tiny [table]font=tiny

archaeologyname=archaeology, description=Michel Foucault defines archaeology as a description that questions the already-said at the level of its existence. Alternately, archaeology describes discourses as practices specified in the element of the archive **Foucault_1972**; **131**

Machine Learning: Archaeology of a Data Practice

Adrian Mackenzie

Contents

List of Figures

List of Tables

Listings

Chapter 1

Introduction: Into the Data

> *Definition*: A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, improves with experience E (Mitchell 1997, 2).

> In the past fifteen years, the growth in algorithmic modeling applications and methodology has been rapid. It has occurred largely outside statistics in a new community—often called machine learning—that is mostly young computer scientists. The advances, particularly over the last five years, have been startling (Breiman 2001, 200)

> The key question isn't 'How much will be automated?' It's how we'll conceive of whatever *can't* be automated at a given time. (Lanier 2013, 77)

A relatively new field of scientific-engineering devices said to 'learn from experience' has become operational in the last three decades. Known by various names – machine learning, pattern recognition, knowledge discovery,

code!machine learning as

machine learner!Skynet

machine learner!logistic regression

data science!relation to machine learning

Google Search

Apple Siri

Google!TensorFlow

Facebook!news feed

data mining – the field and its devices, which all take shape as computer programs or code, seem to have quickly spread across scientific disciplines, business and commercial settings, industry, engineering, media, entertainment and government. Heavily dependent on computation, they are found in breast cancer research, in autonomous vehicles, in insurance risk modelling, in credit transaction processing, in computer gaming, in face and handwriting recognition systems, in astronomy, advanced prosthetics, ornithology, finance, surveillance (see the U.S. Government's `SkyNet` for one example of a machine learning surveillance system (Agency 2012)) or robots(see a Google robotic arm farm learning to sort drawers of office equipment such as staplers, pens, erasers and paperclips (Levine et al. 2016). Sometimes machine learning devices are understood as *scientific models*, and sometimes they are understood as *operational algorithms*. In very many scientific fields, publications mention or describe these techniques as part of their analysis of some experimental or observational data (as in the logistic regression classification models found in a huge number of biomedical papers). They anchor the field of 'data science' (Schutt and O'Neil 2013), as institutionalised in several hundred data science institutes scattered worldwide. Not so recently, they also became mundane mechanisms deeply embedded in other systems or gadgets (as in the decision tree models used in some computer game consoles to recognise gestures, the neural networks used to recognise voice commands by search engine services such as `Google Search` and `Apple Siri` (McMillan 2013) or Google's `TensorFlow` software packages that puts deep convolutional neural nets on Android devices (Google 2015)). In platform settings, they operate behind the scenes as part of the everyday functioning of services ranging from player ranking in online games to border control face recognition, from credit scores to news feeds on Facebook . In all of these settings, applications and fields, machine learning is said to transform the nature of knowledge. Might it transform the practice of

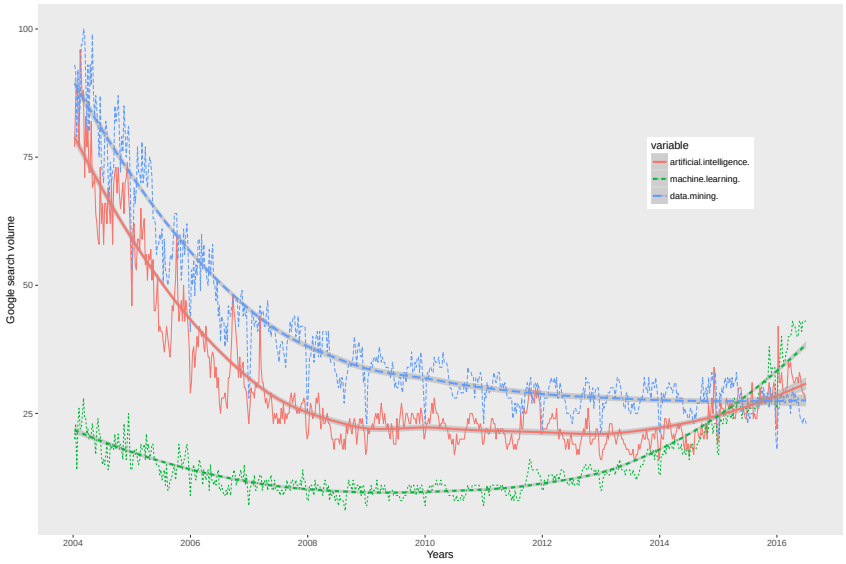


Figure 1.1: Google Trends search volume for 'machine learning' and related query terms in English, globally 2004-2015

critical thought? This book is an experiment in such practice.

Three accumulations: settings, data and devices

Three different accumulations cross-stratify in machine learning: settings, data and devices. The volume and geography of searches on Google Search provides some evidence of the diverse settings or sites doing machine learning. If we search for terms such as `artificial intelligence`, `machine learning` and `data mining` on the Google Trends service , the results for the last decade or so suggest shifting interest in these topics.

In Figure 1.1, two general search terms that had a very high search volume in 2004 – 'artificial intelligence' and 'data mining' – slowly decline over the years before starting to increase again in the last few years. By contrast, `machine learning` loses volume until around 2008, and then gradually rises again so that by mid-2016 it exceeds the long-standing interests in data-mining and artificial intelligence. Whatever the difficulties in under-

critical thought!practice of
Google!Google Trends
artificial intelligence
data mining

accumulation!of settings
Facebook!AI-Flow
machine learner!Skynet
data!plenitude
data!practice
image recognition
kittydar
Arthur, Heather

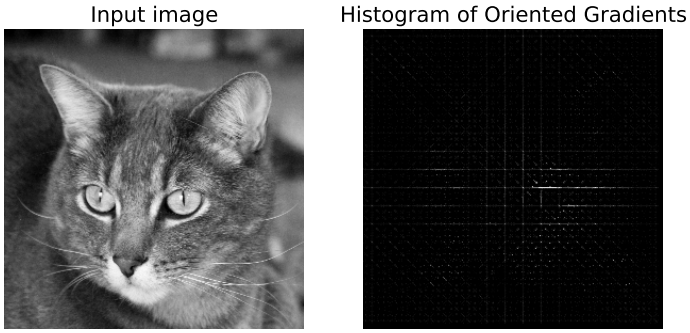


Figure 1.2: Close up of cat. The image on the left is already signal-processed as a JPEG format file. The image on the right is further processed using histogram of oriented gradients (HOG) edge detection. `kittydar` models HOG features. Cat photo courtesy photos-public-domain.com

standing GoogleTrends results, these curves suggest an accumulation of sites and settings turning towards machine learning.[^0.3] What does it mean that machine learning surfaces in so many different places, from fMRIs to Facebook's `AI-Flow` (**Facebook__2016**), , from fisheries management to Al Queda courier network monitoring by `SkyNet`?[^0.0987]

A second accumulation concerns the plenitude of things in the world as data. . If we wanted to describe the general horizon of machine learning as a data practice in its specificity, we might turn to cats. Cat images accumulate on websites and social media platforms as de-centred, highly repetitive data forms. Like the billions of search engine queries, email messages, tweets, or much contemporary scientific data (e.g. the DNA sequence data discussed in chapter **??**), images accumulate in archives of communication. Take the case of `kittydar,` a machine learner in the area of image recognition (see kittydar): 'Kittydar is short for kitty radar. Kittydar takes an image (canvas) and tells you the locations of all the cats in the image' (Arthur 2012) . This playful piece of code demonstrates how machine learning can be amidst mundane accumulation. Heather Arthur, who developed `kittydar` writes:

> Kittydar first chops the image up into many "windows" to test for the presence of a cat head. For each window, kittydar first extracts more tractable data from the image's data. Namely, it computes the Histogram of Orient Gradients descriptor of the image … This data describes the directions of the edges in the image (where the image changes from light to dark and vice versa) and what strength they are. This data is a vector of numbers that is then fed into a neural network … which gives a number from 0 to 1 on how likely the histogram data represents a cat. The neural network … has been pre-trained with thousands of photos of cat heads and their histograms, as well as thousands of non-cats. See the repo for the node training scripts (Arthur 2012). a

data!image as
data!vector
data!vector
machine learner!neural net
code!circulation of
gradient|see gradient descent

This toy device finds cat heads in digital photographs, but also exemplifies many key traits of machine learning. Large accumulations of things become in a dataset. A dataset is used to train a typical machine learning device, a neural net, and the neural net *classifies* subsequent images probabilistically. The code for all this is 'in the repo.' Based on how it locates cats, we can begin to imagine similar pattern recognition techniques in use in self-driving cars (Thrun et al. 2006), border control facial recognition systems, military robots or wherever something seen implies something to do.

Faced with the immense accumulation of cat images on the internet, `kittydar` can do very little. It only detects the presence of cats that face forward. And it sometimes classifies people as cats. As Arthur's description suggests, the software finds cats by cutting the images into smaller windows. For each window, it measures a set of gradients – a spatial order of great significance in machine learning - running from light and dark, and then compares these measurements to the gradients of known cat images

data!training
classification
neural network|see machine learner!neural net
support vector machine|see machine learner!support vector machine
linear regression
machine learner!logistic regression
machine learner!neural net
machine learner!linear discriminant analysis
support vector machine
k-means clustering
\
nearest neighbours|see machine learner!_k_ nearest neighbours
principal component analysis
machine learner!Naive Bayes
data!practices

(the so-called 'training data'). The work of classification according to the simple categories of 'cat' or 'not cat' is given either to a neural network (as discussed in chapter **??**, a typical machine learning technique and one that has recently been heavily developed by researchers at Google (Le et al. 2011), themselves working on images of cats among other things taken from Youtube videos (BBC 2012), or to a support vector machine (a technique first developed in the 1990s by researchers working at IBM; see chapter **??**).

A final accumulation comprises machine learning techniques and devices. Machine learning range from the mundane to the esoteric, from code miniatures such as `kittydar` to the infrastructural sublime of computational clouds and clusters twirling in internet data-streams. Like the images that `kittydar` classifies, the names of machine learning techniques and devices proliferate and accumulate in textbooks, instructional courses, website tutorials, software libraries and code listings: linear regression, logistic regression, neural networks, linear discriminant analysis, support vector machines, k-means clustering, decision treesdecision tree|see {machine learner!decision tree}, k nearest neighbours, random forests, principal component analysis, or naive Bayes classifier to name just some of the most commonly used. Sometimes they have proper names: `RF-ACE`, `Le-Net5`, or `C4.5`. These names refer to predictive models and to computational algorithms of various ilk and provenance. Intricate data practices – normalization, regularization, cross-validation, feature engineering, feature selection, optimisation – embroider datasets into shapes they can recognise. The techniques, algorithms and models are not necessarily startling new or novel. They take shape against a background of more than a century of work in mathematics, statistics, computer science as well as disparate scientific fields ranging from anthropology to zoology. Mathematical constructs drawn from linear

algebra, differential calculus, numerical optimization and probability theory pervade practice in the field. Machine learning itself is an accumulation rather than a radical transformation.

mathematics

machine learner|textbf

human-machine relations

positivity!of knowledge

positivity!threshold of

\

Who or what is a machine learner?

I am focusing on machine learners – a term that refers to both humans and machines or human-machine relations throughout this book – situated amidst these three accumulations of settings, data and devices. While it is not always possible to disentangle machine learners from the databases, infrastructures, platforms or interfaces they work through, I will argue that data practices associated with machine learning delimit a of knowing. The term 'positivity' comes from Michel Foucault's *The Archaeology of Knowledge* (Foucault 1972), and refers to specific forms of accumulation of grouped in a discursive practice. Foucault attributes a lift-off effect to positivity:

> The moment at which a discursive practice achieves individuality and autonomy, the moment therefore at which a single system for the formation of statements is put into operation, or the moment at which this system is transformed, might be called the threshold of positivity. (186)Foucault, Michel!

Machine learners today circulate into domains that lie far afield of the eugenic and psychology laboratories, industrial research institutes or specialized engineering settings in which they first took shape (in some cases, such as the linear regression model or principal component analysis, more than a century ago; in others such as support vector machines or random forests, in the last two decades). If they are not exactly new and have diverse genealogies, the question is: what happen as machine learners shift from

generalization
operational formation
statements!forms of
code!writing of
\

localized mathematical or engineering techniques to an everyday device that can be generalized to locate cats in digital images, the Higgs boson in particle physics experiments or fraudulent credit card transactions? Does the somewhat unruly generalization of machine learning across different epistemic, economic, institutional settings – the pronounced uptick shown in Figure 1.1 – attest to a re-definition of knowledge, decision and control, a new operational formation in which 'a system is transformed'?

Algorithmic control to the machine learners?

Written in code, machine learners operate as programs or computational processes to produce statements that may take the form of numbers, graphs, propositions (see for instance the propositions produced by a recurrent neural net on the text of this book in the concluding chapter **??**). Machine learning can also be viewed as a change in how programs or the code that controls computer operations are developed and operate (see chapter 2 for more detailed discussion of this}. The term 'learning' in machine learning points to this change and many machine learners emphasize it. Pedro Domingos, for instance, a computer scientist at the University of Washington, writes:

> Learning algorithms – also known as learners – are algorithms that make other algorithms. With machine learning, computers write their own programs, so we don't have to.(Domingos 2015, 6)Domingos, Pedro|on machine learning]

Viewed from the perspective of control, and how control is practiced, digital computer programs stem from and epitomise the 'control revolution' (Beniger 1986) that arguably has, since the late nineteenth century, programmatically re-configured production, distribution, consumption, and

bureaucracy by tabulating, calculating and increasingly communicating events and operations. With the growth of digital communication networks in the form of the internet, the late 20th century entered a new crisis of control, no longer centred on logistic acceleration but on communication and knowledge. . Almost all accounts of the operational power of machine learning emphasise its power to inflect the control of processes of communication – border flows, credit fraud, spam email, financial market prices, cancer diagnosis, targetted online adverts – processes whose unruly or transient multiplicity otherwise evades or overwhelm us – with knowledge (classifications and predictions in particular) derived from algorithms that make other algorithms. On this view, `kittydar` can isolate cats amidst the excessive accumulation of images on the internet because neural net learning algorithms (back-propagation, gradient descent) have written a program – 'a pre-trained' neural net – during its training phase.

control!crisis of
Beniger, James
machine learner!\textttkittydar
facial recognition
Amoore, Louise

If a newly programmatic field of knowledge-control takes shape around machine learning, how would we distinguish it from computation more generally? Recent critical research on algorithms offers one lead. In a study of border control systems, which often use machine learners to do profiling and facial recognition , Louise Amoore advocates attention to calculation and algorithms:

> Surely this must be a primary task for critical enquiry – to uncover and probe the moments that come together in the making of a calculation that will automate all future decisions. To be clear, I am not proposing some form of humanist project of proper ethical judgement, but rather calling for attention to be paid to the specific temporalities and norms of algorithmic techniques that rule out, render invisible, other potential futures (Amoore 2011).

