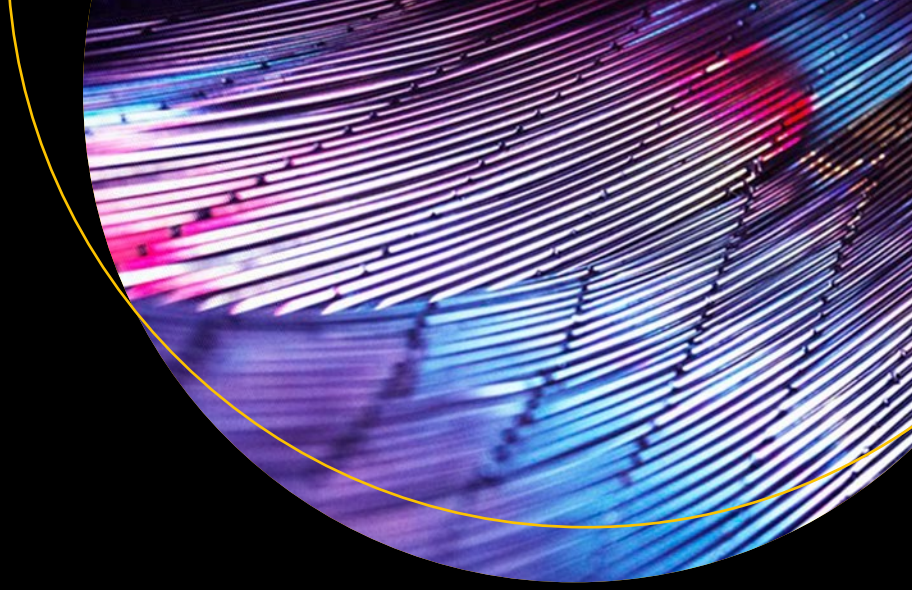

Machine Learning for Oracle Database Professionals

Deploying Model-Driven Applications and Automation Pipelines

Heli Helskyaho
Jean Yu
Kai Yu

Apress®

Machine Learning for Oracle Database Professionals

Deploying Model-Driven Applications and Automation Pipelines

Heli Helskyaho
Jean Yu
Kai Yu

Apress®

Machine Learning for Oracle Database Professionals: Deploying Model-Driven Applications and Automation Pipelines

Heli Helskyaho
Helsinki, Finland

Jean Yu
Austin, TX, USA

Kai Yu
Austin, TX, USA

ISBN-13 (pbk): 978-1-4842-7031-8
ISBN-13 (electronic): 978-1-4842-7032-5
https://doi.org/10.1007/978-1-4842-7032-5

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Jonathan Gennick
Development Editor: Laura Berendson
Coordinating Editor: Jill Balzano

Cover image designed by Freepik (www.freepik.com)

Distributed to the book trade worldwide by Springer Science+Business Media LLC, 1 New York Plaza, Suite 4600, New York, NY 10004. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at http://www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/9781484270318. For more detailed information, please visit http://www.apress.com/source-code.

Printed on acid-free paper

Table of Contents

About the Authors

Heli Helskyaho is the CEO of Miracle Finland Oy. She holds a master's degree in computer science from the University of Helsinki and specializes in databases. She is currently working on her doctoral studies, researching and teaching at the University of Helsinki. Her research areas cover big data, multi-model databases, schema discovery, and methods and tools for utilizing semi-structured data for decision making.

Heli has been working in IT since 1990. She has held several positions, but every role has included databases and database designing. She believes that understanding your data makes using the data much easier. She is an Oracle ACE Director, an Oracle Groundbreaker Ambassador, and a frequent speaker at many conferences. She is the author of several books and has been listed as one of the top 100 influencers in the IT sector in Finland for each year from 2015 to 2020.

Heli can be reached at `www.linkedin.com/in/helihelskyaho/`, `https://helifromfinland.blog`, and `https://twitter.com/HeliFromFinland`.

Jean Yu is a Senior Staff MLOps Engineer at Habana Labs, an Intel company. Prior to that, she was a Senior Data Engineer on the IBM Hybrid Cloud Management Data Science Team. Her current interests include deep learning, model productization, and distributed training of massive transformer-based language models. She holds a master's degree in computer science from the University of Texas at San Antonio. She has more than 25 years of experience in developing, deploying, and managing software applications,

as well as in leading development teams. Her recent awards include an Outstanding Technical Achievement Award for significant innovation in Cloud Brokerage Cost and Asset Management products in 2019 as well as an Outstanding Technical Achievement Award for Innovation in the Delivery of Remote Maintenance Upgrade for Tivoli Storage Manager in 2011.

Jean is a master inventor with 14 patents granted. She has been a voting member of the IBM Invention Review Board from 2006 to 2020. She has been a speaker at conferences such as North Central Oracle Apps User Group Training Day 2019 and Collaborate 2020.

Jean can be reached at www.linkedin.com/in/jean-yu/.

Kai Yu is a Distinguished Engineer in the Dell Technical Leadership Community. He is responsible for providing technical leadership to Dell Oracle Solutions Engineering. He has more than 27 years of experience in architecting and implementing various IT solutions, specializing in Oracle databases, IT infrastructure, the cloud, business analytics, and machine learning.

