FACILITATING THE SPREAD OF KNOWLEDGE AND INNOVATION IN PROFESSIONAL SOFTWARE DEVELOPMENT

# REAL-WORLD MACHINE LEARNING

## CASE STUDIES, TECHNIQUES AND RISKS

eMag Issue 62 - Jul 2018


InfoQ queue

# IN THIS ISSUE




## FOLLOW US

facebook.com
/InfoQ

@InfoQ

linkedin.com
company/infoq


youtube.com
/MarakanaTechTV

## CONTACT US

GENERAL FEEDBACK feedback@infoq.com
ADVERTISING sales@infoq.com
EDITORIAL editors@infoq.com

# A LETTER FROM THE EDITOR

## Srini Penchikala

Machine learning (ML) and deep learning technologies like Apache Spark, Flink, Microsoft CNTK, TensorFlow, and Caffe brought data analytics to the developer community. Whether it's classifying two million sales products received from over 700 multinational retailers at "Love the Sales" organization, building awareness of hindsight bias with customers on Salesforce's Einstein website, or understanding software system behavior with ML and time series data at SumoLogic, machine learning solutions are driving the competitive edge in various companies and industries.

This eMag focuses on the current landscape of ML technologies and presents several associated real-world case studies.

It showcases articles and interviews covering a diverse set of topics, including:

- Using Support Vector Machines (SVMs) algorithm as an effective tool for document classification
- Analyzing & preventing unconscious bias in machine learning
- Exploring how the city of New York City established a task force for obtaining explanation and mitigations for people affected by the use of machine learning algorithms by city agencies

All the publications included in this eMag are authored by practitioners and subject matter experts working in the field of ML. We hope that you agree with us in that these articles are valuable resources to reference, and that the techniques can be used in your own projects and initiatives within your organization.

As the quote from Einstein goes, "education is not the learning of facts, but the training of the mind to think". We at InfoQ hope this magazine helps you to get up to speed with the real-world case studies of how ML is being used by different companies, and also act as a catalyst in finding even more new and innovative use cases to apply ML techniques and algorithms.

Thank you for checking out this InfoQ Machine Learning eMag.

# CONTRIBUTORS

## Srini Penchikala

Srini Penchikala currently works as a senior software architect in Austin, Tex. Penchikala has over 22 years of experience in software architecture, design, and development. He is also the lead editor for AI, ML & Data Engineering community at InfoQ, which recently published his mini-book Big Data Processing with Apache Spark. He has published articles on software architecture, securiaty, risk management, NoSQL, and big data at websites like InfoQ, TheServerSide, the O'Reilly Network (OnJava), DevX's Java Zone, Java.net, and JavaWorld.

### David Bishop

After studying computer science in New Zealand, David Bishop moved to London and led the technical team at top-100 UK employment website reed.co.uk for 10 years. He has gone on to found his own retail technology business, Love the Sales, which seeks to aggregate all of the sales from thousands of retail websites.

### Mayukh Bhaowal

is a director of product management at Salesforce Einstein, working on automated machine learning and data science. Mayukh received his master's in computer science from Stanford University. Prior to Salesforce, Mayukh worked at startups in the domain of machine learning and analytics. He served as head of product at Scaled Inference, a machine-learning-platform startup backed by Khosla Ventures, and led product at an e-commerce startup, Narvar, backed by Accel. He was also a principal product manager at Yahoo and Oracle.

## Roland Meertens

is a computer-vision engineer working in artificial intelligence for the perception side of self-driving vehicles at the Audi subsidiary Autonomous Intelligent Driving. he has worked on interesting things like neural machine translation, obstacle avoidance for small drones, and a social robot for elderly people. Besides putting news about machine learning on InfoQ, he sometimes publishes on his blog Pinch of Intelligence and Twitter. In his spare time, he likes to run through the woods and participate in obstacle runs.

## Michael Stiefe

principal of Reliable Software, Inc. is a consultant on software architecture and development, and the alignment of information technology with business goals. He was a Lecturer in the Aeronautics and Astronautics Department at the Massachusetts Institute of Technology where his research and teaching focus was understanding how people build mental models in order to solve problems. As Adjunct faculty, Stiefel has taught graduate and undergraduate software engineering courses at Northeastern University and Framingham State University.
He explores his interest in the intersection between technology and art in the blog Art and Software.

Read online on InfoQ

## KEY TAKEAWAYS

SVMs are effective tools for classifying documents.

By reducing the size of large data sets/vectors, we make it easier to train your models.

By reusing labelled data via linked relationships, we reduce the cost of training data and increase the accuracy of predictions.

Choosing the right data structures is important in achieving the best results.

Flattening out the hierarchy of data can be helpful in reducing the number of SVMs.

# GET MORE BYTES FOR YOUR BUCK

by **David Bishop**

In many cases, the acquisition of well-labelled training data is a huge hurdle for developing accurate prediction systems with supervised learning.

At Love the Sales, we aggregate sales products from over 700 multinational retailers, which results in over 2 million products a day that need classification. It could take a traditional merchandising team four years to complete this task manually.

Our requirement was to apply this classification to the textual meta-data of these 2 million products (mostly fashion and homewares) into more than a thousand categories represented in a hierarchy such as:

Mens Clothing
- Mens Jeans
- Mens Jumpers

Womens Clothing
- Womens Jeans
- Womens Jumpers
- ...

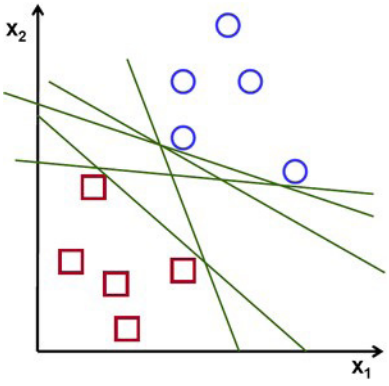
## Support vector machines

For our classification task, we decided upon support-vector machines (SVMs). SVMs are a class of supervised machine-learning algorithm, proficient in the classification of linearly separable data.

Given a large enough set of labelled training data, an SVM essentially will attempt to learn a best discriminative plane between examples: to try to draw a multidimensional line in the sand.

