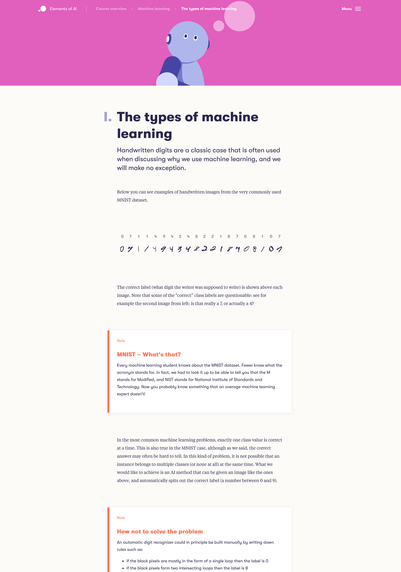


I. The types of machine learning

Handwritten digits are a classic case that is often used when discussing why we use *machine learning*, and we will make no exception.

Below you can see examples of handwritten images from the very commonly used MNIST dataset.

The correct label (what digit the writer was supposed to write) is shown above each image. Note that some of the "correct" class labels are questionable: see for example the second image from left. Is that really a 1 or actually a 4?

Note

MNIST — What's that?

Every machine learning student knows about the MNIST dataset. Fewer know what the acronym stands for. In fact, we had to look it up to be able to tell you that the M stands for Modified, and NIST stands for National Institute of Standards and Technology. Now you probably know something that an average machine learning expert doesn't!

In the most common machine learning problems, exactly one class value is correct at a time. This is also true in the MNIST case, although as we said, the correct answer may often be hard to tell. In this kind of problem, it is not possible that an instance belongs to multiple classes (or none at all) at the same time. What we would like to achieve is an AI method that can be given an image like the ones above, and automatically spits out the correct label (a number between 0 and 9).

Note

How not to solve the problem

An automatic digit recognizer could in principle be built manually, by writing down rules such as:

- If the black pixels are mostly in the form of a single loop then the label is 0
- If the black pixels form two intersecting loops then the label is 8
- If the black pixels are mostly in a straight vertical line in the middle of the figure then the label is 1

and so on...

This was how AI methods were mostly developed in the 1980s (so called "expert systems"). However, even for such a simple task as digit recognition, the task of writing such rules is very laborious. In fact, the above example rules wouldn't be specific enough to be implemented, by programming – we'd have to define precisely what we mean by "mostly", "loop", "line", "middle", and so on.

And even if we did all this work, the result would likely be a bad AI method because as you can see, the handwritten digits are often a bit so-and-so, and every rule would need a dozen exceptions.

Three types of machine learning

The roots of machine learning are in statistics, which can also be thought of as the art of **extracting knowledge from data**. Especially methods such as linear regression and Bayesian statistics, which are both already more than two centuries old (!), are even today at the heart of machine learning. For more examples and a brief history, see the timeline of machine learning (Wikipedia).

The area of machine learning is often divided in subareas according to the kinds of problems being attacked. A rough categorization is as follows:

Supervised learning: We are given an input, for example a photograph with a traffic sign, and the task is to predict the correct output or label, for example which traffic sign is in the picture (speed limit, stop sign, etc.). In the simplest cases, the answers are in the form of yes/no (so called *binary classification problems*).

Unsupervised learning: There are no labels or correct outputs. The task is to discover the structure of the data: for example, grouping similar items to form "clusters", or reducing the data to a small number of important "dimensions". Data visualization can also be considered unsupervised learning.

Reinforcement learning: Commonly used in situations where an AI agent like a self-driving car must operate in an environment and where feedback about good or bad choices is available with some delay. Also used in games where the outcome may be decided only at the end of the game.

The categories are somewhat overlapping and fuzzy, so a particular method can sometimes be hard to place in one category. For example, as the name suggests, so-called **semisupervised learning** is partly supervised and partly unsupervised.

Note

Classification

When it comes to machine learning, we will focus primarily on supervised learning, and in particular, classification tasks. In classification, we observe *n* input, such as a photograph of a traffic sign, and try to infer its "class", such as the type of sign (speed limit 80 km/h, pedestrian crossing, stop sign, etc). Other examples of classification tasks include identification of fake Twitter accounts (input includes the list of followers, and the role at which they have started following the account, and the class is either fake or real account) and handwritten digit recognition (input is an image, class is 0,...,9).

Humans teaching machines: supervised learning

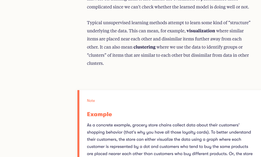
Instead of manually writing down exact rules to do the classification, the point in supervised machine learning is to take a number of examples, label each one by the correct label, and use them to "train" an AI method to automatically recognize the correct label for the training examples as well as (at least hopefully) other unseen images. This of course requires that the correct labels are provided, which is why we talk about supervised learning. The user who provides the correct labels is a supervisor who guides the learning algorithm towards correct answers so that eventually, the algorithm can independently produce them.