Munster, Anna
calculation|see mathematics!calculation
decision
machine learning!human-machine difference
machine learning|as automation
infrastructure
calculation!historical specificity of

As Amoore writes, some potential futures are being 'ruled out' as calculations automate decisions. Anna Munster puts the challenge more bluntly: 'prediction takes down potential' (Munster 2013). I find much to agree with here. Machine learning is a convoluted but nevertheless concrete and historically specific form of calculation (as we will see in exploring algebraic operations in chapter **??**, in finding and optimising certain mathematical functions in chapter **??** or in characterising and shaping probability distributions in chapter **??**). It works to mediate future-oriented decisions (although all too often, very near-future decisions such as ad-click prediction).

I am less certain about treating machine learning as automation. Learning from data, as we will see, does often sidestep and substitute for existing ways of acting, and practices of control, and it thereby re-configures human-machine differences. Yet the notion of automation does not capture well how this comes about. The programs that machine learner 'write' are formulated as probabilistic models, as learned rules or association, and they generate predictive and classificatory statements ('this is a cat'). They render calculable some things which hitherto appeared intractable to calculation (for instance, the argument of a legal case). Such calculation, with all the investment it attracts (in the form of professional lives, in the form of infrastructures , in reorganisation of institutions, corporations and governments, etc.) does rule out some and reinforce other futures. [^0.001] If this transformed calculability is automation, we need to understand the specific contemporary reality of automation as it takes shape in machine learning. We cannot conduct critical enquiry into the calculation that will automate future decisions without opening the very notions of calculation and automation into question.

Does the concept of algorithm help us identify the moments that come together in machine learning without resorting to a-historical concepts

of automation or calculation? In various scholarly and political debates around changes business, media, education, health, government or science, quasi-omnipotent agency has been imputed to algorithms (Pasquinelli 2014; Neyland 2015; Totaro and Ninno 2014; Smith 2013; Beer and Burrows 2013; Fuller and Goffey 2012; Wilf 2013; Barocas, Hood, and Ziewitz 2013; Gillespie 2014; Cheney-Lippold 2011; A. R. Galloway 2004) or sometimes just 'the algorithm.' This growing body of work understands the power of algorithms in the social science and humanities literature in different ways, sometimes in terms of rules, sometimes as functions or mathematical abstractions, and increasingly as a located practice. but there is general agreement that algorithms are powerful, or at least, can bear down heavily on people's lives and conduct, re-configuring, for instance, culture as algorithmic (Hallinan and Striphas 2014).

algorithm!primacy
Pasquinelli, Paolo
algorithm!as abstraction

Some of the critical literature on algorithms identifies abstractions as the source of their power. For instance, in his discussion of the 'metadata society,' Paolo Pasquinelli proposes that

> a progressive political agenda for the present is about moving at the same level of abstraction as the algorithm in order to make the patterns of new social compositions and subjectivities emerge. We have to produce new revolutionary institutions out of data and algorithms. If the abnormal returns into politics as a mathematical object, it will have to find its strategy of resistance and organisation, in the upcoming century, in a mathematical way (Pasquinelli 2015).

'Moving at the same level of abstraction as the algorithm' offers some purchase as a formulation for critical practice, and for experiments in such practice. Since in mathematics let alone critical thought, abstraction can be

mathematics!historicity of
abstraction!algorithm as
human-machine relations!practice in
knowledge!power

understood in many different ways, any direct identification of algorithms with abstraction will, however, be difficult to practice. Which algorithm, what kind of abstraction and which 'mathematical way' should we focus on? Like automation and calculation, abstraction and mathematics have mutable historicities. We cannot 'move at the same level' without taking that into account. Furthermore, given the accumulations of settings, data and devices, there might not be any single level of abstraction to move at, only a torque and flux of different moments of abstraction at work in generalizing, classifying, circulating and stratifying in the midst of transient and plural multiplicities.

The archaeology of operations

Given mathematics and algorithms do loom large in machine learning, how do we address their the workings without pre-emptively ascribing potency to mathematics, or to algorithms? In the chapters that follow, I do explore specific learning algorithms (gradient descent in chapter **??** or recursive partitioning **??**) and mathematical techniques (the sigmoid function in chapter **??** or inner products in chapter **??**) in greater empirical and conceptual depth. Following much scholarship in science and technology studies, I maintain that attention to specificity of practices is an elementary prerequisite to understanding human-machine relations, and their transformations. The archaeology of operations that I will develop combines an interest in machine learning as a form of knowledge production and a strategy of power. Like Foucault, I see no exteriority between techniques of knowledge and strategies of power ('between techniques of knowledge and strategies of power, there is no exteriority, even if they have specific roles and are linked together on the basis of their difference' (Foucault 1998, 98)).

If we understand machine learning as a data practice that re-configures local centres of power-knowledge through a re-drawing of human-machine relations, then the specific roles and differences associated with machine learners in the production of knowledge should be a focus of attention. Differences are a key concern here since many machine learners classify things. They are often simply called '.' Some of the practice of difference works in terms of categories. `Kittydar` classifies images as `cat` with some probability, but categorisation and classification in machine learning occurs much more widely.[^0.007] We might understood the importance of categories sociologically. For instance, in his account of media power, Nick Couldry highlights the importance of categories and categorisation:

> *Category* is a key mechanism whereby certain types of ordered (often 'ritualized') practice produce power by enacting and embodying categories that serve to mark and divide up the world in certain ways. Without *some* ordering feature of practice, such as 'categories', it is difficult to connect the multiplicity of practice to the workings of power, whether in the media or in any other sphere. By understanding the work of categories, we get a crucial insight into why the social world, in spite of its massive complexity still appears to us as a *common* world (Couldry 2012, 62) ,

The orderings of categorical differences undergo a great deal of intensification via machine learning. Categories are often simply an existing set of classifications derived from institutionalised or accepted knowledges (for instance, the categories of customers according to gender or age). Machine learners also generate new categorical workings or mechanisms of differentiation. As we will see (for instance in chapter **??** in relation to scientific data from genomes), machine learners invent or find new sets of

classifier
Couldry, Nick
categories|seealso differences|categories
power

machine learning!regularization

differences!orderings of

categories for a particular purpose (such as cancer diagnosis or prognosis). These differentiations may or may not bring social good. The person who finds themselves paying a higher price for an air ticket by virtue of some unknown combination of factors including age, credit score, home address, previous travel, or educational qualifications experiences something of the classificatory power.

Both the intensification of ordering categories and the invention of new classifications feed directly into the regulation of conduct, communication, and ways of living, as will be familiar to readers of Foucault (Foucault 1977). the mathematical abstractions, whether they come from calculus, linear algebra, statistics, or topology, maximise regularities. The power of machine learning to find patterns, to classify or predict regularizes and normalizes differences, although perhaps via some novel ordering practice.

Asymmetries in a common world

What can critical thought, the kind of enquiry that seeks to identify the conditions that concretely constitute what anyone can say or think or do, learn from machine learning? If we see a massive regularization of order occurring in machine learning, what is at stake in trying to think through those practices? They display moments of formalisation (especially mathematical and statistical), circulation (pedagogically and operationally), generalization (propagating and proliferating in many domains and settings) and stratification (the socially, epistemically, economically and sometimes politically or ontologically loaded re-iterative enactment of categories). I am not sure that understanding how a support vector machine or a random forest orders differences would change how would we relate to what we see, feel, sense, hear or think in the face of a contemporary webpage such as

Amazon's that uses Association Rule Mining , an app, a passport control system that matches faces of arriving passengers with images in a database, a computer game, or a genetic test (all settings in which machine learning is likely to be operating).

Machine learners themselves sometimes complain of the monolithic and homogeneous success of machine learning. Some expert practitioners complain of a uniformity in its applications. Jeff Hammerbacher , previously chief research scientist at Facebook, co-founder of a successful data analytics company called Cloudera, and currently working also on cancer research at Mount Sinai hospital, complained about the spread of machine learning in 2011: 'the best of my generation are thinking about how to make people click ads' (Vance 2011) . Leaving aside debates about the ranking of 'best minds' (a highly competitive and exhaustively tested set of subject positions; see chapter **??**), Hammerbacher was lamenting the flourishing use of predictive analytics techniques in online platforms such as Twitter, Google and Facebook, and on websites more generally, whether they be websites that sell things or advertising space. The mathematical skills of many PhDs from MIT, Stanford or Cambridge were wrangling data in the interests of micro-targeted advertising. As Hammerbacher observers, they were 'thinking about how to make people click ads,' and this 'thinking' mainly took and does take the form of building predictive models that tailored the ads shown on websites to clusters of individual preferences and desires.

Hammerbacher's unhappiness with ad-click prediction resonates with critical responses to the use of machine learning in the digital humanities. Some versions of the digital humanities make extensive use of machine learning. To cite one example, in *Macroanalysis: Digital Methods and Literary History*, Matthew Jockers describes how he relates to one currently popular machine

topic model
Mohr, John
Jockers, Matthew!on topic models
data!latent variables in
Galloway, Alex|on knowledge production

learning or statistical modelling technique, the topic model (itself the topic of discussion in Chapter **??**; see also (Mohr and Bogdanov 2013)):

> If the statistics are rather too complex to summarize here, I think it is fair to skip the mathematics and focus on the end results. We needn't know how long and hard Joyce sweated over *Ulysses* to appreciate his genius, and a clear understanding of the LDA machine is not required in order to see the beauty of the result. (Jockers 2013, 124)

The widely used Latent Dirichlet Allocation or models provide a litmus test of how relations to machine learning is taking shape in the digital humanities. On the one hand, these models promise to make sense of large accumulations of documents (scientific publications, news, literature, online communications, etc.) in terms of underlying themes or latent 'topics.' As we will, large document collections have long attracted the interest of machine learners (see chapter **??**). On the other hand, Jockers signals the practical difficulties of relating to machine learning when he suggests that 'it is fair to skip the mathematics' for the sake of 'the beauty of the result'. While some parts of the humanities and critical social research exhorts closer attention to algorithms and mathematical abstractions, other parts elides its complexity in the name of 'the beauty of the results.'

Critical thought has not always endorsed the use of machine learning in digital humanities. Alex Galloway makes two observations about the circulation of these methods in humanities scholarship. The first points to its marginal status in increasingly machine-learned media cultures:

> When using quantitative methodologies in the academy (spidering, sampling, surveying, parsing, and processing), one must

decision tree

credit scoring|FICO and Equifax

> compete broadly with the sorts of media enterprises at work in the contemporary technology sector. A cultural worker who deploys such methods is little more than a lesser Amazon or a lesser Equifax.110 (A. Galloway 2014, 110)

Galloway highlights the asymmetry between humanities scholars and media enterprises or credit score agencies (Equifax). The 'quantitative methodologies' that he refers to as spidering, sampling, processing and so forth are more or less all epitomised in machine learning techniques (for instance, the Association Rule Mining techniques used by Amazon to recommend purchases, or perhaps the decision tree techniques used by the credit-rating systems at Equifax and FICO (Fico 2015)). Galloway's argument is that the infrastructural scale of these enterprises along with the sometime very large technical workforces they employ to continually develop new predictive techniques dwarfs any gain in efficacy that might accrue to humanities research in its recourse to such methods.

Galloway also observes that even if 'cultural workers' do manage to learn to machine learn, and become adept at re-purposing the techniques in the interests of analyzing culture rather than selling things or generating credit scores,they might actually reinforce power asymmetries and exacerbate the ethical and political challenges posed by machine learning:

> Is it appropriate to deploy positivistic techniques against those self-same positivistic techniques? In a former time, such criticism would not have been valid or even necessary. Marx was writing against a system that laid no specific claims to the apparatus of knowledge production itself—even if it was fueled by a persistent and pernicious form of ideological misrecognition. Yet, today the state of affairs is entirely reversed. The new spirit of capitalism

> is found in brainwork, self-measurement and self-fashioning, perpetual critique and innovation, data creation and extraction. In short, doing capitalist work and doing intellectual work—of any variety, bourgeois or progressive—are more aligned today than they have ever been (A. Galloway 2014, 110).

This perhaps is a more serious charge concerning the nature of any knowledge produced by machine learning. The 'techniques' of machine learning may or may not be positivist, and indeed, given the claims that machine learning transforms the production of knowledge, positivism may not be any more stable than other conceptual abstractions. Hence, it might not be so strongly at odds with critical thought, even if remains complicit – 'aligned' – with capitalist work. Intellectual work of the kind associated with machine learning is definitely at the centre of many governmental, media, business and scientific fields of operation and increasingly they anchor the operations of these fields. Yet neither observation – asymmetries in scale, alignment with a 'positivist' capitalist knowledge economy – exhaust the potentials of machine learning, particularly if, as many people claim, it transforms the nature of knowledge production and hence 'brainwork.'

What cannot be automated?

Jaron Lanier's question – how will we conceive at a given time what cannot be automated? – suggests an alternative angle of approach. Like Galloway, I'm wary of certain deployments of machine learning, particularly the platform-based media empires and their efforts to capture sociality (Gillespie 2010; Van Dijck 2012). Machine learners do seem to be 'laying claim to the apparatus of knowledge production.' Yet even amidst the jarring ephemerality of targeted online advertising or the more elevated analytics of

literary history, the transformations in knowledge and knowing do not simply simply appropriate intellectual work to capitalist production. Empirical work to describe differences, negotiations, modifications and contestation of knowledge would be needed to show the unevenness and variability of that appropriation. As I have already suggested, machine learning practice is not simply automating existing economic relations or even data practices. While Hammerbacher and Galloway are understandably pessimistic about the existential gratifications and critical efficacy of building targeted advertising systems or document classifiers, the 'deployment' of machine learning is not a finished process, but very much in train, constantly subject to revision, re-configuration and alteration.

machine learning!coincidence with critical thought

machine learner!as human-machine relation

experiment!in critical thought

Importantly, the familiar concerns of critical social thought to analyse differences, power, materiality, subject positions, agency, etc. somewhat overlap with the claims that machine learning produces knowledge of differences, of nature, cultural processes, communication and conduct. Unlike other objects of critical thought, machine learners (understood always as human-machine relations) are themselves closely interested in producing knowledge, albeit scientific, governmental or operational. This coincidence of knowledge projects suggests the possibility of some different articulation, of modification of the practice of critical thought in its empirical and theoretical registers. The altered human-machine relations we see as machine learners might shift and be re-drawn through experiments in empiricism and theory.