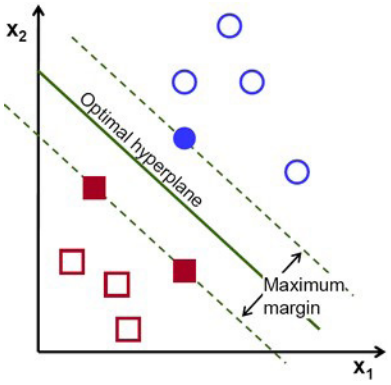
Find more on SVMs from this lecture and at OpenCV.

For example, the green lines in this graph represent possible ways to separate this dataset into blue circles and red squares:



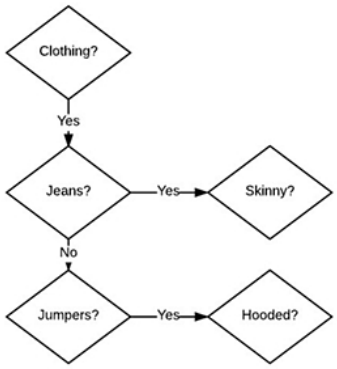

Images from opencv.org.

The SVM will attempt to learn the optimal hyperplane:




Whilst there are many machine-learning algorithms for classification (for example, neural networks, random forest, and naive Bayesian), SVMs really shine for data with many features — in our case to classify documents, wherein each word is treated as a discrete feature.

Whilst SVMs can actually classify into multiple classes, we opted to use a hierarchy of simple two-class SVMs chained together in a hierarchical fashion.




The main reason for this is that it seemed to yield better results and, importantly, it used much less memory on our machine-learning platform because each SVM only had to recognize two classes of data. High memory utilisation for large data sets (300K+ examples) and large input vectors (a million different known words) was a definite stumbling block for us.

Some simple well-known techniques for pre-processing our documents also helped a great

**A key learning for us, is that the way in which these SVM's are structured can actually have a significant impact on how much training data has to be applied**

deal in reducing the feature space, such as lowercasing, stemming, stripping odd characters, and removing noisy words and numbers.

Stemming is a common language-specific technique useful when dealing with large corpuses of textual data that takes different words with a similar meaning and root and crunches them down to similar tokens. For example, the words "clothing" and "clothes" have a very similar meaning, and when they are stemmed through the common Porter stemming algorithm, the result is "cloth". In doing this, we halved the number of words we had to worry about. Using stemming in conjunction with the elimination of noisy words (the removal of common words without domain meaning like "the", "is", "and", etc.) greatly reduced the number of words we had to deal with.

## Creating SVMs

Once we have pre-processed your textual data, the next step is to train our model. In order to do this, we must first transform your textual data into a format the SVM can understand — this is known as "vectorization". A quick, simple description of this process is as follows.

Take the sentence "Men, you'll look fantastic in this great pair of men's skinny jeans."

After pre-processing it with stemming and removing words that we don't care about, that sentence becomes: "men fantastic great pair men skinny jean".

We could encode the data in alphabetical order like so:

| Occurrences | Term |
|---|---|
| 1 | fantastic |
| 1 | great |
| 1 | jean |
| 2 | men |
| 1 | pair |
| 1 | skinny |

This can be represented in a vector such as: `[1,1,1,2,1,1]`.

This is fine for a small set of terms (only one short sample in this case). However, as we add more samples and more terms our vocabulary increases — for example, if we add another training sample that isn't men's skinny jeans: "women bootcut acid wash jean".

We now need to increase the algorithm's vocabulary to `[acid,bootcut,fantastic,great,jean,men,pair,skinny,wash,women]`.

After creating a new alphabetical vocabulary table, the new term vector for the initial men's skinny jeans example changes to: `[0,0,1,1,1,2,1,1,0,0]`.

When dealing with thousands of samples, the vocabulary can become large, and as a result can become quite cumbersome as the encoded training samples become mostly empty and very long: `[0,0,0,0,0,0,0,0,.....,2,0,0,0,0,0,.....1,0,0,0,0,…]`.

Thankfully, many machine-learning libraries allow us to encode our term vectors as sparse vectors, which means we only need to supply non-zero cases and the library (in our case, LIBSVM) will magically figure it out for us and fill in the gaps.

In such a case, we provide the term vectors and the classes they represent as term indices relative to the entire vocabulary for all the training samples we wish you use, for example:

| Term Index | Term |
|---|---|
| 0 | acid |
| 1 | bootcut |
| 2 | fantastic |
| 3 | great |
| 4 | jean |
| 5 | men |
| 6 | pair |
| 7 | skinny |
| 8 | wash |
| 9 | women |

So, we would describe "men fantastic great pair men skinny jean" as:

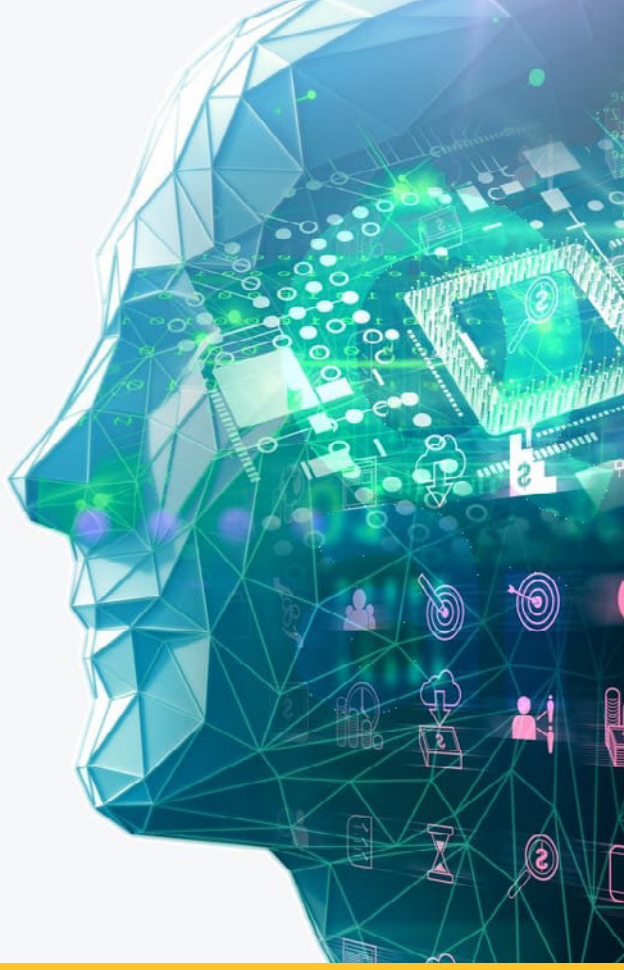
```
Term Index #2 : 1 Occurrence
Term Index #3 : 1 Occurrence
Term Index #4 : 1 Occurrence
Term Index #5 : 2 Occurrences
Term Index #6 : 1 Occurrence
Term Index #7 : 1 Occurrence
```

This can then be encoded succinctly as `[2:1,3:1,4:1,5:2,6:1,7:1]`.

