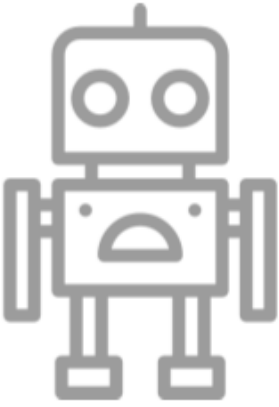
Demystifying Artificial Intelligence



Jeff Leek &
Divya Narayanan

Demystifying Artificial Intelligence

Jeffrey Leek and Divya Narayanan

This book is for sale at http://leanpub.com/demystifyai

This version was published on 2017-02-23

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Also By Jeffrey Leek

The Elements of Data Analytic Style

Executive Data Science

How to be a modern scientist

Contents

Introduction

> "Thinking is a human feature. Will AI someday really think? That's like asking if submarines swim. If you call it swimming then robots will think, yes." Noam Chomsky

Why does artificial intelligence need to be demystified?

Artificial intelligence is a fertile topic that has been covered by a number of books, online tutorials, media articles, videos, and movies. So why on earth does artificial intelligence need to be demystified?

The reason is that almost everything written about AI can be placed in one of two categories:

1. General audience facing media that treats AI as a fully human-like or all-knowing machine.
2. Highly technical resources that focus on the complicated computing and statistics behind artificial intelligence.

The consequence of this separation is that for most people AI is something mysterious and confusing. It is not surprising when artificial intelligence is viewed as a combination of

human-like robots and dense equations and assume it is something that is hard to understand or work with. It is a complete mystery how computers can be trained to recognize faces or drive cars.

The truth about AI is both more mundane and more immediate. Computers are pretty far away from being able to do everything that humans do. At the same time, specific types of AI are already commonplace - from Siri to Amazon Echo to Facebook tagging people in pictures. These types of AI share some common principles and are impacting our world righ tnow.

This book is designed to demystify all the parts of a modern AI application. The target audience is meant to be anyone who is curious about how Google, Facebook, Amazon, and others use artificial intelligence to interact with people right now. In the book we will cover:

- A specific definition of artificial intelligence that covers most of what you read about in the popular press.
- How images, audio files, video, and text are converted from their raw form to data that can be used by computers to "learn" about the world.
- How information is gathered to build artificial intelligence applications and how the collection process influences how the application will behave.
- The concept of machine learning and how computers use data to make decisions or classify examples.
- How neural networks - the most widely used algorithms for modern AI applications - work and how they are implemented.
- How you can use commodity technology to build AI applications without fitting your own neural network.

- What are the ethical and moral implications of a world where data is ubiquitously collected and used?

The book is intended to be instructive in how AI applications work and as the beginning of a multi-part educational path toward building your own AI applications. The target audience is meant to be broad. The goal is that anyone with an interest in artificial intelligence will be able to follow this book from start to finish.

To achieve this goal it is necessary to explain many of the components of the AI process in a careful, step-by-step way. It is likely that people with a technical background in machine learning or statistics may find parts of this book unnecessarily detailed and explicit. Readers with sufficient background are encouraged to move quickly through material that they already understand and may wish to focus their attention on chapters devoted to high-level conceptual issues that are relevant to even people with deep expertise in the area.

At the same time there are some concepts - particularly related to the algorithmic details of neural networks - that may seem daunting to people without a technical background. The goal of this book is to use diagrams, activities, and thought exercises to explain these concepts to a broad audience. Where appropriate we will point to background reading and more in depth material for interested readers.

Our goal for this book is to "demystify" how modern AI applications work. These applications have rapidly become integrated into our lives, from digital assistants on our phones, to smart speakers in our houses, to the search engines we use, and the maps that give us directions. While we are still probably years from any human-like form of AI the potential

implications of already existing AI applications are enormous. In that sense the future is already here - but as is generally the case - it is not evenly distributed. We want to make the world conversant in AI as AI becomes conversant with us.

How the book is organized

This book first introduces the key components of AI applications: (a) large collections of data - how they are obtained and stored, (b) algorithms for learning models of the world from data, and (c) interfaces for applying the learned models to interact with the world. We will use very simple examples to illustrate these ideas so that the reader can get an intuitive understanding of how artificial intelligence works.

The next set of chapters will cover several concrete examples of artificial intelligence. What data are used to build that type of AI, how they are collected, how they are processed, what kind of algorithms they use, and how the interfaces work. These chapters are meant to take the high level conceptual ideas developed in the first set of chapters and make them concrete with real world examples. These examples will be explained concretely, but at a high level. The real details of these applications could each be the subject of multiple books and so we will only explain enough detail to make the AI process concrete in the readers mind.

In general modern AI applications require massive amounts of data and computing to be feasible. Most individuals and all but the largest organizations will not have sufficient data, computing, and expertise to build AI applications from scratch. However, a number of large companies including Amazon, Facebook, and Google have developed AI models for speech,

text, images, and video that you can use to build AI applications even if you don't have massive amounts of data or computing capability. The next chapter will discuss how to design modern AI applications using commodity machine learning models that have already been built by large organizations.

The final chapter will cover some of the ethical, moral, and legal implications of modern AI applications. There are outstanding works devoted entirely to this topic which we will point out during this discussion. However, it would be remiss of us not to point out the large potential for misuse, mistakes, and abuse implied by the combination of large scale data collection coupled with highly accurate predictive models.

Throughout the book we will include illustrations of key ideas using figures and images. We have generated many of these images using the R programming language[1]. If you have background with R and are interested in these examples we have made them available as part of the demistifyai[2] R package which can be installed in R using the commands:

```
devtools::install_github("jtleek/demystifyai")
```

Relationship to the course "Demistifying Artificial Intelligence"

This book represents the educational content for the class Demistifying Artificial Intelligence[3]. If you find the material

[1] r-project.org

[2] https://github.com/jtleek/demistifyai

[3] coursera.org/demistifyai

in the book interesting and would like to test your understanding of the material or obtain a qualification in this material you can join the course online at any time.

About the authors

The authors are Jeff Leek[4] and Divya Narayanan[5].

Jeff is an associate professor of Biostatistics at the Johns Hopkins Bloomberg School of Public Health. He is the instructor of the Coursera courses The Data Scientist's Toolbox[6], Getting and Cleaning Data[7], Practical Machine Learning[8], Introduction to Genomic Technologies[9], and Statistics for Genomic Data Science[10]. He is also the author of the books The Elements of Data Analytic Style[11] and How to be a Modern Scientist[12].

Divya Narayanan is a Master of Science candidate in Epidemiology at the Johns Hopkins Bloomberg School of Public Health with an interest in data science.

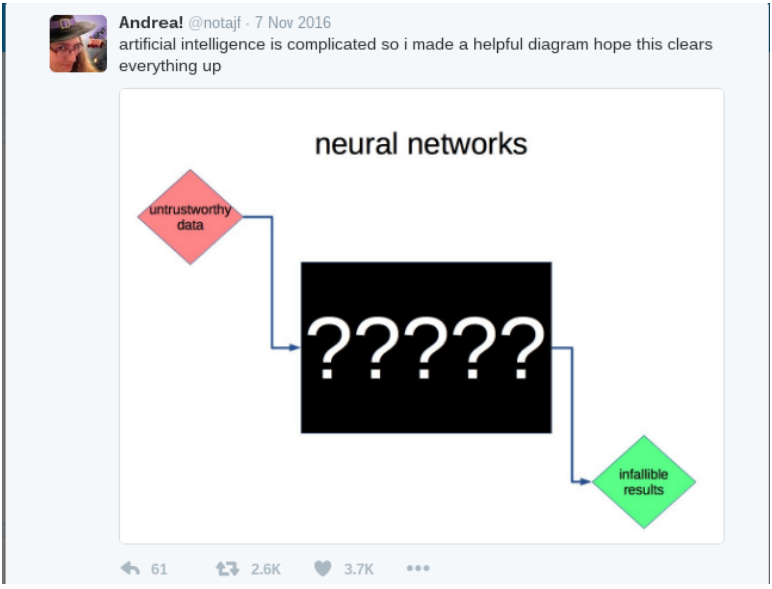
Neither Jeff or Divya are world experts in machine learning or artificial intelligence. It might be surprising that they are writing this book because it isn't their research area. However, as with many educated amateurs, they have learned the material recently enough to know which parts are hard to

[4] jtleek.com
[5] https://twitter.com/data_divya
[6] www.coursera.org/dst
[7] www.coursera.org/gcd
[8] www.coursera.org/pml
[9] www.coursera.org/igt
[10] www.coursera.org/sgds
[11] leanpub.com/datastyle/
[12] leanpub.com/modernscientist/

understand. Hopefully that will mean that difficult concepts will not be glossed over. There are certainly people who are more expert in the field and throughout the book these people and resources will be pointed out.

About the Cover

The cover is based on this amazing tweet[13] by Twitter user (???)(ht //twitter.com/notajf/) which sums up a common concern about AI and neural networks and is one of the author's favorite commentaries on the subject.



