

University of Groningen

Machine learning basic concepts for the movement disorders specialist

van den Brandhof, Elina L.; van der Stouwe, Madelein; Dalenberg, Jelle R.; Tuitert, Inge; de Koning-Tijssen, Marina. A. J.; Biehl, Michael

Published in:
International Review of Movement Disorders

DOI:
[10.1016/bs.irmvd.2023.04.004](https://doi.org/10.1016/bs.irmvd.2023.04.004)

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2023

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

van den Brandhof, E. L., van der Stouwe, M., Dalenberg, J. R., Tuitert, I., de Koning-Tijssen, M. A. J., & Biehl, M. (2023). Machine learning basic concepts for the movement disorders specialist. In A. Sánchez Ferro, & M. H. G. Monje (Eds.), *International Review of Movement Disorders: Digital Technologies in Movement Disorders* (Vol. 5, pp. 21-47). (International Review of Movement Disorders; Vol. 5). Elsevier. <https://doi.org/10.1016/bs.irmvd.2023.04.004>

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.



Machine learning basic concepts for the movement disorders specialist

Elina L. van den Brandhof^{a,b,c,*}, A.M. Madelein van der Stouwe^{a,b},
Jelle R. Dalenberg^{a,b}, Inge Tuitert^{a,b,d}, Marina A.J. Tijssen^{a,b},
and Michael Biehl^{c,e}

^aExpertise Center Movement Disorders Groningen, University Medical Center Groningen, Groningen, The Netherlands

^bDepartment of Neurology, University of Groningen, Groningen, The Netherlands

^cBernoulli Institute for Mathematics, Computer Science and Artificial Intelligence, University of Groningen, Groningen, The Netherlands

^dNHL Stenden University of Applied Sciences, Leeuwarden, The Netherlands

^eSMQB, Institute of Metabolism and Systems Research, College of Medical and Dental Sciences, University of Birmingham, Birmingham, United Kingdom

*Corresponding author: e-mail address: e.l.van.den.brandhof@umcg.nl

Contents

1. Basic vocabulary	22
1.1 Machine learning, statistical modeling, and artificial intelligence	22
1.2 Example	24
1.3 Features	24
1.4 Learning types and tasks	25
2. Generalization in supervised learning	31
2.1 Phases in supervised learning	31
2.2 Validation strategies	35
3. Classification methods	36
3.1 Linear classifiers	36
3.2 Neural networks (NN)	38
3.3 Distance-based classifiers	41
3.4 Tree-based classifiers	43
4. Explainable artificial intelligence (XAI)	44
5. Summary and conclusion	45
References	47

This chapter provides a general overview of the terms and concepts in machine learning and is written for neurology clinicians who are new in this area. We use the classification of fruit and vegetables as an illustrative example but also explain concepts in the context of movement disorders. Our aim

is to give a basic understanding of machine learning. You will learn to understand technical terms and how to evaluate machine learning results. Hopefully, at the end of this chapter, machine learning will feel less like voodoo magic and more like a cool and useful toolbox from which we can benefit.

We will start by introducing the basic vocabulary of machine learning and the workflow of the learning process. Next, we introduce different machine learning methods. Additionally, the section explainable artificial intelligence addresses the importance of transparency of the systems. We will conclude by summarizing the learning points and where to get more in-depth knowledge about the discussed topics.

This section is to a large extent inspired by the lecture notes “The Shallow and the Deep” (Biehl, 2022), and “Deep Learning” by I. Goodfellow et al. (Goodfellow, Bengio, & Courville, 2016). These two sources cover most topics of this chapter, but we refrain from citing them explicitly unless we point to a specific section. The chapter does not aspire to completeness, rather the focus is on providing a clear and understandable overview of what the authors consider to be an important selection of machine learning vocabulary and fundamentals.



1. Basic vocabulary

Let us start by introducing the basic vocabulary. By the end of this first section, we will understand the meaning and relation between machine learning, statistical modeling, artificial intelligence, neural networks, and deep learning. Besides that, we will be familiar with learning types and tasks in machine learning. In Fig. 1, you can find an overview of the relation between the terms and topics that are covered in this chapter.

1.1 Machine learning, statistical modeling, and artificial intelligence

Machine learning (ML) is closely related to statistical modeling: both aim to extract information from data and identify patterns to explain phenomena in a larger group. Differences can be found in the main motivations: statistical modeling focuses on describing, understanding, and interpreting the data, while machine learning emphasizes on generalizing patterns to new data and making predictions. However, there is no clear cut between the two disciplines: frequently, statistical models can be used to make predictions and there are also machine learning techniques that aim at

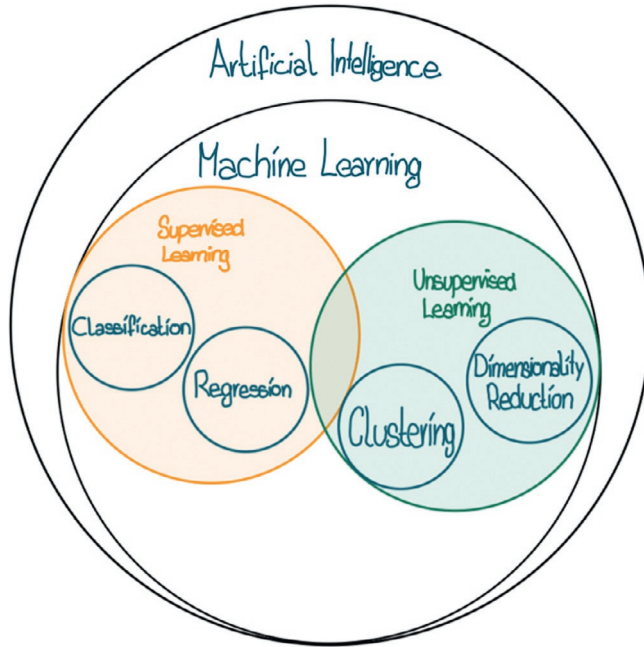


Fig. 1 Overview of terms and topics covered in this chapter. Machine learning is a subdiscipline of artificial intelligence. Supervised and unsupervised learning are two important learning types that can solve learning tasks, such as classification and regression (supervised learning), or clustering and dimensionality reduction (unsupervised learning).

obtaining insights into properties of the data. Moreover, statistical models and machine learning can certainly be applied complementarily.

The terms artificial intelligence (AI) and machine learning are often used synonymously. However, machine learning should be considered a subfield of artificial intelligence, comprising algorithms that can learn from data. A definition for *learning* in this context is provided by Tom M. Mitchell (Mitchell, 1997): “[a] computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.”

Other terms that are commonly confused with artificial intelligence or machine learning are neural networks (NN) and deep learning (DL). In brief, classical machine learning models are typically relatively simple in terms of their structure and their way of processing the data. Neural networks are a type of machine learning systems that have a potentially more complex structure containing layers of trainable elements that combine

many simple transformations to generate a prediction. In the simplest terms, neural networks that have many of such layers are termed deep neural networks and are said to perform deep learning.

1.2 Example

Throughout this chapter, all terms, concepts, and methods will be explained in terms of an imaginary data set representing apples, pears, zucchini, and tomatoes, see Fig. 2. We will refer to this data set as the *food example*.

Additionally, we will consider an imaginary data set of movement disorder phenotypes, such as essential tremor, myoclonus, and dystonia. The data set consists of different data types, such as clinical information, PET images, electromyography (emg) data and accelerometry (acc) data. We will refer to this data set as the *clinical example*.

1.3 Features

The input values for machine learning are called features. These are most frequently numerical values that describe the data. In images, for instance, the features could be pixel grayscale or RGB color values. In the *food example*, we chose to not only use numerical but also descriptive features to make

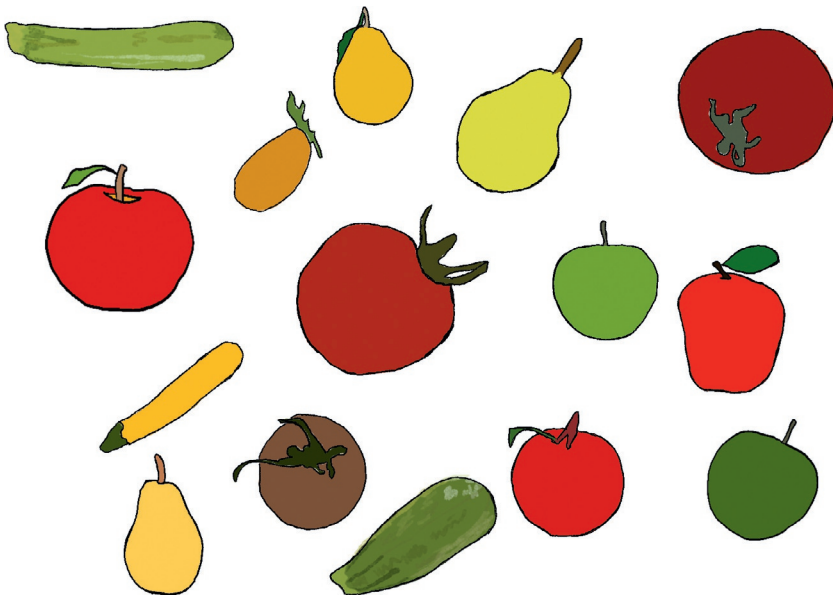


Fig. 2 The data set of the food example.

it more illustrative: the color, shape, height, and width of the food. In practice, such attributes can often also be described mathematically and represented by numerical values. [Table 1](#) shows the food data set with example feature values in columns B through E. In the following, the term sample refers to a single data point, represented by one row of the table.

In the *clinical example* possible features could be the signs and symptoms of the patient, such as age at onset, whether the movement disorders respond to medication, the walking pattern, the affected body parts, the consistency of symptoms during tasks, but also features from additional tests such as clinical neurophysiology.

1.4 Learning types and tasks

Quite generally, the process of finding patterns or structures in the presented data set is called *learning*. We can distinguish two main learning types: *supervised* and *unsupervised* learning.

1.4.1 Supervised learning

In supervised learning, each sample is assigned to a target value, called *label*. The labels can be seen as to represent the ground truth which defines the goal of the learning process. Possible labels for the *food data set* are the type of food, its culinary classification, or the weight. The labels are shown in columns L1 through L3 of [Table 1](#). Possible labels in the *clinical example* could be the phenotype of the movement disorder (i.e., essential tremor, myoclonus, and dystonia), or its severity score.






In the next section, we will look at two important tasks in supervised learning, which are *classification* and *regression*.

1.4.1.1 Classification

In classification, the goal is grouping the samples into predefined classes. Each sample is associated with one of these classes by its label. The task of the algorithm is finding patterns within each class that describe its typical characteristics and its dissimilarities with the other groups. Based on the learned patterns, the classifier predicts the class of a newly presented sample.

When in our food example, the labels are defined as apple, pear, zucchini, and tomato (column L1), the system is trained to classify the samples by food type as shown in [Fig. 3](#). When the labels are fruit and vegetable (column L2), the result would be completely different since the labels define a different target ([Fig. 4](#)).

Table 1 Example features (B–E) and labels (L1–L3) of the food example.

	A	B	C	D	E	L1	L2	L3
Sample-ID	Image	Color	Shape	Height [mm]	Width [mm]	Type	Culinary classification	Weight [g]
S1		Red	Round	43	44	Tomato	Vegetable	56
S2		Red	Round	72	78	Apple	Fruit	123
S3		Orange	Pear-shape	89	53	Pear	Fruit	89
S4		Green	Elongated	277	38	Zucchini	Vegetable	352
...
S15		Red	Round	62	67	Apple	Fruit	89

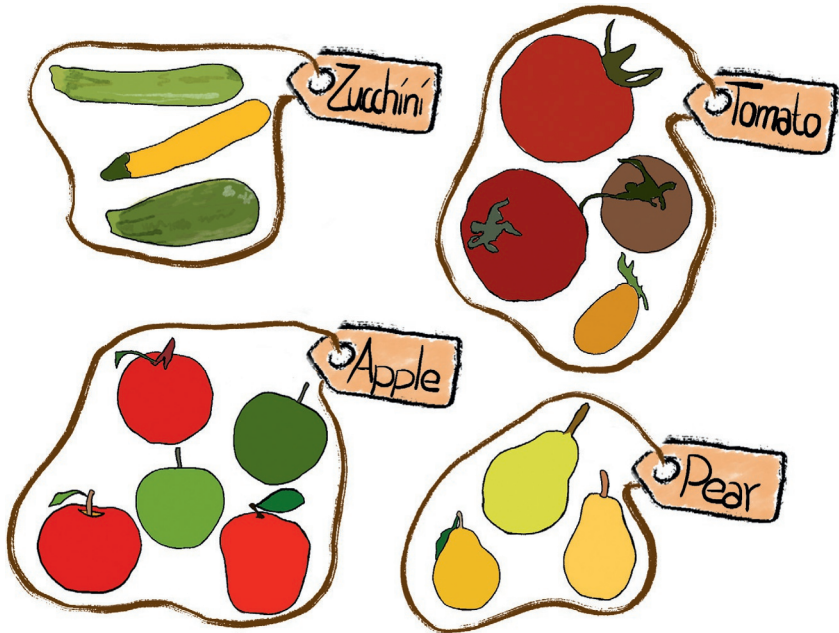


Fig. 3 Classification of the food by its type.

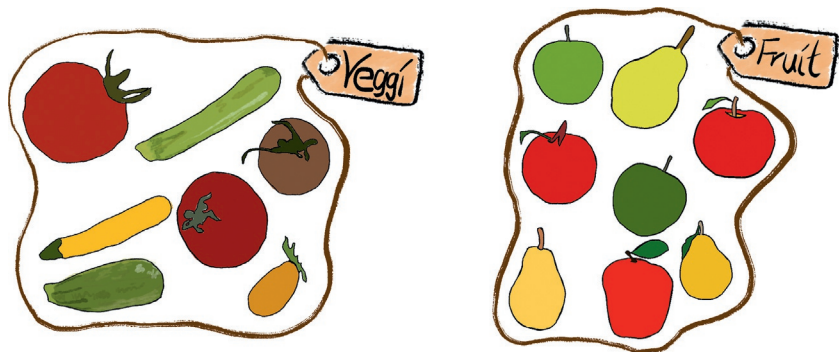


Fig. 4 Classification of the food by its culinary classification.

