BIG DATA & SOCIETY

Original Research Article

Big Data & Society
January–June 2016: 1–12
© The Author(s) 2016
Reprints and permissions:
sagepub.com/journalsPermissions.nav
DOI: 10.1177/2053951715622512
bds.sagepub.com
SAGE

How the machine 'thinks': Understanding opacity in machine learning algorithms

Jenna Burrell

Abstract

This article considers the issue of opacity as a problem for socially consequential mechanisms of classification and ranking, such as spam filters, credit card fraud detection, search engines, news trends, market segmentation and advertising, insurance or loan qualification, and credit scoring. These mechanisms of classification all frequently rely on computational algorithms, and in many cases on *machine learning* algorithms to do this work. In this article, I draw a distinction between three forms of opacity: (1) opacity as intentional corporate or state secrecy, (2) opacity as technical illiteracy, and (3) an opacity that arises from the characteristics of machine learning algorithms and the scale required to apply them usefully. The analysis in this article gets inside the algorithms themselves. I cite existing literatures in computer science, known industry practices (as they are publicly presented), and do some testing and manipulation of code as a form of lightweight code audit. I argue that recognizing the distinct forms of opacity that may be coming into play in a given application is a key to determining which of a variety of technical and non-technical solutions could help to prevent harm.

Keywords

Opacity, machine learning, classification, inequality, discrimination, spam filtering

This article considers the issue of opacity as a problem for socially consequential mechanisms of classification and ranking, such as spam filters, credit card fraud detection, search engines, news trends, market segmentation and advertising, insurance or loan qualification, and credit scoring. These are just some examples of mechanisms of classification that the personal and trace data we generate is subject to every day in network-connected, advanced capitalist societies. These mechanisms of classification all frequently rely on computational algorithms, and lately on *machine learning* algorithms to do this work.

Opacity seems to be at the very heart of new concerns about 'algorithms' among legal scholars and social scientists. The algorithms in question operate on data. Using this data as input, they produce an output; specifically, a classification (i.e. whether to give an applicant a loan, or whether to tag an email as spam). They are opaque in the sense that if one is a recipient of the output of the algorithm (the classification decision), rarely does one have any concrete sense of how or why a particular classification has been arrived at from inputs. Additionally, the inputs themselves may be entirely unknown or known only partially. The question naturally arises, what are the reasons for this state of not knowing? Is it because the algorithm is proprietary? Because it is complex or highly technical? Or are there, perhaps, other reasons?

By distinguishing forms of opacity that are often conflated in the emerging interdisciplinary scholarship on this topic, I seek to highlight the varied implications of algorithmic classification for longstanding matters of concern to sociologists, such as economic inequality and social mobility. Three distinct forms of opacity include: (1) opacity as intentional corporate or

School of Information, UC-Berkeley, Berkeley, CA, USA

Corresponding author:
Jenna Burrell, School of Information, UC-Berkeley, 102 South Hall #4600, Berkeley, CA 94720, USA.
Email: jburrell@berkeley.edu

institutional self-protection and concealment and, along with it, the possibility for knowing deception; (2) opacity stemming from the current state of affairs where writing (and reading) code is a specialist skill and; (3) an opacity that stems from the mismatch between mathematical optimization in high-dimensionality characteristic of machine learning and the demands of human-scale reasoning and styles of semantic interpretation. This third form of opacity (often conflated with the second form as part of the general sense that algorithms and code are very technical and complex) is the particular focus of this article. By examining in depth this form of opacity I point out shortcomings in certain proposals for code or algorithm 'audits' as a way to evaluate for discriminatory classification.

To examine this question of opacity, specifically toward the task of getting inside the algorithms themselves, I cite existing literatures in computer science, known industry practices (as they are publicly presented), and do some testing and manipulation of code as a form of lightweight audit. Along the way, I relate these forms of opacity to technical and non-technical solutions proposed to address the impenetrability of machine learning classification. Each form suggests distinct solutions for preventing harm.

So, what is new?

The word algorithm has recently undergone a shift in public presentation, going from an obscure technical term used almost exclusively among computer scientists, to one attached to a polarized discourse. The term appears increasingly in mainstream media outlets. For example, the professional body National Nurses United produced a radio spot (heard on a local radio station by the author) that starts with a voice that sarcastically declares, ''*algorithms are simple mathematical formulas that nobody understands*'' and concludes with a nurse swooping in to rescue a distressed patient from a disease diagnosis system which makes a series of comically wrong declarations about the patient's condition.[1] The purpose of the public service announcement (PSA) is to champion professional care (by nurses), in this case against error-prone automation. By contrast, efforts at corporate 'branding' of the term algorithm play up notions of algorithmic objectivity over biased human decision-making (Sandvig, 2015). In this way the connotations of the term are actively being shaped as part of advertising culture and corporate self-presentation, as well as challenged by a related counter-discourse tied to general concerns about automation, corporate accountability, and media monopolies (i.e. Tufekci, 2014).

While these new media narratives may be novel, it has long been the case that large organizations (including private sector firms and public institutions) have had internal procedures that were not fully understood to those who were subject to them. These procedures could fairly be described as 'algorithms'. What should we then make of these new uses of the term and the field of critique and analysis emerging along with it? Is this merely 'old wine in new bottles' or are there genuinely new and pressing issues related to patterns of algorithmic design as they are employed increasingly in real-world applications?

In addition to the polarization of a public discourse about algorithms, much of what is *new* in this domain is the more pervasive technologies and techniques of data collection, the more vast archives of personal data including purchasing activities, link clicks, and geospatial movement, an outcome of more universally adopted mobile devices, services, and applications and the reality (in some parts of the world) of constant connectivity. But this does not necessarily have much to do with the algorithms that operate on the data. Often it is about what composes the *data* and new concerns about privacy and the possibility (or troublingly, the impossibility) of opting-out.

Other changes have to do with particular application areas and evolving proposals for a regulatory response. The shift of algorithmic automation into new areas of what were previously white-collar work reflected in headlines like, 'will we need teachers or algorithms?'[2] and into consequential processes of classification that were previously human-determined, such as credit evaluations in an effort to realize cost-savings (as so often fuels shifts toward automation) (Straka, 2000). In the domain of credit and lending, Fourcade and Healy point to a shift from prior practices of exclusionary lending to a select few, to more generous credit offered to a broader spectrum of society, but offered to some on unfavorable, even usurious terms. This shift is made possible by 'the emergence and expansion of methods of tracking and classifying consumer behavior' (Fourcade and Healy, 2013: 560). These methods are (in part) implemented as algorithms in computers. Here the account seems to suggest an expansion of the territory of work claimed by particular algorithmic routines, that they are taking on a broader range of types of tasks at a scale that they were not previously.