A potential concern about neural networks

[13] https://twitter.com/notajf/status/795717253505413122

What is artificial intelligence?

> "If it looks like a duck and quacks like a duck but it needs batteries, you probably have the wrong abstraction" Derick Bailey[14]

This book is about artificial intelligence. The term "artificial intelligence" or "AI" has a long and convoluted history (Cohen and Feigenbaum 2014). It has been used by philosophers, statisticians, machine learning experts, mathematicians, and the general public. This historical context means that when people say *artificial intelligence* the term is loaded with one of many potential different meanings.

Humanoid robots

Before we can demystify artificial intelligence it is helpful to have some context for what the word means. When asked about artificial intelligence, most people's imagination leaps immediately to images of robots that can act like and interact with humans. Near-human robots have long been a source of fascination by humans have appeared in cartoons like the *Jetsons* and science fiction like *Star Wars*. More recently,

[14] https://lostechies.com/derickbailey/2009/02/11/solid-development-principles-in-motivational-pictures/

subtler forms of near-human robots with artificial intelligence have played roles in movies like *Her* and *Ex machina*.

People usually think of artificial intelligence as a human-like robot performing all the tasks that a person could.

The type of artificial intelligence that can think and act like a human is something that experts call artificial general intelligence (Wikipedia contributors 2017a).

> is the intelligence of a machine that could successfully perform any intellectual task that a human being can

There is an understandable fascination and fear associated with robots, created by humans, but evolving and thinking independently. While this is a major area of ressearch (Laird, Newell, and Rosenbloom 1987) and of course the center of most people's attention when it comes to AI, there is no near term possibility of this type of intelligence (Urban, n.d.). There are a number of barriers to human-mimicking AI from

difficulty with robotics (Couden 2015) to needed speedups in computational power (Langford, n.d.).

One of the key barriers is that most current forms of the computer models behind AI are trained to do one thing really well, but can not be applied beyond that narrow task. There are extremely effective artificial intelligence applications for translating between languages (Wu et al. 2016), for recognizing faces in images (Taigman et al. 2014), and even for driving cars (Santana and Hotz 2016).

But none of these technologies are generalizable across the range of tasks that most adult humans can accomplish. For example, the AI application for recognizing faces in images can not be directly applied to drive cars and the translation application couldn't recognize a single image. While some of the internal technology used in the applications is the same, the final version of the applications can't be transferred. This means that when we talk about artificial intelligence we are not talking about a general purpose humanoid replacement. Currently we are talking about technologies that can typically accomplish one or two specific tasks that a human could accomplish.

Cognitive tasks

While modern AI applications couldn't do everything that an adult could do (Baciu and Baciu 2016), they can perform individual tasks nearly as well as a human. There is a second commonly used definition of artificial intelligence that is considerably more narrow (Wikipedia contributors 2017b)

> … the term "artificial intelligence" is applied when

> a machine mimics "cognitive" functions that humans associate with other human minds, such as "learning" and "problem solving".

This definition encompasses applications like machine translation and facial recognition. They are "cognitive" functions that are generally usually only performed by humans. A difficulty with this definition is that it is relative. People refer to machines that can do tasks that we thought humans could only do as artificial intelligence. But over time, as we become used to machines performing a particular task it is no longer surprising and we stop calling it artificial intelligence. John McCarthy, one of the leading early figures in artificial intelligence said (Vardi 2012):

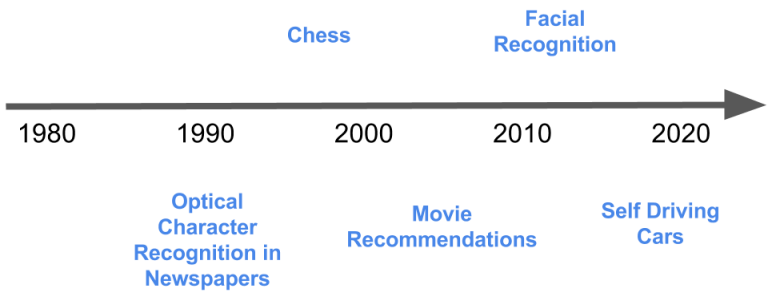
> As soon as it works, no one calls it AI anymore…

As an example, when you send a letter in the mail, there is a machine that scans the writing on the letter. A computer then "reads" the characters on the front of the letter. The computer reads the characters in several steps - the color of each pixel in the picture of the letter is stored in a data set on the computer. Then the computer uses an algorithm that has been built using thousands or millions of other letters to take the pixel data and turn it into predictions of the characters in the image. Then the characters are identified as addresses, names, zipcodes, and other relevant pieces of information. Those are then stored in the computer as text which can be used for sorting the mail.

This task used to be considered "artificial intelligence" (Pavlidis, n.d.). It was surprising that a computer could perform the tasks of recognizing characters and addresses just based on a picture of the letter. This task is now called "optical character

recognition" (Wikipedia contributors 2016). Many tutorials on the algorithms behind machine learning begin with this relatively simple task (Google Tensorflow Team, n.d.). Optical character recognition is now used in a wide range of applications including in Google's effort to digitize millions of books (Darnton 2009).

Since this type of algorithm has become so common it is no longer called "artificial intelligence". This transition happened becasue we no longer think it is surprising that computers can do this task - so it is no longer considered intelligent. This process has played out with a number of other technologies. Initially it is thought that only a human can do a particular cognitive task. As computers become increasingly proficient at that task they are called artificially intelligent. Finally, when that task is performed almost exclusively by computers it is no longer considered "intelligent" and the boundary moves.



Timeline of tasks we were surprised that computers could do as well as humans.

Over the last two decades tasks from optical character recognition, to facial recognition in images, to playing chess have started as artificially intelligent applications. At the time of this writing there are a number of technologies that are currently on the boundary between doable only by a human and doable by a computer. These are the tasks that are considered AI when you read about the term in the media. Examples of tasks that are currently considered "artificial intelligence" include:

- Computers that can drive cars
- Computers that can identify human faces from pictures
- Computers that can translate text from one language to another
- Computers that can label pictures with text descriptions

Just as it used to be with optical character recognition, self-driving cars and facial recognition are tasks that still surprise us when performed by a computer. So we still call them artificially intelligent. Eventually, many or most of these tasks will be performed nearly exclusively by computers and we will no longer think of them as components of computer "intelligence". To go a little further we can think about any task that is repetitive and performed by humans. For example, picking out music that you like or helping someone buy something at a store. An AI can eventually be built to do those tasks provided that: (a) there is a way of measuring and storing information about the tasks and (b) there is technology in place to perform the task if given a set of computer instructions.

The more narrow definition of AI is used colloquially in the news to refer to new applications of computers to perform

tasks previously thought impossible. It is important to know both the definition of AI used by the general public and the more narrow and relative definition used to describe modern applications of AI by companies like Google and Facebook. But neither of these definitions is satisfactory to help demystify the current state of artificial intelligence applications.

A three part definition

The first definition describes a technology that we are not currently faced with - fully functional general purpose artificial intelligence. The second definition suffers from the fact that it is relative to the expectations of people discussing applications. For this book, we need a definition that is concrete, specific, and doesn't change with societal expectations.

We will consider specific examples of human-like tasks that computers can perform. So we will use the definition that artificial intelligence requires the following components:

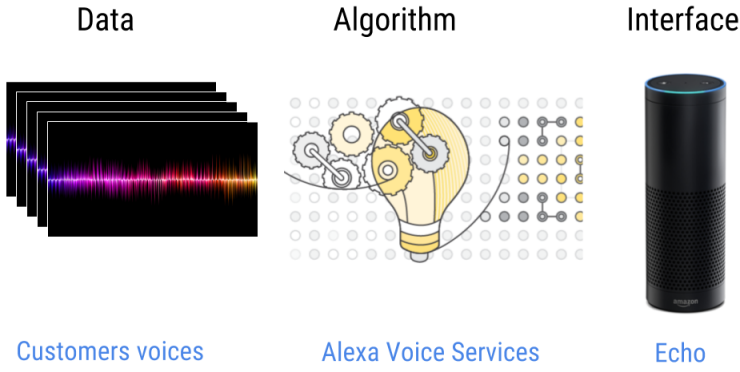
1. *The data set* : A of data examples that can be used to train a statistical or machine learning model to make predictions.
2. *The algorithm* : An algorithm that can be trained based on the data examples to take a new example and execute a human-like task.
3. *The interface* : An interface for the trained algorithm to receive a data input and execute the human like task in the real world.

This definition encompases optical character recognition and all the more modern examples like self driving cars. It is also

intentionally broad, covering even examples where the data set is not large or the algorithm is not complicated. We will use our definition to break down modern artificial intelligence applications into their constituitive parts and make it clear how the computer represents knowledge learned from data examples and then applies that knowledge.

As one example, consider Amazon Echo and Alexa - an application currently considered to be artificially intelligent (Nuñez, n.d.). This combination meets our definition of artificially intelligent since each of the components is in place.