Alexandre Kowalczyk has a great description of vocabulary preparation here, along with other great SVM tutorials.

QCon.ai by InfoQ

**April 15-17, 2019**
San Francisco



# APPLIED AI AND ML CONFERENCE

## for software engineers

*rather than data scientists

## Why should I attend?

Learn how software innovators are applying AI & machine learning in **use-case oriented sessions**

Hear about the **tools and techniques** of AI & machine learning

Go in-depth on key topics in our **1 day of workshops**

**Meet with AI and ML leaders** from innovator and early adopter companies

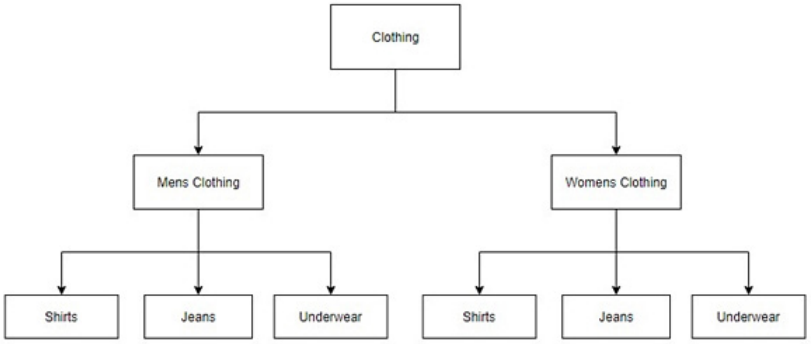
**Gain valuable insights and ideas** to shape your AI and machine learning projects

Learn how innovator companies are **applying AI & machine learning in their businesses**

# Find out more!

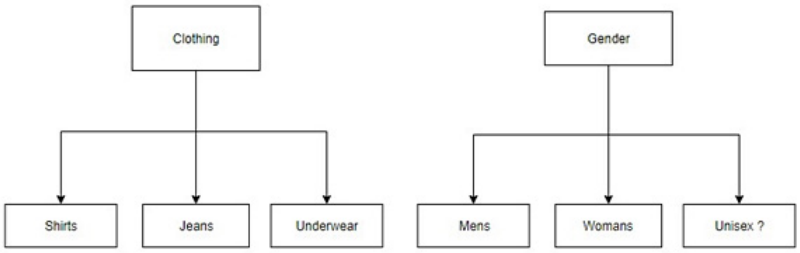
## Hierarchy and data structures

A key lesson is that the way in which these SVMs are structured can significantly affect how much training data we have to apply. For example, a naive approach would be as follows:




This approach requires training two new SVMs for every additional sub-category — for example, the addition of a new class for Swimwear would require an additional SVM under Men's Clothing and Women's Clothing — not to mention the potential complexity of adding a new Unisex class at the top level. Overall, deep hierarchical structures can be too rigid to work with.

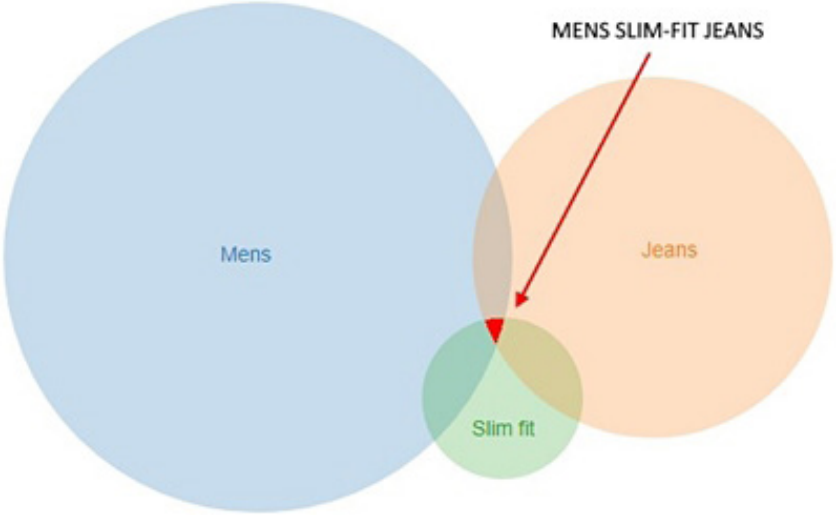
We were able to avoid a great deal of labelling and training work by flattening our data structures into many sub-trees like so:



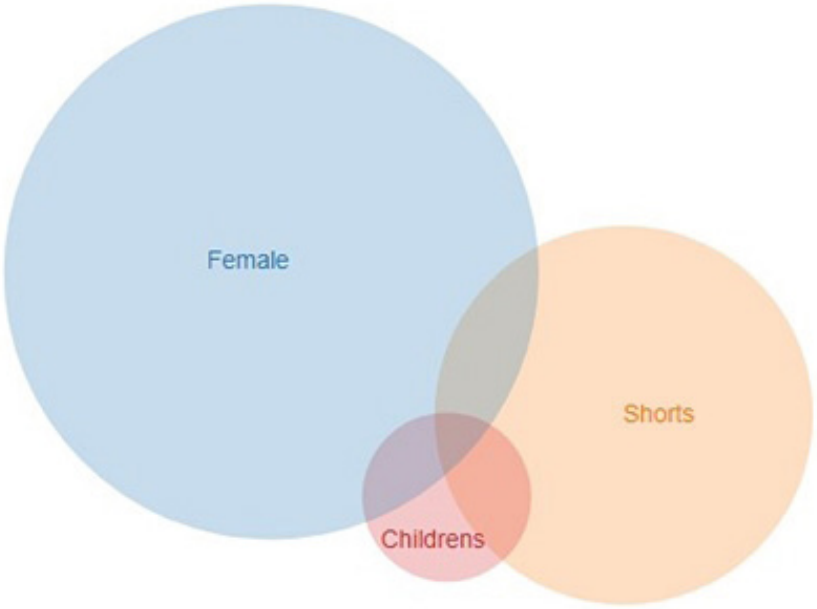

Decoupling our classification structure from the final hierarchy made it possible to generate the final classification by traversing the SVM hierarchy with each document and interrogating the results with simple set-based logic such as:

```
Mens Slim-fit jeans = (Mens and Jeans and Slim Fit) and not Womens
```

This approach vastly reduces the number of SVMs required to classify documents, as the resultant sets can intersect to represent the final classification.