Kai has been a frequent speaker at various IT/Oracle conferences, with over 200 presentations in more than 20 countries. He has authored 36 articles in technical journals such as *IEEE Transactions on Big Data* and co-authored the book *Expert Oracle RAC 12c* (Apress, 2013). He has been an Oracle ACE Director since 2010. He has served on the IOUG/Quest Conference committee, as the IOUG RAC SIG president, and as the IOUG CLOUG SIG co-founder and vice president. He received the 2011 OAUG Innovator of Year award and the 2012 Oracle Excellence Award for Technologist of the Year: Cloud Architect by *Oracle Magazine*. He holds master's degrees in computer science and engineering from the Huazhong University of Science and Technology and the University of Wyoming.

Kai can be reached at https://kyuoracleblog.wordpress.com, www.linkedin.com/in/kaiyu1, and https://twitter.com/ky_austin1.

About the Technical Reviewer

Adrian Png is a seasoned solutions architect with more than 20 years of experience working with clients to design and implement state-of-the-art infrastructure and applications. He earned a Master of Technology (Knowledge Engineering) degree from the National University of Singapore and has applied his knowledge and skills in artificial intelligence and machine learning in practice, notably in his paper "Primer design for Whole Genome Amplification using genetic algorithms" (*In Silico Biology*, 2006; 6(6): 505–14). Adrian is also trained and certified in several Oracle technologies, including Oracle Cloud Infrastructure, Oracle Autonomous Database, Oracle cloud-native services, Oracle Database, and Oracle Application Express. He is an Oracle ACE and is a recognized contributor to the Oracle community. Most recently, he co-authored the book *Getting Started with the Oracle Cloud Free Tier* (Apress, 2020), an indispensable reference for anyone just starting Oracle Cloud and wishing to get the most out of Oracle's cloud offerings.

Acknowledgments

I want to thank my family, Marko, Patrik, and Matias, for their continuous support during this project, and my parents for their encouragement throughout my life. You gave me the confidence to write this book.

Thank you, Jean and Kai, for writing this book with me! It was a great pleasure to work with you. You are great friends and extremely talented.

Special thank you to Charlie Berger and Adrian Png for the extremely valuable comments, guidance, and support during this project. We could not have been able to write this book without your help!

And thank you, Jonathan, Jill, and the rest of the Apress team!

—Heli Helskyaho

I want to thank my parents, my grandfather, and my middle school math teacher for encouraging me to pursue an engineering career. Special thank you to my husband Kai, and to my graduate advisors Dr. Steve Robbins and Dr. Kay Robbins of UTSA for introducing me to the software industry in 1995.

Thank you Heli and Kai for working with me on this amazing project. I greatly admire your dedication and passion for the Oracle Machine Learning community.

I'd like to extend a special thanks to our reviewer Adrian Png and Apress editors Jonathan Gennick and Jill Balzano. I appreciate your extremely valuable review comments. Your guidance and support made this project possible for a novice writer like myself. I learned so much. Thank you!

—Jean Yu

I dedicate this book to the readers of this book. I want to thank all the people who have assisted with this book, especially the technical reviewer, Andrew Peng, Apress editors Jonathan Gennick and Jill Balzano, and the rest of the Apress team, for their great efforts and patience in transforming the technical content into a finished book.

I thank Heli and Jean for their great talents, dedication, and amazing teamwork. It has been my great honor to be part of this great team with them.

I also thank my wife, Jean, and my daughter, Jessica, for their continuous support during this project. I want to dedicate this book to my parents, who encouraged me to pursue my education and career in computer technology.

I thank my mentor, Dell Senior Fellow Jimmy Pike, and the Dell Technical Leadership Community, who have inspired me to pursue technical excellence and expand my expertise in AI and machine learning. I also want to thank my manager, Ibrahim Fashho, for his great inspiration and longtime support.

—Kai Yu

Introduction

This book helps database developers and DBAs gain a conceptual understanding of machine learning, including the methods, algorithms, the process, and deployment. The book covers Oracle Machine Learning (OML) technologies that enable machine learning with Oracle Database, including OML4SQL, OML Notebooks, OML4R, and OML4Py. *Machine Learning for Oracle Database Professionals* focuses on Oracle machine learning in Oracle autonomous databases, such as the Autonomous Data Warehouse (ADW) database as part of the ADW collaborative environment. This book also covers some advanced topics, such as delivery and automation pipelines in machine learning.

This book also provides practical implementation details through hands-on examples to show how to implement machine learning with OML with ADW and how to automate the deployment of machine learning. The primary goal is to bridge the gap between database development/management and machine learning by gaining practical knowledge of machine learning. As a seasoned database professional skilled in managing data, you can apply this knowledge by analyzing data in the same data management system. Through this book, three authors with rich experience in machine learning and database development and management take you on a journey from being a database developer or DBA to a data scientist.

Readers and Audiences

This book is written for

- Database developers and administrators who want to learn about machine learning
- Developers wanting to build models and applications using Oracle Database's built-in machine learning feature set
- Administrators tasked with supporting applications in Oracle Database and ADW that use the machine learning feature set

Readers will learn how to do the following.