In this emerging critique of 'algorithms' carried out by scholars in law and in the social sciences, few have considered in much depth their mathematical design. Many of these critics instead take a broad socio-technical approach looking at 'algorithms in the wild.' The algorithms in question are studied for the way they are situated within a corporation, under the pressure of profit and shareholder value, and as they are applied to particular real-world user populations (and the data these populations produce). Thus something more than the algorithmic logic is being examined. Such analyses are often particular to an implementation (such as Google's search engine) with its specific user base and uniquely accumulated history of problems and failures with resulting parameter setting and manual tweaking by programmers. Such an approach may not surface important broader patterns or risks to be found in particular classes of algorithms.

Investigating opacity: A method and approach

In general, we cannot look at the code directly for many important algorithms of classification that are in widespread use. This opacity (at one level) exists because of proprietary concerns. They are closed in order to maintain competitive advantage and/or to keep a few steps ahead of adversaries. Adversaries could be other companies in the market or malicious attackers (relevant in many network security applications). However, it is possible to investigate the general computational designs that we know these algorithms use by drawing from educational materials.

To do this I draw, in part, from classic illustrative examples of particular machine learning models, of the sort used in undergraduate education. In this case I have specifically examined programming assignments for a Coursera course in machine learning. These examples offer hugely simplified versions of computational ideas scaled down to run on a student's personal computer so that they return output almost immediately. Such examples do not force a confrontation with many thorny, real-world application challenges. That said, the ways that opacity endures in spite of such simplification reveal something important and fundamental about the limits to overcoming it.

Machine learning algorithms do not encompass all of the algorithms of interest to scholars now studying what might be placed under the banner of the 'politics of algorithms.'[3] However, they are interesting to consider specifically because they are typically applied to classification tasks and because they are used to make socially consequential predictions such as 'how likely is this loan applicant to default?' In the broader domain of algorithms implemented in various areas of concern (such as search engines or credit scoring) machine learning algorithms may play either a central or a peripheral role and it is not always easy to tell which is the case. For example, a search engine request is algorithmically driven,[4] but search engine algorithms are not at their core 'machine learning' algorithms. Search engines employ machine learning algorithms for particular purposes, such as detecting ads or blatant search ranking manipulation and prioritizing search results based on the user's location.[5]

While not all tasks that machine learning is applied to are classification tasks, this is a key area of application and one where many sociological concerns arise. As Bowker and Star note in their account of classification and its consequences, 'each category valorizes some point of view and silences another' and there is a long history of lives 'broken, twisted, and torqued by their encounters with classification systems' such as the race classification system of apartheid South Africa and the categorization of tuberculosis patients, as they detail (Bowker and Star, 1999). The claim that algorithms will classify more 'objectively' (thus solving previous inadequacies or injustices in classification) cannot simply be taken at face value given the degree of human judgment still involved in designing the algorithms, choices which become built-in. This human work includes defining features, pre-classifying training data, and adjusting thresholds and parameters.

Opacity

Below I define a typology starting first with the matter of 'opacity' as a form of proprietary protection or as 'corporate secrecy' (Pasquale, 2015). Secondly, I point to opacity in terms of the readability of code. Code writing is a necessary skill for the computational implementation of algorithms, and one that remains a specialist skill not found widely in the general public. Finally, arriving at the major point of this article, I contrast a third form of opacity centering on the mismatch between mathematical procedures of machine learning algorithms and human styles of semantic interpretation. At the heart of this challenge is an opacity that relates to the specific techniques used in machine learning. Each of these forms of opacity may be tackled by different tools and approaches ranging from the legislative, to the organizational or programmatic, to the technical. But importantly, the form (or forms) of opacity entailed in a particular algorithmic application must be identified in order to pursue a course of action that is likely to mitigate its problems.

Forms of opacity

Opacity as intentional corporate or state secrecy

One argument in the emerging literature on the 'politics of algorithms' is that algorithmic opacity is a largely intentional form of self-protection by corporations intent on maintaining their trade secrets and competitive advantage. Yet this is not just about one search engine competing with another to keep their 'secret sauce' under wraps. It is also the case that dominant platforms and applications, particularly those that use algorithms for ranking, recommending, trending, and filtering, attract those who want to 'game' them as part of strategies for securing attention from the general public. The field of 'search engine optimization' does just this. An approach within machine learning called 'adversarial learning' deals specifically with these sorts of evolving strategies. Network security applications of machine learning deal explicitly with spam, scams, and fraud and remain opaque in order to be effective. Sandvig notes that this 'game of cat-and-mouse' makes it entirely unlikely that most algorithms will be (or necessarily should be) disclosed to the general public (Sandvig et al., 2014: 9). That said, an obvious alternative to proprietary and closed algorithms is open source software. Successful business models have emerged out of the open source movement. There are options even in 'adversarial learning' such as the SpamAssassin spam filter for Apache.

On the other hand, Pasquale's more skeptical analysis proposes that the current extent of algorithmic opacity in many domains of application may not be justified and is instead a product of lax or lagging regulations. In his book *The Black Box Society: The Secret Algorithms that Control Money and Information* he argues that a kind of adversarial situation is indeed in play, one where the adversary is regulation itself. 'What if financiers keep their doings opaque on purpose, precisely to avoid or to confound regulation?' he asks (Pasquale, 2015: 2). In reference to this, he defines 'opacity' as 'remediable incomprehensibility.'

The opacity of algorithms, according to Pasquale, could be attributed to willful self-protection by corporations in the name of competitive advantage, but this could also be a cover for a new form of concealing sidestepped regulations, the manipulation of consumers, and/or patterns of discrimination.

For this type of opacity, one proposed response is to make code available for scrutiny, through regulatory means if necessary (Diakopoulos, 2013; Gandy, 2010; Pasquale, 2015). Underlying this particular explanation for algorithmic opacity is an assumption that if corporations were willing to expose the design of the algorithms they use, it would be possible to ascertain problems of consumer manipulation or regulatory violation by reading the code. Pasquale acknowledges that such measures could render algorithms ineffective though suggests that it may still be possible with the use of an independent, 'trusted auditor' who can maintain secrecy while serving the public interest (Pasquale, 2015: 141). In the absence of access to the code, Sandvig et al. (2014) detail and compare several forms of algorithmic audit (carried out with or without corporate cooperation) as a possible response, a way of forcing the issue without requiring access to the code itself.