Where in the algorithms, calculations, abstractions and regularizing practices of machine learning would differences be re-drawn? Machine learning in journalism, in specific scientific fields, in the humanities, in social sciences, in art, media, government or civil society sometimes overflows the platform-based deployments and their trenchantly positivist usages. A fairly

ProPublica

machine learner!Message Machine

natural language processing

Bogost, Ian

Netflix

explicit awareness of the operation of machine learning-driven processes is taking shape in some quarters. And this awareness supports a situationally aware calculative knowledge-practice.

For instance, the campaign to re-elect Barack Obama as U.S. President in 2011-12 relied heavily on micro-targetting of voters in the leadup to the election polls (Issenberg 2012; Mackenzie et al. 2016). In response to the data analytics-driven election campaign run by the US Democrats, data journalists at the non-profit news organisation *ProPublica* reverse engineered the machine learning models that the Obama re-election team used to target individual votes with campaign messages (Larsen 2012). They built their own machine learning model - the 'Message Machine' - using emails sent in by voters to explore the workings of the Obama campaign team's micro-targetting models. While the algorithmic complexity and data infrastructures used in the Message Machine hardly match those at the disposal of the Obama team, it combines natural language processing (NLP) techniques such as measures of document similarity and machine learning models such as decision trees to disaggregate and map the micro-targetting processes .

This reverse engineering work focused on the constitution of subject positions (the position of the 'voter') can be found in other quarters. In response to the personalised recommendations generated by streaming media service Netflix, journalists at *The Atlantic* working with Ian Bogost, a media theorist and programmer, reverse engineered the algorithmic production of around 80,000 micro-genres of cinema used by Netflix.(Madrigal 2014) While Netflix's system to categorise films relies on much manual classification and tagging with meta-data, the inordinate number of categories they use is typical of the classificatory regimes that are developing in machine learning-based settings.

Both cases explore the constitutive contemporary conditions of doing, saying, and thinking of subjects, not only to recognise how subject positions are assigned or made, but to grasp the possibility of change. While these cases may be exceptional achievements, and indeed highlight the dead weight of ad-tech application of machine learning, knowledge production more generally is not easily reducible to contemporary forms of capitalism labour.

Domingos, Pedro
Deleuze, Gilles
diagram
programming!human vs. machine

Chapter 2

Diagramming machines

> Machine learning is not magic; it cannot get something from nothing. What it does is get more from less. Programming, like all engineering, is a lot of work: we have to build everything from scratch. Learning is more like farming, which lets nature do most of the work. Farmers combine seeds with nutrients to grow crops. Learners combine knowledge with data to grow programs. (Domingos 2012, 81)

> The tools or material machines have to be chosen first of all by a diagram (Deleuze 1988, 39)

The 'learning' in machine learning embodies a change in programming practice or indeed the programmability of machines. Our sense of the potentials of machine learning can be understood, according to Pedro Domingos, in terms of a contrast between programming as 'a lot of [building] work' and the 'farming' done by machine learners to 'grow programs.' In characterising machine learning, the tensions between the programming 'we' (programmers, computer scientists?) do and the programming that learners do ('growing') are worth pursuing. While Domingos suggests that machine learners 'get

more from less,' I will propose that an immense constellation of documents, software, publications, blog pages, books, spreadsheets, databases, data centre architectures, whiteboard and blackboard diagrams, and an inordinate amount of talk and visual media orbit around machine learning. There has been lively growth in machine learning, but this liveliness and the sometimes life-like growth of machine learners are a regional expression of a distributed formation. 'Wherever there is a region of nature,' the philosopher Alfred North Whitehead writes 'which is itself the primary field of the expressions issuing from each of its parts, that region is alive' (Whitehead 1956, 31).

In this chapter I attend to the problem of identifying and describing the distributed practices that give rise to a sense of machine learners framed as growth, or liveliness. I will argue these practices can only be traced partially through code written in generic or specialized programming languages such as `Python` and `R`, in libraries of machine learning code such as `R`'s `caret` or `Python`'s `scikit-learn` or `TensorFlow` to do machine learning. Code obscures and reveals multiple transformations at work in the operational formation. Science studies scholars such as Anne-Marie Mol have urged the need to keep practice together with theories of what exists. Mol writes:

> If it is not removed from the practices that sustain it, reality is multiple. This may be read as a description that beautifully fits the facts. But attending to the multiplicity of reality is also an act. It is something that may be done – or left undone (Mol 2003, 6)

Mol advocates thinking reality as multiplicity. Her insistence on the nexus of practice or doing and the plural existence of things suggests a way of handling the code that machine learners produce. Code should be approached multiplicity. In the case of machine learners, this means following through

pedagogical expositions of machine learning focused on both mathematical derivations and the accumulation of scientific or technical research publications, ranging from textbooks to research articles, that vary, explore, experiment and implement machine learners in code. The effect of liveliness or growth issues from many parts. In relation to machine learning, reading and writing code alongside scientific papers, Youtube lectures, machine learning books and competitions is not only a form of observant participation, but directly forms part of the diagrammatic multiplicity.

code!readability of
code!agency of

While machine learning utterly depends on code, I will suggest therefore that no matter how expressive or well-documented it may be, code alone cannot fully capture how machine learners make programs or how they combine knowledge with data. Domingos writes that 'learning algorithms … are algorithms that make other algorithms. With machine learning, computers write their own programs, so we don't have to' (Domingos 2015, 6). Yet the writing performed by machine learners cannot be read textually or procedurally as other programs might be read for instance in work known as critical code studies. The difference between the reading of an Atari computer game console in Nick Montfort and Ian Bogost's *Racing the Beam* (Montfort and Bogost 2009) and the machine learning of Atari's games undertaken by DeepMind in London during recent years (Mnih et al. 2013, 2015) is hard to read from program code. The learning or making by learning is far from homogeneous, stable or automatic in practice. Materially, code is only one element in the diagram of machine learning. It displays, with greater or lesser degrees of visibility relations between a variety of forces (infrastructures, scientific knowledges, mathematical formalisations, etc.). It itself is aligned by and exposes other institutional, infrastructural, epistemic and economic positions.

Coding practices and the pedagogical expositions of machine learning have

Python|seealsoprogramming language

R|seealsoprogramming language!R

programming language!FORTRAN

code!writing of

shifted substantially over the last decade or so due to the growth in open source programming languages and as a part of the broader and well-known expansion of digital media cultures. The fact that data scientists, software developers and other machine learners across scientific and commercial settings use programming languages such as `Python` and `R` more than specialized commercial statistical and data software packages such as Matlab, SAS or SPSS (Muenchen 2014) is perhaps symptomatic of shifts in computational culture. Coding cultures are crucial to the recent growth of machine learning. Although scientific computing languages such as `FORTRAN` – 'Formula Translator' – have long underpinned scientific research and engineering applications in various fields (Campbell-Kelly 2003, 34-35), the development in recent decades of data-analytic and statistical programming languages and coding frameworks has crystallized a repertoire of standard operations, patterns and functions for reshaping data and constructing models that classify and predict events and associations between things, people, processes, etc. This development continues apace, especially in research and engineering driven by social media and internet platforms such as Facebook and Google. While Domingos speaks of 'growing' programs, the accumulating sediment of certain well-established data practices are the soil in which programs take root. The different elements of coding practice are precisely the faceted levels of abstraction which we need to access and traverse in order to know and come to grips empirically with contemporary compositions of power and knowledge in machine learning.

'We don't have to write programs'?

In machine learning, coding changes from what we might call symbolic logical diagrams to statistical algorithmic diagrams. While many machine learning techniques have long statistical lineages (running back to the

1900s in the case of Karl Pearson's development of the still-heavily used Principal Component Analysis (Pearson 1901)), machine learning techniques often embody a certain dissatisfaction with the classical computer science understanding of programs as manipulation of symbols, even as they rely on such symbolic operations to function. Symbolic manipulation, epitomised by deductive logic or predicate calculus, was very much at the centre of many AI projects during the 1950s and 1960s (Dreyfus 1972; Edwards 1996). In machine learning, the privileged symbolic-cognitive forms of logic are subject to a statistical transformation.

Pearson, Karl

machine learner!Principal Component Analysis

artificial intelligence!symbolic manipulation in

function!logical

Take for instance once of the most common operations of the Boolean logical calculus, the `NOT-AND` or `NAND` function shown in table 2.1. The truth table summarises a logical function that combines three input variables `X1`, `X2`, and `X3` and produces the output variable `Y`. Because in Boolean calculus, variables or predicates can only take the values `true` or `false`, they can be coded in as `1` and `0`.

Table 2.1: The truth table for the Boolean function NOT-AND truth

X1	X2	X3	Y
0	0	0	1
0	0	1	1
0	1	1	1
0	1	0	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Now in Foucaultean terms, the truth table and its component propositions

statement!truth table as
table!see data!table
infrastructure!digital circuit as
artificial intelligence

constitutes a statement. This statement has triple relevance for archaeology of machine learning. The spatial arrangement of the table is fundamental (and this is the topic of chapter **??**). Most datasets come as tables, or end up as tables at some point in their analysis. Second, the elements or cells of this table are numbers. The numbers 1 and 0 are the binary digits as well as the 'truth' values 'True' and 'False' in classical logic. These numbers are readable as symbolic logical propositions governed by the rule $Y = \not{X}_1 X_2 X_3$. The table acts as a hinge between numbers and symbolic thought or cognition. Third, the NAND table in particular has an obvious operational relevance in digital logic, since digital circuits of all kinds – memory, processing, and communication – comprise such logical functions knitted together in the intricate gated labyrinths of contemporary calculation.[1]

A pre-machine learning programmer, tasked with implementing the logical NAND function might write:

```
Y = not(X1 & X2 & X3)
```

The trivial simplicity of the code stands out. This looks like the kind of symbol manipulation that that computers can easily be programmed to do. How, by contrast, could such a truth table be learned by a machine, even a machine whose *modus operandi* and indeed whose very fabric is nothing other than a massive mosaic of `NAND` operations inscribed in semiconductors? Machine learning of on elementary truth tables is not a typical operation today, but usefully illustrates something of the diagrammatic transformations that programming or coding (and classical AI) has undergone .

Machine learners such as decision trees or neural nets typically know nothing of the logical calculus and its elementary logical operations. Can they be

1. The philosophy Charles Sanders Peirce himself had first shown that combinations of NAND operations could stand in for any logical expression whatsoever, thus paving the way for the diagrammatic weave of contemporary digital memory and computation in all their permutations. Today, NAND-based logic is norm in digital electronics. NOR – NOT OR – logic is also used in certain applications.

induced to learn it? A perceptron, an elementary machine learner dating from the 1950s, that 'learns' the binary logical operation NAND (Not-AND) is expressed in twenty lines of python code on the Wikipedia 'Perceptron' page (*Perceptron* 2013) (see listing 2). It is a standard machine learner almost always included in machine learning textbooks and usually taught in introductory machine learning classes. The code outputs a series of numbers shown in table 2.2.

machine learner!perceptron!learning logical functions

```
threshold = 0.5
learning_rate = 0.1
weights = [0, 0, 0]
training_set = [((1, 0, 0), 1), ((1, 0, 1), 1), ((1, 1, 0), 1), ((1, 1, 1), 0)]

def dot_product(values):
    return sum(value * weight for value, weight in zip(values, weights))

out = 'weight1, weight2, weight3, error_count, iteration'
print  out
iteration_count = 0
while True:
    error_count = 0
    iteration_count += 1
    for input_vector, desired_output in training_set:
        out = ','.join(map(str,weights)) + ',' + str(error_count)
        out = out + ',' + str(iteration_count)
        result = dot_product(input_vector) > threshold
        error = desired_output - result
        print out
```

machine learning!as transformation in programming

code!brevity in machine learning

```
        if error != 0:
            error_count += 1
            for index, value in enumerate(input_vector):
                weights[index] += learning_rate * error * value
    if error_count == 0:
        break
```

weight1	weight2	weight3	error_count	iteration
0.00	0.00	0.00	0	1
0.10	0.00	0.00	1	1
0.20	0.00	0.10	2	1
0.30	0.10	0.10	3	1
0.30	0.10	0.10	0	2
0.40	0.10	0.10	1	2
0.50	0.10	0.20	2	2
0.50	0.10	0.20	2	2
0.40	0.00	0.10	0	3
0.50	0.00	0.10	1	3
0.50	0.00	0.10	1	3
0.60	0.10	0.10	2	3
0.50	0.00	0.00	0	4
0.60	0.00	0.00	1	4
0.60	0.00	0.00	1	4

Table 2.2: Iterative change in weights as a perceptron learns the NAND function

What does the code in listing 2 show or say about the transformation in programmability or the writing of programs? First of all, we should note the relative conciseness of the code vignette. Much of the code here is familiar, generic programming. It defines variables, sets up data structures (lists of

numerical values), checks conditions, loops through statements or prints results. In citing this code, I am not resorting to a technical publication or scientific literature as such, or even to a machine learning software library or package (in `scikit-learn`, the same model could be written `p = scikit-learn.linearmode.Perceptron(X,Y)`), just to a Wikipedia page, and the relatively generic and widely used programming language `Python`.[^1.31] Whatever the levels of abstraction associated with machine learning, the code is hardly ever hermetically opaque. As statements, everything lies on the surface.

machine learner!perceptron|(

Second, while the shaping of data, the counting of errors, and the optimisation of models are topics of later discussion, the code shows some typical features of a machine learner in the form of elements such the `learning rate`, a `training_set, weights`, an `error count`, and a loop function that multiplies values (`dot_product`). Some of the names such as `learning_rate` or `error_count` present in the code bear the marks of the theory of learning machines that we will discuss.