- Build an automated pipeline that can detect and handle changes in data/model performance
- Develop and deploy machine learning projects in ADW
- Develop machine learning with Oracle Database using the built-in machine learning packages
- Analyze, develop, evaluate, and deploy various machine learning models using OML4R and OML4SQL

CHAPTER 1

Introduction to Machine Learning

We live in exciting times with smartphones and watches, smart clothes, robots, drones, face recognition, smart personal assistants, recommender systems, self-driving autonomous cars, and 24/7 service chatbots, all of which are *artificial intelligence* (AI). But what is intelligence? Intelligence might be defined as the ability to acquire and apply knowledge and skills, in other words, to learn and use the skills learned. Artificial intelligence is exactly that but done by computers and software. In real life, people would like to have intelligent machines that can do things people find boring, do inefficiently, or maybe cannot do at all. It could be an extension of human intelligence through using computers, which is artificial intelligence. The core of artificial intelligence is the ability to learn, acquire knowledge and skills, which is *machine learning*. In machine learning, the machine is learning, reasoning, and self-correcting. Arthur Samuel defined machine learning in 1959 as "a field of study that gives computers the ability to learn without being explicitly programmed," which defines machine learning very well.

Why Machine Learning?

When Arthur Samuel defined machine learning in 1959, a lot of the mathematics and statistics needed was already invented. Still, there was no technology nor enough data to get the theory to practice. Today, there are hardware solutions, including GPUs and TPUs for matrix calculation, inexpensive storage solutions for storing data, open data sets, pre-trained models for transfer learning, and so on. All this makes it possible to use machine learning in the most interesting and useful ways. But it is not only that we are now able to use machine learning; it is also necessary to use it. With its volume, velocity, variety,

H. Helskyaho et al., *Machine Learning for Oracle Database Professionals*,
https://doi.org/10.1007/978-1-4842-7032-5_1

veracity, viability, value, variability, and visualization, big data has made it necessary to change traditional data processing into something more efficient and faster: machine learning.

Machine learning is not a silver bullet, and it should not be seen as such. Machine learning should be used only when it brings value. Typical use cases are when the rules and equations are complex or constantly changing. If the rules are understandable and can be programmed with if-else-then structures, machine learning might not be the best solution.

Classic examples of machine learning use cases are image recognition, speech recognition, fraud detection, predicting shopping trends, spam filters, medical diagnosis, or robotics. Some examples of machine learning to businesses are churn prediction, predicting customer behavior, anticipating voluntary employee attrition, and cross and up-sell opportunities.

An important requirement for machine learning is that you have data; otherwise, it makes no sense. The data is given to the machine, or the machine produces it, as it does in reinforcement learning. The better the quality of the data is, the better it can be used by machine learning. But even though the data is of excellent quality and machine learning works like a charm, a machine learning prediction is never a fact; it is always a sophisticated guess. Sometimes that guess is good and even useful, but sometimes it is not.

Also, a well-working machine learning model will no longer work well if something has changed—perhaps there is more noise in the data, the amount of data is larger, or the quality of data has lessened. In other words, it is important to understand that machine learning models need to monitor their defined metrics to make sure they still work as planned and to tune them if necessary.

What Is Machine Learning?

Machine learning can be divided into different categories based on the nature of the training data, the problem type, and the technique used to solve it. This book divides machine learning into three main categories: supervised learning, unsupervised learning, and semi-supervised learning.

Supervised Learning

Supervised machine learning is supervised by a human. Typically, that means that somebody has labeled the data to show the output or the correct answer. For example, somebody manually checks 1000 pictures and labels them to identify which of the pictures show cats, dogs, or horses.

Supervised learning is used when there is enough high-quality data and you know the target (e.g., the data is labeled). The models are trained and tested on known input and known output data to predict future outputs on new data. When testing the models, the prediction is compared to the true output to evaluate the models. To make this process meaningful, the training data must separate from the data used for testing. Each model is built using a different algorithm. A model maps the data to the algorithm and produces the prediction. So, each algorithm is processing the data differently. Depending on the chosen metrics, the evaluation process defines which algorithm performed the best, and the model using that algorithm can be implemented into production. The selection of an algorithm depends on the data's size, the type of data, the insights you want to get from the data, or how those insights will be used. The decision is a trade-off between many things, such as the predictive accuracy on new data, the speed of training, memory usage, transparency (black box vs. "clear box," how decisions are made), or interpretability (the ability of a human to understand the model).

Regression and classification are the most common methods for supervised learning. Regression predicts numeric values and works with continuous data. Classification works with categorized data and classifies data points. So, if you want to predict a quantity, you should use regression. If you want to predict a class or a group, you should use classification. An example of regression is the price of a house over time. An example of classification is predicting a beer's evaluation by rating it against other beers on a scale of 1 to 5, with 1 being poor quality and 5 being excellent. Figure 1-1 is a simple example of regression. From the line shown in Figure 1-1, you can see that for value 3, the prediction of the target value is 1.5.

