

spam. This would be an example of using an expert-designed rule system to design an “intelligent” application. Manually crafting decision rules is feasible for some applications, particularly those in which humans have a good understanding of the process to model. However, using handcoded rules to make decisions has two major disadvantages:

- The logic required to make a decision is specific to a single domain and task. Changing the task even slightly might require a rewrite of the whole system.
- Designing rules requires a deep understanding of how a decision should be made by a human expert.

One example of where this handcoded approach will fail is in detecting faces in images. Today, every smartphone can detect a face in an image. However, face detection was an unsolved problem until as recently as 2001. The main problem is that the way in which pixels (which make up an image in a computer) are “perceived” by the computer is very different from how humans perceive a face. This difference in representation makes it basically impossible for a human to come up with a good set of rules to describe what constitutes a face in a digital image.

Using machine learning, however, simply presenting a program with a large collection of images of faces is enough for an algorithm to determine what characteristics are needed to identify a face.

Problems Machine Learning Can Solve

The most successful kinds of machine learning algorithms are those that automate decision-making processes by generalizing from known examples. In this setting, which is known as *supervised learning*, the user provides the algorithm with pairs of inputs and desired outputs, and the algorithm finds a way to produce the desired output given an input. In particular, the algorithm is able to create an output for an input it has never seen before without any help from a human. Going back to our example of spam classification, using machine learning, the user provides the algorithm with a large number of emails (which are the input), together with information about whether any of these emails are spam (which is the desired output). Given a new email, the algorithm will then produce a prediction as to whether the new email is spam.

Machine learning algorithms that learn from input/output pairs are called supervised learning algorithms because a “teacher” provides supervision to the algorithms in the form of the desired outputs for each example that they learn from. While creating a dataset of inputs and outputs is often a laborious manual process, supervised learning algorithms are well understood and their performance is easy to measure. If your application can be formulated as a supervised learning problem, and you are able to

create a dataset that includes the desired outcome, machine learning will likely be able to solve your problem.

Examples of supervised machine learning tasks include:

Identifying the zip code from handwritten digits on an envelope

Here the input is a scan of the handwriting, and the desired output is the actual digits in the zip code. To create a dataset for building a machine learning model, you need to collect many envelopes. Then you can read the zip codes yourself and store the digits as your desired outcomes.

Determining whether a tumor is benign based on a medical image

Here the input is the image, and the output is whether the tumor is benign. To create a dataset for building a model, you need a database of medical images. You also need an expert opinion, so a doctor needs to look at all of the images and decide which tumors are benign and which are not. It might even be necessary to do additional diagnosis beyond the content of the image to determine whether the tumor in the image is cancerous or not.

Detecting fraudulent activity in credit card transactions

Here the input is a record of the credit card transaction, and the output is whether it is likely to be fraudulent or not. Assuming that you are the entity distributing the credit cards, collecting a dataset means storing all transactions and recording if a user reports any transaction as fraudulent.

An interesting thing to note about these examples is that although the inputs and outputs look fairly straightforward, the data collection process for these three tasks is vastly different. While reading envelopes is laborious, it is easy and cheap. Obtaining medical imaging and diagnoses, on the other hand, requires not only expensive machinery but also rare and expensive expert knowledge, not to mention the ethical concerns and privacy issues. In the example of detecting credit card fraud, data collection is much simpler. Your customers will provide you with the desired output, as they will report fraud. All you have to do to obtain the input/output pairs of fraudulent and nonfraudulent activity is wait.

Unsupervised algorithms are the other type of algorithm that we will cover in this book. In unsupervised learning, only the input data is known, and no known output data is given to the algorithm. While there are many successful applications of these methods, they are usually harder to understand and evaluate.

Examples of unsupervised learning include:

Identifying topics in a set of blog posts

If you have a large collection of text data, you might want to summarize it and find prevalent themes in it. You might not know beforehand what these topics are, or how many topics there might be. Therefore, there are no known outputs.

Segmenting customers into groups with similar preferences

Given a set of customer records, you might want to identify which customers are similar, and whether there are groups of customers with similar preferences. For a shopping site, these might be “parents,” “bookworms,” or “gamers.” Because you don’t know in advance what these groups might be, or even how many there are, you have no known outputs.

Detecting abnormal access patterns to a website

To identify abuse or bugs, it is often helpful to find access patterns that are different from the norm. Each abnormal pattern might be very different, and you might not have any recorded instances of abnormal behavior. Because in this example you only observe traffic, and you don’t know what constitutes normal and abnormal behavior, this is an unsupervised problem.

For both supervised and unsupervised learning tasks, it is important to have a representation of your input data that a computer can understand. Often it is helpful to think of your data as a table. Each data point that you want to reason about (each email, each customer, each transaction) is a row, and each property that describes that data point (say, the age of a customer or the amount or location of a transaction) is a column. You might describe users by their age, their gender, when they created an account, and how often they have bought from your online shop. You might describe the image of a tumor by the grayscale values of each pixel, or maybe by using the size, shape, and color of the tumor.

Each entity or row here is known as a *sample* (or data point) in machine learning, while the columns—the properties that describe these entities—are called *features*.

Later in this book we will go into more detail on the topic of building a good representation of your data, which is called *feature extraction* or *feature engineering*. You should keep in mind, however, that no machine learning algorithm will be able to make a prediction on data for which it has no information. For example, if the only feature that you have for a patient is their last name, no algorithm will be able to predict their gender. This information is simply not contained in your data. If you add another feature that contains the patient’s first name, you will have much better luck, as it is often possible to tell the gender by a person’s first name.

Knowing Your Task and Knowing Your Data

Quite possibly the most important part in the machine learning process is understanding the data you are working with and how it relates to the task you want to solve. It will not be effective to randomly choose an algorithm and throw your data at it. It is necessary to understand what is going on in your dataset before you begin building a model. Each algorithm is different in terms of what kind of data and what problem setting it works best for. While you are building a machine learning solution, you should answer, or at least keep in mind, the following questions: