

An Introduction to Machine Learning

Miroslav Kubat

An Introduction to Machine Learning

Second Edition

 Springer

Miroslav Kubat
Department of Electrical and Computer Engineering
University of Miami
Coral Gables, FL, USA

ISBN 978-3-319-63912-3 ISBN 978-3-319-63913-0 (eBook)
DOI 10.1007/978-3-319-63913-0

Library of Congress Control Number: 2017949183

© Springer International Publishing AG 2015, 2017

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To my wife, Verunka.

Contents

1	A Simple Machine-Learning Task	1
1.1	Training Sets and Classifiers	1
1.2	Minor Digression: Hill-Climbing Search	5
1.3	Hill Climbing in Machine Learning	8
1.4	The Induced Classifier’s Performance.....	11
1.5	Some Difficulties with Available Data	13
1.6	Summary and Historical Remarks	15
1.7	Solidify Your Knowledge	16
2	Probabilities: Bayesian Classifiers	19
2.1	The Single-Attribute Case	19
2.2	Vectors of Discrete Attributes	22
2.3	Probabilities of Rare Events: Exploiting the Expert’s Intuition	26
2.4	How to Handle Continuous Attributes	30
2.5	Gaussian “Bell” Function: A Standard <i>pdf</i>	33
2.6	Approximating PDFs with Sets of Gaussians	34
2.7	Summary and Historical Remarks	36
2.8	Solidify Your Knowledge	40
3	Similarities: Nearest-Neighbor Classifiers	43
3.1	The <i>k</i> -Nearest-Neighbor Rule	43
3.2	Measuring Similarity	46
3.3	Irrelevant Attributes and Scaling Problems	49
3.4	Performance Considerations	52
3.5	Weighted Nearest Neighbors	55
3.6	Removing Dangerous Examples.....	57
3.7	Removing Redundant Examples.....	59
3.8	Summary and Historical Remarks	61
3.9	Solidify Your Knowledge	62

4	Inter-Class Boundaries: Linear and Polynomial Classifiers	65
4.1	The Essence	65
4.2	The Additive Rule: Perceptron Learning	69
4.3	The Multiplicative Rule: WINNOW	73
4.4	Domains with More Than Two Classes	76
4.5	Polynomial Classifiers	79
4.6	Specific Aspects of Polynomial Classifiers	81
4.7	Numerical Domains and Support Vector Machines	84
4.8	Summary and Historical Remarks	86
4.9	Solidify Your Knowledge	87
5	Artificial Neural Networks	91
5.1	Multilayer Perceptrons as Classifiers	91
5.2	Neural Network's Error	95
5.3	Backpropagation of Error	97
5.4	Special Aspects of Multilayer Perceptrons	100
5.5	Architectural Issues	104
5.6	Radial-Basis Function Networks	106
5.7	Summary and Historical Remarks	109
5.8	Solidify Your Knowledge	110
6	Decision Trees	113
6.1	Decision Trees as Classifiers	113
6.2	Induction of Decision Trees	117
6.3	How Much Information Does an Attribute Convey?	119
6.4	Binary Split of a Numeric Attribute	122
6.5	Pruning	126
6.6	Converting the Decision Tree into Rules	130
6.7	Summary and Historical Remarks	132
6.8	Solidify Your Knowledge	133
7	Computational Learning Theory	137
7.1	PAC Learning	137
7.2	Examples of PAC Learnability	141
7.3	Some Practical and Theoretical Consequences	143
7.4	VC-Dimension and Learnability	145
7.5	Summary and Historical Remarks	148
7.6	Exercises and Thought Experiments	149
8	A Few Instructive Applications	151
8.1	Character Recognition	151
8.2	Oil-Spill Recognition	155
8.3	Sleep Classification	158
8.4	Brain-Computer Interface	161
8.5	Medical Diagnosis	165

8.6	Text Classification	167
8.7	Summary and Historical Remarks	169
8.8	Exercises and Thought Experiments	170
9	Induction of Voting Assemblies	173
9.1	Bagging	173
9.2	Schapire's Boosting	176
9.3	Adaboost: Practical Version of Boosting	179
9.4	Variations on the Boosting Theme	183
9.5	Cost-Saving Benefits of the Approach	185
9.6	Summary and Historical Remarks	187
9.7	Solidify Your Knowledge	188
10	Some Practical Aspects to Know About	191
10.1	A Learner's Bias	191
10.2	Imbalanced Training Sets	194
10.3	Context-Dependent Domains	199
10.4	Unknown Attribute Values	202
10.5	Attribute Selection	204
10.6	Miscellaneous	206
10.7	Summary and Historical Remarks	208
10.8	Solidify Your Knowledge	208
11	Performance Evaluation	211
11.1	Basic Performance Criteria	211
11.2	Precision and Recall	214
11.3	Other Ways to Measure Performance	219
11.4	Learning Curves and Computational Costs	222
11.5	Methodologies of Experimental Evaluation	224
11.6	Summary and Historical Remarks	227
11.7	Solidify Your Knowledge	228
12	Statistical Significance	231
12.1	Sampling a Population	231
12.2	Benefiting from the Normal Distribution	235
12.3	Confidence Intervals	239
12.4	Statistical Evaluation of a Classifier	241
12.5	Another Kind of Statistical Evaluation	244
12.6	Comparing Machine-Learning Techniques	245
12.7	Summary and Historical Remarks	247
12.8	Solidify Your Knowledge	248
13	Induction in Multi-Label Domains	251
13.1	Classical Machine Learning in Multi-Label Domains	251
13.2	Treating Each Class Separately: Binary Relevance	254
13.3	Classifier Chains	256

- 13.4 Another Possibility: Stacking 258
- 13.5 A Note on Hierarchically Ordered Classes..... 260
- 13.6 Aggregating the Classes 263
- 13.7 Criteria for Performance Evaluation..... 265
- 13.8 Summary and Historical Remarks 268
- 13.9 Solidify Your Knowledge 269
- 14 Unsupervised Learning 273**
 - 14.1 Cluster Analysis 273
 - 14.2 A Simple Algorithm: *k*-Means..... 277
 - 14.3 More Advanced Versions of *k*-Means 281
 - 14.4 Hierarchical Aggregation 283
 - 14.5 Self-Organizing Feature Maps: Introduction..... 286
 - 14.6 Some Important Details 289
 - 14.7 Why Feature Maps? 291
 - 14.8 Summary and Historical Remarks 293
 - 14.9 Solidify Your Knowledge 294
- 15 Classifiers in the Form of Rulesets 297**
 - 15.1 A Class Described By Rules 297
 - 15.2 Inducing Rulesets by Sequential Covering..... 300
 - 15.3 Predicates and Recursion 302
 - 15.4 More Advanced Search Operators..... 305
 - 15.5 Summary and Historical Remarks 306
 - 15.6 Solidify Your Knowledge 307
- 16 The Genetic Algorithm..... 309**
 - 16.1 The Baseline Genetic Algorithm 309
 - 16.2 Implementing the Individual Modules 311
 - 16.3 Why It Works..... 314
 - 16.4 The Danger of Premature Degeneration..... 317
 - 16.5 Other Genetic Operators 319
 - 16.6 Some Advanced Versions 321
 - 16.7 Selections in *k*-NN Classifiers 324
 - 16.8 Summary and Historical Remarks 327
 - 16.9 Solidify Your Knowledge 328
- 17 Reinforcement Learning 331**
 - 17.1 How to Choose the Most Rewarding Action 331
 - 17.2 States and Actions in a Game..... 334
 - 17.3 The SARSA Approach 337
 - 17.4 Summary and Historical Remarks 338
 - 17.5 Solidify Your Knowledge 338
- Bibliography 341**
- Index 347**

Introduction

Machine learning has come of age. And just in case you might think this is a mere platitude, let me clarify.

The dream that machines would one day be able to learn is as old as computers themselves, perhaps older still. For a long time, however, it remained just that: a dream. True, Rosenblatt's perceptron did trigger a wave of activity, but in retrospect, the excitement has to be deemed short-lived. As for the attempts that followed, these fared even worse; barely noticed, often ignored, they never made a breakthrough—no software companies, no major follow-up research, and not much support from funding agencies. Machine learning remained an underdog, condemned to live in the shadow of more successful disciplines. The grand ambition lay dormant.

And then it all changed.

A group of visionaries pointed out a weak spot in the knowledge-based systems that were all the rage in the 1970s' artificial intelligence: where was the “knowledge” to come from? The prevailing wisdom of the day insisted that it should take the form of *if-then* rules put together by the joint effort of engineers and field experts. Practical experience, though, was unconvincing. Experts found it difficult to communicate what they knew to engineers. Engineers, in turn, were at a loss as to what questions to ask and what to make of the answers. A few widely publicized success stories notwithstanding, most attempts to create a knowledge base of, say, tens of thousands of such rules proved frustrating.

