Machine Learning: Foundations, Methodologies, and Applications

Books published in this series focus on the theory and computational foundations, advanced methodologies and practical applications of machine learning, ideally combining mathematically rigorous treatments of a contemporary topics in machine learning with specific illustrations in relevant algorithm designs and demonstrations in real-world applications. The intended readership includes research students and researchers in computer science, computer engineering, electrical engineering, data science, and related areas seeking a convenient medium to track the progresses made in the foundations, methodologies, and applications of machine learning.

Topics considered include all areas of machine learning, including but not limited to:

- Decision tree
- Artificial neural networks
- Kernel learning
- Bayesian learning
- Ensemble methods
- Dimension reduction and metric learning
- Reinforcement learning
- Meta learning and learning to learn
- Imitation learning
- Computational learning theory
- Probabilistic graphical models
- Transfer learning
- Multi-view and multi-task learning
- Graph neural networks
- Generative adversarial networks
- Federated learning

This series includes monographs, introductory and advanced textbooks, and state-of-the-art collections. Furthermore, it supports Open Access publication mode.

More information about this series at https://link.springer.com/bookseries/16715

Alexander Jung

Machine Learning

The Basics

Springer

Alexander Jung
Department of Computer Science
Aalto University
Espoo, Finland

ISSN 2730-9908 ISSN 2730-9916 (electronic)
Machine Learning: Foundations, Methodologies, and Applications
ISBN 978-981-16-8192-9 ISBN 978-981-16-8193-6 (eBook)
https://doi.org/10.1007/978-981-16-8193-6

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

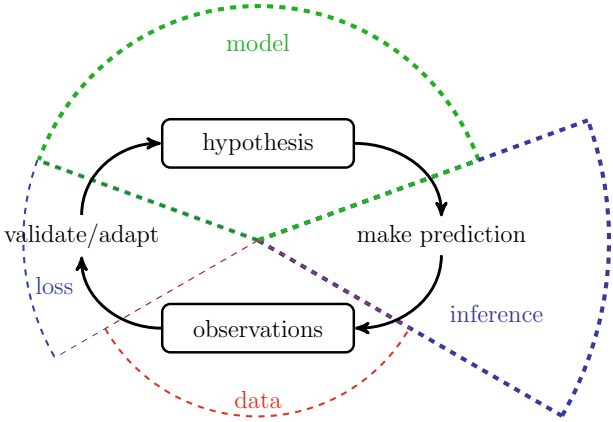

Fig. 1 Machine learning combines three main components: data, model and loss. Machine learning methods implement the scientific principle of "trial and error". These methods continuously validate and refine a model based on the loss incurred by its predictions about a phenomenon that generates data.

Preface

Machine learning (ML) influences our daily lives in several aspects. We routinely ask ML empowered smartphones to suggest lovely restaurants or to guide us through a strange place. ML methods have also become standard tools in many fields of science and engineering. ML applications transform human lives at unprecedented pace and scale.

This book portrays ML as the combination of three basic components: data, model and loss. ML methods combine these three components within computationally efficient implementations of the basic scientific principle "trial and error". This principle consists of the continuous adaptation of a hypothesis about a phenomenon that generates data.

ML methods use a hypothesis map to compute predictions of a quantity of interest (or higher level fact) that is referred to as the label of a data point. A hypothesis map reads in low level properties (referred to as features) of a data point and delivers the prediction for the label of that data point. ML methods choose or learn a hypothesis map from a (typically very) large set of candidate maps. We refer to this set as of candidate maps as the hypospace or model underlying an ML method.

The adaptation or improvement of the hypothesis is based on the discrepancy between predictions and observed data. ML methods use a loss function to quantify this discrepancy.

A plethora of different ML methods is obtained by combining different design choices for the data representation, model and loss. ML methods also differ vastly in their practical implementations which might obscure their unifying basic principles.

Deep learning methods use cloud computing frameworks to train large models on large datasets. Operating on a much finer granularity for data and computation, linear (least squares) regression can be implemented on small embedded systems. Nevertheless, deep learning methods and linear regression use the same principle of iteratively updating a model based on the discrepancy between model predictions and actual observed data.