Third, executing this code (by copying and pasting it into a `Python` terminal, for instance) produces several dozen lines of numbers. They are initially different to each other, but gradually converge on the same values (see table 2.2). These numbers are the 'weights' of the nodes of the perceptron as it iteratively 'learns' to recognise patterns in the input values. None of the workings of the perceptron need concern us at the moment. Again, what runs across all of these observations are the numbers that the algorithm produces as output – they embody the program the perceptron has written. How has the learning happened in code? The NAND truth table has been re-drawn as a dataset (see line 4 of the code that defines the variable `training_set`). The perceptron has learned the data by approaching it as a set of training examples, and then adjusting its internal model – the

weights!see model!parameters

weights that are printed during each loop of the model as the output – repeatedly until the model is producing the correct result values Y of the truth table. The algorithm exits its main loop (`while True:`) when there are no errors.

The perceptron algorithm computes numbers - `0.79999`, `2.0` – as weights or parameters. These numbers display no direct correspondence with the symbolic categories of boolean True and False or the binary digits `1` and `0`. There may be a relation but it is not obvious at first glance. The problem of mapping these calculated parameters – and they truly abound in machine learning – triggers many different diagrammatic movements in machine learning (and these different movements will be discussed in chapters **??** and **??**). These numbers engender much statistical ratiocination (see chapter **??**). Here we need only note the contrast between symbolically organised statements like the NAND truth table of Table 2.1 and the operational statements in table 2.2.

The operation here is recursive: a model or algorithm implemented in digital logic (as `Python` code) has 'learned' – a term we need to explore more carefully – a basic rule of digital logic (the `NAND` function) at least approximately by treating logical propositions as data. This mode of transformation is symptomatic. The learning done in machine learning has few cognitive or symbolic underpinnings. It differs from classical AI in that it treats existing symbolic, control, communicative and increasingly, signifying processes (such as the cat faces that `kittydar` tries to find), and latches onto them programmatically only in the form of weights.

The elements of machine learning

learning!to machine learng
Ng, Andrew!CS229 lectures
programming languages
science!publications!on machine learning
machine learning!topic structure of

If this growing of programs through modelling data is a different mode of coding operations, and a different mode of knowing, how does one learn to do it? In the course of writing this book, as well as reading the academic textbooks, popular how-to books, software manuals, help documents and blog-how-to posts, I attended graduate courses in Bayesian statistics, genomic data analysis, data mining, and missing data. I also participated in online machine learning courses and some machine learning competitions. These are all widely shared activities for people learning to do machine learning. I watched and copied down by hand many statements, equations and propositions from Youtube videos of the eighteen lectures in Andrew Ng's Stanford University lectures CS229 computer science course recorded in 2008. These lectures have cumulative viewing figures of around 500,000 (*Lecture 1 | Machine Learning (Stanford)* 2008). I spent extended hours learning to use various code libraries, packages and platforms in `R`, `Python` and to a limited extent `Torch` . Finally, I gradually accumulated and worked with a set of around 400,000 articles drawn from various fields of science in reponse to search queries on particular machine learners such as `support vector machine` or `expectation maximization`. This accumulation of knowledge appears in figures 2.1 as a network-cloud.

Some broadly shared topic structures help in any navigation of the software libraries, pedagogical and research literatures. The textbooks, the how-to recipe books (Segaran 2007; Kirk 2014; Russell 2011; Conway and White 2012) and the online university courses on machine learning often have a similar topic structure. They nearly always begin with 'linear models' (fitting a line to the data), then move to logistic regression (a way of using the linear model to classify binary outcomes; for example, spam/not-spam; malignant/benign; cat/non-cat), and afterwards move to some selection

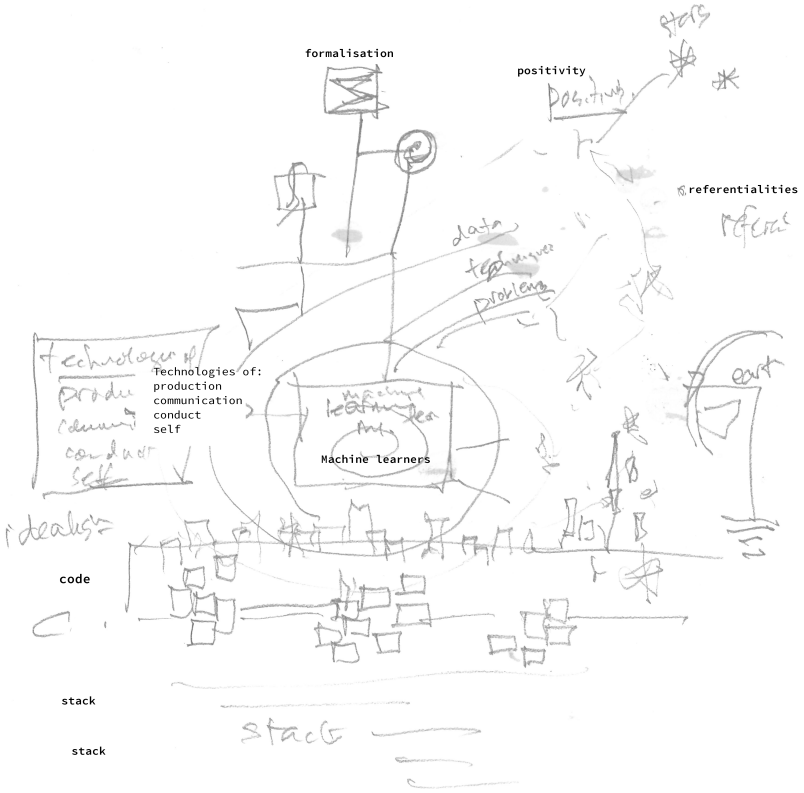


Figure 2.1: A diagram of the elements of learning to machine learn

of neural networks, decision trees, support vector machine and clustering algorithms. They add in some decision theory, techniques of optimization, and ways of selecting predictive models (especially the bias-variance tradeoff).[2] The topic structures have in recent years started to become increasingly uniform. This coagulation around certain topics, problems and mathematical formalisms is both something worth analyzing (since, for instance, it definitely affects how machine learning is taken up in different settings), but should not be taken as obvious or given since it results from many iterations.

linear regression

2. They differ, however, in several important respects. Reading the *Elements of Statistical Learning* textbook or one of machine learning books written for programmers (for example, *Programming Collective Intelligence* or *Machine Learning for Hackers* (Segaran 2007; Conway and White 2012)) does not directly subject the reader to machine learning By contrast, doing a Coursera course on machine learning brings with it an ineluctable sense of being machine-learned, of oneself becoming an object of machine learning. The students on Coursera are the target of machine learning. Daphne Koller and Andrew Ng are leading researchers in the field of machine learning, but they also co-founded the online learning site Coursera. As experts in machine learning, it is hard to imagine how they would not treat teaching as a learning problem. And indeed, Daphne Koller sees things this way:

> There are some tremendous opportunities to be had from this kind of framework. The first is that it has the potential of giving us a completely unprecedented look into understanding human learning. Because the data that we can collect here is unique. You can collect every click, every homework submission, every forum post from tens of thousands of students. So you can turn the study of human learning from the hypothesis-driven mode to the data-driven mode, a transformation that, for example, has revolutionized biology. You can use these data to understand fundamental questions like, what are good learning strategies that are effective versus ones that are not? And in the context of particular courses, you can ask questions like, what are some of the misconceptions that are more common and how do we help students fix them? (*What we're learning from online education | Video on TED.com* 2012)

Whether the turn from 'hypothesis-driven mode to the data-driven mode' has 'revolutionized biology' is debatable (I return to this in a later chapter). And whether or not the data generated by my participation in Coursera's courses on machine learning generates data supports understanding of fundamental questions about learning also seems an open question. Nevertheless, the loopiness of this description interests and appeals to me. I learn about machine learning, a way for computer models to optimise their predictions on the basis of 'experience'/data, but at the same time, my learning is learned by machine learners. This is not something that could happen very easily with a printed text, although versions of it happen all the time as teachers work with students on reading texts. While Coursera and other MOOCs promise something that mass education struggles to offer (individually profiled educational services), it also negatively highlights the possibility that machine learning in practice can, somewhat recursively, help us make sense of machine learning as it develops.

\textitElements of Statistical Learning

Hastie, Jeff

Tibshirani, Rob

Friedman, Jerome

Amidst this avalanching machine learning materials and practice, a single highly cited and compendious textbook, *Elements of Statistical Learning: Data Mining, Inference, and Prediction* dating from from around 2000 and currently in its second edition (Hastie, Tibshirani, and Friedman 2009), can be seen from almost any point of the terrain.[3] At least for narrative purposes, I regard this book as a major archaeological assemblage, and a diagram that presents many important statements, forms of visibility and relations between forces at work in machine learning. The authors of the book, Jeff Hastie, Rob Tibshirani and Jerome Friedman are statisticians working at Stanford and Columbia University.

Elements of Statistical Learning is a massive textual object, densely radiant with equations, tables, algorithms, graphs and references to other scientific literature. From the first pages proper of the book, almost every page has a figure or a table or a formal algorithm (counting these together: 1670 equations; 291 figures; 34 tables; and 94 algorithms, giving a total of 2089 operational statements threaded through the book). Equations rivet the text with mathematical abstractions of varying sophistication. On each page of the book we are seeing, reading, puzzling over and perhaps learning from the products of code execution. The graphic figures are all produced by code. The tables are mostly produced by code. The algorithms specify how to implement code, and the equations diagram various operations, spaces and movements that meant to run as code.

In the range of references, combinations of code, diagram, equation, scientific disciplines and computational elements, and perhaps in the somewhat viscous, inter-objectively diverse referentiality that impinges on any reading

3. The complete text of the book can be downloaded from the website http://statweb.stanford.edu/~tibs/ElemStatLearn/. At the end of short intensive course on data mining at the Centre of Postgraduate Statistics, Lancaster University, the course convenor, Brian Francis, recommended this book as the authoritative text. Some part of me rues that day. That book is a poisoned chalice; that is, something shiny, valuable but also somewhat toxic.

of it, *Elements of Statistical Learning* betrays some hyperobject-like positivity (Morton 2013). It is an accumulation of forms, techniques, practices, propositions and referential relations. *Elements of Statistical Learning* combines statistical science with various algorithms to 'learn from data' (Hastie, Tibshirani, and Friedman 2009, 1). The data range across various kinds of problems (identifying spam email, predicting risk of heart disease, recognising handwritten digits, etc.). The learning takes the form of various machine learning techniques, methods and algorithms (linear regression, k-nearest neighbours, neural networks, support vector machines, the Google Page Rank algorithm, etc.).

positivity
hyperobject
machine learning!textbooks
Ripley, Brian
Mitchell, Tom
Flach, Peter

There are other juggernaut machine learning textbooks. Ethem Alpaydin's *Introduction to Machine Learning* (Alpaydin 2010) (a more computer science-base account), Christopher Bishop's heavily mathematical *Pattern recognition and machine learning* (Bishop 2006), Brian Ripley's luminously illustrated and almost coffee-table formatted *Pattern Recognition and Neural Networks* (Ripley 1996) , Tom Mitchell's earlier artificial intelligence-centred *Machine learning* (Mitchell 1997) , Peter Flach's perspicuous *Machine Learning: The Art and Science of Algorithms that Make Sense of Data* (Flach 2012) , or further afield, the sobering and laconic *Statistical Learning for Biomedical Data* (Malley, Malley, and Pajevic 2011) all cover a similar range of data and approaches. These and quite a few other recent machine learning textbooks display a range of emphases, ranging from the highly theoretical to the very practical, from an orientation to statistical inference to an emphasis on computational processes, from science to commercial applications.

Elements of Statistical Learning!as diagram of abstraction

Elements of Statistical Learning!readerships

science!diversity of fields in machine learning

Who reads machine learning textbooks?

How does such textbook help us assess and engage with claim to learn from data or to produce knowledge differently? While it certainly does not comprehend everything taking place in and around machine learning, it diagrams several *elementary* tendencies or traits. It's readership as we will see is widespread. It has a heterogeneous texture in terms of the examples, formalisms, disciplines and domains it covers. It starkly renders the problems of making sense of mathematical operations, diagrams and transformations carried on through calculation, simulation, deduction or analysis. It draws on a matrix of operational practices, particularly in the form of the `R` code it heavily but somewhat latently relies on. In short, *Elements of Statistical Learning* presents a multi-faceted and somewhat monumental layering of abstractive practice that might be open to archaeological inquiry.