It should also now be evident that adding new classes opens up an exponentially increasing number of final categories. For example, adding a top-level Children's class would immediately allow the creation of an entire dimension of new Children's categories (children's jeans, shirts, underwear, etc.) with minimal additional training data (only one additional SVM):




## Data reuse

Because of the structure we chose, we were able to reuse training data via linked data relationships. Linking data allowed us to reuse our training data nine times, thus massively reducing the cost and increasing the accuracy of predictions.

For each individual class, we obviously want as many training-data examples as possible, covering both possible outcomes. Even though we built some excellent internal tooling — primarily a nice, fast user interface for searching, sorting, and labelling training data in large batches — labelling thousands of examples of each kind of product can still be laborious, costly, and prone to error. We determined that the best way to circumvent these issues was to attempt to reuse as much training data as we could across classes.

For example, given some basic domain knowledge of the categories, we know for certain that washing machines can never be carpet cleaners.


By adding the ability to link "Excluded" data, we can heavily bolster the amount of negative training examples for the "Washing machines" SVM by adding to it the positive training data from the "Car-

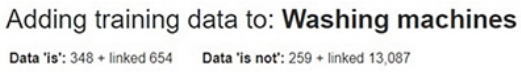
pet cleaners" SVM. Put more simply, given that we know carpet cleaners can never be washing machines, we may as well reuse that training data.

This approach has a nice advantage in that whenever we add additional training data to improve the "Carpet cleaners" SVM, we also inadvertently improve the "Washing machines" class via linked negative data.

Finally, another advantage of reuse when operating in a hierarchy is that the positive training data for any child node is also always positive training data for its parent.

For example, jeans are *always* clothing.


This means that for every positive example of training data added to the "Jeans" SVM, an additional positive example is also added to the "Clothing" SVM via the link. Adding linked data is far more efficient than manually labelling thousands of examples.




## Conclusion

SVMs have helped us to reach a quality and speed of classification that we could have never achieved without a machine-learning approach. We have come to learn that SVMs are an excellent addition to any developer's toolbox and that any investigation should also serve as a nice introduction to some key machine-learning concepts.



Additionally, when it comes to the specifics of hierarchical classification systems, decoupling the classification component from the resulting hierarchy, flattening the data structure, and enabling the reuse of training data will all benefit the process and gain as much efficiency as possible. The approaches outlined above have not only helped reduce the amount of training data we needed to label, they also have given us greater flexibility overall.

Read online on InfoQ

## KEY TAKEAWAYS

Bias in data has created a bottleneck in enterprise artificial intelligence that cannot be solved by excessively optimizing machine-learning algorithms or inventing new ones.

Hindsight bias is the accidental presence of information in the training data that will never legitimately be available in production. In lay terms, it is like Marty McFly (from Back to the Future) retrieving the sports almanac from the future and using it to bet on the games of the present.

There is no silver bullet that solves it. A combination of statistical methods and feature engineering can help to detect and fix it.

Features that exhibit such bias need to be distinguished from true predictors and determining the right threshold is key.

# BACK TO THE FUTURE: DEMYSTIFYING HINDSIGHT BIAS

by **Mayukh Bhaowal**

Once upon a time, there was a sales executive who tracked incoming sales leads by entering the minimal data needed to insert a lead record. He thought, as we all do, that data entry is a pain. As he converted the leads, some of them turned into purchases. At the time of conversion, he filled in additional information, a label of positive outcome, only for those leads that became purchases.

If you, like that executive, train your machine-learning algorithm with years of labeled data, it will correlate those features with a positive label, though they would never really be available before the conversion. The business process builds bias into the data from the ground up.

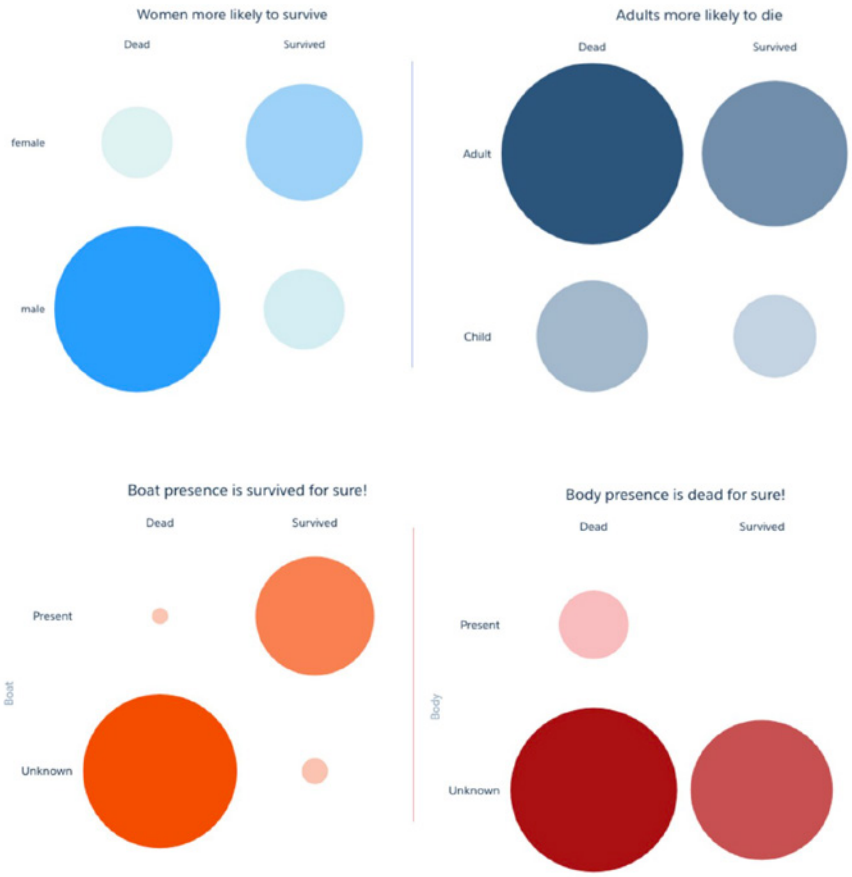
This story repeats itself across different enterprise use cases, users, and data. Machine-learning algorithms often assume that they have consumed some mythical perfect dataset to predict the target label. In reality, there is often a lot of noise in the data. The Achilles' heel in this domain is hindsight bias (also known as label leakage or data leakage). It is the accidental presence of information in the training data that will never legitimately be available in production, causing unrealistic results in the research environment and poor results in the production environment.