1. *The data set* : The large set of data examples consist of all the recordings that Amazon has collected of people talking to their Amazon devices.
2. *The machine learning algorithm* : The Alexa voice service (Alexa Developers 2016) is a machine learning algorithm trained using the previous recordings of people talking to Amazon devices.
3. *The interface* : The interface is the Amazon Echo (Amazon Inc 2016) a speaker that can record humans talking to it and respond with information or music.



The three parts of an artificial intelligence illustrated with Amazon Echo and Alexa

When we break down artificial intelligence into these steps it makes it clearer why there has been such a sudden explosion of interest in artificial intelligence over the last several years.

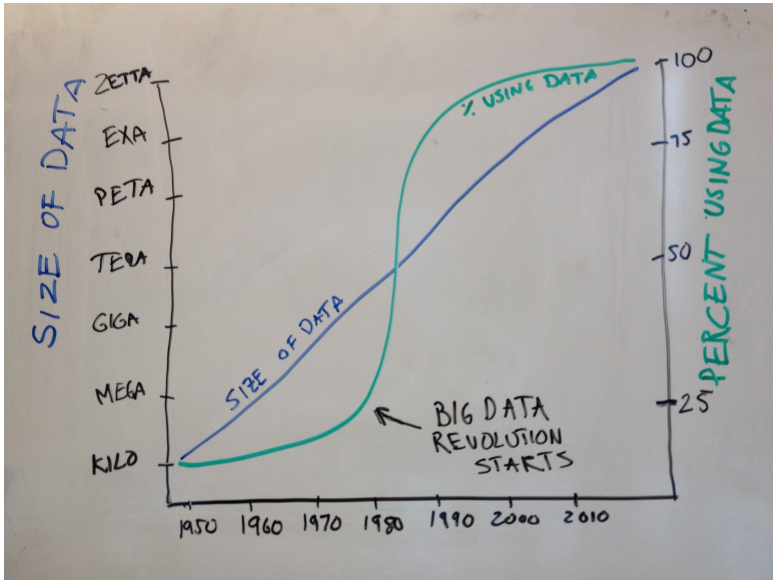
First, the cost of data storage and collection has gone down steadily (Irizarry, n.d.) but dramatically (Quigley, n.d.) over the last several years. As the costs have come down, it is increasingly feasible for companies, governments, and even individuals to store large collections of data (Component 1 - *The Data*). To take advantage of these huge collections of data requires incredibly flexible statistical or machine learning algorithms that can capture most of the patterns in the data and re-use them for prediction. The most common type of algorithms used in modern artificial intelligence are something called "deep neural networks". These algorithms are so flexible they capture nearly all of the important structure in the data. They can only be trained well if huge data sets exist and computers are fast enough. Continual increases in computing speed and power over the last several decades now

make it possible to apply these models to use collections of data (Component 2 - *The Algorithm*).

Finally, the most underappreciated component of the AI revolution does not have to do with data or machine learning. Rather it is the development of new interfaces that allow people to interact directly with machine learning models. For a number of years now, if you were an expert with statistical and machine learning software it has been possible to build highly accurate predictive models. But if you were a person without technical training it was not possible to directly interact with algorithms.

Or as statistical experts Diego Kuonen and Rafael Irizarry have put it:

> The big in big data refers to importance, not size



It isn't about how much data you have, it is about how many people you can get to use it.

The explosion of interfaces for regular, non-technical people to interact with machine learning is an underappreciated driver of the AI revolution of the last several years. Artificial intelligence can now power labeling friends on Facebook, parsing your speech to your personal assistant Siri or Google Assistant, or providing you with directions in your car, or when you talk to your Echo. More recently sensors and devices make it possible for the instructions created by a computer to steer and drive a car.

These interfaces now make it possible for hundreds of millions of people to directly interact with machine learning algorithms. These algorithms can range from exceedingly simple to mind bendingly complex. But the common result is that the interface allows the computer to perform a human-like action

and makes it look like artificial intelligence to the person on the other side. This interface explosion only promises to accelerate as we are building sensors for both data input and behavior output in objects from phones to refrigerators to cars (Component 3 - *The interface*).

This definition of artificial intelligence in three components will allow us to demystify artificial intelligence applications from self driving cars to facial recognition. Our goal is to provide a high-level interface to the current conception of AI and how it can be applied to problems in real life. It will include discussion and references to the sophisticated models and data collection methods used by Facebook, Tesla, and other companies. However, the book does not assume a mathematical or computer science background and will attempt to explain these ideas in plain language. Of course, this means that some details will be glossed over, so we will attempt to point the interested reader toward more detailed resources throughout the book.

References

Alexa Developers. 2016. "Alexa Voice Service." https://developer.amazon.com/alexa-voice-service.

Amazon Inc. 2016. "Amazon Echo." https://www.amazon.com/Amazon-Echo-Bluetooth-Speaker-with-WiFi-Alexa/dp/B00X4WHP5E.

Baciu, Assaf, and Assaf Baciu. 2016. "Artificial Intelligence Is More Artificial Than Intelligent." *Wired*, 7∼dec.

Cohen, Paul R, and Edward A Feigenbaum. 2014. *The Handbook of Artificial Intelligence*. Vol. 3. Butterworth-Heinemann. https://goo.gl/wg5rMk.

Couden, Craig. 2015. "Why It's so Hard to Make Humanoid Robots | Make:" http://makezine.com/2015/06/15/hard-make-humanoid-robots/.

Darnton, Robert. 2009. *Google & the Future of Books*. na.

Google Tensorflow Team. n.d. "MNIST for ML Beginners | TensorFlow." https://www.tensorflow.org/tutorials/mnist/beginners/.

Irizarry, Rafael. n.d. "The Big in Big Data Relates to Importance Not Size · Simply Statistics." http://simplystatistics.org/2014/05/28/the-big-in-big-data-relates-to-importance-not-size/.

Laird, John E, Allen Newell, and Paul S Rosenbloom. 1987. "Soar: An Architecture for General Intelligence." *Artificial Intelligence* 33 (1). Elsevier: 1–64.

Langford, John. n.d. "AlphaGo Is Not the Solution to AI « Machine Learning (Theory)." http://hunch.net/?p=3692542.

Nuñez, Michael. n.d. "Amazon Echo Is the First Artificial Intelligence You'll Want at Home." http://www.popsci.com/amazon-echo-first-artificial-intelligence-youll-want-home.

Pavlidis, Theo. n.d. "Computers Versus Humans - 2002 Lecture." http://www.theopavlidis.com/comphumans/comphuman.htm.

Quigley, Robert. n.d. "The Cost of a Gigabyte over the Years." http://www.themarysue.com/gigabyte-cost-over-years/.

Santana, Eder, and George Hotz. 2016. "Learning a Driving Simulator," 3∼aug.

Taigman, Y, M Yang, M Ranzato, and L Wolf. 2014. "DeepFace: Closing the Gap to Human-Level Performance in Face Verification." In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 1701–8.

Urban, Tim. n.d. "The AI Revolution: How Far Away Are Our Robot Overlords?" http://gizmodo.com/the-ai-revolution-how-far-away-are-our-robot-overlords-1684199433.

Vardi, Moshe Y. 2012. "Artificial Intelligence: Past and Future." *Commun. ACM* 55 (1). New York, NY, USA: ACM: 5–5.

Wikipedia contributors. 2016. "Optical Character Recognition." https://en.wikipedia.org/w/index.php?title=Optical_character_recognition&oldid=757150540.

———. 2017a. "Artificial General Intelligence." https://en.wikipedia.org/w/index.php?title=Artificial_general_intelligence&oldid=758867755.

———. 2017b. "Artificial Intelligence." https://en.wikipedia.org/w/index.php?title=Artificial_intelligence&oldid=759177704.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, et al. 2016. "Google's Neural Machine Translation System: Bridging the Gap Between Human and Machine Translation," 26∼sep.

An example that isn't that artificial or intelligent

> "I am so clever that sometimes I don't understand a single word of what I am saying." Oscar Wilde

As we have described it artificial intelligence applications consist of three things:

1. A large collection of data examples
2. An algorithm for learning a model from that training set.
3. An interface with the world.

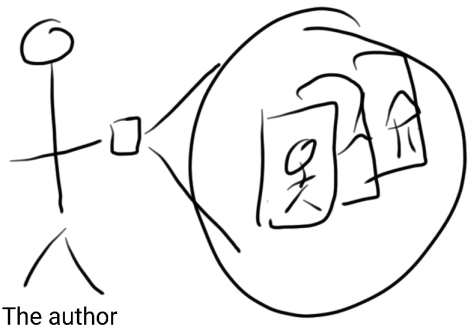
In the following chapters we will go into each of these components in much more detail, but lets start with a a couple of very simple examples to make sure that the components of an AI are clear. We will start with a completely artificial example and then move to more complicated examples.

Building a album

Lets start with a very simple hypothetical example that can be understood even if you don't have a technical background.

We can also use this example to define some of the terms we will be discussing later in the book.

In our simple example the goal is to make an album of photos for a friend. For example, suppose I want to take the photos in my photobook and find all the ones that include pictures of myself and my son Dex for his grandmother.