The proposition made by the visionaries was both simple and audacious. If it is so hard to tell a machine exactly how to go about a certain problem, why not provide the instruction indirectly, conveying the necessary skills by way of examples from which the computer will—yes—*learn!*

Of course, this only makes sense if we can rely on the existence of algorithms to do the learning. This was the main difficulty. As it turned out, neither Rosenblatt's perceptron nor the techniques developed after it were very useful. But the absence of the requisite machine-learning techniques was not an obstacle; rather, it was a challenge that inspired quite a few brilliant minds. The idea of endowing computers with learning skills opened new horizons and created a large amount of excitement. The world was beginning to take notice.

The bombshell exploded in 1983. *Machine Learning: The AI Approach*¹ was a thick volume of research papers which proposed the most diverse ways of addressing the great mystery. Under their influence, a new scientific discipline was born—virtually overnight. Three years later, a follow-up book appeared and then another. A soon-to-become-prestigious scientific journal was founded. Annual conferences of great repute were launched. And dozens, perhaps hundreds, of doctoral dissertations, were submitted and successfully defended.

In this early stage, the question was not only *how* to learn but also *what* to learn and *why*. In retrospect, those were wonderful times, so creative that they deserve to be remembered with nostalgia. It is only to be regretted that so many great thoughts later came to be abandoned. Practical needs of realistic applications got the upper hand, pointing to the most promising avenues for further efforts. After a period of enchantment, concrete research strands crystallized: induction of the *if-then* rules for knowledge-based systems; induction of classifiers, programs capable of improving their skills based on experience; automatic fine-tuning of Prolog programs; and some others. So many were the directions that some leading personalities felt it necessary to try to steer further development by writing monographs, some successful, others less so.

An important watershed was Tom Mitchell's legendary textbook.² This summarized the state of the art of the field in a format appropriate for doctoral students and scientists alike. One by one, universities started offering graduate courses that were usually built around this book. Meanwhile, the research methodology became more systematic, too. A rich repository of machine-learning test beds was created, making it possible to compare the performance of learning algorithms. Statistical methods of evaluation became widespread. Public domain versions of most popular programs were made available. The number of scientists dealing with this discipline grew to thousands, perhaps even more.

Now, we have reached the stage where a great many universities are offering machine learning as an undergraduate class. This is quite a new situation. As a rule, these classes call for a different kind of textbook. Apart from mastering the baseline techniques, future engineers need to develop a good grasp of the strengths and weaknesses of alternative approaches; they should be aware of the peculiarities and idiosyncrasies of different paradigms. Above all, they must understand the circumstances under which some techniques succeed and others fail. Only then will they be able to make the right choices when addressing concrete applications. A textbook that is to provide all of the above should contain less mathematics, but a lot of practical advice.

These then are the considerations that have dictated the size, structure, and style of a teaching text meant to provide the material for a one-semester introductory course.

¹Edited by R. Michalski, J. Carbonell, and T. Mitchell.

²T. Mitchell, *Machine Learning*, McGraw-Hill (1997).

The first problem is the choice of material. At a time when high-tech companies are establishing machine-learning groups, universities have to provide the students with such knowledge, skills, and understanding that are relevant to the current needs of the industry. For this reason, preference has been given to Bayesian classifiers, nearest-neighbor classifiers, linear and polynomial classifiers, decision trees, the fundamentals of the neural networks, and the principle of the boosting algorithms. Significant space has been devoted to certain typical aspects of concrete engineering applications. When applied to really difficult tasks, the baseline techniques are known to behave not exactly the same way they do in the toy domains employed by the instructor. One has to know what to expect.

The book consists of 17 chapters, each covering one major topic. The chapters are divided into sections, each devoted to one critical problem. The student is advised to proceed to the next section only after having answered the set of 2–4 “control questions” at the end of the previous section. These questions are here to help the student decide whether he or she has mastered the given material. If not, it is necessary to return to the previous text.

As they say, only practice makes perfect. This is why at the end of each chapter are exercises to encourage the necessary practicing. Deeper insight into the diverse aspects of the material will then be gained by going through the thought experiments that follow. These are more difficult, but it is only through hard work that an engineer develops the right kind of understanding. The acquired knowledge is then further solidified by suggested computer projects. Programming is important, too. Nowadays, everybody is used to downloading the requisite software from the web. This shortcut, however, is not recommended to the student of this book. It is only by being forced to flesh out all the details of a computer program that you learn to appreciate all the subtle points of the machine-learning techniques presented here.