Opacity as technical illiteracy

This second level of opacity stems from an acknowledgement that, at present, writing (and reading) code and the design of algorithms is a specialized skill. It remains inaccessible to the majority of the population. Courses in software engineering emphasize the writing of clean, elegant, and intelligible code. While code is implemented in particular programming languages, such as C or Python, and the syntax of these languages must be learned, they are in certain ways quite different from human languages. For one, they adhere strictly to logical rules and require precision in spelling and grammar in order to be 'read' by the machine.

Good code does double-duty. It must be interpretable by humans (the original programmer or someone adding to or maintaining the code) as well as by the computational device (Mateas and Montfort, 2005). Writing for the computational device demands a special exactness, formality, and completeness that communication via human languages does not. The art and 'craft'[6] of programming is partly about managing this mediating role and entails some well-known 'best practices' like choosing sensible variable names, including 'comments' (one-sided communication to human programmers omitted when the code is compiled for the machine), and choosing the simpler code formulation, all things being equal.

Recent calls for greater diversity in STEM fields and for general efforts toward developing 'computational thinking' at all levels of education (Lee et al., 2011; Wing, 2006) are relevant. Diakopoulos (2013) likewise suggests ways that journalists might play a valuable role in reverse engineering algorithms to inform the general public, but notes that this poses a challenge of 'human resource' development, one of developing code and computational literacy in journalists or others who wish to do this sort of examination. To address this form of opacity, widespread educational efforts would ideally make the public more knowledgeable about these mechanisms that impact their life opportunities and put them in a better position to directly evaluate and critique them.

Opacity as the way algorithms operate at the scale of application

Scholars have noted that algorithms (such as that underlying the Google search engine) are often multi-component systems built by teams producing an opacity that programmers who are 'insiders' to the algorithm must contend with as well (Sandvig et al., 2014; Seaver, 2014). A call for code 'audits' (where this means reading the code) and the employment of 'auditors' may underestimate what this would entail as far as the number of hours required to untangle the logic of the code within a complicated software system. This valid critique is nevertheless non-specific about different classes of algorithms and their particular logics.

I further argue that there are certain challenges of scale and complexity that are distinctive to machine learning algorithms. These challenges relate not simply to total number of lines or pages of code, the number of team members on the engineering team, and the multitude of interlinkages between modules or subroutines. These are challenges not just of reading and comprehending code, but being able to understand the algorithm in action, operating on data. Though a machine learning algorithm can be implemented simply in such a way that its logic is almost fully comprehensible, in practice, such an instance is unlikely to be particularly useful. Machine learning models that prove useful (specifically, in terms of the 'accuracy' of classification) possess a degree of unavoidable complexity.

Machine learning in particular is often described as suffering from the 'curse of dimensionality' (Domingos, 2012). In a 'Big Data' era, billions or trillions of data examples and thousands or tens of thousands of properties of the data (termed 'features' in machine learning) may be analyzed. The internal decision logic of the algorithm is altered as it 'learns' on training data. Handling a huge number especially of *heterogeneous* properties of data (i.e. not just words in spam email, but also email header info) adds complexity to the code. Machine learning techniques quickly face computational resource limits as they scale and may manage this, using techniques written into the code (such as 'principal component analysis') which add to its opacity. While datasets may be extremely large but possible to comprehend and code may be written with clarity, the interplay between the two in the mechanism of the algorithm is what yields the complexity (and thus opacity). Better understanding this complexity (and the barriers to overcoming the opacity it effects) is the concern of the following examples.

Machine learning: A very brief primer

Machine learning algorithms are used as powerful generalizers and predictors. Since the accuracy of these algorithms is known to improve with greater quantities of data to train on, the growing availability of such data in recent years has brought renewed interest to these algorithms.

A given machine learning algorithm generally includes two parallel operations, or two distinct algorithms: a 'classifier' and a 'learner' (see, for example, Figure 3). Classifiers take input (referred to as a set of 'features') and produce an output (a 'category'). For example, a classifier that does spam filtering takes a set of features (such as email header information, words in the body of the email, etc.) and produces one of two output categories ('spam' or 'not spam'). A decision support system that does disease diagnosis may take input (clinical presentation/symptoms, blood test results) and produce a disease diagnosis as output ('hypertension,' 'heart disease,' 'liver cancer'). However, machine learning algorithms called 'learners' must first train on test data.[7] The result of this training is a matrix of weights that will then be used by the classifier to determine the classification for new input data. This training data could, for example, be emails that have been pre-sorted and labeled as 'spam' or 'not spam.'

Machine learning encompasses a number of models that are implemented in code in different ways. Some popular machine learning models include neural networks, decision trees, Naïve Bayes, and logistic regression. The choice of model depends upon the domain (i.e. loan default prediction vs. image recognition), its demonstrated accuracy in classification, and available computational resources, among other concerns. Models may also be combined into 'model ensembles,' an approach often used in machine learning competitions that seek to maximize accuracy in classification. Two applications of machine learning using separate models will be considered below.

Visualizing opacity in a neural network

The first model and application of machine learning I wish to consider is a 'neural network' applied to an image recognition task. Because this is an image recognition task, it lends itself to an attempt to 'see' the weights output by the training algorithm. The classic example for teaching neural networks to computer science undergraduates is handwriting recognition.[8] To simplify the computational task for educational purposes, the code is implemented to recognize handwritten digits only (the numbers 0 through 9). To further simply the task, these digits are drawn within the boundaries of a space-constrained box. Viewing Figure 1 you can see some of the 'fuzziness' and ambiguity of the data that is to be classified. If you take a single handwritten number in an 8 × 8 pixel square, each pixel (and a grayscale value associated with it) becomes an input (or 'feature') to the classifier which ultimately outputs what number it recognizes (in the case of Figure 2 it should be the number 6).