In the *clinical example*, the classification goal could be phenotyping movement disorders. In this case, the classes and labels could be the phenotypes, e.g., tremor, myoclonus, and dystonia. A classifier is then trained to learn the patterns based on features such as tremor frequency, location of the movement disorder, age of onset or comorbidities. After that, the fully trained classifier may be used to assign a newly presented patient to one of the classes based on the previously learned patterns.

In classification, the labels specify a goal and the set of classes into which to classify the data. This comes with the limitation that any new data that is presented to the classifier will be forced on the trained space: when a classifier is not aware of the existence of cauliflowers, it is not capable of classifying them. Instead, the classifier will try to assign a cauliflower to one of the predefined classes, apple, pear, zucchini, or tomato. In practice, methods have been suggested to circumvent this difficulty, see e.g., (Fischer, Hammer, & Wersing, 2016).

A characteristic aspect of supervised learning is that the labels provide the possibility to measure the performance of the machine learning system. Assuming that the labels are a faithful representation of a ground truth, also a *right* and *wrong* exists. When labeled data points are presented to the machine learning system, the prediction can be compared to the target label, allowing to quantify the performance of the system. Further details and examples of the validation and testing process and strategies are provided in [Section 2](#).

1.4.1.2 Machine learning regression

Another common supervised learning task is called regression. Its main difference from classification is its output type. Instead of assigning the samples to one class of a set of predefined classes, it outputs a numerical value. A possible goal in the *food example* could be estimating the weight of the food based on the shape, height, and width. The corresponding label would then be the weight of the food (column L3 in [Table 1](#)). In the *clinical example*, a possible regression goal could be the severity assessment of the movement disorders on a continuous scale. As with classification, model performance can also be quantified in regression by comparing the predicted value with the target value (see [Section 2](#)).

1.4.1.3 Other supervised learning tasks

Besides classification and regression, other supervised learning tasks exist that can only partially be assigned to these two types. In ordinal regression, for example, the task could be the severity estimation of a movement disorder with a fixed scale from one to five. The numbers could be interpreted as predefined classes, but, at the same time, the target values are numerical and ordered, making the problem similar to a regression task.

1.4.2 Unsupervised learning

In unsupervised learning, the samples are not labeled. Instead, the algorithm is asked to learn properties of the data set and find patterns without guidance

toward a specific, predefined goal. This is useful for understanding, exploring, or preprocessing data sets. Common forms of unsupervised learning are clustering and dimensionality reduction.

1.4.2.1 Clustering

A possible application of unsupervised learning is clustering. In a sense, it is similar to classification, regarding the goal of grouping the samples. However, the difference is that the groups (i.e., classes) are not predefined. Instead, the algorithm seeks similarities and dissimilarities within the data and groups the samples accordingly.

In unsupervised learning, there is no ground truth, no right or wrong and thus also no straightforward performance measure. Getting back to our example, the food could be clustered by the color and shape and thus, groups of apples, pears, and zucchini would form, while the tomatoes arrange close to the apples and pears as shown in [Fig. 5](#). However, if the food is clustered only by color, the result would be completely different: the green apples, pears, zucchini, and tomatoes, the red apples and tomatoes, and the yellow food would group up ([Fig. 6](#)). As there is no predefined goal and hence no direct feedback, both results are neither correct nor incorrect.

Clustering can be a helpful approach to explore and analyze a dataset. In the clinical example, groups of patients who share similar characteristics might be present. This could potentially result in groups of different types of movement disorders without explicitly training the system to distinguish those.

1.4.2.2 Other unsupervised learning tasks

Other unsupervised learning tasks exist, such as dimensionality reduction that reduces the number of variables of the dataset while keeping as much information as possible. A prominent example of dimensionality reduction is principal component analysis (PCA). Projecting data on lower dimensional space, e.g., two dimensions, enables data visualization and thereby allows visual inspection and interpretation of high dimensional data sets ([Bunte, Biehl, & Hammer, 2012](#)).

1.4.3 Other learning types

Supervised and unsupervised learning are the two main learning types in machine learning. However, more learning types and variants of these two types exist. Semi-supervised learning ([Chapelle, Schölkopf, & Zien, 2006](#)) for instance, can handle partially labeled data. This can be helpful

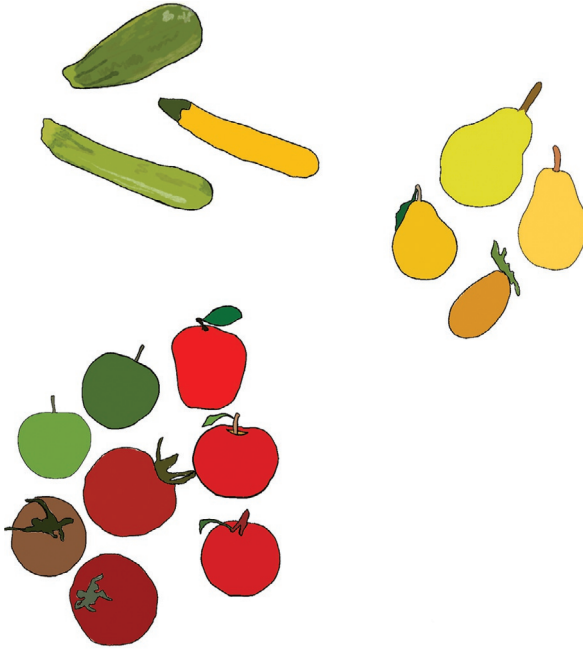


Fig. 5 Clustering by color and shape.

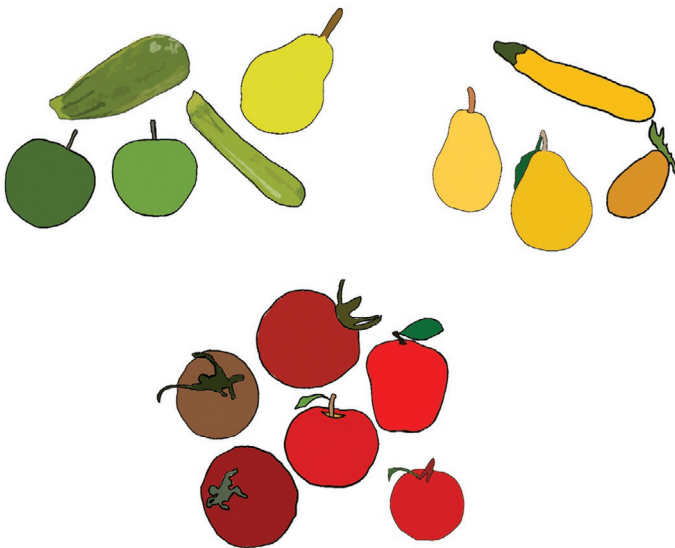


Fig. 6 Clustering by color only.

in the medical field, e.g., when part of the data is missing a label because the patients do not have a clear diagnosis (yet).