Who reads the *Elements of Statistical Learning*? It is often cited by academic machine learning practitioners as an authoritative guide. On the other hand, students participating in new data science courses often come from different disciplinary backgrounds and find the tome unhelpful (see the comment by students during an introductory data science course documented in (Schutt and O'Neil 2013)). Whether the citations are friendly or not, i is hard to find a field of contemporary science, engineering, natural, applied, health and indeed social science that has not cited it. A Thomson-Reuters Scientific 'Web of Science'(TM) search for references citing either the first or second edition of (Hastie, Tibshirani, and Friedman 2009) yields around 9000 results. These publications sprawl across over 100 different fields of research. While computer science, mathematics and statistics dominate, a very diverse set of references comes from disciplines from archaeology, through fisheries and forestry, genetics, robotics, telecommunications and toxicology ripple out from this book since 2001. Table 2.3 shows the top 20 fields

positivity!as form of accumulation

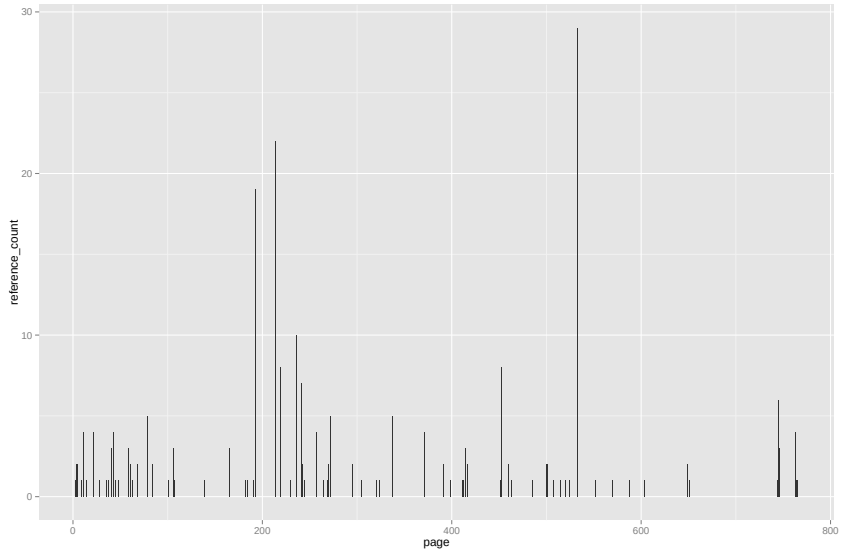


Figure 2.2: Pages cited from *Elements of Statistical Learning* by academic publications in all fields.

by count. One could learn something about the diagrammatic movement of machine learners from that reference list, which itself spans biomedical, engineering, telecommunications, ecology, operations research and many other fields. While it is not surprising to see computer science, mathematics and engineering appearing at highest concentration in the literature, molecular biology, control and automation, operation research, business and public health soon appear, suggesting something of the propagating accummulation or positivity of machine learning.

```
##   File "<string>", line 88
##     <!--hastie = df.CR.str.findall('Hastie.*?;')-->
##     ^
## SyntaxError: invalid syntax
```

So we know that *Elements of Statistical Learning* passes into many fields. But what do people read in the textual environment of book? In general, the thousands of citations of the book themselves compose a diagram of

machine learner!Non-negative matrix factorization

'kittydar'

face recognition

the book's intersection with different domains of knowledge. The relative concentrations and sparsities of these citations suggest there may be specific sites of engagement in the techniques, approaches and machines that the book documents. Of the around 760 pages in the two editions ((Hastie, Tibshirani, and Friedman 2009) and (Hastie, Tibshirani, and Friedman 2001)), around 77 distinct pages are referenced in the citing literature. As figure 2.2 indicates, certain portions of the book are much more heavily cited than others. This distribution of page references in the literature that cites *Elements of Statistical Learning* is a rough guide to how the book has been read in different settings. For instance, the most commonly cited page in the book is page 553. That page begins a section called 'Non-negative Matrix Factorization', a technique frequently used to process digital images to compress their visual complexity into a simpler set of visual signals (Hastie, Tibshirani, and Friedman 2009, 553). (The underlying reference here is the highly cited paper (Lee and Seung 1999)) Like `kittydar`, it, as Hastie and co-authors write, 'learns to represent faces with a set of basis images resembling parts of faces' (Hastie, Tibshirani, and Friedman 2009, 555). (So `kittydar`, which doesn't use NMF, might do better if it did, because it could work just with parts of the images that lie somewhere near the parts of a cat's face – its nose, its eyes, its ears.)

Conversely, what do the authors of *Elements of Statistical Learning* read? The book gathers elements from many different quarters and seeks to integrate them in terms of statistical theory. The hyperobject-like aspect of the book comes from the thick weave of equations, diagrams, tables, algorithms, bibliographic apparatus, and numbers wreathed in typographic ornaments drawn from many other sources. For instance, in terms of outgoing references or the literature that it cites, *Elements of Statistical Learning* webs together a field of scientific and technical work with data and predictive

archaeology!reading practices in

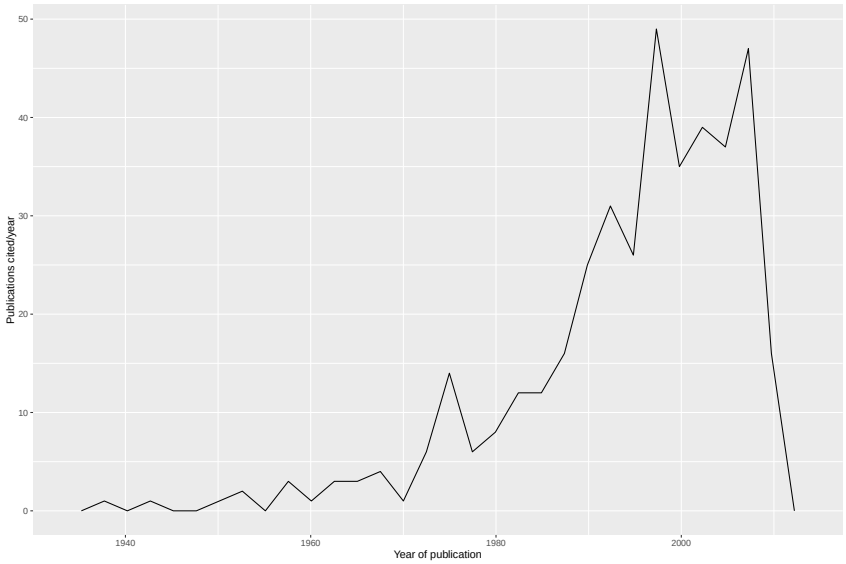


Figure 2.3: Year of publications cited in*Elements of Statistical Learning*. The references range over almost 80 years, with peaks in late 1970s, late 1980s, mid-1990s and mid-2000s. These peaks relate to different mixtures of cybernetics, statistics, computer science, medicine, biology and other fields running through machine learning. Regression-related publications form the main body of citation.

models ranging across half a century. The reference list beginning at page 699 (699) runs for around 35 pages, and the five hundred or so references there point in many directions. The weave of these elements differs greatly from citational patterns in the humanities or social sciences. Reading this book almost necessitates an archaeological approach since it comprsies so many parts and fragments.

The citational fabric of *Elements of Statistical Learning* is woven with different threads, some reaching back into early twentieth statistics, some from post-WW2 cybernetics, many from information theory and then in the 1980s onwards, increasingly, from cognitive science and computer science (see figure 2.3 to get some sense of their distribution over time). While some of these references either point to Hastie, Tibshirani or Friedman's own publications, or that of their statistical colleagues, the references rove quite widely in other fields and over time. *Elements of Statistical Learning* as a text is processing, assimilating and recombining techniques, diagrams and

Kuhn, Thomas
critical thought
programming language!R|(
operational formation!code as operational practice in
code!as operational practice

data from many different places and times. (The different waves appearing in the references cited in (Hastie, Tibshirani, and Friedman 2009) will shape discussion in later chapters in certain ways; for instance, late 1990s biology is the topic of chapter **??** and optimization functions dating from the 1950s are discussed in chapter **??**).

Both the inward and outward movements of citation suggest that *Elements of Statistical Learning*, like much in field of machine learning, has a matrix-like character that constantly superimposes and transposes elements across boundaries and barriers. The implication here is that machine learning as a knowledge practice has a highly interwoven texture, and in this respect differs somewhat from the classical understandings of scientific disciplines as bounded by communities of practice, norms and problems (as for instance, in Thomas Kuhn's account of normal science (Kuhn 1996). This aggregate or superimposed character of machine learning should definitely figure in any sense we make of it, and will inevitably affect how critical thought might test itself against it. .

`R`: a matrix of transformations

Although barely a single line of code appears in *Elements of Statistical Learning*, all of the learners presented there are implemented in a single programming language, `R`. Coding is the operational practice that links the different planes and elements of machine learning in an operational formation. The authors say 'we used the R and S-PLUS programming languages in our courses' (Hastie, Tibshirani, and Friedman 2009, 9) but many elements of the book derive from `R` code.[4] The proliferation of programming

4. In a later book by some of the same authors with the very similar sounding title *An Introduction to Statistical Machine Learning with Applications in* `R` (James et al. 2013), `R` does appear in abundance. This book, however, is much shorter and less inclusive in various ways. There are in any case many online manuals, guides and tutorials relating

languages such as FORTRAN (dating from the 1950s), C (1970s), C++ (1980s), then Perl (1990s), Java (1990s), Python (1990s) and R (1990s), and computational scripting environments such as Matlab, multiplied the paths along which machine learners move through operational formations. It would be difficult to comprehend the propagation of machine learners across domains of science, business and government without paying attention to coding practices. Even if textbooks and research articles are not read, software packages and libraries for machine learning are used. Code has a mobility that extends the diagrammatic practices of machine learning into a variety of settings and places where the scientific reading apparatuses of equations, statistical plots, and citations of research articles would not be operative.

programming languages
code!mobility of

\begin{table}[!htb]

to R (Wikibooks 2013). For present purposes, I draw mainly on semi-popular books such as *R in a Nutshell* (Adler and Beyer 2010), *The Art of R Programming* (Matloff 2011), *R Cookbook* (Teetor 2011), *Machine Learning with R* (Lantz 2013) or *An Introduction to Statistical Learning with Applications in R* (James et al. 2013). These books are not written for academic audiences, although academics often write them and use them in their work. They are largely made up of illustrations and examples of how to do different things with different types of data, and their examples are typically oriented towards business or commercial settings, where, presumably, the bulk of the readers work, or aspire to work. Given a certain predisposition (that is, geekiness), these books and other learning materials can make for enjoyable reading. While they vary highly in quality, it is sometimes pleasing to see the economical way in which they solve common data problems. This genre of writing about programming specialises in posing a problem and solving it directly. In these settings, learning takes place largely through following or emulating what someone else has done with either a well known dataset (Fisher's iris) or a toy dataset generated for demonstration purposes. The many frictions, blockages and compromises that often affect data practice are largely occluded here in the interests of demonstrating the neat application of specific techniques. Yet are not those frictions, neat compromises and strains around data and machine learning precisely what we need to track diagrammatically? In order to demonstrate both the costs and benefits of approaching R through such materials, rather than through ethnographic observation of people using R, it will be necessary to stage encounters here with data that has not been completely transformed or cleaned in the interests of neatly demonstrating the power of a technique. One way to do this is by writing about code while coding.

programming languages!R!popularity of

	Package	How often depended on
methods	methods	75
stats	stats	33
survival	survival	30
MASS	MASS	26
mvtnorm	mvtnorm	21
Matrix	Matrix	13
ggplot2	ggplot2	12
lattice	lattice	9
rJava	rJava	9
grid	grid	7
igraph	igraph	7
tcltk	tcltk	7
XML	XML	7
rgl	rgl	6
graphics	graphics	5
methods, stats	methods, stats	5
nlme	nlme	5
utils	utils	5
vegan	vegan	5
ape	ape	4

\caption[`ElemStatLearn` depends on R packages]{`R` packages depended on by the `ElemStatLearn` package} \end{table}

How should we think of the `R` code in *Elements of Statistical Learning* in its operational specificity? Its growth is perhaps just as important as its operation (Mackenzie 2014). An open source programming language, according to surveys of business and scientific users, at the time of writing, `R` has replaced popular statistical software packages such as SPSS, SAS and

Stata as the statistical and data analysis tool of choice for many people in business, government, and sciences ranging from political science to genomics, from quantitative finance to climatology (RexerAnalytics 2015). Developed in New Zealand in the mid-1990s, and like many open source software projects, emulating `S`, a commercialised predecessor developed at AT&T Bell Labs during the 1980s, `R` is now extremely widely used across life and physical sciences, as well as quantitative social sciences. John Chambers, the designer of `S`, was awarded the Association for Computing Machinery (ACM) 'Software System Award' in 1998 for 'the S system, which has forever altered how people analyze, visualize, and manipulate data' (ACM 2013). Many undergraduate and graduate students today earn `R` as a basic tool for statistics. Skills in `R` are often seen as essential pre-requisite for scientific researchers, especially in the life sciences. (In engineering, `Matlab` is widely used.) Research articles and textbooks in statistics commonly both use `R` to demonstrate methods and techniques, and create `R` packages to distribute the techniques and sample data. Nearly all of these publication-related software packages, including quite a few from the authors of *Elements of Statistical Learning* are soon or later available from the 'Comprehensive `R` Archive Network (CRAN)' (CRAN 2010). Estimates of its number of users range between 250000 and 2 million. Increasingly, `R` is integrated into commercial services and products (for instance, SAS, a widely used business data analysis system now has an `R` interface; Norman Nie, one of the original developers of the SPSS package heavily used in social sciences, now leads a business, Revolution, devoted to commercialising `R`; `R` is heavily used at Google, at FaceBook, and by quantitative traders in hedge funds; in 2013 'R usage is sky-rocketing'; etc.). In general terms, `R` has a kind of disciplinary polyglot currency as a form of expression, and exhibits a fine-grained relationality with many different epistemic and operational situations associated with machine learning.

Chambers, John

Two early proponents of `R` and `S` describe the motivation for the language:

> `R` is an increasing well-known and widely used statistical programming language and environment (R Development Core Team 2010). >The goal of the S language … is "to turn ideas into software, quickly and faithfully" … it is the duty of the responsible data analysts to engage in this process … the exercise of drafting an algorithm to the level of precision that programming requires can in itself clarify ideas and promote rigorous intellectual scrutiny. … Turning ideas into software in this way need not be an unpleasant duty. (Venables and Ripley 2002, 2)

Bill Venables and Brian Ripley, statisticians working on developing `S`, the almost identical commercial predecessor to `R`, wrote in the early 1990s of the responsibility of data analysts to write not just use software. They write 'software' here not in the sense of a product, but in the sense that today would more likely be called 'code.' Their sense of coding and programming as clarifying and concretising ideas with precision – as abstractions – has thoroughly taken hold in contemporary data analysis. If code, as they suggest, entails a threshold of idealisation, it differs from mathematical formalization in that it changes the positions and relations of knowledge to include machines, devices and infrastructures.