In the design of a neural network, a set of input nodes connects to a second set of nodes called the 'hidden' layer (like interlinked neurons in the brain) and then to an output layer (see Figure 3). Each input node is connected to a hidden layer node and

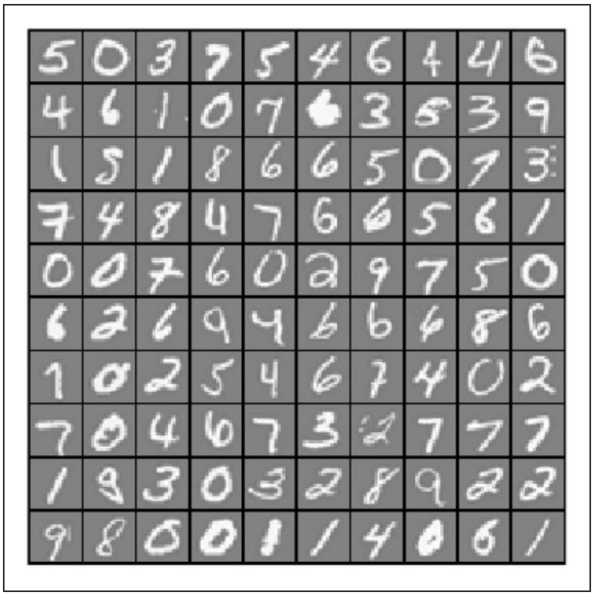

Figure 1. A set of examples of handwritten numbers that a machine learning algorithm (a ‘learner’) and, in this case, a neural network could be trained on.

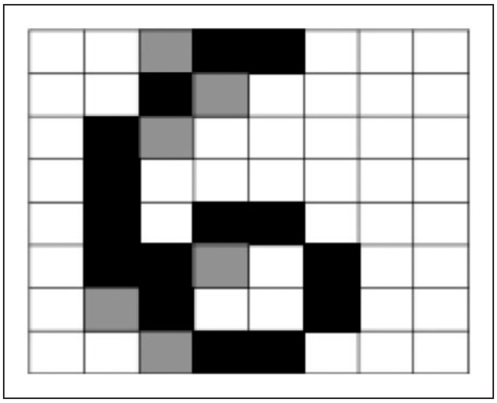

Figure 2. A handwritten number in an 8 x 8 pixel square.

each node in the hidden layer is connected to an output in the design of the neural network in Figure 3. A value or weight is associated with each of these connecting lines. The optimal values for the matrix of weights are what the learning algorithm *learns*. What is ‘optimal’ is defined by the set of weights that produce the most accurate possible classification of inputs (the individual pixels and their intensity ranging from white to black in an 8 × 8 matrix) to outputs (the handwritten numbers these pixels represent).

Because this is an image recognition task, we can actually visualize the optimized weights coming into the hidden layer node. In this way we can *see* the way a neural network breaks down the problem of recognizing a handwritten number (see Figure 4).

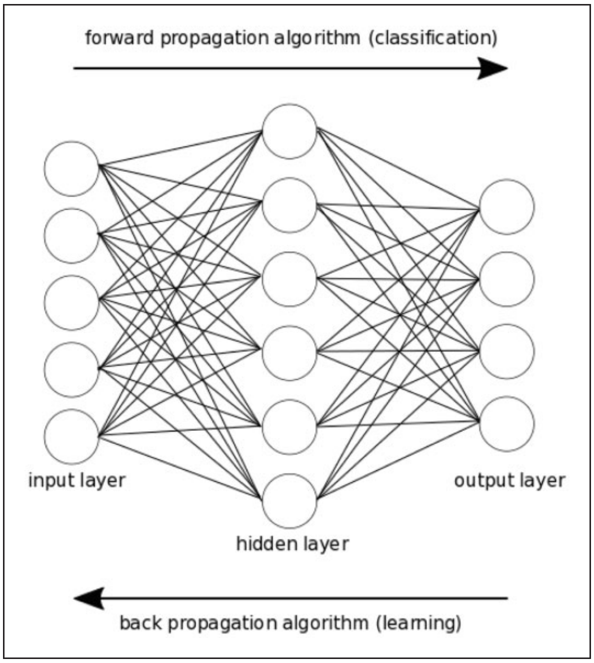
forward propagation algorithm (classification)
input layer
hidden layer
output layer
back propagation algorithm (learning)

Figure 3. Graphical depiction of a neural network.

Figure 4(a) illustrates the hidden layer in a neural network. If you look at one of the 25 boxes you can see which part of a handwritten number it cues in on. Each box represents a single node in the hidden layer and each pixel within the box illustrates the value of the weight coming from one input layer node into that particular hidden layer node. In sum, each box shows the set of weights for a simplified neural network with only one hidden layer. The regions in the box that are black are the specific pixels the node in question is most sensitive to. The top left box, for example, shows a hidden layer node that cues in on darkened pixels sort of in the lower left part of the quadrant and a little bit in the middle. A combination of the calculations coming out of these hidden layer nodes yields a classification of the inputs to a number from 0 to 9.

What is notable is that the neural network doesn’t, for example, break down handwritten digit recognition into subtasks that are readily intelligible to humans, such as identifying a horizontal bar, a closed oval shape, a diagonal line, etc. This outcome, the apparent non-pattern in these weights, arises from the very notion of computational ‘learning.’ Machine learning is applied to the sorts of problems for which encoding an explicit logic of decision-making functions very poorly. In his machine learning Coursera course, Andrew Ng describes this as the domain of *‘[applications] we cannot program “by hand”.’*[9] The ‘hand’ is implied to be a human one.[10] As noted, the craft of code writing (by humans) is two-sided communication, for fellow human programmers on the one hand and