The author's drawing of the author's phone album. Don't make fun, he's a data scientist, not an artist

If you are anything like the author of this book, then you probably have a very large number of pictures of your family on your phone. So the first step in making the photo alubm would be to stort through all of my pictures and pick out the ones that should be part of the album.

This is a typical example of the type of thing we might want to train a computer to do in an artificial intelligence application. Each of the components of an AI application is there:

1. **The data**: all of the pictures on the author's phone (a

big training set!)
2. **The algorithm**: finding pictures of me and my son Dex
3. **The interface**: the album to give to Dex's grandmother.

One way to solve this problem is for me to sort through the pictures one by one and decide whether they should be in the album or not, then assemble them together, and then put them into the album. If I did it like this then I myself would be the AI! That wouldn't be very artificial though…imagine we instead wanted to teach a computer to make this album..

> But what does it mean to "teach" a computer to do something?

The terms "machine learning" and "artificial intelligence" invoke the idea of teaching computers in the same way that we teach children. This was a deliberate choice to make the analogy - both because in some ways it is appropriate and because it is useful for explaining complicated concepts to people with limited backgrounds. To teach a child to find pictures of the author and his son, you would show her lots of examples of that type of picture and maybe some examples of the author with other kids who were not his son. You'd repeat to the child that the pictures of the author and his son were the kinds you wanted and the others weren't. Eventually she would retain that information and if you gave her a new picture she could tell you whether it was the right kind or not.

To teach a machine to perform the same kind of recognition you go through a similar process. You "show" the machine many pictures labeled as either the ones you want or not. You repeat this process until the machine "retains" the information

and can correctly label a new photo. Getting the machine to "retain" this information is a matter of getting the machine to create a set of step by step instructions it can apply to go from the image to the label that you want.

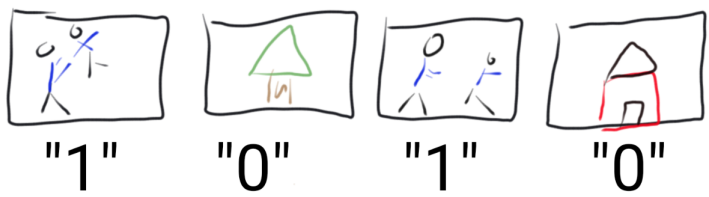
The data

The images are what people in the fields of artificial intelligence and machine learning call *"raw data"* (Leek, n.d.). The categories of pictures (a picture of the author and his son or a picture of something else) are called the *"labels"* or *"outcomes"*. If the computer gets to see the labels when it is learning then it is called *"supervised learning"* (Wikipedia contributors 2016) and when the computer doesn't get to see the labels it is called *"unsupervised learning"* (Wikipedia contributors 2017a).

Going back to our analogy with the child, supervised learning would be teaching the child to recognize pictures of the author and his son together. Unsupervised learning would be giving the child a pile of pictures and asking them to sort them into groups. They might sort them by color or subject or location - not necessarily into categories that you care about. But probably one of the categories they would make would be pictures of people - so she would have found some potentially useful information even if it wasn't exactly what you wanted. One whole field of artificial intelligence is figuring out how to use the information learned in this "unsupervised" setting and using it for supervised tasks - this is sometimes called *"transfer learning"* (Raina et al. 2007) by people in the field since you are transferring information from one task to another.

Returning to the task of "teaching" a computer to retain information about what kind of pictures you want we run into a problem - computers don't know what pictures are! They also don't know what audio clips, text files, videos, or any other kind of information is. At least not directly. They don't have eyes, ears, and other senses along with a brain designed to decode the information from these senses.

So what can a computer understand? A good rule of thumb is that a computer works best with numbers. If you want a computer to sort pictures into an album for you, the first thing you need to do is to find a way to turn all of the information you want to "show" the computer into numbers. In the case of sorting pictures into albums - a supervised learning problem - we need to turn the labels and the images into numbers the computer can use.



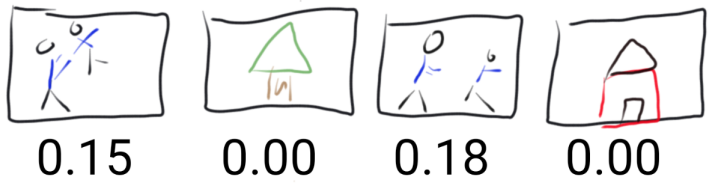
Label each picture as a one or a zero depending on whether it is the kind of picture you want in the album

One way to do that would be for you to do it for the computer.

You could take every picture on your phone and label it with a 1 if it was a picture of the author and his son and a 0 if not. Then you would have a set of 1's and 0's corresponding to all of the pictures. This takes some thing the computer can't understand (the picture) and turns it into something the computer can understand (the label).

This process would turn the labels into something a computer could understand, it still isn't something we could teach a computer to do. The computer can't actually "look" at the image and doesn't know who the author or his son are. So we need to figure out a way to turn the images into numbers for the computer to use to generate those labels directly.

This is a little more complicated but you could still do it for the computer. Let's suppose that the author and his son always wear matching blue shirts when they spend time together. Then you could go through and look at each image and decide what fraction of the image is blue. So each picture would get a number ranging from zero to one like 0.30 if the picture was 30% blue and 0.53 if it was 53% blue.



Calculate the fraction of each image that is the color blue as a "feature" of the image that is numeric

The fraction of the picture that is blue is called a *"feature"* and the process of creating that feature is called *"feature engineering"* (Wikipedia contributors 2017b). Until very recently feature engineering of text, audio, or video files was best performed by an expert human. In later chapters we will discuss how one of the most exciting parts about AI application is that it is now possible to have computers perform feature engineering for you.

The algorithm

Now that we have converted the images to numbers and the labels to numbers, we can talk about how to "teach" a computer to label the pictures. A good rule of thumb when thinking about algorithms is that a computer can't "do" anything without being told very explicitly what to do. It needs a step by step set of instructions. The instructions

should start with a calculation on the numbers for the image and should end with a prediction of what label to apply to that image. The image (converted to numbers) is the *"input"* and the label (also a number) is the *"output"*. You may have heard the phrase:

> "Garbage in, garbage out"

What this phrase means is if the inputs (the images) are bad - say they are all very dark or hard to see. Then the output of the algorithm will also be bad - the predictions won't be very good.

A machine learning *"algorithm"* can be thought of as a set of instructions with some of the parts left blank - sort of like mad-libs. One example of a really simple algorithm for sorting pictures into the album would be:

1. Calculate the fraction of blue in the image.
2. If the fraction of blue is above X label it 1
3. If the fraction of blue is less than X label it 0
4. Put all of the images labeled 1 in the album

The machine *"learns"* by using the examples to fill in the blanks in the instructions. In the case of our really simple algorithm we need to figure out what fraction of blue to use (X) for labeling the picture.

To figure out a guess for X we need to decide what we want the algorithm to do. If we set X to be too low then all of the images will be labeled with a 1 and put into the album. If we set X to be too high then all of the images will be labeled 0

and none will appear in the album. In between there is some grey area - do we care if we accidentally get some pictures of the ocean or the sky with our algorithm?

But the number of images in the album isn't even the thing we really care about. What we might care about is making sure that the album is mostly pictures of the author and his son. In the field of AI they usually turn this statement around - we want to make sure the album has a very small fraction of pictures that are not of the author and his son. This fraction - the fraction that are incorrectly placed in the album is called the *"loss"*. You can think about it like a game where the computer loses a point every time it puts the wrong kind of picture into the album.

Using our loss (how many pictures we incorrectly placed in the album) we can now use the data we have created (the numbers for the labels and the images) to fill in the blanks in our mad-lib algorithm (picking the cutoff on the amount of blue). We have a large number of pictures where we know what fraction of each picture is blue and whether it is a picture of the author and his son or not. We can try each possible X and calculate the fraction of pictures in the album that are incorrectly placed into the album (the loss) and find the X that produces the smallest fraction.

Suppose that the value of X that gives the smallest faction of wrong pictures in the album is 30. Then our "learned" model would be:

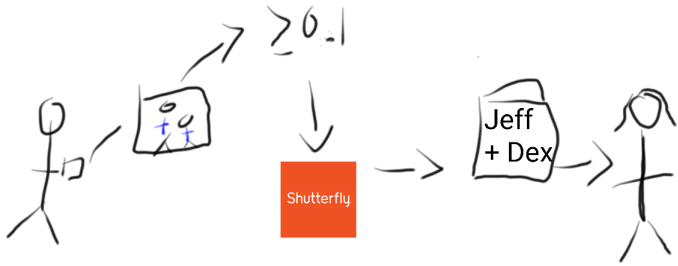
1. Calculate the fraction of blue in the image
2. If the fraction of blue is above 0.1 label it 1
3. If the fraction of blue is less than 0.1 label it 0
4. Put all of the images labeled 1 in the album

The interface

The last part of an AI application is the interface. In this case, the interface would be the way that we share the pictures with Dex's grandmother. For example we could imagine uploading the pictures to Shutterfly[15] and having the album delivered to Dex's grandmother.