In addition to learning how to predict the correct label in a classification problem, supervised learning can also be used in situations where the predicted outcome is a number. Examples include predicting the number of people who will click a Google ad based on the ad content and data about the user's prior online behavior, predicting the number of traffic accidents based on road conditions and speed limit, or predicting the selling price of real estate based on its location, size, and condition. These problems are called *regression*. You probably recognize the term *linear regression*, which is a classical, still very popular technique for regression.

Note

Example

Suppose we have a data set consisting of apartment sales data. For each purchase, we would obviously have the price that was paid, together with the size of the apartment in square meters (or square feet, if you like), and the number of bedrooms, the year of construction, the condition (on a scale from "disaster" to "spick and span"). We could then use machine learning to train a regression model that predicts the selling price based on these features. See a real-life example here



Caveat: careful with that machine learning algorithm

There are a couple potential mistakes that we'd like to make you aware of. They are related to the fact that unless you are careful with the way you apply machine learning methods, you could become too confident about the accuracy of your predictions, and be heavily disappointed when the accuracy turns out to be worse than expected.

The first thing to keep in mind in order to avoid big mistakes, is to split your data set into two parts: the **training data** and the **test data**. We first train the algorithm using only the training data. This gives us a model or a rule that predicts the output based on the input variables.

To assess how well we can actually predict the outputs, we can't count on the training data. While a model may be a very good predictor in the training data, it is no proof that it can **generalize** to any other data. This is where the test data comes in handy: we can apply the trained model to predict the outputs for the test data and compare the predictions to the actual outputs (for example, future apartment sale prices).

Note

Too fit to be true! Overfitting alert

It is very important to keep in mind that the accuracy of a predictor learned by machine learning can be quite different in the training data and in separate test data. This is the so-called **overfitting** phenomenon, and a lot of machine learning research is focused on avoiding it one way or another. Intuitively, overfitting means trying to be too smart. When predicting the success of a new song by a known artist, you can look at the track record of the artist's earlier songs, and come up with a rule like "If the song is about love, and includes a catchy chorus, it will be top-20". However, maybe there are two love songs with catchy choruses that didn't make the top-20, so you decide to continue the rule "...except if Sweden or yoga are mentioned" to improve your rule. This could make your rule fit the past data perfectly, but it could in fact make it work **worse on future test data**.

Machine learning methods are especially prone to overfitting because they can try a huge number of different "rules" until one that fits the training data perfectly is found. Especially methods that are very flexible and can adapt to almost any pattern in the data can overfit unless the amount of data is enormous. For example, compared to quite restricted linear models obtained by linear regression, neural networks can require massive amounts of data before they produce reliable prediction.

Learning to avoid overfitting and choose a model that is not too restricted, nor too flexible, is one of the most essential skills of a data scientist.

Learning without a teacher: unsupervised learning

Above we discussed supervised learning where the correct answers are available, and the task of the machine learning algorithm is to find a model that predicts them based on the input data.

In unsupervised learning, the correct answers are not provided. This makes the situation quite different since we can't build the model by making it fit the correct answers on training data. It also makes the evaluation of performance more complicated since we can't check whether the learned model is doing well or not.

Typical unsupervised learning methods attempt to learn some kind of "structure" underlying the data. This can mean, for example, **visualization** where similar items are placed near each other and dissimilar items further away from each other. It can also mean **clustering** where we use the data to identify groups or "clusters" of items that are similar to each other but dissimilar from data in other clusters.

Note

Example

As a concrete example, grocery store chains collect data about their customers' shopping behavior (that's why you have all those loyalty cards). To better understand their customers, the store can either visualize the data using a graph where each customer is represented by a dot and customers who tend to buy the same products are placed nearer each other than customers who buy very different products. Or, the store could apply clustering to obtain a set of customer groups such as "low budget health food enthusiasts", "high-end fish lovers", "soda and pizza 6 days a week", and so on. Note that the machine learning method would only group the customers into clusters, but it wouldn't automatically generate the cluster labels ("fish lovers" and so on). This task would be left for the user.

Yet another example of unsupervised learning can be termed **generative modeling**. This has become a prominent approach since the last few years as a deep learning technique called generative adversarial networks (GANs) has lead to great advances. Given some data, for example, photographs of people's faces, a generative model can generate more of the same: more real-looking but artificial images of people's faces.

We will return to GANs and the implications of being able to produce high-quality artificial image content a bit later in the course, but next we will take a closer look at supervised learning and discuss some specific methods in more detail.

Next section

II. The nearest neighbor classifier →

Course overview | About | FAQ | Privacy Policy | My profile | Sign out

University of Helsinki
Department of Computer Science
Reaktor