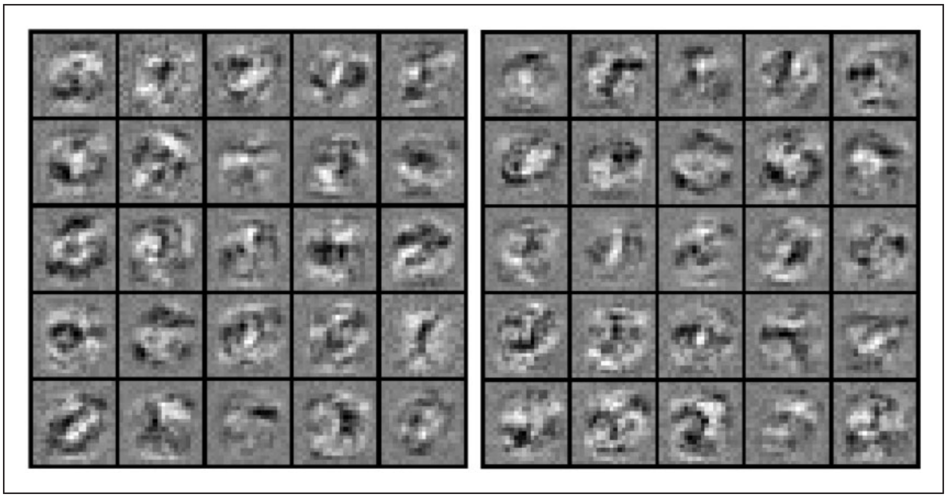

Figure 4. (a) The hidden layer: the black areas in each box are the areas (strokes or other patterns) that a particular hidden layer node cues in on in a handwritten digit. (b) This shows the result of the same learning algorithm being run a second time with the same training data. The reason (a) and (b) are not identical is because of the random initialization step that defines the set of weights initially to very small random numbers.

for the computer processor on the other. Where an algorithm does the 'programming' (i.e. optimally calculates its weights) then it logically follows that being intelligible to humans (part of the art of writing code) is no longer a concern, at least, not to the non-human 'programmer.'

The primary purpose of this first example is to give a quick, visual sense of how the machine 'thinks.' Figure 4(a) should appear unintuitive, random, and disorganized. However, handwriting recognition specifically is not a 'conscious' reasoning task in humans either. Humans recognize visual elements in an immediate and subconscious way (thus there is certainly a kind of opacity in the human process of character recognition as well). Such an example may not seem to provide much insight into broader real-world questions about discrimination in classification. However, a recent case where automated classification in Google Photos labeled a set of photos of African-American people as 'Gorillas' suggests otherwise.[11] To further the argument, my next example, spam filtering, looks at the automation of a task that calls upon a more conscious form of human reasoning. As a matter relating to core communication capabilities of the Internet, I show how spam filtering is of relevance to questions of classificatory discrimination.

The opacity of spam filtering

Spam has no fixed and indisputable definition (Brunton, 2013). It is generally understood to be unwelcome emails, especially those sent in bulk, but this is, in part, a designation by network administrators concerned particularly with overtaxing network resources. Spam filtering is, for this reason among others, a better application domain for thinking about machine learning based classification as socially consequential. Messages that are categorized as spam are messages that do not get through to their intended recipients. Consequently, this example relates more directly to ongoing conversations about the politics of search, ranking, and filtering content. Where a legitimate message is categorized as spam (a 'false positive'), this is a message that has, in effect, been unwittingly censored. One question is whether the design of spam filters could make certain individuals more susceptible to having their legitimate messages diverted to spam folders. For example, does being located in a hotbed of Internet fraud or spam activity, say West Africa (Nigeria or Ghana) or Eastern Europe, create a tendency for one's messages to be *mis*labeled as spam?

In Ng's Coursera course, support vector machines (SVMs) are the machine learning model used to implement spam filtering. SVMs are another type of machine learning model like neural networks and either model could be used for spam filtering. The simplified version used in the Coursera course does not use the 'kernel trick,' a computational technique characteristic of SVMs, so it is essentially a form of linear regression; in technical terms it uses a 'linear kernel.' As an additional simplification, the programming exercise relies solely on the contents of the email to train a spam classifier, that is, the words contained in the message alone, and no email header information. These words are analyzed by the 'learner' algorithm to determine a set of weights. These weights measure the degree to which a given word is associated with 'spam' vs. 'ham' (non-spam) emails. Such an approach is described as a 'bag of words.' There is no posited semiotic relationship between the words and no meaning in the messages is extracted, nor is there an attempt in the algorithm at narrative analysis.

I offer a lightweight 'audit' of the algorithm and an examination of the weights produced for each word and how we might make sense of them. In particular, I focus on a type of spam email, the Nigerian 419 scam, a genre with which I have deep familiarity (Burrell, 2012). The 419 scam raises an interesting concern as far as network access and spam 'false positives.' In particular, are place names, particularly 'Nigeria,' a trigger leading to a greater likelihood of categorizing as spam?

In fact after running the training algorithm on an (admittedly) very dated public corpus[12] a list of place names can be produced with their associated 'weights.' These weights are in a range from −1 (highly associated with non-spam emails) to 1 (highly associated with spam emails). Reassuringly perhaps for the general population of Nigerian email users, the weight associated with the word 'Nigeria' contained within an email is −0.001861. This means the word 'Nigeria' is essentially a neutral term.[13] Looking at the totality of spam, this makes a certain amount of sense. On the balance of it, the vast majority of spam does not originate in nor make mention of Nigeria. Presumably, the number of totally legitimate emails that mention Nigeria would further dilute an association between the country and spam email.

The words that *are* in fact most associated with spam (note, these have been destemmed so that groups of words such as guarantee, guarantees, and guaranteed can be handled as equivalent terms) are the following:

our (0.500810)
click (0.464474)
remov (0.417698)
guarante (0.384834)
visit (0.369730)
basenumb[14] (0.345389)
dollar (0.323674)
price (0.268065)
will (0.264766)
most (0.261475)
pleas (0.259571)

In many cases these are terms we *would* expect to cut across genres of spam. They seem to suggest generic appeals, pleading and promises ('guarantee'), the authority of a collective ('our'), and concrete and quantified gain or benefit (especially monetary).

Consider below a specific example of spam in the Nigerian 419 style recently caught in the author's gmail account spam filter, which is indeed categorized as spam by the simplified SVM spam filter (for the full email see Appendix 1):

> My Dearest,
> Greetings to you my Dear Beloved, I am Mrs Alice Walton, a citizen of United State. I bring to you a proposal worth $ 1,000,000,000.00 which I intend to use for CHARITY but I am so scared because it is hard to find a trust worthy human on earth...

In reading this email, I notice the formality of language and words like 'dearest' and 'beloved.' The mention of being a 'citizen,' offering 'charity' and looking for someone 'trust worthy' as well as a reference to 'fraud' also strike a note of suspicion. *None* of these words, however, are cued in on by the SVM spam filter. Rather it is the mention of money, the words 'please,' and 'contact' that are the most heavily weighted terms found in this particular email. In fact after removing the mention of money and the word 'please' from the email and running it through the 'classifier' algorithm again, it is no longer classified as spam.