Albert Einstein said, "If I had an hour to solve a problem, I'd spend 55 minutes thinking about the problem and five minutes thinking about solutions." So, let's look at the problem a bit more, with an example.

## Demystifying hindsight bias with *Titanic*

In the machine-learning community, the Titanic passenger-survivability trope is well known. The lack of lifeboats was responsible for many lost lives. Specific groups of passengers such as women, children, and the upper class were more likely to survive than others. Machine learning is used to learn such signals and predict which passengers survived the tragedy.

What many do not know is that the Kaggle challenge based on the *Titanic* data uses a filtered, cleaned-up version of the dataset. The original data had additional features, two of which were particularly problematic: "boat" and "body" fields. Passengers were assigned a boat number if they safely made it to a lifeboat or a body number if they were eventually found dead.

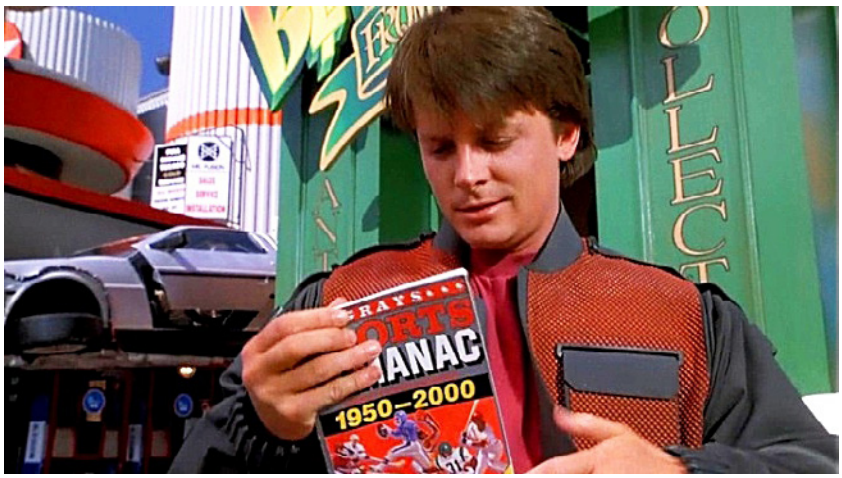



If an entry has a body number, the passenger is dead. You do not need a fancy machine-learning algorithm to tell you that.

When using the original, unaltered dataset, information about the target label crept into the training data. A passenger's boat or body number is only known in the future, after the event has occurred. They are not known in the present, when making the prediction. If we train the model with such data, it will perform poorly in the present, as that piece of information would not legitimately be available.

This problem is known formally as hindsight bias and it is predominant in real-world data, which we've witnessed

Cabin class B/C more likely to survive




first-hand while building predictive applications at Salesforce Einstein. Here is an actual example in the context of predicting sales-lead conversion: the data had a field called "deal value" that was populated intermittently when a lead was converted or was close to being converted (similar to the boat and body fields in the *Titanic* story).

In lay terms, it is like Marty McFly (from *Back to the Future*) retrieving the sports almanac from the future and using it to bet on the games of the present. Since time travel is still a few years away, hindsight bias is a serious problem today.

## Hindsight bias versus modeling algorithm

Machine-learning algorithms take center stage in current AI applications. There is a race to improve model accuracy a fraction of a percentage by optimizing modeling algorithms or inventing new ones. While that is useful, we can achieve a bigger bang for our buck by focusing on the bottlenecks for applied machine learning, specifically with enterprise data. Hindsight bias is one such area mostly unexplored. So, how can we address this problem?

## Mitigation strategies

**1. Statistical analysis of input features**

There is a suite of statistical tests that we can run on the input features to detect strong association between the features and the target label. Pearson correlation provides a numeric measure in the range (-1,1) that expresses the strength and direction of the association between the feature and the label. It works great for numeric features and it can work for categorical features as well once they are vectorized. However, if categorical features have a large number of unique values (e.g., cities in the world), correlation misses association with labels due to dilution of the feature across several columns during vectorization. This can be addressed by Cramer's V, making it a more preferred statistical test for categorical features.

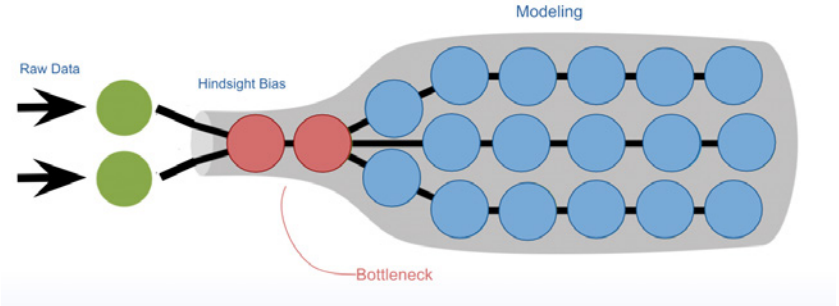
The impact of such biased features might be trickier when it affects a small fraction of the examples. Imagine a set of global geographic data: the portion of rows in which City = San Francisco might be one in a thousand. Lift is an alternative measure that catches such sparse hindsight bias.

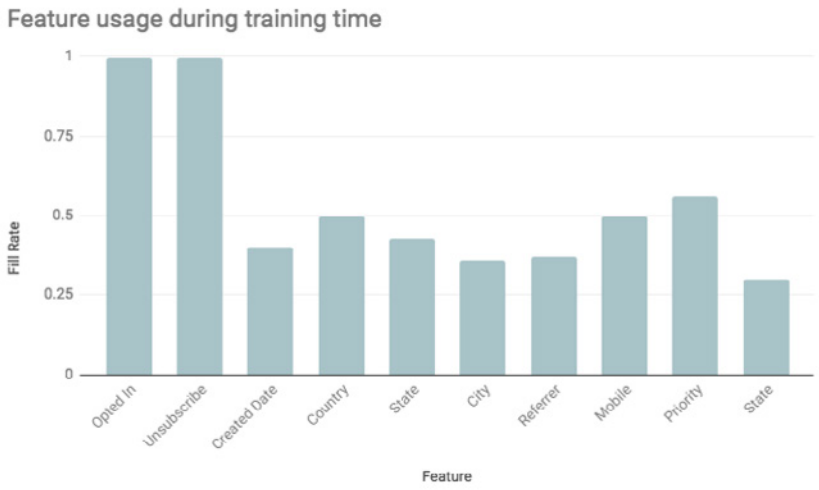
**2. Statistical analysis of derived features**

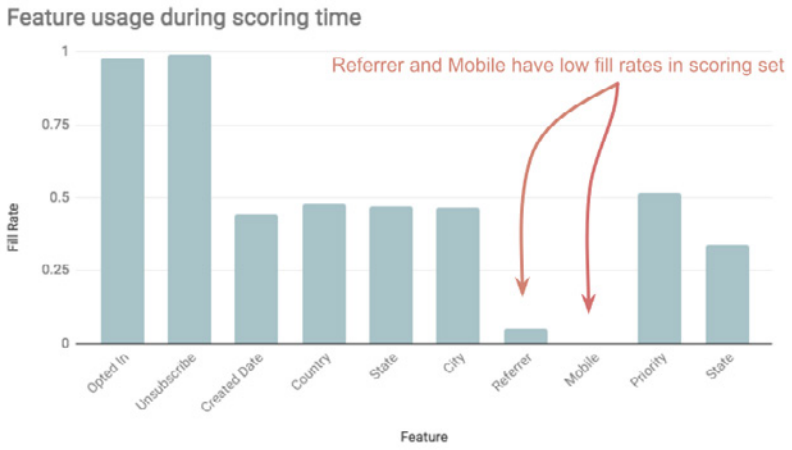
One useful strategy is to perform some preliminary feature engineering before running statistical tests on the input features.

For instance, many categorical features with hindsight bias follow the pattern of being null until the target label is determined. They tend to have some value filled in close to when the label is specified. The boat and body fields from the *Titanic* data are examples of this pattern. The way to bust them is to add a null-indicator (isNull) derived feature and use Cramer's V as a statistical test.

Correlation does not always catch numeric features with hindsight bias. For example, we used a feature called "expected revenue" to predict whether a sales opportunity would be won or lost. The system filled in the value after






the salesperson closed the opportunity. When the salesperson lost the opportunity, the system calculated expected revenue as 0 or 1. Otherwise, the system calculated it as a large number. A decision tree can be used to discover the two bins: [0,1] and [2, infinity]. Once we bin a numeric feature, we can treat it as a categorical feature. A statistical test like Cramer's V can then reveal the strong association between the specific bin and the label, thus exposing the bias.

The other noteworthy pattern that we observed was categorical features disguised as text. For instance, we used a feature called "lost in stage" while predicting whether a deal would be lost or won. Clearly heavily biased, it was defined as a text feature with only three possible values. A cardinality check on such features, converting them to categorical features, and then trying statistical tests like Cramer's V can reveal hindsight bias.

**3. Training versus scoring distribution**

Sometimes, the most elusive hindsight bias may not get exposed to the techniques above by only looking at the training data. One of the principal assumptions behind training a machine-learning algorithm is that the data used for training is similar to the data used for scoring.

Since features with hindsight bias contain information about the label at the time or right before the actual label is determined, we can look at the distribution of features in training data and the scoring data (before knowing the actual label). If any of the features indicate a statistically significant gap in the two distributions, that is a candidate for hindsight bias.

Temporal or timestamp cutoff is a related technique. Here, we determine a cutoff timestamp as the moment when the predicted event is supposed to occur, based on current and past records. We then exclude all data before the event of interest, so we are not using any data that we collected close to the prediction or after, i.e., in the future.

**4. Cross-validation folds and data preparation**

It is crucial to perform all data preparation and feature engineering within each cross-validation fold. As an example, if we use the label info in any feature-engineering step such as binning, we inherently introduce hindsight bias to the data. The same applies for feature selection, outlier removal, encoding, and feature-scaling methods for dimensionality reduction. If we perform any of these on the entire data before cross-validation, then the test data in each fold of the cross-validation procedure

played a role in choosing the features, and this introduces hindsight bias in the data.

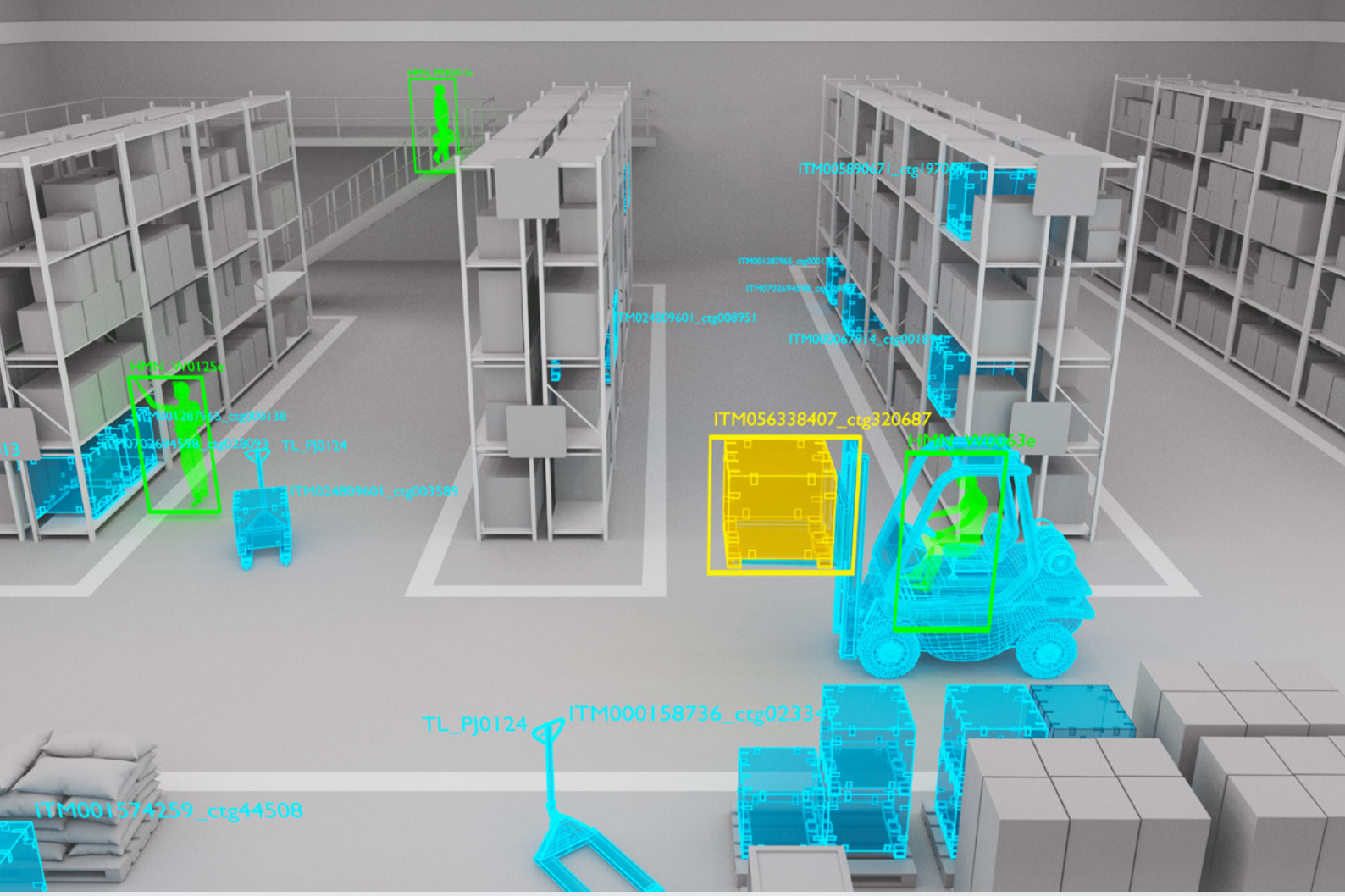
## Hindsight bias or true predictor?