We believe that thinking about ML as combinations of three components given by data, model and lossfunc helps to navigate the steadily growing offer for ready-to-use

ML methods. Our three-component picture allows a unified treatment of ML techniques, such as early stopping, privacy-preserving ML and xml, that seem quite unrelated at first sight. For example, the regularization effect of the early stopping technique in gradient-based methods is due to the shrinking of the effective hypospace. Privacy-preserving ML methods can be obtained by particular choices for the features used to characterize data points (see Sect. 9.5). Explainable ML methods can be obtained by particular choices for the hypospace and lossfunc (see Chap. 10).

To make good use of ML tools it is instrumental to understand its underlying principles at the appropriate level of detail. It is typically not necessary to understand the mathematical details of advanced optimization methods to successfully apply deep learning methods. On a lower level, this tutorial helps ML engineers choose suitable methods for the application at hand. The book also offers a higher level view on the implementation of ML methods which is typically required to manage a team of ML engineers and data scientists.

Espoo, Finland Alexander Jung

Acknowledgements

This book grew from lecture notes prepared for the courses CS-E3210 “Machine Learning: Basic Principles”, CS-E4800 “Artificial Intelligence”, CS-EJ3211 “Machine Learning with Python”, CS-EJ3311 “Deep Learning with Python” and CS-C3240 “Machine Learning” offered at Aalto University and within the Finnish university network fitech.io. This tutorial is accompanied by practical implementations of ML methods in MATLAB and Python https://github.com/alexjungaalto/.

This text benefited from the numerous feedback of the students within the courses that have been (co-)taught by the author. The author is indebted to Shamsiiat Abdurakhmanova, Tomi Janhunen, Yu Tian, Natalia Vesselinova, Linli Zhang, Ekaterina Voskoboinik, Buse Atli, Stefan Mojsilovic for carefully reviewing early drafts of this tutorial. Some of the figures have been generated with the help of Linli Zhang. The author is grateful for the feedback received from Jukka Suomela, Väinö Mehtola, Oleg Vlasovetc, Anni Niskanen, Georgios Karakasidis, Joni Pääkkö, Harri Wallenius and Satu Korhonen.

Contents

1 Introduction 1
1.1 Relation to Other Fields 4
1.1.1 Linear Algebra 5
1.1.2 Optimization 6
1.1.3 Theoretical Computer Science 6
1.1.4 Information Theory 7
1.1.5 Probability Theory and Statistics 9
1.1.6 Artificial Intelligence 10
1.2 Flavours of Machine Learning 12
1.2.1 Supervised Learning 12
1.2.2 Unsupervised Learning 13
1.2.3 Reinforcement Learning 14
1.3 Organization of this Book 15
References 17

2 Components of ML 19
2.1 The Data 19
2.1.1 Features 21
2.1.2 Labels 26
2.1.3 Scatterplot 28
2.1.4 Probabilistic Models for Data 28
2.2 The Model 30
2.2.1 Parametrized Hypothesis spaces 32
2.2.2 The Size of a Hypothesis Space 35
2.3 The Loss 37
2.3.1 Loss Functions for Numeric Labels 39
2.3.2 Loss Functions for Categorical Labels 40
2.3.3 Loss Functions for Ordinal Label Values 43
2.3.4 Empirical Risk 44
2.3.5 Regret 47
2.3.6 Rewards as Partial Feedback 48

2.4 Putting Together the Pieces . . . 48
2.5 Exercises . . . 50
References . . . 55

3 The Landscape of ML . . . 57
3.1 Linear Regression . . . 57
3.2 Polynomial Regression . . . 59
3.3 Least Absolute Deviation Regression . . . 60
3.4 The Lasso . . . 61
3.5 Gaussian Basis Regression . . . 62
3.6 Logistic Regression . . . 64
3.7 Support Vector Machines . . . 66
3.8 Bayes Classifier . . . 68
3.9 Kernel Methods . . . 69
3.10 Decision Trees . . . 70
3.11 Deep Learning . . . 72
3.12 Maximum Likelihood . . . 74
3.13 Nearest Neighbour Methods . . . 75
3.14 Deep Reinforcement Learning . . . 76
3.15 LinUCB . . . 77
3.16 Exercises . . . 78
References . . . 79