Now for comparison, consider this email from a friend and research collaborator of the author, an email that has many of the same markers of the scam email genre (formality, religiosity, expressions of gratitude, etc.) but is *not* a scam email:

> Dear prof. Thank you for continually restoring hope and bringing live back to me when all hopes seem to be lost. With tears and profound gratitude I say thank you.... Am able to get a big generator, air-conditioned, a used professional Panasonic 3ccd video camera and still have about 150 dollars in my account for taking care of my health. ...I pray you continually prosper. Much regards and Bye.

The spam classifier accurately categorizes this email as *not* spam, again based purely on the words it contains (with no knowledge about the author's pre-existing relationship with the sender). Nonetheless, when run through the classifier algorithm there are certain trigger words present in the email (including 'want' and 'will') and, most incriminatingly, there is mention of money. The ranking of words by 'weight' seems to offer a kind of lever for sense-making for the interpreting human mind, but the overall classification, even in this highly simplified example, cannot easily be ascertained by a brief skimming of the words and their associated weights in a particular email. It is the aggregate of *all* of the weights of the words found in the email matched to a dictionary of 1899 of the most frequently used words. Minute differences and key words (i.e. 'visit' or 'will') that cannot easily be made sense of as part of spam strategies of social engineering and persuasion may tip the balance of the classification.

Humans likely recognize and evaluate spam according to genre: the phishing scam, the Nigerian 419 email, the Viagra sales pitch. By contrast, the 'bag of words' approach breaks down texts into atomistic collections of units, words whose ordering is irrelevant. The algorithm surfaces very general terms characteristic of spam emails, often terms that are (in isolation) quite mundane and banal. My semantic analysis attempted to reconcile the statistical patterns the algorithm surfaces with a meaning that relates to the implicit strategy of the text as a whole, but this is decidedly *not* how the machine 'thinks.'

Reconsidering 'interpretability'

The example provided of classifying Nigerian 419 style spam emails gives some insights into the strengths and shortcomings of a code audit. Finding ways to reveal something of the internal logic of an algorithm can address concerns about lack of 'fairness' and discriminatory effects, sometimes with reassuring evidence of the algorithm's objectivity, as in the case of the neutral weighting of the word 'Nigeria.' On the other hand, probing further into the 'why' of a particular classification decision yielded suggestive evidence that seemed sufficient as an explanation, but this imposed a process of human interpretive reasoning on a mathematical process of statistical optimization. In other words, machine thinking was resolved to human interpretation. Yet ambiguities remained, such as the weighting of innocuous words like 'visit' and 'want' as indicators for spam. This raises doubts about whether an explanation produced in this way to satisfy the 'why' question was necessarily a particularly correct one.

Computer scientists term this opacity issue a problem of 'interpretability.' One approach to building more interpretable classifiers is to implement an end-user facing component to provide not only the classification outcome, but also exposing some of the logic of this classification. A real-world implementation of this in the domain of spam filtering is found in Google's gmail 'spam' folder. If you select a spam message in this folder, a yellow alert box with the query 'why is this message in Spam?' above the text of the email itself provides one reason why it has been placed in this folder.[15] Messages include 'it contains content that's typically used in spam messages' (perhaps a reference to a 'bag of words' type of approach) and 'many people marked similar messages as phishing scams, so this might contain unsafe content.' And yet, explanations that bring forward a human-manageable list of key criteria (i.e. the 10 most heavily weighted/spammy words present in an email or a single sentence description) provide an understanding that is at best incomplete[16] and at worst false reassurance.

Further complicating the attempts to draw a direct line between 'weighted' inputs and classification outcomes is the mathematical manipulation that happens in between. Unlike the examples of handwriting recognition and spam filtering presented here, it is often the case that the relationship between a feature and a dimension in the model is not one-to-one. Ways of manipulating dimensionality (through principal component analysis or the 'kernel trick' in SVMs, to give two examples) are often employed to manage computational constraints or to improve accuracy.

The continuing expansion of computational power has produced certain optimization strategies that exaggerate this particular problem of *opacity as the complexity of scale* even further. With greater computational resources, and many terabytes of data to mine (now often collected opportunistically from the digital traces of users' activities), the number of possible features to include in a classifier rapidly grows way beyond what can be easily grasped by a reasoning human. In an article on the folk knowledge of applying machine learning, Domingos (2012) notes that 'intuition fails at high-dimensions.' In other words, reasoning about, debugging, or improving the algorithm becomes more difficult with more qualities or characteristics provided as inputs, each subtly and imperceptibly shifting the resulting classification.

For handling this fundamental opacity there are various proposed approaches. One approach, perhaps surprisingly, is to avoid using machine learning algorithms in certain critical domains of application.[17] There are also ways of simplifying machine learning models such as 'feature extraction', an approach that analyses what features actually matter to the classification outcome, removing all other features from the model. Some solutions perhaps wisely abandon answering the 'why' question and devise metrics that can, in other ways, evaluate discrimination (i.e. Datta et al., 2015). For example, in 'Fairness Through Awareness' a discriminatory effect in classification algorithms can be detected without extracting the 'how' and 'why' of particular classification decisions (Dwork et al., 2011). In some ways this extends the approach of the external audit proposed by Sandvig et al. (2014) and by Diakopoulos (2013) using sophisticated algorithmic implementations.

Conclusion

> There may be something in the end impenetrable about algorithms. (Gillespie, 2012)

The goal of this article was to look more deeply into machine learning algorithms and the nature of their 'opacity', relating this to sociological interests in classification and discrimination. This is part of an ongoing reorientation of the scholarship on 'digital inequality' which has frequently focused on the distribution of computational resources and skills (Hargittai, 2008) but not, until recently, the question of how people may be subject to computational classification, privacy invasions, or other surveillance in ways that are unequal across the general population and could be in violation of existing regulatory protections (Barocas and Selbst, 2016; Eubanks, 2012; Fourcade and Healy, 2013).

Legal critiques of algorithmic opacity often focus on the capacity for intentional secrecy and lead to calls for regulations to enforce transparency. Pasquale (2015) argues for the use of auditors who have access to the code and can assure that classifications are non-discriminatory. Another approach is to educate a broader swathe of society in code writing and computational skills to lessen the problem of a homogenous and elite class of technical people making consequential decisions that cannot be easily assessed by non-members. However, the opacity of machine learning algorithms is challenging at a more fundamental level. When a computer learns and consequently builds its own representation of a classification decision, it does so without regard for human comprehension. Machine optimizations based on training data do not naturally accord with human semantic explanations. The examples of handwriting recognition and spam filtering helped to illustrate how the workings of machine learning algorithms can escape full understanding and interpretation by humans, even for those with specialized training, even for computer scientists.