While the view that code is a precise expression of ideas makes sense, it does not capture the relational complexity of `R` code as it operates in a setting such as *Elements of Statistical Learning*. In machine learning, code, along with mathematics, is a primary operational form that ideas take as they become machine learners. But code as an expressive operation by which 'an individual formulates an idea' or as 'a rational activity that may operate in a system of inference' (Foucault 1972, 117) does not exhaust and should not be conflated with operational practice in machine learning. Similarly, the intersection of `R` with machine learning also lies somewhat at

odds with Pedro Domingos' characterisation of machine learning as a shift away from people building to learners growing programs.

```
‘install.packages('ElemStatLearn', dependencies='Suggests', repos = 'http://cran
```

What in R (let alone other programming languages) overflows both the ideas of code as expression of ideas and code as automation? Alongside expression and automation, much R code furnishes a matrix of practice crossing network infrastructures, display screens, statistical techniques, software engineering architectures as well as publication and documentation standards. For instance, the line of R code shown in the listing 2 when executed opens another way of reading *Elements of Statistical Learning* and getting a feel for the dragnet of practical relations and infrastructural configurations running through it. Take the part of the line `dependencies = 'Suggests'`. When the line of code executes, the stipulation of `dependencies` leads to a quite wide-ranging installation event that installs many R packages.`If the installation works (and that assumes quite a lot of configuration work has already taken place; for instance, installing a recent version of the`Rplatform`), then _Elements of Statistical Learning_ is now augmented by various pieces of code, and by various datasets that in some ways echo or mimic the book but in other ways extend it operationally (see tables \ref{tab:elemstat_learn_depends} and \ref{tab:elemstat_learn_suggests}).[^1.92] \index{_Elements of Statistical Learning_!code elements of} These code elements are often stunningly specialised. As Karl Marx wrote of the 500 different hammers made in Birmingham, 'not only is each adapted to one particular process, but several varieties often serve exclusively for the different operations in one and the same process' [@Marx_1986, 375]`

R packages!variety of
Bollas, Christopher
programming languages!R|)
thinking
critical thought!operational modes of

`\index{Marx, Karl!on hammers}. Something similar holds in`R‘: thousands of software packages in online repositories suggest that a highly specialised division of labour and possibly refined co-operative labour processes operate around data.

Because of this almost incoherent plurality, and its labile status as both a programming environment and a statistical analysis package, `R` is an evocative object, to use the psychoanalyst Christopher Bollas' term (Bollas 2008) , an object through which many different ways of thinking circulate. Standing somewhere at the intersection of statistics and computing, modelling and programming, many different disciplines, techniques, domains and actors intersect in `R`. It engages immediately, practically and widely with words, numbers, images, symbols, signals, sensors, forms, instruments and above all abstract forms such as mathematical functions like probability distributions and many different architectural forms (vectors, matrices, arrays, etc.), as it employs data. If, as Bollas suggests, ‘our encounter, engagement with, and sometimes our employment of, actual things is a *way* of thinking' (92), then it plausible that `R` not only gathers a plurality of data practices – working with measurements, numbers, text, images, models and equations, with techniques for sampling and sorting, with probability distributions and random numbers – but that it embodies the kernel of a mode of thought relevant to contemporary realities.

The obdurate mathematical glint of machine learning

If scientific research literature and operational `R` code constitute the elements of an operational formation presented in *Elements of Statistical Learning*, what of the mathematics? While references from many different

places flow in and out of *Elements of Statistical Learning*, they are nearly all articulated in mathematical form. Machine learning as a grouping of statements relies heavily on mathematics. Given that mathematics is itself diverse and multi-stranded, what kind of mathematics matters here? While later chapters will explore some of the main mathematical practices (linear algebra, statistical inference, etc.), machine learners in *Elements of Statistical Learning* coalesce around a single exemplary technique: linear regression models or fitting a line to points. The linear regression model is pivotal, not just in *Elements of Statistical Learning* but in much of the scientific and engineering literature. The linear regression model pushes up some of the citational peaks in Figure 2.3. Even though it is an old technique dating back to Francis Galton in the 1890s (see (Stigler 1986, chapter 8)), it remains perhaps the central working element of machine learning.

linear regression model|(
statistics!history of
machine learner!linear regression model

Elements of Statistical Learning acknowledges the statistical legacy and inheritance in machine learning:

> The linear model has been a mainstay of statistics for the past 30 years and remains one of our most important tools. Given a vector of inputs $X^T = (X_1, X_2, ..., X_p)$, we predict the output Y via the model
>
> $$\hat{Y} = \hat{\beta}_0 + \sum_{(}^{p} j = 1) X_j \hat{\beta_j})$$
>
> The term $\hat{\beta}_0$ is the intercept, also known as the *bias* in machine learning (Hastie, Tibshirani, and Friedman 2009, 11).

In the course of the book, this mathematical expression appears in many variations, iterations, expansions and modifications ('ridge regression'; 'least angle regression'; 'project pursuit,' etc.). But this introduction of the 'mainstay of statistics,' the linear model, already introduces a

Pasquinelli, Paolo!on mathematics as abstraction

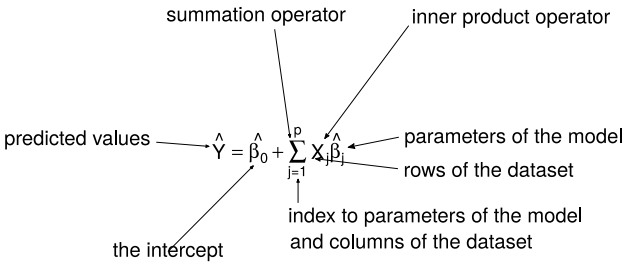


Figure 2.4: The linear regression model

diagrammatic element – the mathematical equation – that is perhaps the most prominent feature in the text.

Any reading of the book has to work out a way to traverse the forms show in equation **??**.

In its relatively compressed typographic weave, expressions such as Equation 2.4 operationalize movements through data or 'a vector of inputs X^T.' These expressions, which are not comfortable reading for non-technical readers, are however worth looking at carefully if we want to move 'at the same level of abstraction as the algorithm' (Pasquinelli 2015). They can be found in hundreds in (Hastie, Tibshirani, and Friedman 2009), but also in many other places. While their presence distinguishes machine learning from parts of computer science where mathematical equations are

less common, mathematical formalizations also allow the book to suture a panoply of scientific publications and datasets in fields of statistics, artificial intelligence and computer science. Along with the citations, the graphical plots, and the code relationality, these equations are integral connective tissue in machine learning.

In dealing with the obscuring dazzle of mathematical formalisation, I find it useful here to follow Charles Sanders Peirce account of mathematics. 'Mathematical reasoning,' he writes, 'is diagrammatic' (Peirce 1998, 206). For Peirce, we should see mathematics, whether it takes an algebraic or geometrical form, whether it appears in symbols, letters, lines or curves as diagrams. For Peirce, a diagram is a kind of 'icon.' The icon is a sign that resembles the object it refers to: it has a relation of likeness. What likeness appears in equation 2.4? Peirce says: 'many diagrams resemble their objects not all in their looks; it is only in respect to the relations of their parts that their likeness consists' (13). As we have seen in the perceptron code, machine learners can be expressed in statements in a programming language like `Python` or `R`. In code, however, the relations between parts cannot be observed in the same way as they can in the algebraic form. The 'very idea of the art,' as Peirce puts it (Peirce 1992, 228), of algebraic expressions is that the formulae can be manipulated, or that component elements can be moved without much effort through substitutions, transformations and variations. The graphic form of the expression include the various classical Greek operator symbols such as or , as well as the letters x, y, z and the indices (indexical signs) such as j that appear in subscript or superscript, as well as the spatial arrangement of all these elements in lines and sometimes arrays. A variety of relations run between these different symbols and spatial arrangements. For instance, in all such expressions, the relation between the left hand side of the '=' and the right

data!as variable in equations

diagram!transformation of

hand side is very important. By convention, the left hand side of the expression is the value that is predicted or calculated (the 'response' variable) and the right hand side are the input variables or 'features' that contribute data to the model or algorithm. This spatial arrangement fundamentally affects the design of algorithms. In the case of equations **??**, the '^' over $\hat{Y}$ symbolises a predicted value rather than a value that can be known completely through deduction, derivation or calculation. This distinction between predicted and actual values organizes a panoply of different practices and imperatives (for instance, to investigate the disparities between the predicted and actual values – machine learning practitioners spend a lot of time on that problem).

The broad point is that the whole formulae is a diagram, or an icon that '*exhibits*, by means of the algebraical signs (which are not themselves icons), the relations of the quantities concerned' (Peirce 1998, 13). Because diagrams suppress so many details, they allow one to focus on a selected range of relations between parts. This affordance of diagrammatic mathematical forms is extremely important in the operational formation of machine learning. Diagrams can diagram other diagrams. Operations can themselves become the subject of operations. The nesting of diagram is highly generative since it allows what Peirce calls 'transformations' (212) or the construction of 'a new general predicate'(Peirce 1992, 303).[5]

5. Félix Guattari makes direct use of Peirce's account of diagrams as icons of relation in his account of 'abstract machines' (Guattari 1984). He writes that 'diagrammaticism brings into play more or less de-territorialized trans-semiotic forces, systems of signs, of code, of catalysts and so on, that make it possible in various specific ways to cut across stratifications of every kind' (145). Here the 'trans-semiotic forces' include mathematical formulae and operations (such as the banking system of Renaissance Venice, Pisa and Genoa). They are trans-semiotic because they are not tethered by the signifying processes that code experience or speaking positions according to given stratifications such as class, gender, nation and so forth. While Guattari (and Deleuze in turn in their co-written works (Guattari and Deleuze 1988)) is strongly critical of the way which signification territorializes (we might think of cats patrolling, marking and displaying in order to maintain their territories), he is much more affirmative of diagrammatic processes. He calls them 'a-signifying' to highlight their difference from the signifying processes that order social strata. He suggests that diagrams become the foundation for 'abstract

While I seek to relate to the equations as diagrams, and will present a selection of them (nowhere near as many as found in *Elements of Statistical Learning*) in the following pages, I am not assuming their operation is transparent or fully legible. Just as much as the analysis of a photograph, a literary work or an ethnographic observation, their diagrammatic composition calls for repeated consideration. Peirce advises not to begin with examples that are too simple: 'in simple cases, the essential features are often so nearly obliterated that they can only be discerned when one knows what to look for' (Peirce 1998, 206). He also suggests 'it is of great importance to return again and again to certain features' (206). Looking at these diagrammatic expressions repeatedly might allows us to map something of how transformations, generalisations or intensification flow across disciplinary boundaries, across social stratifications, and sometimes, generate potentially different ways of thinking about collectives, inclusion and belonging.

CS229, 2007: returning again and again to certain features

If we were to follow Peirce's injunction to 'return again and again to certain features,' how would we do that? *Elements of Statistical Learning* is a difficult book to read in isolation (although it does pay re-reading). Even after several years, the diagrammatic density of its 'elements' or statements (equations, citations, tables, datasets, plots) leaves me with a refractory feeling of 'not quite understanding.' This feeling is inevitable because the book condenses finished work from several disciplines, and partly because it

machines' and the 'simulation of physical machinic processes.' Writing in the 1960s, Guattari powerfully anticipates the abstract machines and their associated diagrams that have taken shape and physical form in the succeeding decades.

machine learning!epistemic threshold of

Ng, Andrew!CS229

linear regression

seeks to organize a great diversity of materials *epistemically.* Indeed, the book might seen as evidence that machine learning has crossed an epistemic threshold formulated in a statistical apparatus. (The statistical aspects of machine learning are the main topic of chapter **??**).

Does a computer science course offer a more easily followed route? Andrew Ng's course 'Machine Learning' CS229 at Stanford (http://cs229.stanford.edu/) might provide a supplementary path into machine learning (*Lecture 1 | Machine Learning (Stanford)* 2008).[6] The course description runs as follows:

> This course provides a broad introduction to machine learning and statistical pattern recognition. Topics include supervised learning, unsupervised learning, learning theory, reinforcement learning and adaptive control. Recent applications of machine learning, such as to robotic control, data mining, autonomous navigation, bioinformatics, speech recognition, and text and web data processing are also discussed (*Lecture 1 | Machine Learning (Stanford)* 2008)

CS229 is in many ways a typical computer science pedagogical exposition of machine learning. Machine learning expositions usually begin with simple datasets and the simplest possible statistical models and machine learners (linear regression), and then, with a greater or lesser degree of attention to issues of implementation, move through a succession of increasingly sophisticated and specialised techniques. This pattern is found in many of the how-to books, in the online courses, and in the academic textbooks, including (Hastie, Tibshirani, and Friedman 2009).

6. A heavily shortened version of this course has been delivered under the title 'Machine Learning' on Coursera.org, a MOOC (Massive Open Online Course) platform.

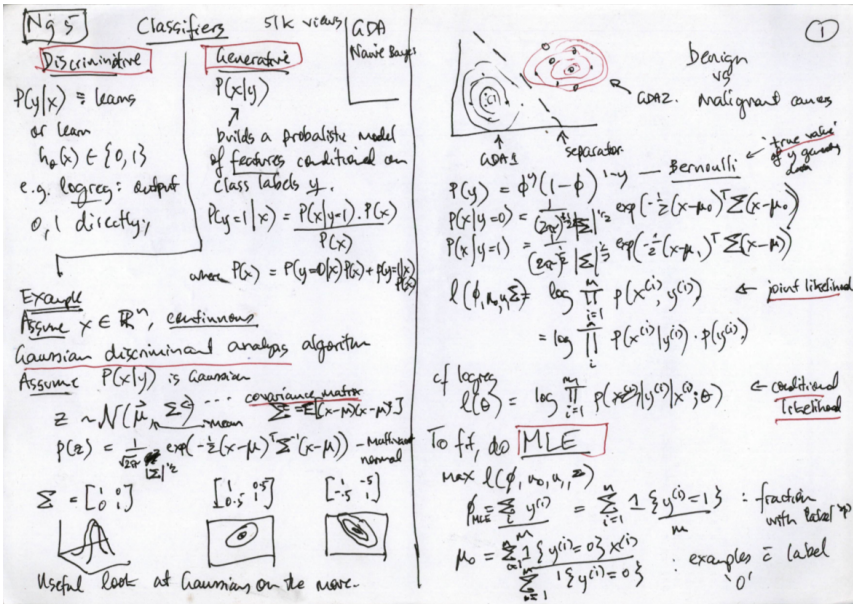


Figure 2.5: Class notes lecture 5, Stanford CS229, 2007

Ng's CS229 lectures differ from most other pedagogical materials in that we see someone writing and deriving line after line of equations using chalk on a blackboard. Occasionally, questions come from students in the audience (not shown on the Youtube videos), but mostly Ng's transcription of equations and other diagrams from the paper notes he holds to blackboard continues uninterrupted.[7] Ng's Youtube anachronistic but popular lectures have a certain diagrammatic density that comes from the many hours of chalked writing he generates during the course deriving equations. In a time when PowerPoint presentations or some other electronic textuality would very much have been the norm (2007), why is a computer science professor, teaching a fairly advanced postgraduate course, writing on a chalkboard by hand?