Across all the methods discussed so far, the hardest aspect is discovering the right threshold for our data and use case, which would help reveal hindsight bias. What should be the correlation measure, beyond which we mark a feature as biased? Is 0.9 a good threshold or should it be 0.75? At what point do we say a feature is biased versus actually being a true predictor? We need to make the same decision on all other statistical measures, including the difference in the training and scoring distribution and so forth.

At Salesforce Einstein, our experience in building models for a wide variety of use cases and data of different shapes and sizes helps inform acceptable thresholds. However, it is far from set in stone. We are continuously iterating on the thresholds to reflect the real-world data and problems.

## Conclusion

Hindsight bias in enterprise AI is a more prevalent problem than in consumer AI or academia. The most significant challenge we faced was building awareness of it with our customers. Once we got past that, understanding the business processes and data patterns that introduce such bias was crucial. This journey helped us develop solutions that automated hindsight bias detection and mitigation. The result was more reliable machine-learning predictions.




Read online on InfoQ

## KEY TAKEAWAYS

Before diving into the machine learning of software system behaviour, you have to know the traditional approaches to time series.

Missing values in your time series can lead to unexpected results when analysing them. The pandas library can help you to fill missing values in your data in a sensible matter.

When humans are using your service, you should expect seasonality in your data. Take this into account when designing your predictive algorithms.

Be careful of what threshold you set for anomaly detection. Events that are unlikely for a single server become very likely when scaling your application.

# SOFTWARE SYSTEM BEHAVIOUR WITH MACHINE LEARNING AND TIME-SERIES DATA

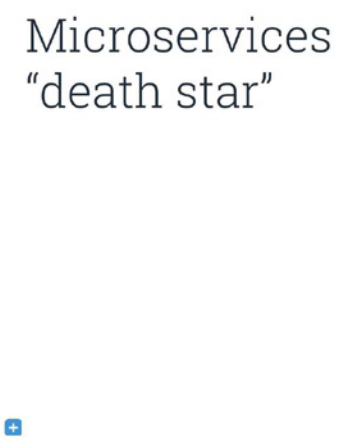
by **Roland Meertens**

At QCon.ai 2018, David Andrzejewski presented "Understanding Software System Behaviour With ML and Time Series Data". David is an engineering manager at Sumo Logic, a cloud-based platform for machine-data analysis. Developers who are already running a software system (like an app or cloud cluster) can use Sumo Logic as the back end for their logging. Sumo Logic provides continuous intelligence for this machine data.

A lot of things run on software, and artificial-intelligence techniques are moving into the software world. Before diving deep into machine learning's impact on software system behaviour, you have to understand the traditional approaches to time series. Knowing the limitations of traditional methods lets you make informed trade-offs in choosing techniques. First, ask yourself if you know what you are trying to accomplish. Once you know, try to ask yourself if you can accomplish this with simple or deterministic analysis. Only look at machine learning when other methods are impossible.

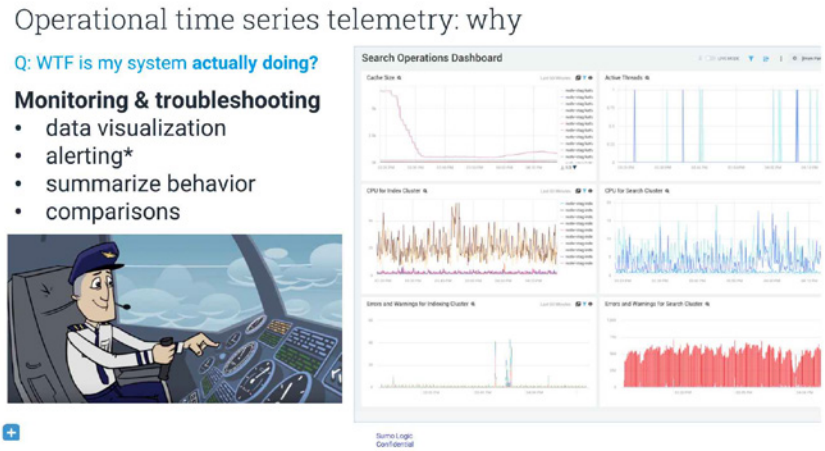
Understanding what your software is doing and why it is failing can be difficult. Companies that deploy services that rely on many microservices on multiple hosts can benefit from a diagram that lists dependencies among those microservices. When drawing this out, you might get an image that people refer to as the microservices death star.