In this first section, we learned that machine learning aims to learn from training data and make predictions on novel data. We distinguished several learning types; the two main types being supervised and unsupervised learning. Applications of supervised learning are classification and regression, while typical applications of unsupervised learning are dimensionality reduction, visualization, preprocessing, or clustering. Given the scope of this book, the remainder of this chapter focuses on supervised classification problems.



2. Generalization in supervised learning

Machine learning aims to extract information from data and identify patterns (learning). In the subsequent generalization step, the identified patterns are applied to novel data in order to make predictions. This section addresses the workflow of the learning process and generalization in supervised learning. Additionally, it briefly introduces validation strategies to estimate the performance of the system.

The process usually undergoes at least three phases: training, working, and validation phase. In the training phase, the system extracts information from the data and learns rules to solve a task. In the working phase, the rules are applied to novel data in order to make predictions. Frequently, a validation phase is inserted, in which the expected working phase performance is estimated. Potentially, an additional testing phase is added to this workflow. For the sake of clarity, the following explanation is presented in terms of classification only. Analogous concepts can be applied in the context of regression.

2.1 Phases in supervised learning

Let us walk through the four phases with the *food example*. For simplicity, we will only consider a part of the data comprising pears and zucchini. Of course, in a real setting, the data set would have to comprise more samples in order to generalize properly on novel observations.

First, the data set is usually divided into a training set and a validation set, as shown in [Fig. 7](#). More details about methods that are suitable for splitting the data can be found in [Section 2.2](#).

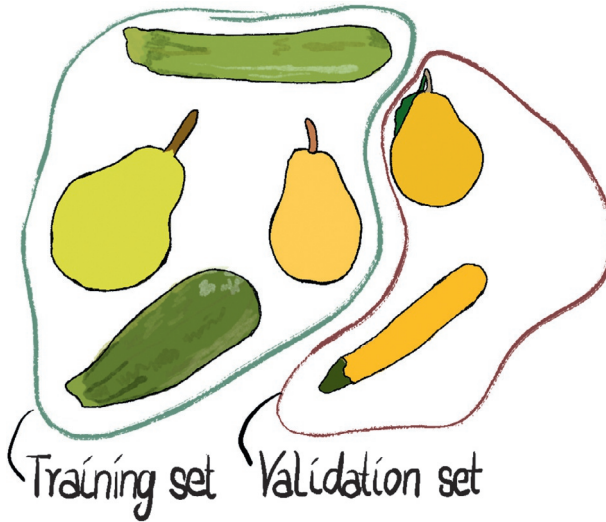


Fig. 7 Dividing the data into a training set and a validation set.

2.1.1 Training

During the training phase, the training set is presented to the system to approximate a target rule, function, or model that solves the desired task. To find a suitable approximation, the system adapts its parameters according to the presented example data. More detailed information about possible learning methods is provided in [Section 3](#).

Let us illustrate this on a very simplified level in our *food example* and classify pears and zucchini (column L1 in [Table 1](#)). The conceptual idea of the training process is visualized in [Fig. 8](#). During the training, the features of the training set and the corresponding labels are presented to the model to approximate a target rule. In our example, the system learns that all pears are yellowish, and all zucchini are green. It thus assumes that the food can be distinguished based on the feature “color.”

In the clinical example, the goal could be to distinguish phenotypes of movement disorders. In this scenario, the training set is presented to the system and patterns that are typical for the phenotypes are learned. The classifier could for example learn, that essential tremor patients are shaking with a very consistent tremor frequency in postural and dynamic tasks, but the tremor disappears in rest with full relaxation. This so-called activation pattern is different in patients with Parkinson’s disease, in whom tremor is most severe at rest.

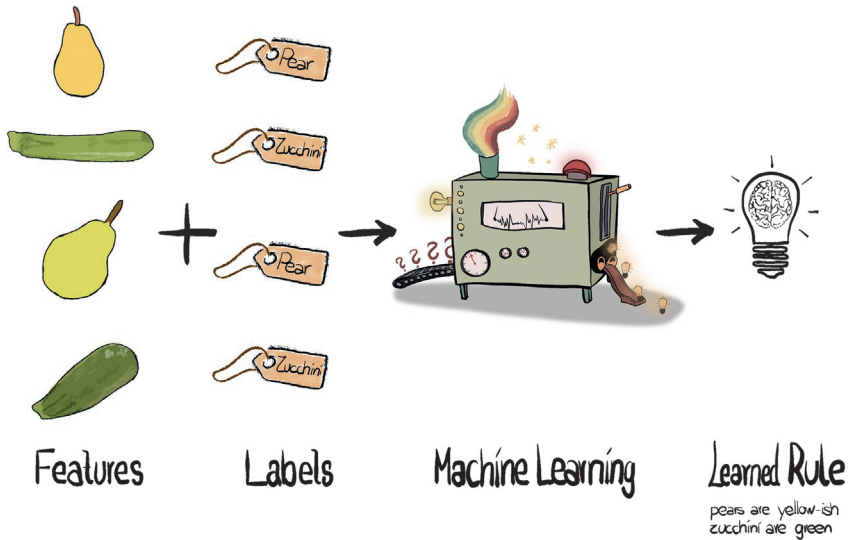


Fig. 8 Conceptual idea of the training process.

2.1.2 Validation

The goal of the validation phase is estimating the model's generalization ability. This is usually done by comparing the predicted values to the real values (labels) of the validation set as follows.

Previously, we learned that the data set is usually divided into a training and a validation set and while the training set is used to fit the model, the validation set is left out. Therefore, it was not used to fit the model and can provide an estimate of the expected performance on novel data.

In the validation phase, the model is applied to the validation set and predicts the class membership for each sample. In supervised learning, we do know the correct classes of the samples as each sample comes with a (hopefully faithful) label. Therefore, we have the possibility to compare the predicted class of a sample to its label and verify whether a sample was classified correctly. If we do this for many samples, we can estimate the generalization ability of the model by calculating, for example, the overall accuracy as: $accuracy = \frac{\text{correct classifications}}{\text{all classifications}}$.

In a two-class problem, where there is a positive (+1) and a negative (-1) class, we speak about a *true positive* when both, the predicted class and the real label of a sample are positive (1). Accordingly, a *false positive* is a sample that is predicted as belonging to the positive class (1), but its real label is negative (-1). The corresponding rates can be calculated as the true positive rate

($tpr = \frac{\text{true positives}}{\text{false negatives} + \text{true positives}}$), which in medicine is also called the *sensitivity*, or the *false positive rate* ($fpr = \frac{\text{false positives}}{\text{false positives} + \text{true negatives}}$), which corresponds to (1-specificity). The same holds for *true negatives* and *false negatives*.

In multiclass problems, a confusion matrix can help to identify the class-specific performance (Table 2). In an n-class problem, the confusion matrix is an (n × n) matrix, where the row indices represent the true class labels and the column indices the predicted classes. Accordingly, the diagonal elements capture the correctly classified samples, while the off-diagonal elements show misclassifications.

The conceptual idea of the validation process is shown in Fig. 9. The food classifier applies the learned rule “all pears are yellow-ish and all zucchini are green” to the validation set. Since our example classifier has

Table 2 Confusion matrix of an imaginary 4-class problem classifying apples, pears, zucchini, and tomatoes.