7. After sitting through 20 hours of Ng's online lectures, and attempting some of the review questions and programming exercises, including implementing well-known algorithms using `R`, one comes to know datasets such as the San Francisco house price dataset and Fisher's `iris` (Fisher 1936) quite well. Like the textbook problems that the historian of science Thomas Kuhn long ago described as one of the anchor points in scientific cultures (Kuhn 1996), these iconic datasets provide an entry point to the 'disciplinary matrix' of machine learning. Through them, one gains some sense of how predictive models are constructed, and what kinds of algorithmic architectures and forms of data are preferred in machine learning.

diagram!hand-drawn
mathematics!equations!derivation of
diagram!hand-drawn

Figure 2.5 shows a brief portion of around 100 pages of notes I made on this course. The act of writing down these equations and copying the many hand-drawn graphs Ng produced was a deliberative diagrammatic experiment in 'returning again and again' to what is perhaps overly hardened in *Elements of Statistical Learning*. Like the 50,000 or so other people who had watched this video, I partly complied with Ng's injunction to 'copy it, write it out, cover it, and see if you can reproduce it' (*Lecture 10 | Machine Learning (Stanford)* 2008). While it occasions much writing and drawing, and many struggles to keep up with the transformations and substitutions that Ng narrates as he writes, it seems to me that writing out derivations, with all their substitutions and variations, alongside the graphic sketches of intuitions about the machine learners, accesses something of the *diagrammatic composition* of machine learning that is quite hard to track in *Elements of Statistical Learning*. There the diagrammatic weave between the expressions of linear algebra, calculus, statistics, and the off-stage implementation in code is almost too tight to work with. In Ng's CS229 lectures, by contrast, the weaving, while still complex, is much more open. The lectures lack the citational tapestry of *Elements of Statistical Learning*. They are not able to wield the datasets and the panoply of graphic forms found there, and virtually no machine learning code appears on the blackboard (although the CS229 student assignments, also to be found online, are code implementations of the algorithms and models). But Ng's lectures move more slowly, and we begin to see some of the different groupings and associations comprising the operational formation. More importantly perhaps, this absorbing process of writing derivations might begin to transform ways of thinking, saying and knowing.

The visible learning of machine learning

learning

learning!as function-finding

artificial intelligence!relation to machine learning

learning

Rosenblatt, Frank

machine learner!perceptron|(

For a book with 'learning' in its title, *Elements of Statistical Learning* has visibly little to say about how to learn machine learning. 'Learning' is briefly discussed on the first page of *Elements of Statistical Learning*, but the book hardly ever returns to the topic or even that term explicitly. We learn on page 2 that a 'learner' in machine learning is a model that predicts outcomes. (As I discuss in chapter **??**, learning is comprehensively understood in machine learning as finding a mathematical *function* that could have generated the data, and optimising the search for that function as much as possible.) The notion of learning in machine learning derives from the field of artificial intelligence. The broad project of artificial intelligence, at least as envisaged in its 1960s-1970s heyday as a form of symbolic reasoning, is today largely regarded as a dead-end.

How then did learning get into the title of *Elements of Statistical Learning*? Does 'learning' anthropomorphize statistical modelling or computer programming? The so-called 'learning problem' and the theory of learning machines developed by researchers in the 1960-1970s was largely based on work already done in the 1950s on cybernetic devices such as the perceptron, the prototypical neural network model developed by the psychologist Frank Rosenblatt in the 1950s (Rosenblatt 1958). Drawing on the McCulloch-Pitts model of the neurone and mathematical techniques of optimization (Bellman 1961), Rosenblatt implemented the perceptron, which today would be called a single-layer neural network (Hastie, Tibshirani, and Friedman 2009, 393) as an electro-mechanical device at the Cornell University Aeronautical Laboratory in 1957. For present purposes, it is interesting to see what diagrams Rosenblatt used and how they differ from contemporary diagrams.

Vapnik, Vladimir

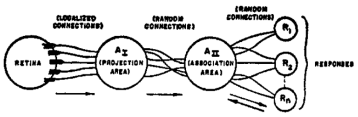
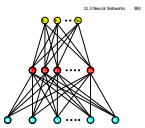


FIG. 1. Organization of a perceptron.

(a) The neurological perceptron (Rosenblatt, 1958, 389)



(b) Neural network from Hastie (2009); Single layer feedforward artificial neural network.

Figure 2.6: 1958 perceptron and 2001 neural net compared

If we compare the diagram of Rosenblatt's perceptron shown in Figure 2.6a to the typical contemporary diagram of a neural network shown in Figure 2.6b, the differences are not that great in many ways. The diagram of a neural network found in Rosenblatt's paper (Rosenblatt 1958) has no mathematical symbols on it, but the one from (Hastie, Tibshirani, and Friedman 2009) does. Rosenblatt's retains neuro-cognitive-anatomical reference points (retina, association area, projection area) whereas Hastie et. al.'s replaces them with the symbols that we have already seen in play in the expression for the linear model **??**. What has happened between the two diagrams? As Vladimir Vapnik, a leading machine learning theorist, observes: 'the perceptron was constructed to solve pattern recognition problems; in the simplest case this is the problem of constructing a rule for separating data of two different categories using given examples' (Vapnik 1999, 2). While computer scientists in artificial intelligence of the time, such as Marvin Minsky and Seymour Papert, were sceptical about the capacity of the perceptron model to distinguish or 'learn' different patterns (Minsky and Papert 1969), later work showed that perceptrons could 'learn

machine learner!perceptron|)

machine learner!neural net

universally'[8]. For present purposes, the key point is not that neural networks have turned out several decades later to be extremely powerful algorithms in learning to distinguish patterns, and that intense research in neural networks has led to their ongoing development and increasing sophistication in many 'real world' applications (see for instance, for their use in sciences (Hinton and Salakhutdinov 2006), or in commercial applications such as drug prediction (Dahl 2013) and above all in the current surge of interest in 'deep learning' by social media platform and search engines such as Facebook and Google). For our purposes, the important point is that the notion of the learning machine began to establish an ongoing diagonalization that transforms basic diagrammatic pattern through substitutions of increasingly convoluted or nested operations. The whole claim that machines 'learn' rests on this

8. In describing the entwined elements of machine learning techniques, and citing various machine learning textbooks, I'm not attempting to provide any detailed history of their development. My archaeology of these developments does not explore the archives of institutions, laboratories or companies where these techniques took shape. It derives much more either from following citations back out of the highly distilled textbooks into the teeming collective labour of research on machine learning as published in hundreds of thousands of articles in science and engineering journals, or from looking at, experimenting with and implementing techniques in code. For instance, (Olazaran 1996) offers a history of the perceptron controversy from a science studies perspective. During the 1980s, artificial intelligence and associated approaches (expert systems, automated decision support, neural networks, etc.) were a matter of some debate in the sociology and anthropology of science. The work of Harry Collins would be one example of this (Collins 1990), Paul Edwards on artificial intelligence and Cold War (Edwards 1996), Nathan Ensmenger on chess (Ensmenger 2012), and Lucy Suchman on plans and expert systems (L. A. Suchman 1987) would be others. Philosophers such as Hubert Dreyfus (*What Computers Can't Do* (Dreyfus 1972, 1992) had already extensively criticised AI. In the 1990s, the appearance of new forms of simulation, computational fields such as a-life and new forms of robotics such as Rodney Brook's much more insect-like robots at MIT attracted the interest of social scientists (Helmreich 2000) amongst many others. Sometimes this interest was critical of claims to expertise (Collins), and at other times, interested in how to make sense of the claims without foreclosing their potential novelty (Helmreich). By and large, I don't attend to controversies in machine learning as a scientific field, and I don't directly contest the epistemic authority of the proponents of the techniques. I share with Lucy Suchman an interest in how the 'effect of machine-as-agent is generated' and in how 'translations … render former objects as emergent subjects'(L. Suchman 2006, 2). I diverge around the site of empirical attention. I'm persevering with the diagrams in each of the following chapters in order to track the movement of tendencies that are not so visible in terms of either the controversies or the assumptions of agency embodied in many AI systems of the 1980s. The agency of machine learning, in short, might not reside so much in any putative predictive or classificatory power if manifests, but rather its capacity to mutate and migrate across contexts.

diagrammatization
diagram|(

diagrammatization that recurrently and sometimes recursively transforms the icon of relations, sometimes in the graphic forms shown above and more often in the algebraic patterns.

The changes in graphics suggest transformations in the operational formation. I am suggesting, then, that we should follow the transformations associated with machine learning diagrammatically, provided we maintain a rich understanding of diagram, and remain open to multiple vectors of abstraction. Following Peirce, we might begin to see machine learning as a diagrammatic practice in which different semiotic forms – lines, numbers, symbols, operators, patches of colour, words, images, marks such as dots, crosses, ticks and arrowheads – are constantly connected, substituted, embedded or created from existing diagrams. The diagrams we have already seen from *Elements of Statistical Learning* - algebraic formulae and network topology - don't exhaust the variations at all. Just a brief glance through this book or almost any other in the field shows not only many formulae, but tables, matrices, arrays, line graphs, contour plots, scatter plots, dendrograms and trees, as well as algorithms expressed as pseudo-code. The connections between these diagrams are not always very tight or close. Learning to machine learning (whether you are a human learner or a learner in the sense of a machine) means dancing between diagrams. This dance is relatively silent and sometimes almost motionless as signs slide between different diagrammatic articulations. Diagrammatization offers then a way to track the ongoing project which tries treat data like farmers treat crops (see epigraph from Domingos in this chapter). To understand what machines can learn, we need to look at how they have been drawn, designed, or formalised. But what in this work of designing and formalising predictive models is like farming? Some very divergent trajectories open up here. On the one hand, the diagrams

become machines when they are implemented. On the other hand, the machines generate new diagrams when they function. We need to countenance both forms of movement in order to understand any of the preceding diagrams – the algebraic expressions or the diagrams of models such as the perceptron or its descendants, the neural network. This means going downstream from the textbooks into actual implementations and places where people, algorithms, and machines mingle more than they do in the relatively neat formality of the textbooks. Rather than history or controversies in the field, I focus on the migratory patterns of methods, and the many configurational shifts associated with their implementations as the same things appears in different places.

The diagram of an operational formation

Does machine learning radically change programming practice? Programmability does change but only through gyrations between epistemic, infrastructural and discursive heterogeneities. I have been suggesting that we can get a sense of the heterogeneities and regularities of machine learning by treating *Elements of Statistical Learning* as a diagram that links and indexes the operations of machine learners in publication, in computation, in code. Mapped through research publications, pedagogical materials or code libraries in `R`, these operations form a primary field of expressions issuing from many parts. These parts include the accumulation or positivity of scientific literature with all its referentiality. They include mathematical derivation and formalization as an accelerated diagrammatic movement. The parts include code as materially weaving of infrastructures, conventions, standards, techniques, devices and collective relations. Statements (especially linear algebra, calculus, statistics), problems and techniques from multiple scientific disciplines (especially computer science, but also

data!table

biology, medicine and others), devices such as computing platforms, data formats, and code repositories populate the operational formation. The operational power of machine learning does not stem from a single layer of abstraction. The diagrammatic forms of movement we have begun to discern in the polymorphic *Elements of Statistical Learning* suggest key lines and paths worth following in opening up that engagement. Like the perceptron calculating weights that allow it to express the logic of the NAND function, we might first of all turn to the table, the ordering and aligning of numbers on which nearly all machine learning depends.

Cita-tions	Field
556	Engineering
520	Computer Science
383	Engineering", "Computer Science
369	Biotechnology & Applied Microbiology
344	Mathematical & Computational Biology
202	Mathematics", "Computer Science
175	Chemistry
150	Remote Sensing
116	Instruments & Instrumentation
100	Imaging Science & Photographic Technology", "Computer Science
82	Engineering", "Automation & Control Systems
78	Mathematics", "Biochemistry & Molecular Biology
77	Public, Environmental & Occupational Health
75	Medical Informatics
72	Operations Research & Management Science", "Computer Science
69	Mathematics
67	Research & Experimental Medicine
66	Geology
62	Marine & Freshwater Biology
60	Engineering", "Engineering

Table 2.3: Subject categories of research publications citing *Elements of Statistical Learning* 2001-2015. The subject categories derive from Thomson-Reuter *Web of Science*.

	How often suggested
testthat	325
knitr	163
MASS	78
knitr, rmarkdown	71
testthat, knitr	52
testthat, knitr, rmarkdown	38
RUnit	37
knitr, testthat	30
R.rsp	26
lattice	25
parallel	25
knitr, rmarkdown, testthat	22
ggplot2	18
mvtnorm	16
survival	16
rgl	12
testthat, covr	8
testthat, roxygen2	8
xtable	8
akima	7

Table 2.4: R packages suggested by the ElemStatLearn package

Glossary

classifier A machine learner that assigns instances to classes or categories.. 6

operational formation is a variation on Michel Foucault's discursive formation that highlights the collective human-machine regularities of power-knowledge. While operation and operational fields are intrinsic to Foucault's account of discursive practice, they are somewhat overshadowed by the figures of the document, the utterance, and the proposition.. 8

vectorised Operations on data that transform vectors of values.. *see also* vector

Bibliography

ACM. 2013. "John M. Chambers - Award Winner." Accessed December 12, 2013. http://awards.acm.org/award_winners/chambers_6640862.cfm.

Adler, Joseph, and Jörg Beyer. 2010. *R in a Nutshell.* O'Reilly Germany.

Agency, National Security. 2012. "SKYNET: Courier Detection via Machine Learning." Accessed October 29, 2015. https://theintercept.com/document/2015/05/08/skynet-courier/.

Alpaydin, Ethem. 2010. *Introduction to machine learning.* Cambridge, Massachusetts; London: MIT Press.

Amoore, Louise. 2011. "Data Derivatives On the Emergence of a Security Risk Calculus for Our Times." *Theory, Culture & Society* 28 (6): 24–43.

Arthur, Heather. 2012. "harthur/kittydar." Accessed September 16, 2014. https://github.com/harthur/kittydar.

Barocas, Solon, Sophie Hood, and Malte Ziewitz. 2013. *Governing Algorithms: A Provocation Piece.* SSRN SCHOLARLY PAPER ID 2245322. Rochester, NY: Social Science Research Network.

