ and we recall that the training loss is $L_{\mathcal{D}}(w) = N^{-1} \|t_{\mathcal{D}} - X_{\mathcal{D}}w\|^2$.

ML vs. MAP. Observing (2.27), it is important to note the following *general property* of the MAP solution: As the number N of data points grows large, the MAP estimate tends to the ML estimate, given that the contribution of the prior information term decreases as $1/N$. When N is large enough, any prior credence is hence superseded by the information obtained from the data.

Problem (2.27), which is often referred to as ridge regression, modifies the ML criterion by adding the quadratic (or Tikhonov) *regularization* function

$$R(w) = \|w\|^2 \quad (2.28)$$

multiplied by the term λ/N . The regularization function forces the norm of the solution to be small, particularly with larger values of the hyperparameter λ , or equivalently α . The solution of problem (2.27) can be found by using standard LS analysis, yielding

$$w_{MAP} = (\lambda I + X_{\mathcal{D}}^T X_{\mathcal{D}})^{-1} X_{\mathcal{D}}^T t_{\mathcal{D}}. \quad (2.29)$$

This expression confirms that, as N grows large, the term λI becomes negligible and the solution tends to the ML estimate (2.18) (see [85] for a formal treatment).

Fig. 2.6 shows the squared root of the generalization loss $L_p(w_{MAP})$ and of the training loss $L_{\mathcal{D}}(w_{MAP})$ as a function of λ (in logarithmic scale) for the training data set in Fig. 2.1 with $M = 9$. The generalization loss is estimated using validation. We observe that increasing λ , and hence the relevance of the regularization term, has a similar impact as decreasing the model order M . A larger λ reduces the effective capacity of the model. Stated in different words, increasing λ reduces overfitting but may entail a larger bias.

Other standard examples for the prior distribution include the Laplace pdf, which yields the l_1 norm regularization function $R(w) = \|w\|_1 = \sum_{j=0}^M |w_j|$. This term promotes the sparsity of the solution, which is useful in many signal recovery algorithms [14] and in non-parametric function estimation [146]. The corresponding optimization

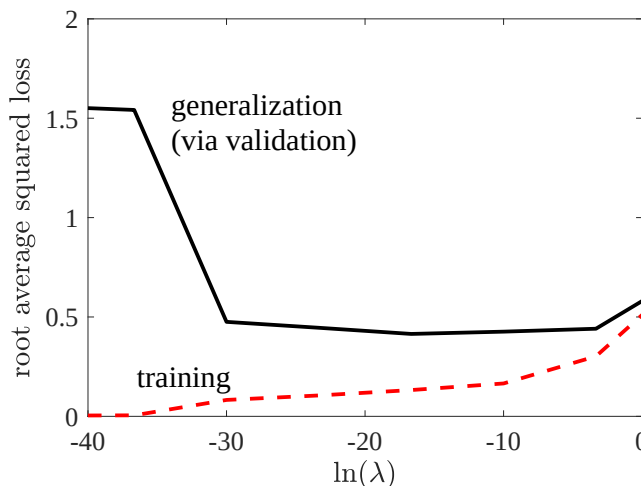


Figure 2.6: Square root of the generalization loss $L_p(w_{MAP})$ and of the training loss $L_D(w_{MAP})$ as a function of the regularization parameter λ for the training data set in Fig. 2.1 with $M = 9$.

problem

$$\min_w L_D(w) + \frac{\lambda}{N} \|w\|_1 \quad (2.30)$$

is known as LASSO (Least Absolute Shrinkage and Selection Operator).

2.3.5 Regularization

We have seen above that the MAP learning criterion amounts to the addition of a regularization function $R(w)$ to the ML or ERM learning losses. This function penalizes values of the weight vector w that are likely to occur in the presence of overfitting, or, generally, that are improbable on the basis of the available prior information. The net effect of this addition is to effectively decrease the capacity of the model, as the set of values for the parameter vector w that the learning algorithm is likely to choose from is reduced. As a result, as seen, regularization can control overfitting and its optimization requires validation.

Regularization generally refers to techniques that aim at reducing overfitting during training. The discussion in the previous subsection has focused on a specific form of regularization that is grounded in

a probabilistic interpretation in terms of MAP learning. We note that the same techniques, such as ridge regression and LASSO, can also be introduced independently of a probabilistic framework in an ERM formulation. Furthermore, apart from the discussed addition of regularization terms to the empirical risk, there are other ways to perform regularization.