Putting this all together we could imagine an application using our trained AI. The author uploads his unlabeled photos. The photos are then passed to the computer program which calculates the fraction of the image that is blue, then applies a label according to the algorithm we learned, then takes all the images predicted to be of the author and his son and sends them off to be a Shutterfly album mailed to the authors' mother.

[15] https://www.shutterfly.com/



Whoa that computer is smart - from the author's picture to grandma's hands!

If the algorithm was good, then from the perspective of the author the website would look "intelligent". I just uploaded pictures and it created an album for me with the pictures that I wanted. But the steps in the process were very simple and understandable behind the scenes.

References

Leek, Jeffrey. n.d. "The Elements of Data Analytic Style." {https://leanpub.com/datastyle}[16].

Raina, Rajat, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. 2007. "Self-Taught Learning: Transfer Learning from Unlabeled Data." In *Proceedings of the 24th International Conference on Machine Learning*, 759–66. ICML '07. New York, NY, USA: ACM.

[16] \protect\char"007B\relaxhttps://leanpub.com/datastyle\protect\char"007D\relax

Wikipedia contributors. 2016. "Supervised Learning." https://en.wikipedia.org/w/index.php?title=Supervised_learning&oldid=752493505.

———. 2017a. "Unsupervised Learning." https://en.wikipedia.org/w/index.php?title=Unsupervised_learning&oldid=760556815.

———. 2017b. "Feature Engineering." https://en.wikipedia.org/w/index.php?title=Feature_engineering&oldid=760758719.

Turning data into numbers

> "It is a capital mistake to theorize before one has data." Arthur Conan Doyle

Data, data everywhere

I already have some data about you. You are reading this book. Does that seem like data? It's just something you did, that's not data is it? But if I collect that piece of information about you, it actually tells me a surprising amount. It tells me you have access to an internet connection, since the only place to get the book is online. That in turn tells me something about your socioeconomic status and what part of the world you live in. It also tells me that you like to read, which suggests a certain level of education.

Whether you know it or not, everything you do produces data - from the websites you read to the rate at which your heart beats. Until pretty recently, most of the data you produced wasn't collected, it floated off unmeasured. Data were painstakingly gathered by scientists one number at a time in small experiments with a few people. This laborious process meant that data were expensive and time-consuming to collect. Yet many of the most amazing scientific discoveries over the last two centuries were squeezed from just a few data points. But over the last two decades, the unit price of data

has dramatically dropped. New technologies touching every aspect of our lives from our money, to our health, to our social interactions have made data collection cheap and easy.

To give you an idea of how steep the drop in the price of data has been, in 1967 Stanley Milgram did an experiment to determine the number of degrees of separation between two people in the U.S. (Travers and Milgram 1969). In his experiment he sent 296 letters to people in Omaha, Nebraska and Wichita, Kansas. The goal was to get the letters to a specific person in Boston, Massachusetts. The trick was people had to send the letters to someone they knew, and they then sent it to someone they knew and so on. At the end of the experiment, only 64 letters made it to the individual in Boston. On average, the letters had gone through 6 people to get there.

This is an idea that is so powerful it even became part of the popular consciousness. For example it is the foundation of the internet meme "the 6-degrees of Kevin Bacon" (Wikipedia contributors 2016a) - the idea that if you take any actor and look at the people they have been in movies with, then the people those people have been in movies with, it will take you at most six steps to end up at the actor Kevin Bacon. This idea, despite its popularity was originally studied by Milgram using only 64 data points. A 2007 study updated that number to "7 degrees of Kevin Bacon". The study was based on 30 billion instant messaging conversations collected over the course of a month or two with the same amount of effort (Leskovec and Horvitz 2008).

Once data started getting cheaper to collect, it got cheaper fast. Take another example, the human genome. The genome is the unique DNA code in every one of your cells. It consists of a set of 3 billion letters that is unique to you. By many

measures, the race to be the first group to collect all 3 billion letters from a single person kicked off the data revolution in biology. The project was completed in 2000 after a decade of work and $3 billion to collect the 3 billion letters in the first human genome (Venter et al. 2001). This project was actually a stunning success, most people thought it would be much more expensive. But just over a decade later, new technology means that we can now collect all 3 billion letters from a person's genome for about $1,000 in about a week ("The Cost of Sequencing a Human Genome," n.d.), soon it may be less than $100 (Buhr 2017).

You may have heard that this is the era of "big data" from The Economist or The New York Times. It is really the era of cheap data collection and storage. Measurements we never bothered to collect before are now so easy to obtain that there is no reason not to collect them. Advances in computer technology also make it easier to store huge amounts of data digitally. This may not seem like a big deal, but it is much easier to calculate the average of a bunch of numbers stored electronically than it is to calculate that same average by hand on a piece of paper. Couple these advances with the free and open distribution of data over the internet and it is no surprise that we are awash in data. But tons of data on their own are meaningless. It is understanding and interpreting the data where the real advances start to happen.

This explosive growth in data collection is one of the key driving influences behind interest in artificial intelligence. When teaching computers to do something that only humans could do previously, it helps to have lots of examples. You can then use statistical and machine learning models to summarize that set of examples and help a computer make decisions what to do. The more examples you have, the more flexible your

computer model can be in making decisions, and the more "intelligent" the resulting application.

What is data?

Tidy data

"What is data"? Seems like a relatively simple question. In some ways this question is easy to answer. According to Wikipedia[17]:

> Data (/ˈdeɪtə/ day-tə, /ˈdætə/ da-tə, or /ˈdɑːtə/ dah-tə)[1] is a set of values of qualitative or quantitative variables. An example of qualitative data would be an anthropologist's handwritten notes about her interviews with people of an Indigenous tribe. Pieces of data are individual pieces of information. While the concept of data is commonly associated with scientific research, data is collected by a huge range of organizations and institutions, ranging from businesses (e.g., sales data, revenue, profits, stock price), governments (e.g., crime rates, unemployment rates, literacy rates) and non-governmental organizations (e.g., censuses of the number of homeless people by non-profit organizations).

When you think about data, you probably think of orderly sets of numbers arranged in something like an Excel spreadsheet. In the world of data science and machine learning this type

[17] https://en.wikipedia.org/wiki/Data

of data has a name - "tidy data" (Wickham and others 2014). Tidy data has the properties that all measured quantities are represented by numbers or character strings (think words). The data are organized such that.

1. Each variable you measured is in one column
2. Each different measurement of that variable is in a different row
3. There is one data table for each "type" of variable.
4. If there are multiple tables then they are linked by a common ID.

This idea is borrowed from data management schemas that have long been used for storing data in databases. Here is an example of a tidy data set of swimming world records.

year	time	sex
1905	65.8	M
1908	65.6	M
1910	62.8	M
1912	61.6	M
1918	61.4	M
1920	60.4	M
1922	58.6	M
1924	57.4	M
1934	56.8	M
1935	56.6	M

This type of data, neat, organized and nicely numeric is not the kind of data people are talking about when they say the "era of big data". Data almost never start their lives in such a neat and organized format.

Raw data

The explosion of interest in AI has been powered by a variety of types of data that you might not even think of when you think of "data". The data might be pictures you take and upload to social media, the text of the posts on that same platform, or the sound captured from your voice when you speak to your phone.

Social media and cell phones aren't the only area where data is being collected more frequently. Speed cameras on roads collect data on the movement of cars, electronic medical records store information about people's health, wearable devices like Fitbit collect information on the activity of people. GPS information stores the location of people, cars, boats, airplanes, and an increasingly wide array of other objects.

Images, voice recordings, text files, and GPS coordinates are what experts call "raw data". To create an artificial intelligence application you need to begin with a lot of raw data. But as we discussed in the simple AI example from the previous chapter - a computer doesn't understand raw data in its natural form. It is not always immediately obvious how the raw data can be turned into numbers that a computer can understand. For example, when an artificial intelligence works with a picture the computer doesn't "see" the picture file itself. It sees a set of numbers that represent that picture and operates on those numbers. The first step in almost every artificial intelligence application is to "pre-process" the data - to take the image files or the movie files or the text of a document and turn it into numbers that a computer can understand. Then those numbers can be fed into algorithms that can make predictions and ultimately be used to make an interface look intelligent.