BBC. 2012. "Google 'brain' machine spots cats." *BBC News: Technology* (June 26).

Beer, David, and Roger Burrows. 2013. "Popular culture, digital archives and the new social life of data." *Theory, Culture & Society.*

Bellman, Richard. 1961. *Adaptive control processes: a guided tour.* Vol. 4. Princeton N.J.: Princeton University Press.

Beniger, James R. 1986. *The control revolution: technological and economic origins of the information society.* Harvard University Press.

Bishop, Christopher M. 2006. *Pattern recognition and machine learning.* Vol. 1. New York: Springer.

Bollas, Christopher. 2008. *The evocative object world.* London & New York: Routledge.

Breiman, Leo. 2001. "Statistical modeling: The two cultures (with comments and a rejoinder by the author)." *Statistical Science* 16 (3): 199–231.

Campbell-Kelly, Martin. 2003. *From Airline Reservations to Sonic the Hedgehog: A History of the Software Industry.* Cambridge, MA: MIT Press.

Cheney-Lippold, John. 2011. "A new algorithmic identity soft biopolitics and the modulation of control." *Theory, Culture & Society* 28 (6): 164–181.

Collins, Harry M. 1990. *Artificial experts: Social knowledge and intelligent machines.* Inside technology. Cambridge, MA.: MIT Press.

Conway, Drew, and John Myles White. 2012. *Machine learning for hackers.* Sebastopol, CA: O'Reilly.

Couldry, Nick. 2012. *Media, society, world: Social theory and digital media practice.* Cambridge ; Malden, MA: Polity.

CRAN. 2010. "The Comprehensive R Archive Network." Accessed May 5, 2010. http://www.stats.bris.ac.uk/R/.

Dahl, George. 2013. "Deep Learning How I Did It: Merck 1st place interview." Accessed June 17, 2013. http://blog.kaggle.com/2012/11/01/deep-learning-how-i-did-it-merck-1st-place-interview/.

Deleuze, Gilles. 1988. *Foucault.* Translated by Seân Hand. Minneapolis: University of Minnesota Press.

Domingos, Pedro. 2012. "A few useful things to know about machine learning." *Communications of the ACM* 55 (10): 78–87.

———. 2015. *The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World.* New York: Basic Civitas Books, September 1.

Dreyfus, Hubert L. 1972. *What computers can't do.* New York: Harper & Row.

———. 1992. *What computers still can't do: a critique of artificial reason.* Cambridge, MA: MIT Press.

Edwards, Paul N. 1996. *The closed world : computers and the politics of discourse in Cold War.* Inside technology. Cambridge, MA; London: MIT Press.

Ensmenger, Nathan. 2012. "Is Chess the Drosophila of Artificial Intelligence? A Social History of an Algorithm." *Social Studies of Science* 42, no. 1 (February 1): 5–30.

Fico. 2015. "FICO® Analytic Modeler Decision Tree Professional | FICO™." Accessed November 1, 2015. http://www.fico.com/en/products/fico-analytic-modeler-decision-tree-professional.

Fisher, Ronald A. 1936. "The use of multiple measurements in taxonomic problems." *Annals of eugenics* 7 (2): 179–188.

Flach, Peter. 2012. *Machine learning: the art and science of algorithms that make sense of data.* Cambridge University Press.

Foucault, Michel. 1972. *The archaeology of knowledge and the discourse on language.* Translated by Allan Sheridan-Smith. New York: Pantheon Books.

———. 1977. *Discipline and punish: The birth of the prison.* Translated by Allan Sheridan-Smith. New York: Vintage.

———. 1998. *The Will to Knowledge: The History of Sexuality.* Translated by Robert Hurley. Vol. 1. London: Penguin.

Fuller, Matthew, and Andrew Goffey. 2012. *Evil Media.* Cambridge, Mass: MIT Press.

Galloway, Alexander. 2014. "The Cybernetic Hypothesis." *differences* 25, no. 1 (January 1): 107–131.

Galloway, Alexander R. 2004. *Protocol: how control exists after decentralization.* Leonardo (Series) (Cambridge, Mass.) Cambridge, Mass.: MIT Press.

Gillespie, Tarleton. 2010. "The politics of 'platforms'." *New Media & Society* 12 (3): 347–364.

———. 2014. "The Relevance of Algorithms." In *Media technologies: Essays on communication, materiality, and society,* edited by Tarleton Gillespie, Pablo Boczkowski, and Kirsten A. Foot, 167–194. Cambridge, MA: MIT Press.

Google. 2015. "TensorFlow – an Open Source Software Library for Machine Intelligence." Accessed June 7, 2016. https://www.tensorflow.org/.

Guattari, Félix. 1984. *Molecular revolution : psychiatry and politics.* Harmondsworth, Middlesex, England ; New York, N.Y., U.S.A.: Penguin.

Guattari, Felix, and Gilles Deleuze. 1988. *A thousand plateaus: capitalism and schizophrenia.* London: Athlone, 1988.

Hallinan, Blake, and Ted Striphas. 2014. "Recommended for you: The Netflix Prize and the production of algorithmic culture." *New Media & Society* (June 23): 1–21.

Hastie, Trevor, Robert Tibshirani, and Jerome H. Friedman. 2001. *The elements of statistical learning: data mining, inference, and prediction.* 1st edition. New York: Springer.

———. 2009. *The elements of statistical learning: data mining, inference, and prediction.* 2nd edition. New York: Springer.

Helmreich, Stefan. 2000. *Silicon second nature : culturing artificial life in a digital world.* Berkeley, Calif. ; London: University of California Press.

Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. 2006. "Reducing the dimensionality of data with neural networks." *Science* 313 (5786): 504–507.

Issenberg, Sasha. 2012. "The Definitive Story of How President Obama Mined Voter Data to Win A Second Term | MIT Technology Review." Accessed January 9, 2013. http://www.technologyreview.com/featuredstory/509026/how-obamas-team-used-big-data-to-rally-voters/.

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An introduction to statistical learning.* Springer.

Jockers, Matthew L. 2013. *Macroanalysis: Digital Methods and Literary History.* Urbana: University of Illinois Press.

Kirk, Matthew. 2014. *Thoughtful Machine Learning: A Test-Driven Approach.* 1 edition. Sebastopol, Calif.: O'Reilly Media.

Kuhn, Thomas S. 1996. *The structure of scientific revolutions.* Chicago, IL: University of Chicago Press.

Lanier, Jaron. 2013. *Who owns the future?* London: Allen Lane.

Lantz, Brett. 2013. *Machine Learning with R.* Birmingham: Packt Publishing.

Larsen, Jeff. 2012. "How ProPublica's Message Machine Reverse Engineers Political Microtargeting." Accessed August 28, 2014. http://www.propublica.org/nerds/item/how-propublicas-message-machine-reverse-engineers-political-microtargeting.

Le, Quoc V., Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg S. Corrado, Jeff Dean, and Andrew Y. Ng. 2011. "Building high-level features using large scale unsupervised learning" (December 28). arXiv: 1112.6209.

Lecture 1 | Machine Learning (Stanford). 2008.

Lecture 10 | Machine Learning (Stanford). 2008. In collaboration with Andrew Ng. July 23.

Lee, D. D., and H. S. Seung. 1999. "Learning the parts of objects by non-negative matrix factorization." *Nature* 401, no. 6755 (October 21): 788–791. pmid: 10548103.

Levine, Sergey, Peter Pastor, Alex Krizhevsky, and Deirdre Quillen. 2016. "Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection" (March 7). arXiv: 1603.02199 [cs].

Mackenzie, Adrian. 2014. "UseR! Aggression, Alterity and Unbound Affects in Statistical Programming." In *Fun and Software: Exploring Pleasure, Paradox and Pain in Computing,* edited by Olga Goriunova. New York: Bloomsbury Academic.

Mackenzie, Adrian, Matthew Fuller, Andrew Goffey, Mills , Richard, and Stuart Sharples. 2016. "Code repositories as expressions of urban life." In *Code and the City,* edited by Rob Kitchin. London: Routledge.

Madrigal, Alexis C. 2014. "How Netflix Reverse Engineered Hollywood." Accessed August 28, 2014. http://www.theatlantic.com/technology/archive/2014/01/how-netflix-reverse-engineered-hollywood/282679/.

Malley, James D., Karen G. Malley, and Sinisa Pajevic. 2011. *Statistical Learning for Biomedical Data.* 1st ed. Cambridge University Press.

Matloff, Norman S. 2011. *Art of R programming.* San Francisco: No Starch Press.

McMillan, Robert. 2013. "How Google Retooled Android With Help From Your Brain." February 18. Accessed August 4, 2015. http://www.wired.com/2013/02/android-neural-network/.

Minsky, Marvin, and Seymour Papert. 1969. "Perceptron: an introduction to computational geometry." *The MIT Press, Cambridge, expanded edition* 19:88.

Mitchell, Tom M. 1997. *Machine learning*. New York, NY [u.a.: McGraw-Hill.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. "Playing Atari with Deep Reinforcement Learning" (December 19). arXiv: 1312.5602 [cs].

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, et al. 2015. "Human-level control through deep reinforcement learning." *Nature* 518, no. 7540 (February 26): 529–533.

Mohr, John W., and Petko Bogdanov. 2013. "Introduction—Topic models: What they are and why they matter." *Poetics* 41, no. 6 (December): 545–569.

Mol, Annemarie. 2003. *The Body Multiple: Ontology in Medical Practice.* Durham, N.C: Duke University Press.

Montfort, Nick, and Ian Bogost. 2009. *Racing the Beam: The Atari Video Computer System.* MIT Press, January 9.

Morton, Timothy. 2013. *Hyperobjects: Philosophy and Ecology After the End of the World.* Univ Of Minnesota Press.

Muenchen, Robert A. 2014. "The Popularity of Data Analysis Software." Accessed September 2, 2015. http://r4stats.com/articles/popularity/.

Munster, Anna. 2013. *An Aesthesia of Networks: Conjunctive Experience in Art and Technology.* MIT Press.

Neyland, Daniel. 2015. "On Organizing Algorithms." *Theory, Culture & Society* 32, no. 1 (January 1): 119–132.

Olazaran, Mikel. 1996. "A Sociological Study of the Official History of the Perceptrons Controversy." *Social Studies of Science* 26, no. 3 (January 8): 611–659.

Pasquinelli, Matteo. 2014. "Italian Operaismo and the Information Machine." *Theory, Culture & Society* (February 2): 1–20.

———. 2015. "Anomaly Detection: The Mathematization of the Abnormal in the Metadata Society." Berlin.

Pearson, Karl. 1901. "LIII. On lines and planes of closest fit to systems of points in space." *Philosophical Magazine Series 6* 2, no. 11 (November 1): 559–572.

Peirce, Charles Sanders. 1992. *The Essential Peirce: 1867-1893 v. 1: Selected Philosophical Writings.* John Wiley & Sons.

———. 1998. *The Essential Peirce - Volume 2: Selected Philosophical Writings: (1893-1913) v. 2.* Indiana University Press.

R Development Core Team. 2010. "The R Project for Statistical Computing." Accessed June 11, 2010. http://www.r-project.org/.

RexerAnalytics. 2015. "Rexer Analytics 7th Annual Data Miner Survey - 2015." Accessed May 9, 2011. http://www.rexeranalytics.com/Data-Miner-Survey-2015-Intro.html.

Ripley, Brian. 1996. *Pattern recognition and neural networks. 1996.* Cambridge ; New York: Cambridge University Press.

Rosenblatt, F. 1958. "The perceptron: A probabilistic model for information storage and organization in the brain." *Psychological Review* 65 (6): 386–408.

Russell, Matthew A. 2011. *Mining the social web.* Sebastopol, CA: O'Reilly.

Schutt, Rachel, and Cathy O'Neil. 2013. *Doing data science.* Sebastopol, Calif.: O'Reilly & Associates Inc.

Segaran, Toby. 2007. *Programming collective intelligence: building smart web 2.0 applications.* Sebastapol CA.: O'Reilly.

Smith, Marquard. 2013. "Theses on the Philosophy of History: The Work of Research in the Age of Digital Searchability and Distributability." *Journal of Visual Culture* 12, no. 3 (December 1): 375–403.

Stigler, Stephen M. 1986. *The history of statistics: the measurement of uncertainty before 1900.* Cambridge, Mass.: Harvard University Press.

Suchman, Lucy. 2006. *Human and Machine Reconfigurations: Plans and Situated Actions.* 2nd ed. Cambridge University Press, December 4.

Suchman, Lucy A. 1987. *Plans and situated actions : the problem of human-machine communication.* Cambridge: Cambridge University Press.

Teetor, Paul. 2011. *R cookbook.* O'Reilly Media, Incorporated.

Thrun, Sebastian, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, and Gabriel Hoffmann. 2006. "Stanley: The robot that won the DARPA Grand Challenge." *Journal of field Robotics* 23 (9): 661–692.

Totaro, Paolo, and Domenico Ninno. 2014. "The concept of algorithm as an interpretative key of modern rationality." *Theory, Culture & Society:* 29–49.

Van Dijck, José. 2012. "Facebook and the engineering of connectivity: A multi-layered approach to social media platforms." *Convergence: The International Journal of Research into New Media Technologies:* 1354856512457548.

Vance, Ashlee. 2011. "This Tech Bubble Is Different." *BusinessWeek: magazine* (April 14).

Vapnik, Vladimir. 1999. *The Nature of Statistical Learning Theory.* 2nd ed. 2000. Springer, December 1.

Venables, William N., and Brian D. Ripley. 2002. *Modern applied statistics with S.* Springer.

What we're learning from online education | Video on TED.com. 2012. In collaboration with Daphne Koller. August.

Whitehead, Alfred North. 1956. *Modes of thought; six lectures delivered in Wellesley College, Massachusetts, and two lectures in the University of Chicago.* New York, Cambridge University Press.

Wikibooks. 2013. "R Programming - Wikibooks, open books for an open world." Accessed June 27, 2013. http://en.wikibooks.org/wiki/R_Programming.

Perceptron. 2013. In *Wikipedia, the free encyclopedia,* by Wikipedia.

Wilf, Eitan. 2013. "Toward an Anthropology of Computer-Mediated, Algorithmic Forms of Sociality." *Current Anthropology* 54, no. 6 (December 1): 716–739. JSTOR: 10.1086/673321.

Index