		Predicted class			
		Apple	Pear	Zucchini	Tomato
True class	Apple	17	8	0	2
	Pear	6	23	0	1
	Zucchini	0	0	11	0
	Tomato	3	0	0	6

The diagonal elements show the correctly classified samples while the off-diagonal elements capture misclassifications.

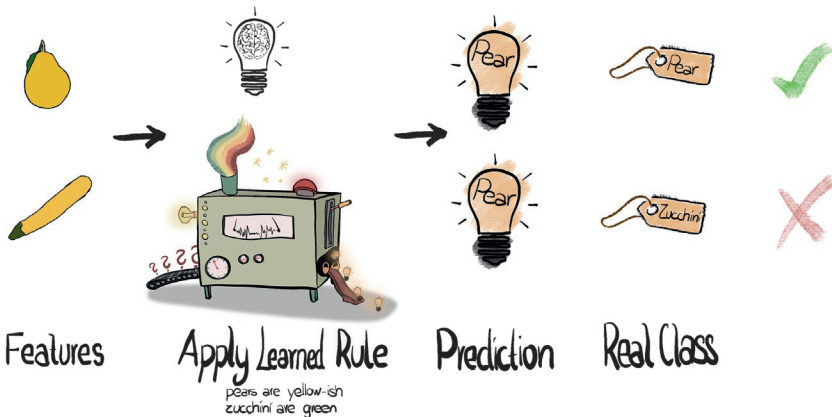


Fig. 9 Conceptual idea of the validation process.

learned to distinguish food by its color, it would classify both samples as “pear” because their color is yellow. The predicted labels are then compared with the real labels “pear,” and “zucchini,” showing that one of the two validation samples was classified correctly. Evaluating multiple samples provides the possibility to measure the performance of the system on a desired task. The training and validation phases can be repeated, and the model parameters adjusted accordingly until a suitable approximation of the target rule is found.

In the clinical example, the validation process operates in a similar fashion: the rule learned from the training set is employed on the validation set, which was previously not used for training, and the individuals are assigned to the appropriate classes. Afterwards, the predicted labels are compared to the actual labels in order to evaluate the system’s overall performance.

2.1.3 Testing

Testing is closely related to validation, in terms of its goal of estimating the model performance on novel data. The difference can be found in the data that is used for testing. The so-called test set must be independent from the training and validation set and the samples from the test set must not be used for parameter adaption of the model. Its purpose is to test if the model indeed can be generalized to novel data and does not just perform well on the data that had been used to build and validate the model.

2.1.4 Working phase

In the working phase, the fully trained classifier is applied to novel data and predicts target values (labels) accordingly.

2.2 Validation strategies

In real world scenarios, the number of available samples is limited, and especially in medical applications frequently quite small. Splitting the available data into a training set and a validation set only once would not result in a reliable performance estimate. Therefore, this process is often repeated several times with different training and validation sets. Two example techniques that allow repeated training and validation with a limited number of samples are k-fold cross validation and, as an extreme case, leave-one-out.

2.2.1 *K-fold cross validation*

K-fold cross validation is a technique of creating several training and validation sets from a limited number of samples. The data set is divided into k groups of approximately equal size. Then, $k-1$ of the groups form the training set and the left out group the validation set. This procedure is repeated k times until each group had been used as validation set once. The generalization ability of the model can be estimated by analyzing the k validation runs.

In a clinical example, we could think of a data set that comprises 50 patients. In 10-fold cross validation, this data set is divided into 10 groups of 5 patients each. Then, the system is trained on 9 groups (45 patients) and validated on the left-out group (5 patients). This can be repeated until each group has been left out once for validation.

2.2.2 *Leave-one-out*

Leave-one-out is the extreme case of k -fold cross validation where the number of k equals the size of the data set. Hence, each group contains exactly one sample and at each run, the performance is validated based on this single sample. A variation of the leave-one-out is to leave one sample out from each class to keep the classes balanced.



3. Classification methods

In the previous section, we got familiar with the learning and generalization process of supervised learning and introduced a selection of validation strategies to estimate a model's performance. In this section, we will have a closer look on the model itself. We start by introducing the general idea of linear classifiers and methods that can identify suitable decision boundaries between classes. Next, we will learn about neural networks and understand what makes them so powerful before moving on to the very intuitive frameworks of distance-based classifiers and tree-based classifiers.

3.1 Linear classifiers

Linear classifiers identify linear decision boundaries between classes of data points in an n -dimensional space. Let us illustrate the meaning of this sentence in a simple scenario: The task is to solve a two-class problem, e.g., classifying food in the two classes apples and zucchini. The input space is two-dimensional ($n = 2$), meaning, that for each sample, two features are presented, e.g., the height and the width of the food.

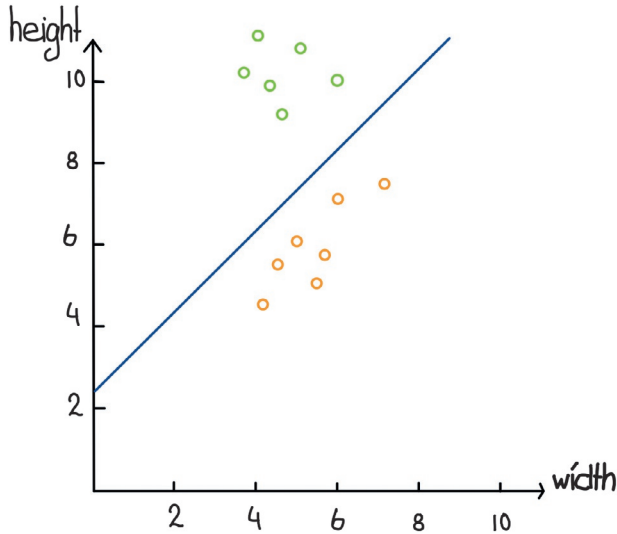


Fig. 10 Projection of apples and zucchini in a two-dimensional feature space with a decision boundary separating the two types.

Projecting the datapoints into the two-dimensional space suggests a grouping of apples and zucchini according to the ratio between width and height of the food (Fig. 10). The two groups can easily be separated by a simple linear function of the form $w_1 \cdot \text{width} + w_2 \cdot \text{height} - \theta = 0$. Inserting the values of the samples into the left side of the equation, the result will be zero if it is exactly on the separating line, positive if it is above it, and negative if it is below it. Classifying a new datapoint works accordingly: computing the expression on the left side and assign it to the positive (zucchini) or negative (apple) class accordingly.

Linear classifiers can also seek decision boundaries in higher dimensional spaces, and thus larger feature sets. In a three-dimensional case, the decision boundary would be a plane, and in higher dimensions a hyperplane. Mathematically, the classification can be expressed as $s = \text{sign}(w_1x_1 + w_2x_2 + \dots + w_nx_n - \theta)$ where the weights $\mathbf{w} = (w_1, w_2, \dots, w_n)$ and the threshold θ define the decision boundary, and $s = \pm 1$ indicates the response of the classifier. During the training phase, linear classifiers learn which weights are appropriate to define a suitable decision boundary for the classification problem. One possibility of learning suitable weights is applying the perceptron algorithm (Kubat, 2017; Rosenblatt, 1958).