One approach is to modify the optimization scheme by using techniques such as *early stopping* [56]. Another is to *augment* the training set by generating artificial examples and hence effectively increasing the number N of training examples. Related to this idea is the technique known as *bagging*. With bagging, we first create a number K of *bootstrap* data sets. Bootstrap data sets are obtained by selecting N data points uniformly *with replacement* from $\mathcal{D}$ (so that the same data point generally appears multiple times in the bootstrap data set). Then, we train the model K times, each time over a different bootstrap set. Finally, we average the results obtained from the models using equal weights. If the errors accrued by the different models were independent, bagging would yield an estimation error that decreases with K . In practice, significantly smaller gains are obtained, particularly for large K , given that the bootstrap data sets are all drawn from $\mathcal{D}$ and hence the estimation errors are not independent [23].

2.4 Bayesian Approach

The frequentist approaches discussed in the previous section assume the existence of a true distribution, and aim at identifying a specific value for the parameters θ of a probabilistic model to derive a predictor (cf. (2.3)). ML chooses the value θ that maximizes the probability of the training data, while MAP includes in this calculation also prior information about the parameter vector. With the frequentist approach, there are hence two distributions on the data: the true distribution, approximated by the empirical distribution of the data and the model (see further discussion in Sec. 2.8).

The Bayesian viewpoint is conceptually different: (i) It assumes that all data points are jointly distributed according to a distribution that is known except for some hyperparameters; and (ii) the model

parameters θ are jointly distributed with the data. As a result, as it will be discussed, rather than committing to a single value to explain the data, the Bayesian approach considers the explanations provided by all possible values of θ , each weighted according to a generally different, and data-dependent, “belief”.

More formally, the Bayesian viewpoint sees the vector of parameters as rvs that are jointly distributed with the labels $t_{\mathcal{D}}$ in the training data $\mathcal{D}$ and with the new example t . We hence have the joint distribution $p(t_{\mathcal{D}}, w, t | x_{\mathcal{D}}, x)$. We recall that the conditioning on the domain points $x_{\mathcal{D}}$ and x in the training set and in the new example, respectively, are hallmarks of discriminative probabilistic models. The Bayesian solution simply takes this modeling choice to its logical end point: in order to predict the new label t , it directly evaluates the posterior distribution $p(t | x_{\mathcal{D}}, t_{\mathcal{D}}, x) = p(t | \mathcal{D}, x)$ given the available information $(\mathcal{D}, x)$ by applying the marginalization rules of probability to the joint distribution $p(t_{\mathcal{D}}, w, t | x_{\mathcal{D}}, x)$.

As seen, the posterior probability $p(t | \mathcal{D}, x)$ can be used as the predictive distribution in (2.3) to evaluate a predictor $\hat{t}(x)$. However, a fully Bayesian solution returns the entire posterior $p(t | \mathcal{D}, x)$, which provides significantly more information about the unobserved label t . As we will discuss below, this knowledge, encoded in the posterior $p(t | \mathcal{D}, x)$, combines both the assumed prior information about the weight vector w and the information that is obtained from the data $\mathcal{D}$.

To elaborate, in the rest of this section, we assume that the precision parameter β is fixed and that the only learnable parameters are the weights in vector w . The joint distribution of the labels in the training set, of the weight vector and of the new label, conditioned on the domain points $x_{\mathcal{D}}$ in the training set and on the new point x , is given as

$$p(t_{\mathcal{D}}, w, t | x_{\mathcal{D}}, x) = \underbrace{p(w)}_{\text{a priori distribution}} \underbrace{p(t_{\mathcal{D}} | x_{\mathcal{D}}, w)}_{\text{likelihood}} \underbrace{p(t | x, w)}_{\text{distribution of new data}}. \quad (2.31)$$

In the previous equation, we have identified the a priori distribution of the data; the likelihood term $p(t_{\mathcal{D}} | x_{\mathcal{D}}, w) = \prod_{n=1}^N \mathcal{N}(t_n | \mu(x_n, w), \beta^{-1})$

in (2.13)⁴; and the pdf of the new label $p(t|w, x) = \mathcal{N}(t|\mu(x, w), \beta^{-1})$. It is often useful to drop the dependence on the domain points $x_{\mathcal{D}}$ and x to write only the joint distribution of the random variables in the model as

$$p(t_{\mathcal{D}}, w, t) = \underbrace{p(w)}_{\text{a priori distribution}} \underbrace{p(t_{\mathcal{D}}|w)}_{\text{likelihood}} \underbrace{p(t|w)}_{\text{distribution of new data}}. \quad (2.32)$$

This factorization can be represented graphically by the Bayesian Network (BN) in Fig. 2.7. The significance of the graph should be clear by inspection, and it will be discussed in detail in Chapter 7.