Turning raw data into numbers

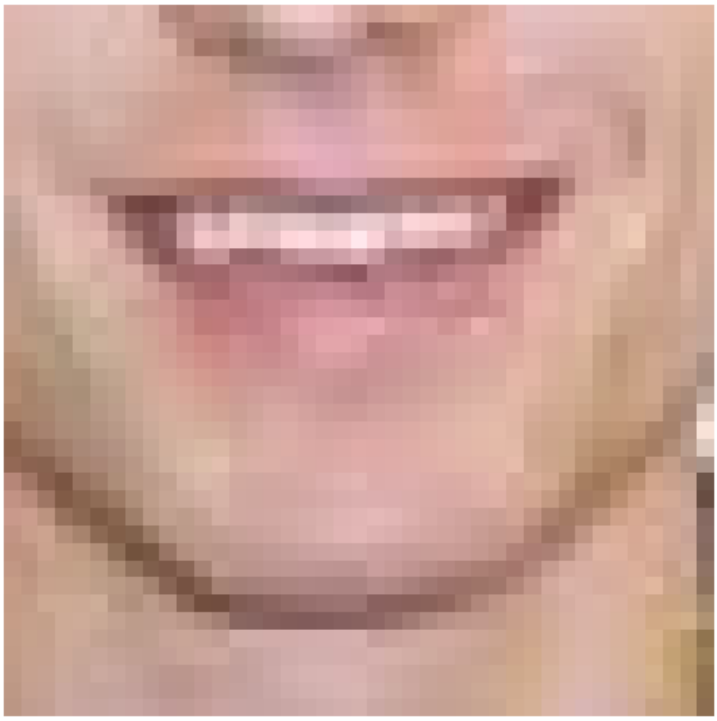
So how do we convert raw data into a form we can work with? It depends on what type of measurement or data you have collected. Here I will use two examples to explain how you can convert images and the text of a document into numbers that an algorithm can be applied to.

Images

Suppose that we were developing an AI to identify pictures of the author of this book. We would need to collect a picture of the author - maybe an embarrassing one.

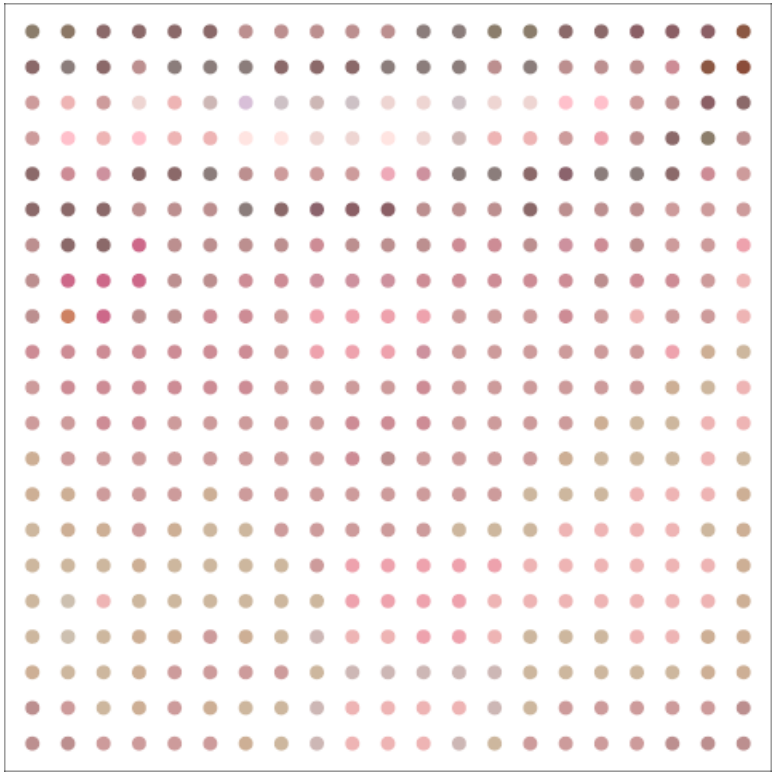
An embarrassing picture of the author

This picture is made of pixels. You can see that if you zoom in very close on the image and look more closely. You can see that the image consists of many hundreds of little squares, each square just one color. Those squares are called pixels and they are one step closer to turning the image into numbers.



A zoomed in view of the author's smile - you can see that each little square corresponds to one pixel and has an individual color

You can think of each pixel like a dot of color. Let's zoom in a little bit more and instead of showing each pixel as a square show each one as a colored dot.



A zoomed in view of the author's smile - now each of the pixels are little dots one for each pixel.

Imagine we are going to build an AI application on the basis of lots of images. Then we would like to turn a set of images into "tidy data". As described above a tidy data set is defined as the following.

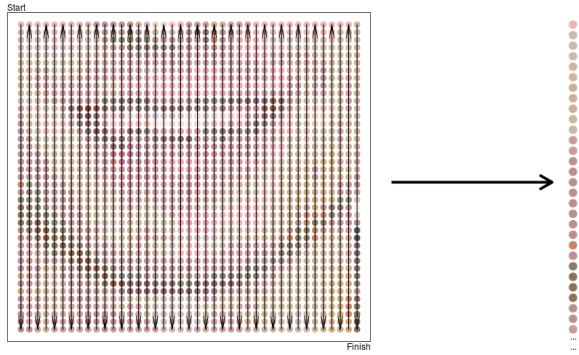
1. Each variable you measured is in one column
2. Each different measurement of that variable is in a different row
3. There is one data table for each "type" of variable.

4. If there are multiple tables then they are linked by a common ID.

A translation of tidy data for a collection of images would be the following.

1. *Variables*: Are the pixels measured in the images. So the top left pixel is a variable, the bottom left pixel is a variable, and so on. So each pixel should be in a separate column.
2. *Measurements*: The measurements are the values for each pixel in each image. So each row corresponds to the values of the pixels for each row.
3. *Tables*: There would be two tables - one with the data from the pixels and one with the labels of each image (if we know them).

To start to turn the image into a row of the data set we need to stretch the dots into a single row. One way to do this is to snake along the image going from top left corner to bottom right corner and creating a single line of dots.



Follow the path of the arrows to see how you can turn the two dimensional picture into a one dimensional picture

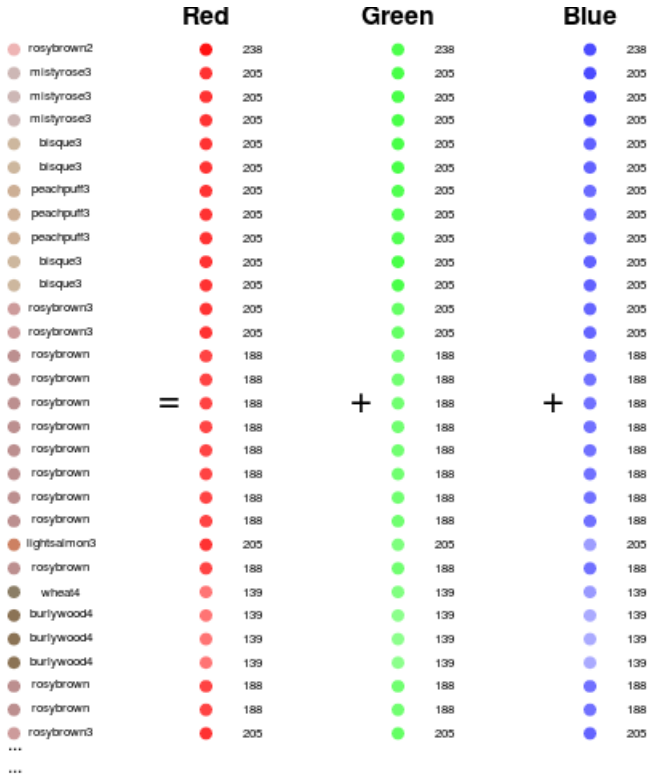
This still isn't quite data a computer can understand - a computer doesn't know about dots. But we could take each dot and label it with a color name.

Labeling each color with a name

We could take each color name and give it a number, something like `rosybrown = 1`, `mistyrose = 2`, and so on. This approach runs into some trouble because we don't have names for every possible color and because it is pretty inefficient to have a different number for every hue we could imagine.

But that would be both inefficient and not very understandable by a computer. An alternative strategy that is often used is to encode the intensity of the red, green, and blue colors for each pixel. This is sometimes called the rgb color model (Wikipedia contributors 2016b). So for example we can take

these dots and show how much red, green, and blue they have in them.



Breaking each color down into the amount of red, green and blue

Looking at it this way we now have three measurements for each pixel. So we need to update our tidy data definition to be:

1. *Variables*: Are the three colors for each pixel measured in the images. So the top left pixel red value is a variable, the top left pixel green value is a variable and so on. So each pixel/color combination should be in a separate

column.

2. *Measurements*: The measurements are the values for each pixel in each image. So each row corresponds to the values of the pixels for each row.
3. *Tables*: There would be two tables - one with the data from the pixels and one with the labels of each image (if we know them).

So a tidy data set might look something like this for just the image of Jeff.

id	label	p1red	p1green	p1blue	p2red	...
1	"jeff"	238	180	180	205	...

Each additional image would then be another row in the data set. As we will see in the chapters that follow we can then feed this data into an algorithm for performing an artificial intelligence task.

Notes

Parts of this chapter from appeared in the Simply Statistics blog post "The vast majority of statistical analysis is not performed by statisticians"[18] written by the author of this book.

[18] http://simplystatistics.org/2013/06/14/the-vast-majority-of-statistical-analysis-is-not-performed-by-statisticians/

References

Buhr, Sarah. 2017. "Illumina Wants to Sequence Your Whole Genome for $100." https://techcrunch.com/2017/01/10/illumina-wants-to-sequence-your-whole-genome-for-100/.

Leskovec, Jure, and Eric Horvitz. 2008. "Planetary-Scale Views on an Instant-Messaging Network," 6∼mar.