3.1.1 Perceptron

The perceptron is an algorithm that learns the weights of a suitable decision boundary from the training data. Each sample is presented separately one after the other and classified based on the current decision boundary. Then, the predicted class and the true class (label) are compared. If the sample is not classified correctly, the weights of the current decision boundary are adapted accordingly. These steps are repeated for all samples in the training set. When the whole training set had been presented to the system, the first epoch is finished. Usually, several epochs are performed until all samples are classified correctly. If the data is linearly separable, the perceptron will find the solution in a finite number of epochs. Proofs for this fundamental result can be found in [Minsky and Papert \(2017\)](#).

3.1.2 Support vector machines (SVM)

[Fig. 10](#) shows one solution of a decision boundary that separates the two groups. In fact, there can be many solutions for a linear decision boundary that differ in their *margins*. The margin is the distance between the decision boundary and the closest input vectors. However, the decision boundary with the largest margin, and thus the greatest distance to the input vectors is most likely to perform well on future data ([Kubat, 2017](#)).

The input vectors that are the closest to the decision boundary have the strongest direct influence on its properties. One important aspect of the training process of support vector machines is determining these so-called *support vectors*. Appropriate training algorithms exist that yield the optimum margin and identify these support vectors.

A major advantage of the support vector machine approach is the capability of solving classification problems that are not linearly separable in terms of the original features. This can be achieved by applying the elegant *kernel trick*. The so-called kernel function implicitly projects the data to a higher dimensional space in a non-linear fashion. Then the optimal hyperplane is identified that separates the data in the higher dimensional space with optimal margin. In practice, the selection of a suitable kernel function for error-free classification or the extension of the formalism to tolerating errors is essential ([Schoelkopf & Smola, 2002](#)).

3.2 Neural networks (NN)

Neural networks are motivated by the idea of solving more complex tasks by combining many computing elements in a network. These elements are called nodes, units, or neurons, as they are loosely inspired by neurons in

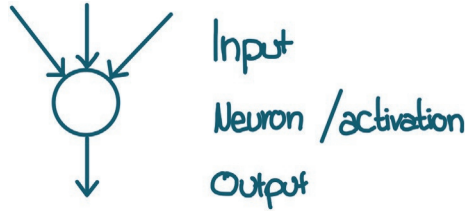


Fig. 11 Single artificial neuron.

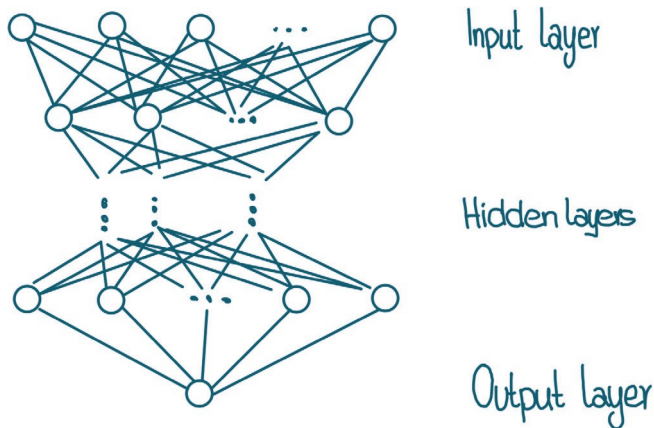


Fig. 12 Layered architecture of a neural network.

the brain. This is because they work as neurons in the way that they receive input from other units, apply some arithmetic operation to determine its activation, e.g., as a function of the weighted sum of inputs, and return an output signal (Fig. 11).

Besides that, the units are interconnected, similar to the way neurons are connected in the brain. Frequently, the connections are organized in a layered architecture as shown in Fig. 12: an input layer receives input data from the *outside world* and passes it through to one or more hidden layers. The neurons in the hidden layer perform some transformation to their individual inputs, before passing it on to an output layer that represents the network output, i.e., the response of the system to a specific input.

In feedforward networks, the neurons receive input only from units in the previous layer. The connections between the layers can be organized according to several patterns. For example, in fully connected layers, every neuron of a layer is connected to every unit in the previous layer, as it is the case in Fig. 12.

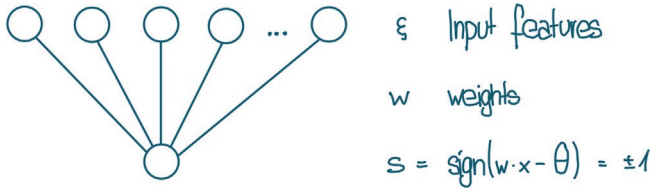


Fig. 13 Single layer perceptron.

All connections are characterized by weights, i.e., real numbers which determine the strength of the connection. In the learning or training process, these weights are adjusted in a similar way to the previously discussed perceptron. In fact, the perceptron can even be seen as a very simple neural network with just one layer (and no hidden layer) as shown in Fig. 13: The features represent the input layer that is connected to the output layer. The weights of the connections represent the weights of the decision boundary, and the arithmetic operation (activation function) of the output layer is the sign function that transforms all inputs (w , x , θ) to the binary output (label) ± 1 .

Combining several perceptrons into a more complex network can make the classifier more powerful. In such a network, the neurons are represented by perceptrons and the output of the network results from a weighted combination of the perceptrons. In practice, however, the neurons usually implement continuous, differentiable activation functions rather than simple thresholds units.

The architecture of neural networks can be very complex by combining many units with relatively simple individual activations, allowing it to identify and learn non-trivial patterns in the data. Neural networks with many hidden layers are, on a very simplified level, called deep neural networks. Accordingly, deep learning can be considered a variant of neural networks that have many layers. In *Deep Learning*, Goodfellow et al. state that “[there is no] consensus about how much depth a model requires to qualify as “deep.” However, deep learning can safely be regarded as the study of models that either involve a greater amount of composition of learned functions or learned concepts than traditional machine learning does. [...] Deep learning is a particular kind of machine learning that achieves great power and flexibility by learning to represent a world as a nested hierarchy of concepts, with each concept defined in relation to simpler concepts, and more abstract representations computed in terms of less abstract ones.”

3.3 Distance-based classifiers

In the particularly transparent and intuitive framework of distance-based classification, novel observations are directly compared with previously stored data. To this end, a suitable measure of similarity or distance has to be identified and employed. Frequently, the comparison is based on a few typical representatives of previously observed data, so-called exemplars, or prototypes (Biehl, Hammer, & Villmann, 2016).

3.3.1 *K*-nearest neighbors

K-nearest neighbors can be used for both classification and regression tasks. This method does not require any actual training. Instead, it simply stores the training samples and corresponding labels. In the working phase, the *k* nearest training samples (neighbors) are determined to generate the output for a newly presented data point. This could be the average value of the corresponding neighbors in a regression task, or the majority class in a classification problem. Fig. 14 shows an illustrative example of a *k*-nearest neighbor classifier with three classes. The gray dots represent novel data points that are classified to one of the three classes according to their five nearest neighbors ($k=5$).