4 Empirical Risk Minimization . . . 81
4.1 The Basic Idea of Empirical Risk Minimization . . . 83
4.2 Computational and Statistical Aspects of ERM . . . 84
4.3 ERM for Linear Regression . . . 86
4.4 ERM for Decision Trees . . . 89
4.5 ERM for Bayes Classifiers . . . 91
4.6 Training and Inference Periods . . . 94
4.7 Online Learning . . . 94
4.8 Exercise . . . 96
References . . . 97

5 Gradient-Based Learning . . . 99
5.1 The GD Step . . . 100
5.2 Choosing Step Size . . . 102
5.3 When to Stop? . . . 103
5.4 GD for Linear Regression . . . 103
5.5 GD for Logistic Regression . . . 106
5.6 Data Normalization . . . 107
5.7 Stochastic GD . . . 108
5.8 Exercises . . . 111
References . . . 112

6 Model Validation and Selection . . . 113
6.1 Overfitting . . . 115
6.2 Validation . . . 117
6.2.1 The Size of the Validation Set . . . 119
6.2.2 k-Fold Cross Validation . . . 120
6.2.3 Imbalanced Data . . . 120
6.3 Model Selection . . . 122
6.4 A Probabilistic Analysis of Generalization . . . 126
6.5 The Bootstrap . . . 130
6.6 Diagnosing ML . . . 131
6.7 Exercises . . . 133
References . . . 134

7 Regularization . . . 135
7.1 Structural Risk Minimization . . . 137
7.2 Robustness . . . 140
7.3 Data Augmentation . . . 141
7.4 Statistical and Computational Aspects of Regularization . . . 144
7.5 Semi-Supervised Learning . . . 147
7.6 Multitask Learning . . . 148
7.7 Transfer Learning . . . 149
7.8 Exercises . . . 150
References . . . 151

8 Clustering . . . 153
8.1 Hard Clustering with k-Means . . . 155
8.2 Soft Clustering with Gaussian Mixture Models . . . 162
8.3 Connectivity-Based Clustering . . . 167
8.4 Clustering as Preprocessing . . . 169
8.5 Exercises . . . 170
References . . . 170

9 Feature Learning . . . 173
9.1 Basic Principle of Dimensionality Reduction . . . 174
9.2 Principal Component Analysis . . . 176
9.2.1 Combining PCA with Linear Regression . . . 178
9.2.2 How to Choose Number of PC? . . . 179
9.2.3 Data Visualisation . . . 179
9.2.4 Extensions of PCA . . . 179
9.3 Feature Learning for Non-numeric Data . . . 181
9.4 Feature Learning for Labeled Data . . . 183
9.5 Privacy-Preserving Feature Learning . . . 185
9.6 Random Projections . . . 186
9.7 Dimensionality Increase . . . 187
9.8 Exercises . . . 187
References . . . 188

10 Transparent and Explainable ML 189
10.1 A Model Agnostic Method 191
10.1.1 Probabilistic Data Model for XML 193
10.1.2 Computing Optimal Explanations 194
10.2 Explainable Empirical Risk Minimization 196
10.3 Exercises 199
References 199

Glossary 201

Index 211

Symbols

Sets

$a := b$	This statement defines a to be shorthand for b.
$\mathbb{N}$	The set of natural numbers 1, 2, ...
$\mathbb{R}$	The set of real numbers x [2].
$\mathbb{R}_+$	The set of non-negative real numbers $x \geq 0$.
$\{0, 1\}$	The set consisting of two real number 0 and 1.
$[0, 1]$	The closed interval of real numbers x with $0 \leq x \leq 1$.