Ultimately partnerships between legal scholars, social scientists, domain experts, along with computer scientists may chip away at these challenging questions of fairness in classification in light of the barrier of opacity. Additionally, user populations and the general public can give voice to exclusions and forms of experienced discrimination (algorithmic or otherwise) that the 'domain experts' may lack insight into.[18] Alleviating problems of black boxed classification will not be accomplished by a single tool or process, but some combination of regulations or audits (of the code itself and, more importantly, of the algorithms functioning), the use of alternatives that are more transparent (i.e. open source), education of the general public as well as the sensitization of those bestowed with the power to write such consequential code. The particular combination of approaches will depend upon what a given application space requires.

Acknowledgments

Thank you to the many who reviewed and provided comments in the early stages of writing this paper including Sebastian Benthall, Laura Devendorf, Shreeharsh Kelkar, Marion Fourcade, Michael Carl Tshantz, Solon Barocas, David Bamman, Steve Weber and the members of the UC-Berkeley Social Science Matrix seminar on 'algorithms as computation and culture.'

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

Notes

1. https://soundcloud.com/national-nurses-united/radio-ad-algorithms
2. Khosla (2012).
3. Most scholars in this space focus on particular application spaces without specifying the technical categories of algorithms that are used. Gillespie looks at search, trending, and other content filtering and ranking algorithms (2012), Pasquale looks at reputation, search, and finance algorithms (2015), Brunton considers spam filtering (2013) while Diakopoulos's (2013) consideration is wide-ranging but tied to data journalism. Sandvig looks at search while briefly considering basic sorting algorithms taught in introductory computer science courses (2015). Solon Barocas's work focusing on machine learning algorithms specifically is a major exception to this trend (Barocas, 2014a; Barocas, 2014b; Barocas and Selbst, 2016).
4. Except for the part (generally totally invisible to users) that may be done manually by human workers who do content moderation, cross-checking, ground truthing and correction—http://www.wired.com/2014/12/google-maps-ground-truth/
5. See the question and response on this Reddit AMA with Andrew Ng about why companies make their algorithmic techniques public (https://www.reddit.com/r/MachineLearning/comments/32ihpe/ama_andrew_ng_and_adam_coates/cqbkmyb) and this quora question and response about how machine learning contributes to the Google search engine—http://www.quora.com/Why-is-machine-learning-used-heavily-for-Googles-ad-ranking-and-less-for-their-search-ranking
6. See also Ensmenger (2003) on programming as craft and programmers as a profession.

[7] This refers to the subset of machine learning approaches called ‘supervised’ learning which, for the sake of clarity of argument, is what is specifically considered here.

[8] Giving some sense of perhaps how *little* the algorithms themselves have changed, this is the exact same example used to teach neural networks in the course I took as an undergraduate in 2001 as well as in the Coursera course I completed in 2013.

[9] ‘Welcome’ video, Coursera course on machine learning. Available at: https://www.coursera.org/learn/machine-learning/lecture/RKFpn/welcome

[10] To program ‘by hand’ (in the context of classification decisions) would also entail outlining explicitly a logic of decision-making, specifically about which category to place a piece of data into. This ‘rationalistic’ approach known as symbolic AI (Olazaran, 1996) was once dominant. It is sometimes referred to, with more than a bit of nostalgia, as Good Old Fashioned AI (GOFAI) (Winograd, 2006) and it entailed the symbolic representation of knowledge in a way that is strictly formalized. However, this approach failed to live up to its early promise and underperformed on many tasks, leading to an AI ‘winter’ when interest and funding waned (Grudin, 2006).

[11] The incident is described at: http://www.slate.com/blogs/future_tense/2015/06/30/google_s_image_recognition_software_returns_some_surprisingly_racist_results.html

[12] The SpamAssassin public corpus dates from 2002; see https://spamassassin.apache.org/publiccorpus/readme.html

[13] Place names in order from least associated with spam to most: Ireland (−0.190707), American (−0.108162), Washington (−0.076769), Boston (−0.032227), America (−0.015666), India (−0.012690), European (−0.007351), Indian (−0.006872), Europ (−0.005295), Nigeria (−0.001861), French (0.001398), Kingdom (0.027125), Foreign (0.031424), Africa (0.049945), Irish (0.062301), California (0.067122), Unit (0.067960), Franc (0.097339), State (0.101561), and China (0.112738).

[14] All numbers in the text are replaced with ‘basenumb’ in the pre-processing of the email contents.

[15] A list of these explanations is available here: https://support.google.com/mail/answer/1366858?hl=en&expand=5

[16] One computer scientist likewise reminds us that ‘the whole reason we turn to machine learning rather than handcrafted decision rules is that for many problems, simple, easily understood decision theory is insufficient’ Lipton (2015).

[17] One social scientist doing field work among researchers developing a self-driving car found that these researchers avoid using machine learning entirely because ‘you don’t know what it learns.’ The innumerable situations not represented in the training set lead to unpredictable and possibly potentially life-threatening consequences (Both, 2014). In personal conversations with the author, researchers at Yahoo! and Fair Isaacs Corporation (the source for FICO scores) have also described an avoidance of machine learning algorithms for this reason. In the credit market this isn’t just a preference, but enforced through the Fair Credit Reporting Act which requires that reasons be provided to consumers for denied credit. ‘Alternative credit scoring’ or ‘consumer scoring’ agencies, however, do freely use ML models and are not (yet) subject to these regulations. For details see this tutorial by Cathy O’Neill on ‘Doing Data Science’: http://bclt.me/audio/Intro%20and%20Keynote.mp3

[18] See for example the many different groups who have experienced problems with Facebook’s ‘real names’ policy and mechanism for reporting and verification—https://www.eff.org/deeplinks/2015/02/facebooks-name-policy-strikes-again-time-native-americans

References

Barocas S (2014a) Data mining and the discourse on discrimination. In: *Proceedings of the Data Ethics Workshop, Conference on Knowledge Discovery and Data Mining (KDD)*, 24–27 August, New York City.

Barocas S (2014b) *Panic Inducing: Data Mining, Fairness, and Privacy,* PhD Thesis, New York University, USA.