3.3.2 *Learning vector quantization (LVQ)*

Learning vector quantization (LVQ) is a family of prototype and distance-based classifiers that is introduced by Kohonen (Kohonen, 1990). Instead of

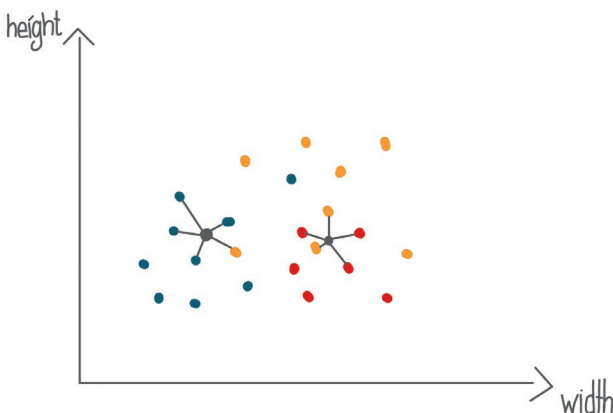


Fig. 14 Illustration of a *k*-nearest neighbor classifier. The yellow, blue, and red dots represent the training samples that belong to three classes according to their color. The gray dots represent novel data points that are classified according to their five nearest neighbors ($k=5$).

using all individual datapoints as in k -nearest neighbors, a few representatives of the classes are determined. These so-called prototypes are learned during the training phase. After initializing prototypes (e.g., randomly), a datapoint is presented to the system and the distance (e.g., Euclidean distance) to each prototype is calculated. In many LVQ variants, the closest prototype with the same label is then moved closer to the presented data point, while the closest prototype with a different label is pushed further away. This process continues until the prototypes are considered suitable representatives for the classes. In the working phase, a new datapoint is presented to the classifier and assigned to the class of the closest prototype, called nearest prototype classification (NPC). The distance can be seen as similarity measure between the presented subject and the prototype. An illustrative example of a prototype-based classifier is shown in Fig. 15.

While several variants of LVQ exist, one important group of extensions incorporates weighting coefficients into the distance measure that quantify the contribution of both single features and combinations of features. The adaption and optimization of these weights are an integral part of the training process. As the weights additionally provide us with information about the significance of the features in the classification process, they help to understand which features are important for the distinction and might help to detect biases. An example for a variant of LVQ that includes such an adaptive distance measure is Generalized Matrix Learning Vector Quantization

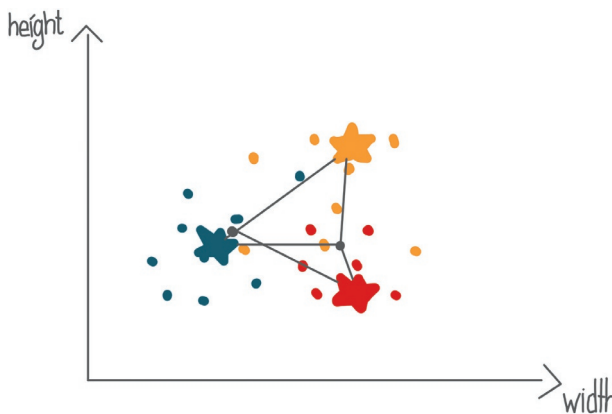


Fig. 15 Illustration of a prototype-based classifier. The yellow, blue, and red dots represent the training set consisting of three classes. The stars are the prototypes that represent the classes, and the gray dots are novel samples, that are classified according to their distance to the prototypes.

(GMLVQ), introduced and extended in (Biehl et al., 2016; Bunte et al., 2012; Schneider, Biehl, & Hammer, 2009; Schneider et al., 2010). For examples of applications in the medical domain, we recommend (van Veen, 2022) and (Biehl, 2017).

3.4 Tree-based classifiers

3.4.1 Decision trees

Decision trees are a simple and easy to interpret method for regression and classification tasks. They consist of a set of splitting rules that can be summarized in a tree-like structure as shown in Fig. 16: The nodes represent the input values (features), the edges are associated with defined splitting rules and end nodes (leaves) are assigned to a class label. Classifying a new sample, starts at the root of the tree, which is the topmost node. Based on the input value of the node, the edges lead the example along a certain path based on its input values until it reaches a leaf node that assigns it to the corresponding label. A strong advantage of decision trees is their interpretability. It allows tracing the splitting rules and provides precise explanation for the decision making (Kubat, 2017).

3.4.2 Random forests

However, the best supervised learning approaches typically outperform single decision trees in terms of prediction accuracy. A great improvement in prediction accuracy can be achieved by combining large numbers of trees and combine the results from all trees to conclude the final prediction as in random forests. The individual trees are trained on different subsets of

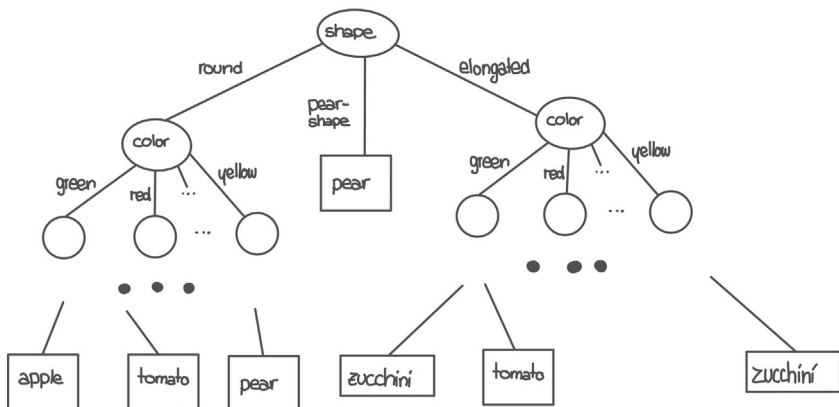


Fig. 16 Illustrative example of a decision tree.

the data to ensure that the random forest is composed of different trees (James, Witten, Hastie, & Tibshirani, 2013).

Moreover, random forests can also reduce the risk of overfitting. We speak about overfitting, when a model becomes too specific to the training data and might not generalize well on novel data. The risk of overfitting is reduced in random forests due to the randomness of the subsets and features that the trees are trained on.

A drawback is that the interpretability of random forest is more difficult compared to single decision trees. However, random forests provide importance measures of the significance of the features that nevertheless allow interpretation of the model.



4. Explainable artificial intelligence (XAI)

In the previous section, we got familiar with different types of classifiers including linear classifiers, neural networks, distance based and tree based classifiers, and learned about a selection of classification methods (SVM, KNN, LVQ, RF). This next section focuses on explainable artificial intelligence (XAI) which is a subdiscipline of machine learning. XAI is motivated by the idea that machine learning systems are more likely to be trusted if it is possible to understand and interpret its decisions (at least to a certain degree). We will start by understanding the importance of interpretability in machine learning before giving examples for methods that allow some transparency and interpretation possibilities.

In the medical field, where our decisions can have a major impact on a person's life, we usually critically examine whether we can trust the tools at our disposal. However, machine learning systems can certainly have drawbacks. For instance, they are very good artifact detectors and decisions can be based on subtle or hidden biases in the data.

We could think of the background of the photo in our food example that could be blurred for all tomatoes or that all apples are alighted from the right side. The classifier might then perfectly work on the dataset it is trained on, even though it has not learned any pattern of the food itself.