Matrices and Vectors

$\mathbf{I}$	The identity matrix having diagonal entries equal to one and every off diagonal entry equal to zero.
$\mathbb{R}^n$	The set of vectors that consist of n real-valued entries.
$\mathbf{x} = (x_1, \ldots, x_n)^T$	A vector of length n. The jth entry of the vector is denoted as x_j.
$\|\mathbf{x}\|_2$	The Euclidean (or "ℓ_2") norm of the vector $\mathbf{x} = (x_1, \ldots, x_n)^T$ given as $\|\mathbf{x}\|_2 := \sqrt{\sum_{j=1}^n x_j^2}$.
$\|\mathbf{x}\|$	Some norm of the vector $\mathbf{x}$ [1]. Unless specified otherwise, we mean the Euclidean norm $\|\mathbf{x}\|_2$.
$\mathbf{x}^T$	The transpose of a vector $\mathbf{x}$ that is considered as a single column matrix. The transpose can be interpreted as a single-row matrix $(x_1, \ldots, x_n)$.
$\mathbf{A}^T$	The transpose of a matrix $\mathbf{A}$. A square matrix is called symmetric if $\mathbf{A} = \mathbf{A}^T$
$\mathbb{S}_+^n$	The set of all (psd) $n \times n$ matrices.

Machine Learning

i	A generic index $i = 1, 2, \ldots,$ used to enumerate the data points within a dataset.
m	The number of data points in (the size of) a dataset.
n	The number of individual features used to characterize a data point.
x_j	The jth individual feature of a data point.
$\mathbf{x}$	The feature vector $\mathbf{x} = (x_1, \ldots, x_n)^T$ of a data point whose entries are the individual features of the data point.
$\mathbf{z}$	Beside the symbol x, we sometimes use as another symbol to denote a vector whose entries are features of a data point. We need two different symbols to denote feature vectors for the discussion feature learning methods in Chap. 9.
$\mathrm{x}^{(i)}$	The feature vector of the ith data point within a dataset.
$x_j^{(i)}$	The jth feature of the ith data point within a dataset.
y	The label (quantity of interest) of a data point.
$y^{(i)}$	The label of the ith data point.
$(\mathbf{x}^{(i)}, y^{(i)})$	The features and the label of the ith data point within a dataset.
$h(\cdot)$	A hypothesis map that reads in the features $\mathbf{x}$ of a data point and outputs the predicted label $\hat{y} = h(\mathbf{x})$.
x_j	The j-th feature of a data point. The first feature of a given data point is denoted as x_1, the second feature x_2 and so on.
$L((\mathbf{x}, y), h)$	The loss incurred by predicting the label y of a data point with feature vector $\mathbf{x}$ using the value $\hat{y} = h(\mathbf{x})$ obtained from evaluating the hypothesis $h \in \mathcal{H}$ at the feature vector $\mathbf{x}$.
E_v	The validation error of a hypothesis, which is the average loss computed on a validation set.
$\widehat{L}(h\|\mathcal{D})$	The emprisk or average loss incurred by the predictions of hypothesis h for the data points in the dataset $\mathcal{D}$.
E_t	The trainer of a hypothesis h, which is the average loss incurred by h on labeled data points that form a training set.
t	A discrete-time index $t = 0, 1, \ldots$ used to enumerate a sequence to temporal events (time instants).
t	A generic index used to enumerate a finite set of learning tasks within a multi-task learning problem (see Sect. 7.6).
λ	A regularization parameter that is used to scale the regularization term that is added to the empirical risk in structural risk minimization (SRM).
$\lambda_j(\mathbf{Q})$	The jth eigenvalue (sorted either ascending or descending) of a psd matrix $\mathbf{Q}$. We also use the shorthand λ_j if the corresponding matrix is clear from context.
$f(\cdot)$	The activation function used by an artificial neuron within an artificial neural network (ANN).

References

1. G.H. Golub, C. F. Van Loan. Matrix Computations. (Johns Hopkins University Press, Baltimore, MD, 3rd edition, 1996)
2. W. Rudin. Real and Complex Analysis. (McGraw-Hill, New York, 3rd edition, 1987)