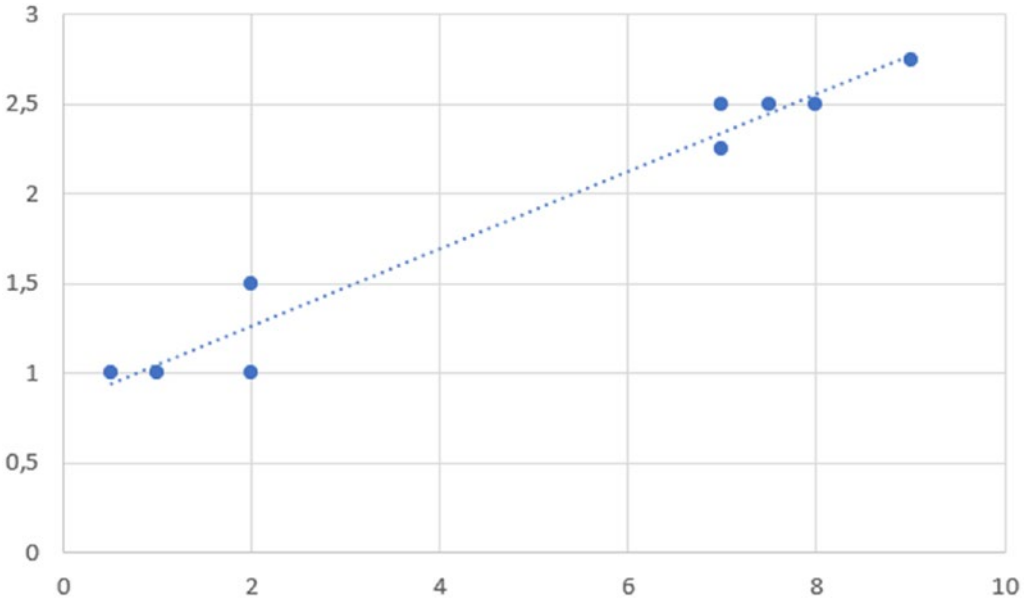

Figure 1-1. *An example of regression*

Figure 1-2 is an example of classification. The data points are classified in orange and blue. The red line shows in which category each data point belongs. You can see that point (4,1) belongs to the orange group, and point (9,2) belongs to the blue group.

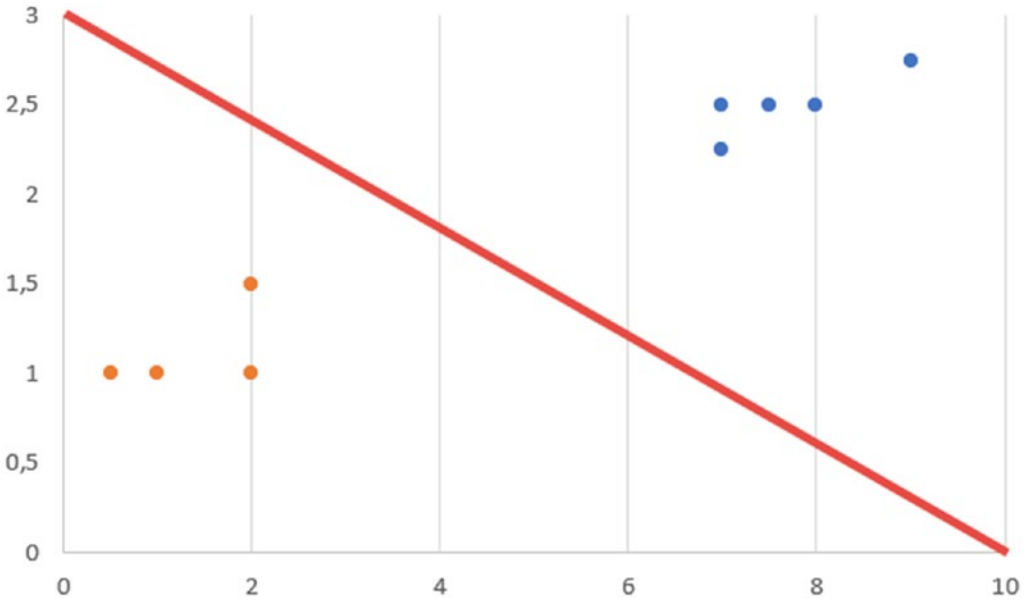

Figure 1-2. *An example of classification*

Time series forecasting can be a supervised learning problem. The machine learning model predicts the value of the next time step by using the value of a previous time step. You need data that is suitable for the purpose. This method is called the *sliding window method*. For example, the following is a small part of a data set.

Time	Measure
1	100
2	150
3	170
4	250
5	330

You can reconstruct this data set to be useful in supervised learning by setting the next value as the prediction of the value, as follows.

X	Y
?	100
100	150
150	170
170	250
250	330
330	?

The first and the last rows cannot be used because some of the information is missing, so we remove those rows. Afterward, there is a solid data set that can be used in supervised machine learning.

Time series forecasting can be used in weather forecasting, inventory planning, or resource allocation, for example. Time series prediction can be very complex, and understanding the data is very important. For example, trends in data might be different in summer than in winter, or on weekdays than on weekends. That must be considered when building the model or maybe several models for different trends.

Deep learning has become very popular as a technique for mainly supervised machine learning. Deep learning is typically used with more complex machine learning tasks on text, voice, recommender systems, or images and videos. Text can be transformed into speech using deep learning. Speech can be transformed into text, which can be used as input to another machine learning task, such as translating from the Finnish language to English.