Barocas S and Selbst A (forthcoming) Big Data’s disparate impact. *California Law Review*.

Both G (2014) What drives research in self-driving cars? (Part 2: Surprisingly not machine learning). Available at: http://blog.castac.org/2014/04/what-drives-research-in-self-driving-cars-part-2-surprisingly-not-machine-learning/.

Bowker GC and Star SL (1999) *Sorting Things Out: Classification and Its Consequences*. Cambridge, MA: The MIT Press.

Brunton F (2013) *Spam*. Cambridge, MA: The MIT Press.

Burrell J (2012) *Invisible Users: Youth in the Internet Cafes of Urban Ghana*. Cambridge, MA: The MIT Press.

Datta A, Tschantz MC and Datta A (2015) Automated experiments on ad privacy settings: A tale of opacity, choice, and discrimination. In: *Proceedings on Privacy Enhancing Technologies*, 30 June–2 July, Philadelphia, PA.

Diakopoulos N (2013) *Algorithmic Accountability Reporting: On the Investigation of Black Boxes*. Report, Tow Center for Digital Journalism, Columbia University.

Domingos P (2012) A few useful things to know about machine learning. *Communications of the ACM* 55(10): 78.

Dwork C, Hardt M, Pitassi T, et al. (2012) Fairness through awareness. In: *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference,* 8–10 January, Cambridge, MA, pp. 214–226.

Ensmenger NL (2003) Letting the “computer boys” take over: Technology and the politics of organizational transformation. *International Review of Social History* 48(S11): 153–180.

Eubanks V (2012) *Digital Dead End: Fighting for Social Justice in the Information Age*. Cambridge, MA: The MIT Press.

Fourcade M and Healy K (2013) Accounting, organizations and society classification situations: Life-chances in the neoliberal era. *Accounting, Organizations and Society* 38(8): 559–572.

Gandy OH (2010) Engaging rational discrimination: Exploring reasons for placing regulatory constraints on decision support systems. *Ethics and Information Technology* 12(1): 29–42.

Gillespie T (2012) The relevance of algorithms. In: Gillespie T, Boczkowski P and Foot K (eds) *Media Technologies: Essays on Communication, Materiality, and Society*. Cambridge, MA: The MIT Press.

Grudin J (2006) Turing maturing: The separation of artificial intelligence and human–computer interaction. *Interactions* 13(5): 54–57.

Hargittai E (2008) The digital reproduction of inequality. In: Gursky D (ed.) *Social Stratification*. Boulder, CO: Westview Press, pp. 936–944.

Khosla (2012) Will we need teachers or algorithms? In: *TechCrunch*. Available at: http://techcrunch.com/2012/01/15/teachers-or-algorithms/ (accessed 11 December 2015).

Lee I, Martin F, Denner J, et al. (2011) Computational thinking for youth in practice. *ACM Inroads* 2(1): 32–37.

Lipton Z (2015) The myth of model interpretability. Available at: http://www.kdnuggets.com/2015/04/model-interpretability-neural-networks-deep-learning.html (accessed 11 December 2015).

Mateas M and Montfort N (2005) A box, darkly: Obfuscation, weird languages, and code aesthetics. In: *Proceedings of the 6th Annual Digital Arts and Culture Conference*, 1–3 December, Copenhagen, Denmark..

Olazaran M (1996) A sociological study of the official history of the perceptrons controversy. *Social Studies of Science* 26(3): 611–659.

Pasquale F (2015) *The Black Box Society: The Secret Algorithms that Control Money and Information*. Cambridge, MA: Harvard University Press.

Sandvig C (2014) Seeing the sort: The aesthetic and industrial defense of ‘‘the algorithm’’. *Journal of the New Media Caucus* 10(3): 1–21.

Sandvig C, Hamilton K, Karahalios K, et al. (2014) Auditing algorithms: Research methods for detecting discrimination on internet platforms. In: *Annual Meeting of the International Communication Association,* Seattle, WA, pp. 1–23.

Seaver N (2014) Knowing algorithms. Presented at Media in Transition 8, Cambridge, MA.

Straka JW (2000) A shift in the mortgage landscape: The 1990s move to automated credit evaluations. *Journal of Housing Research* 11(2): 207–232.

Tufekci Z (2014) The year we get creeped out by the algorithms. Available at: http://www.niemanlab.org/2014/12/the-year-we-get-creeped-out-by-algorithms/ (accessed 17 June 2015).

Wing JM (2006) Computational thinking. *Communications of the ACM* 49(3): 33–35.

Winograd T (2006) Shifting viewpoints: Artificial intelligence and human–computer interaction. *Artificial Intelligence* 170(18): 1256–1258.

Appendix 1

Spam email from spam folder in the author’s gmail account:

> My Dearest,
>
> Greetings to you my Dear Beloved, I am Mrs Alice Walton, a citizen of United State. I bring to you a proposal worth $ 1,000,000,000.00 which I intend to use for CHARITY but I am so scared because it is hard to find a trust worthy human on earth. I am happy to know you, but God knows you better and he knows why he has directed me to you at this point in time so do not be afraid. I know there are lots of fraud out there sending messages like these or in other form. I saw your email contact at the ministries of commerce and foreign trade departments.
>
> I am writing this mail to you with heavy sorrow in my heart,
>
> It is painful now to let you know that I have been suffering from a Heart disease for the past 22 years and just few weeks ago my Doctor told me that I won’t survive the illness for long.
>
> I am contacting you as I was touched to open up to you about my project.
>
> Please reply me back if you are interested, but if not please ignore this message.
>
> God Bless You.
>
> Please reply me back if you are interested, so I can provide you with further details.
>
> Email: alice.walton2@yandex.com

Non-spam email from Ghanaian researcher and friend:

> Dear prof. Thank you for continually restoring hope and bringing live back to me when all hopes seem to be lost. With tears and profound gratitude I say thank you. I have delayed in responding because I didn’t want to tell you am still not better. Am responding to treatment even though am not that well. Am able to get a big generator, air-conditioned, a used professional Panasonic 3ccd video camera and still have about 150 dollars in my account for taking care of my health. Also I have changed my phone from a problematic old Nokia to an h 6 tecno [advanced China phone]. The doctors say I have malaria parasites. Only God knows when I will be very okay. I can’t imagine how life would have been without you. I pray you continually prosper. Much regards and Bye.