"The Cost of Sequencing a Human Genome." n.d. https://www.genome.gov/sequencingcosts/.

Travers, Jeffrey, and Stanley Milgram. 1969. "An Experimental Study of the Small World Problem." *Sociometry* 32 (4). [American Sociological Association, Sage Publications, Inc.]: 425–43.

Venter, J Craig, Mark D Adams, Eugene W Myers, Peter W Li, Richard J Mural, Granger G Sutton, Hamilton O Smith, et al. 2001. "The Sequence of the Human Genome." *Science* 291 (5507). American Association for the Advancement of Science: 1304–51.

Wickham, Hadley, and others. 2014. "Tidy Data." *Under Review*.

Wikipedia contributors. 2016a. "Six Degrees of Kevin Bacon." https://en.wikipedia.org/w/index.php?title=Six_Degrees_of_Kevin_Bacon&oldid=748831516.

———. 2016b. "RGB Color Model." https://en.wikipedia.org/w/index.php?title=RGB_color_model&oldid=756764504.

Learning about Machine Learning with an Earthquake Example

> "A learning machine is any device whose actions are influenced by past experience." - Nils John Nilsson

Machine learning describes exactly what you would think: a machine that learns. As we described in the previous chapter a machine "learns" just like humans from previous examples. With certain experiences that give them an understanding about a particular concept, machines can be trained to have similar experiences as well, or at least mimic them. With very routine tasks, our brains become attuned to characteristics that define different objects or activities.

Before we can dive into the algorithms - like neural networks - that are most commonly used for artificial intelligence, lets consider a real example to understand how machine learning works in practice.

The Big One

Earthquakes occur when the surface of the Earth experiences a shake due to displacement of the ground, and can read-

ily occur along fault lines where there have already been massive displacements of rock or ground(Wikipedia 2017a). For people living in places like California where earthquakes occur relatively frequently, preparedness and safety are major concerns. One famous fault in southern California, called the San Andreas Fault, is expected to produce the next big earthquake in the foreseeable future, often referred to as the "Big One". Naturally, some residents are concerned and may like to know more so they are better prepared.

The following data are pulled from **fivethirtyeight**, a political and sports blogging site, and describe how worried people are about the "Big One" (Hickey 2015). Here's an example of the first few observations in this dataset:

	worry_-general	**worry_-bigone**	**will_occur**
1004	Somewhat worried	Somewhat worried	TRUE
1005	Not at all worried	Not at all worried	FALSE
1006	Not so worried	Not so worried	FALSE
1007	Not at all worried	Not at all worried	FALSE
1008	Not at all worried	Not at all worried	FALSE
1009	Not at all worried	Not at all worried	FALSE
1010	Not so worried	Somewhat worried	FALSE
1011	Not so worried	Extremely worried	FALSE
1012	Not at all worried	Not so worried	FALSE
1013	Somewhat worried	Not so worried	FALSE

Just by looking at this subset of the data, we can already get a feel for how many different ways it could be structured. Here, we see that there are 10 observations which represent 10 individuals. For each individual, we have information on 11 different aspects of earthquake preparedness and experience (only 3 of which are shown here). Data can be stored as text, logical responses (true/false), or numbers. Sometimes, and quite often at that, it may be missing; for example, observation 1013.

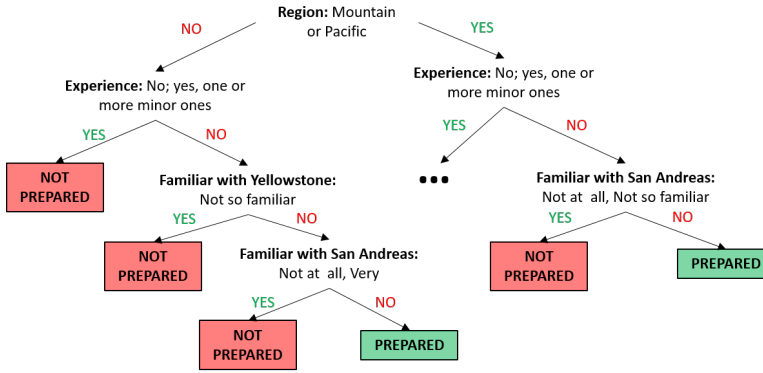
So what can we do with this data? For example, we could predict - or classify - whether or not someone was likely to have taken any precautions for an upcoming earthquake, like bolting their shelves to the wall or come up with an evacuation plan. Using this idea, we have now found a question that we're interested in analyzing: are you prepared for an earthquake or not? And now, based on this question and the data that we have, we can see that you can either be prepared (seen above as "true") or not (seen above as "false").

> Our question: How well can we predict whether or not someone is prepared for an earthquake?

An Algorithm – what's that?

With our question in tow, we want to design a way for our machine to determine if someone is prepared for an earthquake or not. To do this, the machine goes through a flowchart-like set of instructions. At each fork in the flowchart, there are different answers which take the machine on a different path to get to the final answer. If you go through the correct series of questions and answers, it can correctly

identify a person as being prepared. Here's a small portion of the final flowchart for the San Andreas data which we will proceed to dissect (note: the ellipses on the right-hand side of the flowchart indicate where the remainder of the algorithm lies. This will be expanded later in the chapter):



The steps that we take through the flowchart, or **tree** make up the **classification algorithm**. An algorithm is essentially a set of step-by-step instructions that we follow to organize, or in other words, to make a prediction about our data. In this case, our goal is to classify an individual as prepared or not by working our way through the different branches of the tree. So how did we establish this particular set of questions to be in our framework of identifying a prepared individual?

CART, or a classification and regression tree, is one way to assess which of these characteristics is the most important in terms of splitting up the data into prepared and unprepared individuals (Wikipedia 2017b, Breiman et al. (1984)). There are multiple ways of implementing this method, often times with the earlier branches making larger splits in the data, and later branches making smaller splits.

Within an algorithm, there exists another level of organization composed of **features** and **parameters**.

In order to tell if someone is prepared for an earthquake or not, there have to be certain characteristics, known as **features**, that separate those who are prepared from those who are not. Features are basically the things you measured in your dataset that are chosen to give you insight into an individual and how to best classify them into groups. Looking at our sample data, we can see that some of the possible features are: whether or not an individual is worried about earthquakes in general, prior experiences with earthquakes, or their gender. As we will soon see, certain features will carry more weight in separating an individual into the two groups (prepared vs. unprepared).

If we were looking at how important previously experiencing an earthquake was in classifying someone as prepared, we'd say it plays a pretty big role, since it's one of the first features that we encounter in our flowchart. The feature that seems to make a bigger split to our data is region, as it appears as the first feature in our algorithm shown above. We would expect that people in the Mountain and Pacific regions have more experience and knowledge about earthquakes, as that part of the country is more prone to experiencing an earthquake. However, someone's age may not be as important in classifying a prepared individual. Since it doesn't even show up in the top of our flowchart, it means it wasn't as crucial to know this information to decide if a person is prepared or not and it didn't separate the data much.

The second form of organization within an algorithm are the questions and cutoffs for moving one direction or another at each node. These are known as **parameters** of our algorithm.

These parameters give us insight as to how the features we have established define the observation we are trying to identify. Let us consider an example using the feature of region. As we mentioned earlier, we would expect that those in the Mountain and Pacific regions would have more experience with earthquakes, which may reflect in their level of preparedness. Looking back at our abbreviated classification tree, the first node in our tree has a parameter of "Mountain or Pacific" for the feature region, which can be split into "yes" (those living in one of these regions) or "no" (living elsewhere in the US).

If we were looking at a continuous variable, say number of years living in a region, we may set a threshold instead, say greater than 5 years, as opposed to a yes/no distinction. In supervised learning, where we are teaching the machine to identify a prepared individual, we provide the machine multiple observations of prepared individuals and include different parameter values of features to show how a person could be prepared. To illustrate this point, let us consider the 10 sample observations below, and specifically examine the outcome, preparedness, with respect to the features: will_occur, female, and household income.

	prepared	**will_-occur**	**female**	**hhold_-income**
1004	TRUE	TRUE	FALSE	$50,000 to $74,999
1005	FALSE	FALSE	TRUE	$10,000 to $24,999
1006	TRUE	FALSE	TRUE	$200,000 and up

	prepared	will_occur	female	hhold_income
1007	FALSE	FALSE	FALSE	$75,000 to $99,999
1008	FALSE	FALSE	TRUE	Prefer not to answer
1009	FALSE	FALSE	FALSE	Prefer not to answer
1010	TRUE	FALSE	TRUE	$50,000 to $74,999
1011	FALSE	FALSE	TRUE	Prefer not to answer
1012	FALSE	FALSE	TRUE	$50,000 to $74,999
1013	FALSE	FALSE	NA	NA

Of these ten observations, 7 are not prepared for the next earthquake and 3 are. But to make this information more useful, we can look at some of the features to see if there are any similarities that the machine can use as a classifier. For example, of the 3 individuals that are prepared, two are female and only one is male. But notice we get the same distribution of males and females by looking at those who are not prepared: you have 4 females and 2 males, the same 2:1 ratio. From such a small sample, the algorithm may not be able to tell how important gender is in classifying preparedness. But, by looking through the remaining features and a larger sample, it can start to classify individuals. This is what we mean when we say a machine learning algorithm

learns.