For biases in the medical field, one can think of the age or gender distribution among the classes. Moreover, the classification can be based on batch effects resulting from various pre-processing pipelines, from the use of different machines, or from the collection of the data at different institutes.

Especially in medical applications, clinicians are not only interested in the final classification result, but also in the reasons that lead to the decision or

the degree to which the system is certain about it. A lack of interpretability of so-called black-box systems and, particularly in deep learning, high model complexity can negatively impact acceptance. The subdiscipline of machine learning that is called explainable artificial intelligence addresses these needs by expanding the focus on performance perfection (Arrieta et al., 2020; Ghosh, 2021).

A variety of so-called white-box methods are more transparent and interpretable for humans. Example methods are, among others, random forest, or GMLVQ. Those methods belong to the subdiscipline of relevance learning, since they provide information about the significance of the features and thus indicate the extent to which the features contribute to the decision making. As relevance learning allows the inspection of the features' importance, it can help to uncover possible biases in the data.

When applying relevance learning to train a classifier that distinguishes food types, it offers the opportunity to inspect the relevance profile of the features. If the classifier would classify the tomatoes based on the background, the relevance profile would indicate that the feature “background color” is the most important marker. But if it classifies the tomatoes based on the shape, and color, and the feature relevances seem comprehensible, it will increase our trust in the system.

Moreover, prototype-based classifiers learn representatives of the classes that are defined in the same dataspace as the datapoints themselves, meaning that if the input data are e.g., images, the prototypes are images as well. Inspecting the prototypes can give insights into properties of the classes and structure of the dataset.

In the clinical example, one could train a prototype-based classifier on PET scan images to distinguish movement disorders. The prototypes will then represent phenotypical PET images of the brain and can serve as a reference to gain insights into the properties of classes, detect biases, and facilitate discussion.



5. Summary and conclusion

The purpose of this chapter was introducing movement disorder specialists to machine learning and creating a fundamental understanding of important terms and methods. By its end, we are familiar with the basic vocabulary and know that machine learning is a part of artificial intelligence where the system learns from sample data, and that the input to machine learning are usually numerical values, called features.

Furthermore, we now know that we can distinguish two main learning types which are supervised and unsupervised learning. In supervised learning, the samples are paired with labels that provide a target value, defining the goal and ground truth of the task. Such a task could, besides others, be regression (estimating a numerical value such as the weight of an apple) or classification (estimating the affiliation of the sample to one of a set of predefined classes).

Besides the basic vocabulary, we now have an idea about the phases in supervised machine learning: training, working, validation, and testing phase. In the training phase, an approximation of the target rule is learned and in the working phase the learned rule is applied to novel data. To evaluate how well the model can solve the desired task, we usually add a validation and testing phase where the values predicted by the system are compared to the correct values. The main difference between validation and testing lies in the data that we are using: The validation set is formed by splitting the data set into a training and validation set, such that a different part of the data set is held out for validation on each run. There are suitable techniques that can be applied to efficiently split the data set. The test set on the other hand, consists of data that had never been used to train the model. This way, we test the model's generalization ability on completely novel, previously unseen data.

Besides that, we were introduced to a number of classification methods: starting with linear classifiers that identify linear decision boundaries between data points, before getting to neural networks that combine many computing elements and therefore can solve more complex tasks which unfortunately is on the expend of transparency. However, there are very powerful methods that are more transparent and that do offer interpretation possibilities. Example methods are distance-based classifiers like GMLVQ and tree-based classifiers like random forest. The discipline that focuses on interpretability and transparency in machine learning is called explainable artificial intelligence (XAI) and is particularly in the medical domain important to uncover biases, increase the trust into the system and facilitate discussions with domain experts.

Readers who are interested in more detailed and in-depth introductions to machine learning are advised to consider the texts “The Shallow and the Deep” (Biehl, 2022), and “Deep Learning” by I. Goodfellow et al. (Goodfellow et al., 2016) or other textbooks and reviews like “The elements of statistical learning” by Hastie et al. (Hastie, Tibshirani, & Friedman, 2009), or “Pattern recognition and machine learning” of Bishop (Bishop, 2006).

References

- Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Sijam, T., Alberto, B., et al. (2020). Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58, 82–115.
- Biehl, M. (2017). Biomedical applications of prototype based classifiers and relevance learning. In *International conference on algorithms for computational biology* (pp. 3–23). Springer.
- Biehl, M. (2022). *The shallow and the deep: A biased introduction to neural networks and old school machine learning*. University of Groningen.
- Biehl, M., Hammer, B., & Villmann, T. (2016). Prototype-based models in machine learning. *Wiley Interdisciplinary Reviews: Cognitive Science*, 7(2), 92–111.
- Bishop, C. (2006). *Pattern recognition and machine learning*. Springer.
- Bunte, K., Biehl, M., & Hammer, B. (2012). A general framework for dimensionality-reducing data visualization mapping. *Neural Computation*, 24(3), 771–804.
- Bunte, K., Schneider, P., Hammer, B., Schleif, F.-M., Villmann, T., & Biehl, M. (2012). Limited rank matrix learning, discriminative dimension reduction and visualization. *Neural Networks*, 26, 159–173.
- Chapelle, O., Schölkopf, B., & Zien, A. (2006). *Semi-supervised learning*. Cambridge, MA: MIT Press.
- Fischer, L., Hammer, B., & Wersing, H. (2016). Optimal local rejection for classifiers. *Neurocomputing*, 214, 445–457.
- Ghosh, S. (2021). *Intrinsically interpretable machine learning in computer aided diagnosis* (PhD thesis). University of Groningen. <https://doi.org/10.33612/diss.175627883>.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. Cambridge, MA: MIT Press.
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction. *Springer Series in Statistics: Vol. 2* (2 ed.). New York, NY: Springer.
- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning (Vol. 112)*. Springer.
- Kohonen, T. (1990). Improved versions of learning vector quantization. In *Proc. international joint conference on neural networks (IJCNN): Vol. 1* (pp. 545–550). IEEE.
- Kubat, M. (2017). *An introduction to machine learning (Vol. 2)*. Springer.
- Minsky, M., & Papert, S. (2017). *Perceptrons: An introduction to computational geometry*. Cambridge, MA: MIT press.
- Mitchell, T. M. (1997). *Machine learning (Vol. 1)*. New York, NY: McGraw-Hill.
- Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), 386–408.
- Schneider, P., Biehl, M., & Hammer, B. (2009). Adaptive relevance matrices in learning vector quantization. *Neural Computation*, 21, 3532–3561.
- Schneider, P., Bunte, K., Stiekema, H., Hammer, B., Villmann, T., & Biehl, M. (2010). Regularization in matrix relevance learning. *IEEE Transactions on Neural Networks*, 21, 831–840.
- Schoelkopf, B., & Smola, A. (2002). *Learning with kernels. Adaptive computation and machine learning*. Cambridge, MA: MIT Press.
- van Veen, R. (2022). *Learning vector quantization with applications in neuroimaging and biomedicine* (PhD thesis). University of Groningen. <https://doi.org/10.33612/diss.211419033>.