Automatic speech recognition or natural language processing might also be tasks for deep learning. Recommender systems are producing recommendations for users to make their decision process easier and more fluent. There are three kinds of recommender systems: collaborative filtering, content-based, and hybrid recommender systems. A collaborative filtering recommender system uses the decisions of other users with a similar profile as a base for a recommendation for another user. Content-based recommender systems create recommendations based on similarities of new items to those that the user liked in the past. Hybrid recommender systems use multiple approaches when creating recommendations. Visual recognition and computer vision are very typical and useful tasks for deep learning. Image or action classification, object detection or recognition, image captioning, or image segmentation are useful in machine learning.

One difference between classical supervised learning and deep learning is that in deep learning you do not need to perform feature extraction at all, it is done by the machine as part of the deep learning process. In supervised learning, feature extracting is time-consuming manual work. Of course, that means that deep learning needs more data to do it and, in general, more resources and time. Deep learning has become more popular and useful because of so many improvements in different areas. There is a lot of digital data (photos, videos, voice, etc.) available. The technology has improved: existing data sets and pre-trained models, transfer learning, research such as combining convolutional layers to a neural network, and much more is available. Things that were difficult or nearly impossible to perform using deep learning have become easy and almost trivial. There are plenty of example codes that programmers can use and start building their first deep learning projects.

Deep learning uses neural networks for the prediction process and backpropagation to learn (e.g., tune the network). A neural network consists of neurons. Each input is multiplied by its weight, and a bias is added to that. When using an activation function, an output is passed to the next layer until the last layer and the prediction are reached. The weight and the bias are called *hyperparameters*. Their values are defined

before the machine learning process starts. The first values are a guess, but by using backpropagation and an optimizer function, the process tunes those hyperparameters to have a better-performing model.

In a neural network, there are plenty of hyperparameters that need to be defined before starting the process, and they need to be tuned during it. Some examples of hyperparameters are the number of layers, number of epochs, the batch size, number of neurons in each layer, or what activation function, optimizer, and loss function to use. The backpropagation computes the loss function for the initial guess and the gradient of the loss function. Using that information the optimizer takes the steps to a negative gradient direction to reduce loss. This is done as long as needed to get the weights as good as possible. A convolutional neural network complements the neural network with convolutional layers. Convolutional neural networks are especially useful with image processing. A convolutional neural network consists of several convolutional layers (filter, output, pooling) and a flattening layer to pass the data to a neural network for further processing.

Algorithms for Supervised Learning

A model uses an algorithm to produce a prediction. The goal is to find the best algorithm for the use case. There are plenty of algorithms to be used with supervised learning.

For classification, examples of algorithms include k-nearest neighbors (kNN), naïve Bayes, neural networks, decision trees, or support-vector machine (SVM). kNN categorizes objects based on the classes of their nearest neighbors that have already been categorized. It assumes that objects near each other are similar. kNN is a simple algorithm, but it consumes a lot of memory, and the prediction speed can be slow if the amount of data is large or several dimensions are used. Naïve Bayes assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature when the class is defined. It classifies new data based on the highest probability of its belonging to a particular class. For example, if a fruit is red, it could be an apple, and if a fruit is round, it could be an apple, but if it is both red and round, there is a stronger probability that the fruit is an apple.

Naïve Bayes works well for a data set containing many features (e.g., the dimensionality of the inputs is high). It is simple to implement and easy to interpret. A neural network imitates the way biological nervous systems and the brain process information. A large number of highly interconnected processing elements (neurons)

work together to solve specific problems. Neural networks are good for modeling highly nonlinear systems when the interpretability of the model is not important. They are useful when data is available incrementally, you wish to constantly update the model, and unexpected changes in your input data may occur.

Decision trees are very typical classification algorithms. Decision trees, bagged decision trees, or boosted decision trees are tree structures that consist of branching conditions. They predict responses to data by following the decisions in the tree from the root down to a leaf node.

A *bagged decision tree* consists of several trees that are trained independently on data. Boosting involves reweighting misclassified events and building a new tree with reweighted events. Decision trees are used when there is a need for an algorithm that is easy to interpret and fast to fit, and you want to minimize memory usage but high predictive accuracy is not a requirement and the time taken to train a model is less of a concern. A *support-vector machine* (SVM) classifies data by finding the linear decision boundary, or hyperplane, that separates all the data points of one class from those of another class. The best hyperplane for an SVM is the one with the largest margin between the two classes when the data is linearly separable. If the data is not linearly separable, a loss function penalizes points on the wrong side of the hyperplane. Sometimes SVMs use a kernel to transform nonlinearly separable data into higher dimensions where a linear decision boundary can be found. SVMs work the best for high-dimensional, nonlinearly separable data that has exactly two classes. For multiclass classification, it can be used with a technique called *error-correcting output codes*. It is very useful as a simple classifier, it is easy to interpret, and it is accurate.

For regression tasks, some examples of algorithms are linear regression, nonlinear regression, generalized linear model (GLM), Gaussian process regression (GPR), regression tree, or support-vector regression (SVR).

Linear regression describes a continuous response variable as a linear function of one or more predictor variables. Linear regression could be used when you need an algorithm that is easy to interpret and fast to fit. It is often the first model to be fitted to a new data set and could be used as a baseline for evaluating other, more complex, regression models.

Nonlinear regression describes nonlinear relationships in data. It can be used when data has nonlinear trends and cannot be easily transformed into a linear space.

GLM is a special nonlinear model that uses linear methods. It fits a linear combination of the input to a nonlinear function of the output. It could be used when the response variables have non-normal distributions.