Now, let us take a closer look at observations 1005, 1011, and 1012, and more specifically the household income feature:

	prepared	will_occur	female	hhold_income
1005	FALSE	FALSE	TRUE	$10,000 to $24,999
1011	FALSE	FALSE	TRUE	Prefer not to answer
1012	FALSE	FALSE	TRUE	$50,000 to $74,999

All three of these observations are females and believe that the "Big One" won't occur in their lifetime. But despite the fact that they are all unprepared, they each report a different household income. Based on just these three observations, we may conclude that household income is not as important in determining preparedness. By showing a machine different examples of which features a prepared individual has (or unprepared, as in this case), it can start to recognize patterns and identify the features, or combination of features, and parameters that are most indicative of preparedness.

In summary, every flowchart will have the following components:

1. **The algorithm** - The general workflow or logic that dictates the path the machine travels, based on chosen features and parameter values. In turn, the machine

classifies or predicts which group an observation belongs to
2. **Features** - The variables or types of information we have about each observation
3. **Parameters** - The possible values a particular feature can have

Even with the experience of seeing numerous observations with various feature values, there is no way to show our machine information on every single person that exists in the world. What will it do when it sees a brand new observation that is not identified as prepared or unprepared? Is there a way to improve how well our algorithm performs?

Training and Testing Data

You may have heard of the terms *sample* and *population*. In case these terms are unfamiliar, think of the population as the entire group of people we want to get information from, study, and describe. This would be like getting a piece of information, say income, from every single person in the world. Wouldn't that be a fun exercise!

If we had the resources to do this, we could then take all those incomes and find out the average income of an individual in the world. But since this is not possible, it might be easier to get that information from a smaller number of people, or *sample*, and use the average income of that smaller pool of people to represent the average income of the world's population. We could only say that the average income of the sample is *representative* of the population if the sample

of people that we picked have the same characteristics of the population.

For example, if we assumed that our population of interest was a company with 1,000 employees, where 200 of those employees make $60,000 each and 800 of them make $30,000 each. Since we have this information on everyone, we could easily calculate the average income of an employee in the company, which would be $36,000. Now, say we randomly picked a group of 100 individuals from the company as our sample. If all of those 100 individuals came from the group of employees that made $60,000, we might think that the average income for an employee was actually much higher than the true average of the population (the whole company). The opposite would be true if all 100 of those employees came from the group making less money - we would mistakenly think the average income of employees is lower. In order to accurately reflect the distribution of income of the company employees through our sample, or rather to have a *representative* sample, we would try to pick 20 individuals from the higher income group and 80 individuals from the lower income group to get an accurate representation of this company population.

Now heading back to our earthquake example, our big picture goal is to be able to feed our algorithm a brand new observation of someone who answered information about themselves and earthquake preparedness, and have the machine be able to correctly identify whether or not they are prepared for a future earthquake.

One definition of a population could consist of all individuals in the world. However, since we can't get access to data on all these individuals, we decide to sample 1013 respondents and ask them earthquake related questions. Remember that in

order for our machine to be able to accurately identify an individual as prepared or unprepared, it needs to have had some experience seeing some observations where features identify the individual as prepared, as well as some observations that aren't. This seems a little counterintuitive though. If we want our algorithm to identify a prepared individual, why wouldn't we show it all the observations that are prepared?

By showing our machine observations of respondents that are not prepared, it can better strengthen its idea of what features identify a prepared individual. But we also want to make our algorithm as *robust* as possible. For an algorithm to be robust, it should be able to take in a wide range of values for each feature, and appropriately go through the algorithm to make a classification. If we only show our machine a narrow set of experiences, say people who have an income of between $10,000 and $25,000, it will be harder for the algorithm to correctly classify an individual who has an income of $50,000.

One way we can give our machine this set of experiences is to take all 1013 observations and randomly split them up into two groups. Note: for simplification, any observations that had missing data (total: 7) for the outcome variable were removed from the original dataset, leaving 1006 observations for our analysis.

1. **Training data** - This serves as the wide range of experiences that we want our machine to see to have a better understanding of preparedness
2. **Testing data** - This data will allow us to evaluate our algorithm and see how well it was able to pick up on features and parameter values that are specific to prepared individuals and correctly label them as such

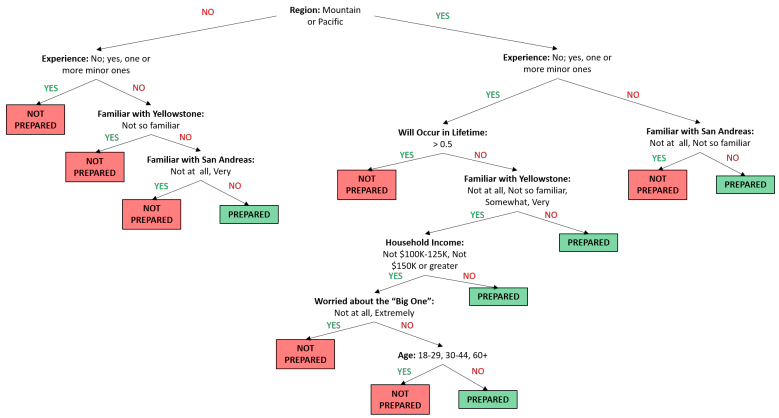
So what's the point of splitting up our data into training and testing? We could have easily fed all the data that we have into the algorithm and have it detect the most important features and parameters we have based on the provided observations. But there's an issue with that, known as **overfitting**. When an algorithm has overfit the data, it means that it has been fit specifically to the data at hand, and only that data. It would be harder to give our algorithm data that does not fit within the bounds of our training data (though it would perform very well in this sample set). Moreover, the algorithm would only accurately classify a very narrow set of observations. This example nicely illustrates the concept we introduced earlier - *robustness* - and demonstrates the importance of exposing our algorithm to a wide range of experiences. We want our algorithm to be useful, which means it needs to be able to take in all kinds of data with different distributions, and still be able to accurately classify them.

To create training and testing sets, we can adopt the following idea:

1. Split the 1006 observations in half: roughly 500 for training, and the remainder for testing
2. Feed the 500 training observations through the algorithm for the machine to understand what features best classify individuals as prepared or unprepared
3. Once the machine is trained, feed the remaining test observations through the algorithm and see how well it classifies them

Algorithm Accuracy

Now that we've built up our algorithm and split our data into training and test sets, let's take a look at the full classification algorithm:



Recall the question we set out to answer with respect to the earthquake data: **How well can we predict whether or not someone is prepared for an earthquake?** In a binary (yes/no) case like this, we can set up our results in a 2x2 table, where the rows represent predicted preparedness (based on the features of our algorithm) and the columns represent true preparedness (what their true label is).

		True Outcome		
		Prepared	Unprepared	
Predicted Outcome	Prepared	34	36	70
	Unprepared	75	358	433
		109	394	503

This simple 2x2 table carries quite a bit of information. Essentially, we can grade our machine on how well it learned to tell whether individuals are prepared or unprepared. We can see how well our algorithm did at classifying new observations by calculating the **predictive accuracy**, done by summing cells A and C and dividing by the total number of observations, or more simply, (A + C) / N. Through this calculation, we see that the algorithm from our example correctly classified individuals as prepared or unprepared 77.9% of the time. Not bad!

When we feed the roughly 500 test observations through the algorithm, it is the first time the machine has seen those observations. As a result, there is a chance that despite going through the algorithm, the machine **misclassified** someone as prepared, when in fact they were unprepared. To determine how often this happens, we can calculate the **test error rate** from the 2x2 table from above. To calculate the test error rate, we take the total number of observations that are *discordant*, or dissimilar between true and predicted status, and divide this total by the total number of observations that were assessed. Based on the above table, the test error rate would be (B + C) / N, or 22.1%.

There are a number of reasons that a test error rate would be high. Depending on the data set, there might be different methods that are better for developing the algorithm. Additionally, despite randomly splitting our data into training and testing sets, there may be some inherent differences between the two (think back to the employee income example above), making it harder for the algorithm to correctly label an observation.

References

Breiman, Leo, Jerome H Friedman, Richard A Olshen, and Charles J Stone. 1984. "Classification and Regression Trees. Wadsworth & Brooks." *Monterey, CA*.

Hickey, Walt. 2015. "The Rock Isn't Alone: Lots of People Are Worried About 'the Big One'." *FiveThirtyEight*. FiveThirtyEight. https://fivethirtyeight.com/datalab/the-rock-isnt-alone-lots-of-people-are-worried-about-the-big-one/.

Wikipedia. 2017a. "Earthquake — Wikipedia, the Free Encyclopedia." http://en.wikipedia.org/w/index.php?title=Earthquake&oldid=762614740.

———. 2017b. "Predictive analytics — Wikipedia, the Free Encyclopedia." http://en.wikipedia.org/w/index.php?title=Predictive%20analytics&oldid=764577274.