It is worth pointing out that, by treating all quantities in the model – except for the hyperparameter α – as rvs, the Bayesian viewpoint does away with the distinction between learning and inference. In fact, since the joint distribution is assumed to be known in a Bayesian model, the problem at hand becomes that of inferring the unknown rv t . To restate this important point, the Bayesian approach subsumes all problems in the general inference task of estimating a subset of rvs given other rvs in a set of jointly distributed rvs with a known joint distribution.

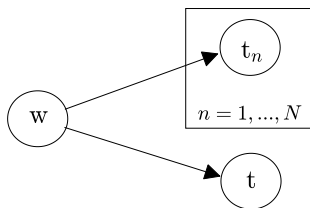


Figure 2.7: Bayesian Network (BN) describing the joint distribution (2.32) of the weight vector w , of the labels $t_{\mathcal{D}}$ in the training data $\mathcal{D}$ and t in the new example, as used in the Bayesian approach.

As mentioned, we are interested in computing the posterior probability $p(t|\mathcal{D}, x)$ of the new label t given the training data $\mathcal{D}$ and the new domain point $x = x$. Dropping again the domain variables to simplify the notation, we apply standard rules of probability to obtain

$$p(t|\mathcal{D}) = \frac{p(t_{\mathcal{D}}, t)}{p(t_{\mathcal{D}})} = \int \frac{p(w)p(t_{\mathcal{D}}|w)}{p(t_{\mathcal{D}})} p(t|w) dw$$

⁴The likelihood is also known as sampling distribution within the Bayesian framework [92].

$$= \int p(w|t_{\mathcal{D}})p(t|w)dw, \quad (2.33)$$

where the second equality follows from the marginalization rule $p(t_{\mathcal{D}}, t) = \int p(t_{\mathcal{D}}, w, t)dw$ and the last equality from Bayes' theorem. Putting back the dependence on the domain variables, we obtain the predictive distribution as

$$p(t|x, \mathcal{D}) = \int \underbrace{p(w|\mathcal{D})}_{\text{posterior distribution of } w} p(t|x, w)dw. \quad (2.34)$$

This is the key equation. Accordingly, the Bayesian approach considers the predictive probability $p(t|x, w)$ associated with each value of the weight vector w weighted by the posterior belief

$$p(w|\mathcal{D}) = \frac{p(w)p(t_{\mathcal{D}}|x_{\mathcal{D}}, w)}{p(t_{\mathcal{D}}|x_{\mathcal{D}})}. \quad (2.35)$$

The posterior belief $p(w|\mathcal{D})$, which defines the weight of the parameter vector w , is hence proportional to the prior belief $p(w)$ multiplied by the correction $p(t_{\mathcal{D}}|x_{\mathcal{D}}, w)$ due to the observed data.

Computing the posterior $p(w|\mathcal{D})$, and a fortiori also the predictive distribution $p(t|x, \mathcal{D})$, is generally a difficult task that requires the adoption of approximate inference techniques covered in Chapter 8. For this example, however, we can directly compute the predictive distribution as [23]

$$t|x, \mathcal{D} \sim \mathcal{N}(\mu(x, w_{MAP}), s^2(x)), \quad (2.36)$$

where $s^2(x) = \beta^{-1}(1 + \phi(x)^T (\lambda I + X_{\mathcal{D}}^T X_{\mathcal{D}})^{-1} \phi(x))$. Therefore, in this particular example, the optimal predictor under ℓ_2 loss is MAP. This is consequence of the fact that mode and mean of a Gaussian pdf coincide, and is not a general property. Even so, as discussed next, the Bayesian viewpoint can provide significantly more information on the label t than the ML or MAP.

ML and MAP vs. Bayesian approach. The Bayesian posterior (2.36) provides a finer prediction of the labels t given the explanatory variables x as compared to the predictive distribution $p(t|x, \theta_{ML}) = \mathcal{N}(\mu(x, w_{ML}), \beta^{-1})$ returned by ML and similarly by MAP. To see this,

note that the latter has the same variance for all values of x , namely β^{-1} . Instead, the Bayesian approach reveals that, due to the uneven distribution of the observed values of x , the accuracy of the prediction of labels depends on the value of x : Values of x closer to the existing points in the training sets generally exhibit a smaller variance.

This is shown in Fig. 2.8, which plots a training set, along with the corresponding predictor $\mu(x, w_{MAP})$ and the high-probability interval $\mu(x, w_{MAP}) \pm s(x)$ produced by the Bayesian method. We set $M = 9$, $\beta^{-1} = 0.1$ and $\alpha^{-1} = 0.2 \times 10^5$. For reference, we also show the interval $\mu(x, w_{MAP}) \pm \beta^{-1/2}$ that would result from the MAP analysis. This intervals illustrate the capability of the Bayesian approach to provide information about the uncertainty associated with the estimate of t .

This advantage of the Bayesian approach reflects its conceptual difference with respect to the frequentist approach: The frequentist predictive distribution refers to a hypothetical new observation generated with the same mechanism as the training data; instead, the Bayesian predictive distribution quantifies the statistician's belief in the value of t given the assumed prior and the training data.

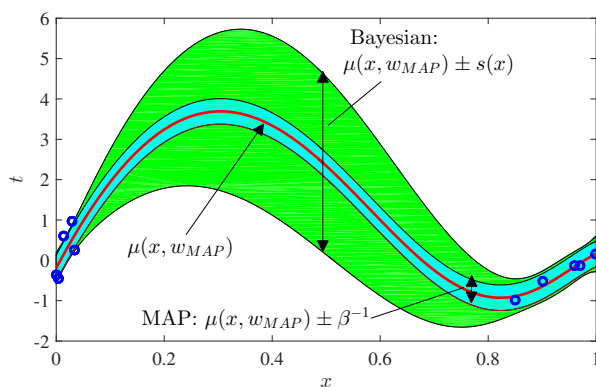


Figure 2.8: Illustration of the predictive distribution $p(t|x, \mathcal{D})$ produced by the Bayesian method for the training set shown in the figure as compared to that obtained with the MAP criterion. The larger interval corresponds to $\mu(x, w_{MAP}) \pm s(x)$ for the Bayesian method, while the smaller interval to $\mu(x, w_{MAP}) \pm \beta^{-1/2}$ for MAP ($M = 9$, $\beta^{-1} = 0.1$ and $\alpha^{-1} = 0.2 \times 10^5$).

From (2.36), we can make another important general observation about the relationship with ML and MAP concerning the asymptotic behavior when N is large. In particular, when $N \rightarrow \infty$, we have already seen that, informally, the limit $w_{MAP} \rightarrow w_{ML}$ holds. We now observe that it is also the case that the variance $s^2(x)$ of the Bayesian predictive distribution tends to β^{-1} . As a result, we can conclude that the Bayesian predictive distribution approaches that returned by ML when N is large. A way to think about this conclusion is that, when N is large, the posterior distribution $p(w|\mathcal{D})$ of the weights tends to concentrate around the ML estimate, hence limiting the average (2.34) to the contribution of the ML solution.

Marginal likelihood. Another advantage of the Bayesian approach is that, in principle, it allows model selection to be performed without validation. Toward this end, compute the *marginal likelihood*

$$p(t_{\mathcal{D}}|x_{\mathcal{D}}) = \int p(w) \prod_{n=1}^N p(t_n|x_n, w) dw, \quad (2.37)$$

that is, the probability density of the training set when marginalizing over the weight distribution. With the ML approach, the corresponding quantity $p(t_{\mathcal{D}}|x_{\mathcal{D}}, w_{ML})$ can only increase by choosing a larger model order M . In fact, a larger M entails more degrees of freedom in the optimization (2.16) of the LL. A similar discussion applies also to MAP. However, this is not the case for (2.37): a larger M implies a more “spread-out” prior distribution of the weights, which may result in a more diffuse distribution of the labels in (2.37). Hence, increasing M may yield a smaller marginal likelihood.

To illustrate this point, Fig. 2.9 plots the marginal likelihood for the data set in Fig. 2.1 for $\beta = 10$ and three different values of α as a function of M . The marginal likelihood in this example can be easily computed since we have

$$t_{\mathcal{D}}|x_{\mathcal{D}} = x_{\mathcal{D}} \sim \mathcal{N}(0, \alpha^{-1} X_{\mathcal{D}} X_{\mathcal{D}}^T + \beta^{-1} I). \quad (2.38)$$

It is observed that the marginal likelihood presents a peak at a given value of M , while decreasing when moving away from the optimal value. Therefore, we could take the value of M at which the marginal likelihood is maximized as the selected model order.