GPR is for nonparametric models used to predict the value of a continuous response variable; for example, to interpolate spatial data, as a surrogate model to optimize complex designs such as automotive engines, or to forecast mortality rates.

Regression trees are similar to decision trees for classification, but they are modified to predict continuous responses. They could be used when predictors are categorical (discrete) or behave nonlinearly.

SVM regression algorithms (SVR) work like SVM classification algorithms but are modified to predict a continuous response. Instead of finding a hyperplane that separates data, SVR algorithms find the decision boundaries and data points inside those boundaries. SVR can be useful with high-dimensional data.

Unsupervised Learning

Unsupervised learning is machine learning with unlabeled data, with an unknown target, to find something useful from the data. Unsupervised learning finds hidden patterns or intrinsic structures in input data.

Clustering is one of the most common methods for unsupervised learning. It is used for exploratory data analysis to find hidden patterns or groupings in data. There are typically two ways of clustering: hard and soft. In hard clustering, each data point belongs to only one cluster, whereas in soft clustering, each data point can belong to more than one cluster.

In Figure 1-3, you can see data points, and in Figure 1-4, you see how they have been clustered in two clusters: green and blue. The idea of clustering is that you tell the algorithm that you want to break the data into two groups, and it finds things that are common to the data points and things that are different. Using that information, the algorithm decides which group (cluster) a particular data point belongs to.

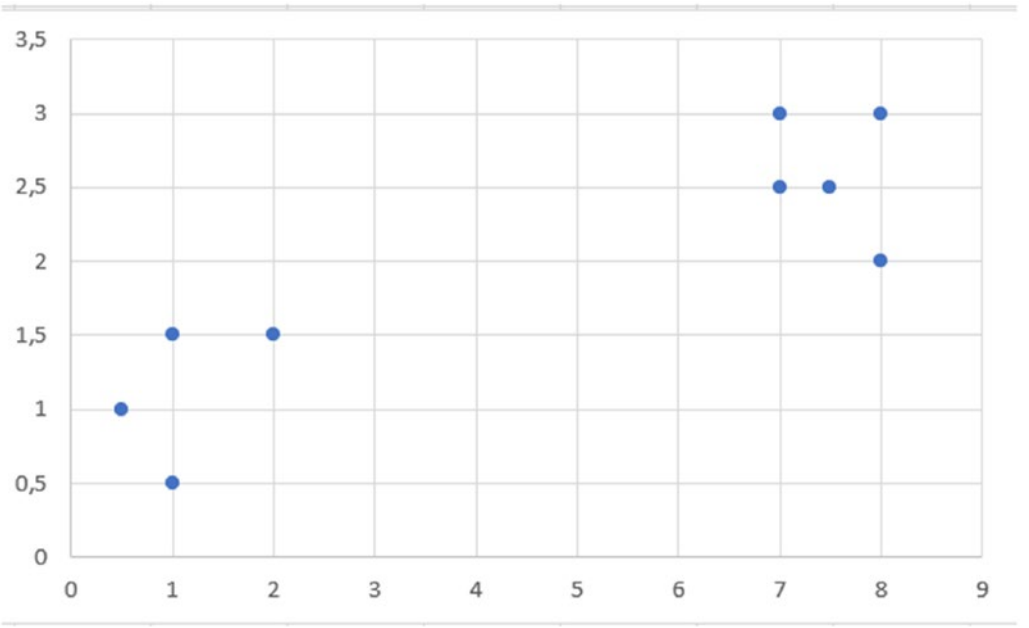

Figure 1-3. *Data points for clustering*

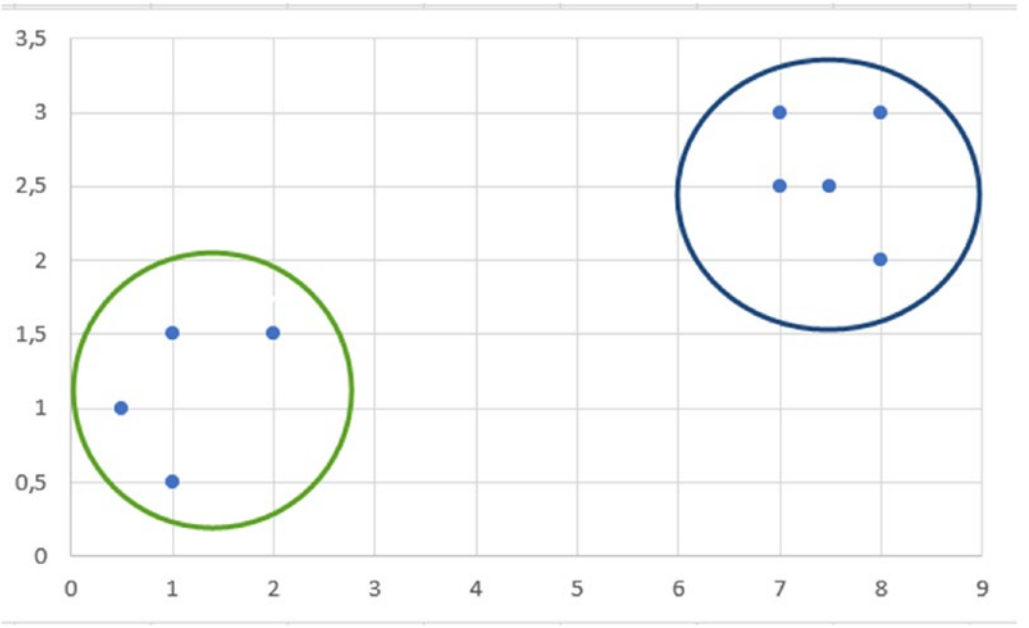

Figure 1-4. *Data points clustered in two clusters: green and blue*

Association rule mining/learning is also a very useful unsupervised machine learning method. It is for identifying hidden patterns in the data set. A typical example of association rules is shopping basket data analysis. Using association rules, you can find interesting patterns on what people typically buy. Using that information, you can better plan the layout of a shop, product placement in a supermarket, or launch better market campaigns to different customer groups. A classic example is that diapers and beer are often bought at the same time. The historical explanation was that fathers stop by the store to buy diapers on their way home from the office and often end up buying beer as well.

Anomaly detection is for finding anomalies in a data set. It can be used with unsupervised, supervised, or semi-supervised machine learning. Figure 1-5 is an example of an anomaly (marked in red). Anomaly detection can be used for finding exceptions that must be handled, or it can be used for data preparation for machine learning to find outliers. For instance, if an anomaly has been detected in a log file, it can alert the database administrator to check what is going on.

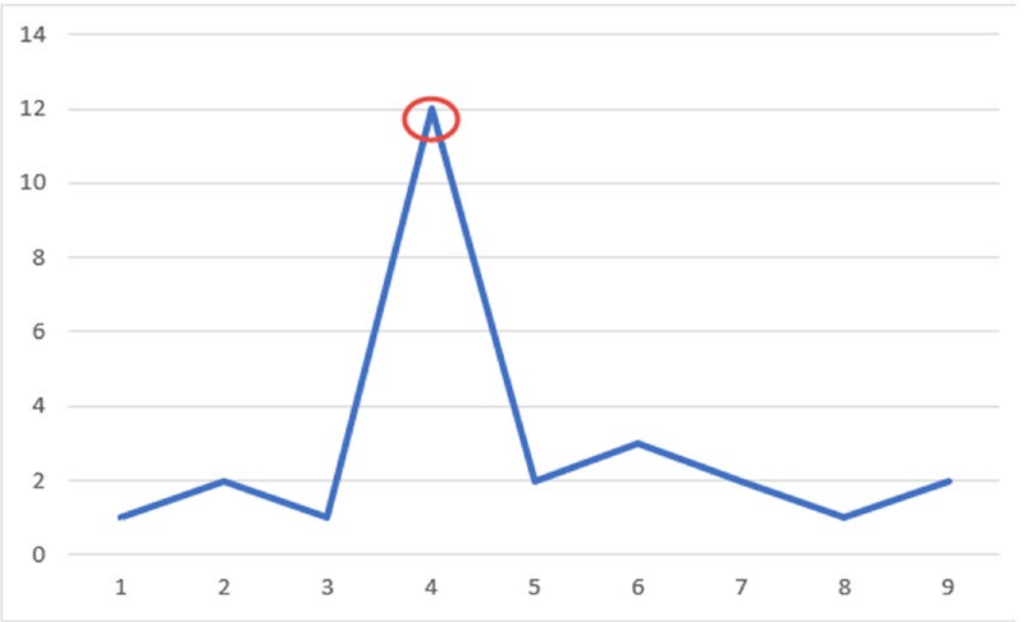

Figure 1-5. *An example of an anomaly*

Dimensionality reduction is an unsupervised machine learning method that can be used when there is a need to reduce the data set's features as preprocessing for supervised learning or to tune the model. Feature selection is a technique to reduce the number of features in a data set to improve the model accuracy or performance.

It ranks the importance of the existing features in the data set and discards less important ones. Feature extraction also aims to reduce the number of features in a data set, but it creates new features from the existing ones and then discards the original features. A term used for both methods is *dimensionality reduction* because they aim to reduce the dimensions/features of the data set.

Algorithms for Unsupervised Learning

There are plenty of algorithms for unsupervised learning. For hard clustering, several algorithms are available, including k-means, k-medoids, and hierarchical clustering. K-means (Lloyd's algorithm) partitions data into k number of mutually exclusive clusters (centroids) and assigns each observation to the closest cluster. Then it moves the centroids to the true mean of its observations. *K-means* works well when the number of clusters is known, and there is a need for fast clustering of large data sets.

K-medoids is similar to k-means but requires that the cluster centers coincide with points in the data. In other words, it chooses datapoints as centers, medoids. k-medoids can be more robust to noise and outliers than k-means. It works well when the number of clusters is known, and there is a need for fast clustering of categorical data.

Hierarchical clustering is used when the number of clusters is unknown and/or you want visualization to guide the selection. It works either as a divisive or agglomerative method. A divisive method assigns all observations to one cluster and then partitioning that cluster into several clusters. An agglomerative method sets each observation to its own cluster and merges similar clusters. Of these two types of methods, the agglomerative method is used more often.

For soft clustering, there are algorithms like Fuzzy C-means (FCM) or Gaussian mixture model. FCM is similar to k-means, but data points may belong to more than one cluster. It could be used when the number of clusters is known and they overlap. A typical use case is pattern recognition. A Gaussian mixture model is a partition-based clustering where data points come from different multivariate normal distributions with certain probabilities, such as home prices in different areas. It could be used when data points might belong to more than one cluster, and those clusters have different sizes and correlation structures within them.

For anomaly detection, kNN is a simple algorithm, and for each data point, its distance to its k-nearest neighbor could be viewed as the outlier score.

An Apriori algorithm is for association rule-learning problems. It identifies items that often occur together.

Semi-Supervised Learning

There is also a combination of unsupervised and supervised machine learning called *semi-supervised learning*. In semi-supervised learning, a small part of the data is supervised data with labeled data, and a large part of the data is unsupervised data with unlabeled data. The idea in semi-supervised learning is that you should find a way to let the algorithm label the data automatically. These algorithms are often based on assumptions on the relationships of the data distribution. Some algorithms assume that the points close to each other are likely to have similar or the same labels (continuity assumption). Some assume that data in the same cluster most likely have the same labels (cluster assumption). For high-dimensional data, manifold assumption might help find the label.

There are many algorithms and techniques to automatically label the data, and more techniques will soon be invented.

A simple way to label the unlabeled data is called *pseudo-labeling*. The process of pseudo-labeling is simple.

1. Train the model with a small set of labeled data.
2. Use the model to label unlabeled data. Set the prediction as the label. This is called a *pseudo-label*.
3. Link the labels from the labeled training data with the pseudo-labeled data. Link the input data from the labeled data with the unlabeled data.
4. Train the model with that data set.

After this process, the accuracy of the model should be better than simply training it with the labeled data set.

Active learning is a technique in which the learning algorithm chooses a subset of unlabeled data and queries a user interactively to label it. And since the algorithms chose the data set to be labeled, it is assumed this data set is very useful in learning the algorithm.

Many interesting and promising areas in machine learning research have something to do with semi-supervised learning. Examples of those areas are reinforcement learning and self-supervised learning (self-supervision). Some people include reinforcement learning into semi-supervised learning. Some say it is a different concept because it is more about learning a skill than labeling data and should have its own category. This book includes it in semi-supervised learning to keep the concepts simple.

Reinforcement Learning

Reinforcement learning (RL) has a different approach to machine learning. Its goal is to get the maximum reward, not to predict the input data's output. RL is typically used in use cases where the computer needs to learn a skill, such as playing chess or parking a car. RL works well with problems like games, optimization, or navigation because data can be generated easily. A computer can play against itself or a human player and generate a lot of data about the game's strategies. It learns while playing, and it can also use transfer learning to learn faster. RL's best feature is that it can become better than a human since humans are not teaching it with their limited skills. It takes hundreds of thousands of mistakes to learn, which is not possible in all use cases.

A *Markov decision process* (MDP) is the basis of reinforcement learning (RL) and the mathematical formulation for an RL problem. MDP provides a mathematical framework (a discrete-time stochastic control process) for modeling decision making when outcomes are partly random and partly under the control of a decision-maker, just like in RL. The RL process is shown in Figure 1-6.

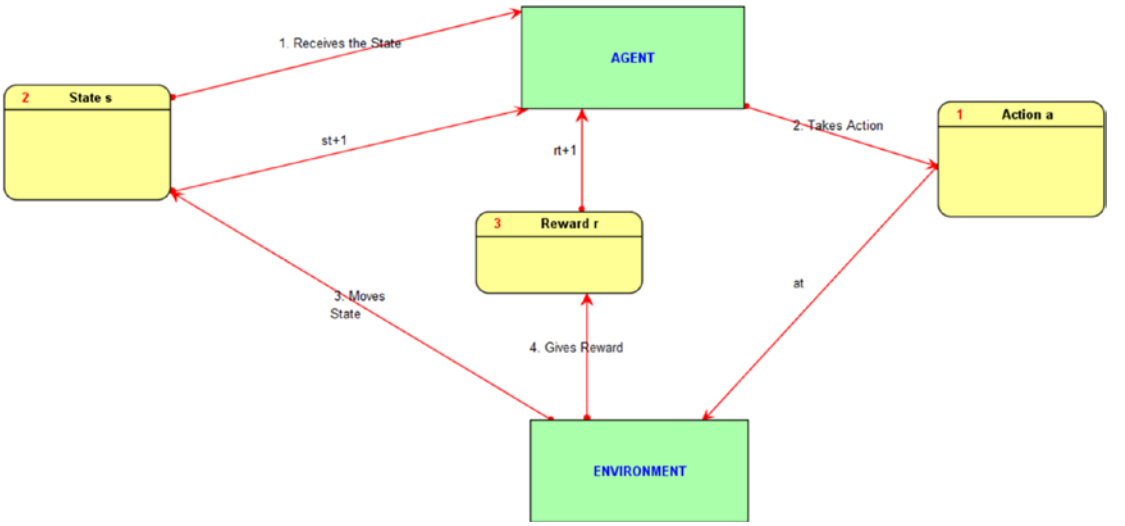

Figure 1-6. *The reinforcement learning process*

A decision-maker/learner is defined as an agent, and anything outside the agent is defined as an environment. The interactions between an agent and an environment are described by three core elements: state (s), action (a), and reward (r). The agent receives as input the current state of the environment. Then the agent chooses an action.