Many applications generate terabytes of logs per day, consist of gigabytes of source code, and output millions of metrics per minute. Analysing this data by hand is impossible, so you need machine intelligence. However, analysing the data to find out only what your system is REALLY doing is a difficult if not impossible task. An interesting paper that goes deeper into the granularity of data and at what level you need it, is "Could a neuroscientist understand a microprocessor?" The authors of this paper use a simulator to play the old game of Donkey Kong. Because they own the memory of the simulation, they have access to the complete state of the system. Theoretically, this means it is possible to analyse the data to try to reverse-engineer what is going on at a higher level of understanding, just from looking at the underlying data. Although this tactic can provide small insights, it is unlikely that only looking at the data will allow you to completely understand the higher level of Donkey Kong.



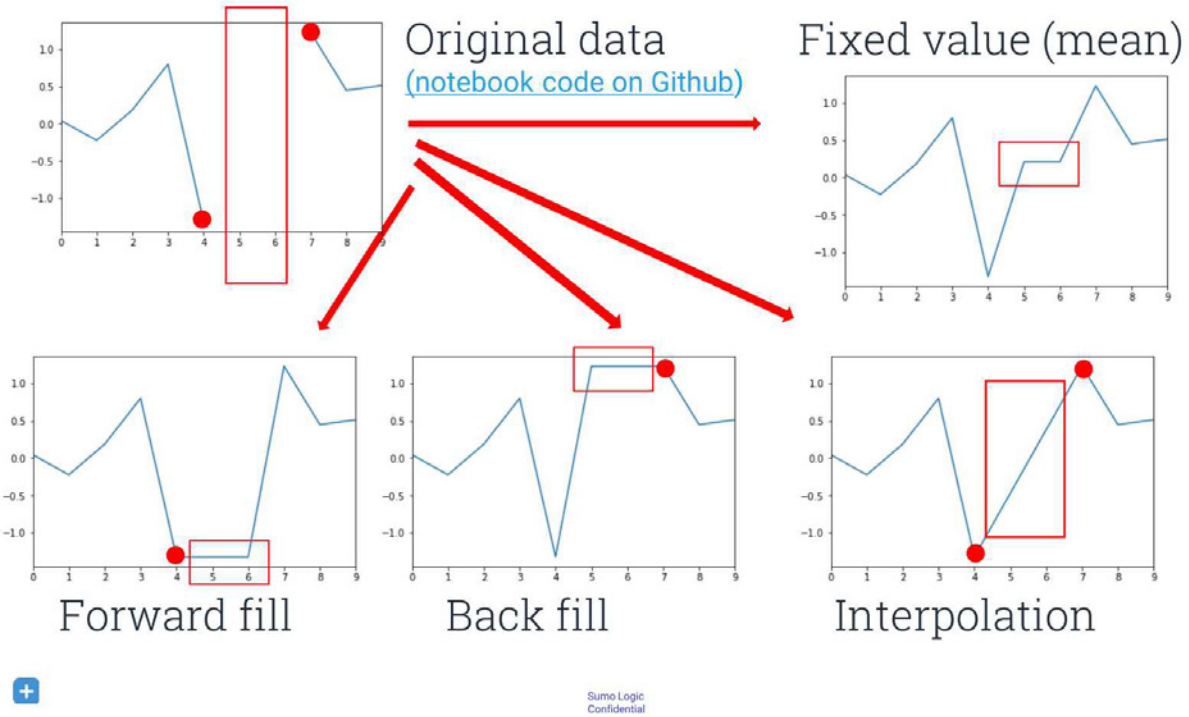

This analogy becomes really important when you are using only raw data to understand complex, dynamic, multiscale systems. Aggregating the raw data into a time-series view makes the problem more approachable. A good resource for this is the book *Site Reliability Engineering,* which can be read online for free.

Understanding complex, dynamic, multiscale systems is especially important for an engineer who is on call. When a system goes down, he or she has to go in to discover what the system is actually doing. For this, the engineer needs both the raw data and the means to visualise it, as well as higher-level metrics that are able to summarise the data. An engineer in this situation often wants to know how this server is behaving compared to another server, to itself yesterday, or to itself before that one software update.




## Upsides and downsides of percentiles

When looking at a long history of log data, you don't go into continuous millisecond detail. You quantize your data in time. The most basic ways to do this is with functions such as min, max, average, sum, and count. Many people who aggregate data like to use percentiles as well. The advantage of percentiles is that they can express your data in an unambiguous language. An ex-




ample of a sentence without a percentile is "The maximum time to load a request was 4,300 milliseconds." This sentence is precise but does not help to determine how far outside the bounds of normal operation it falls. However, saying that "p99 is less than 2,000 milliseconds" indicates that no more than 1% of customer requests take longer than two seconds to load.

The downside of percentiles is that they make it difficult to combine data into something meaningful. Although values around the 50th percentile tend to be stable, higher percentiles will vary a lot and have a long-tailed distribution of possible values. Another problem is that it is easy to aggregate the simple analyses of several datasets. You can calculate the minimum of two datasets by looking only at the minima of both. However, you can't as simply use the methods with percentiles. It is mathematically impossible to combine the p95 of dataset X and the p95 of dataset Y. This means that it is difficult to say something meaningful about a combination of multiple datasets without further work.

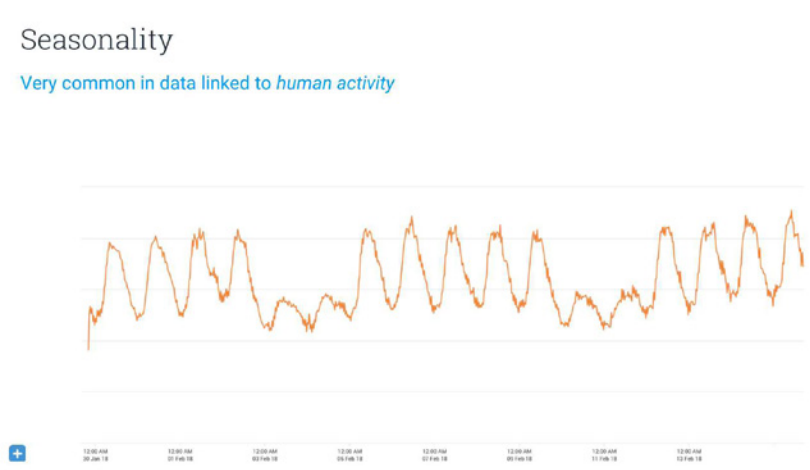
## Important time-series concepts

A basic monitoring aspect for time series is time-shifted comparisons. This is particularly important if you want to compare the write latency of one cluster to the write latency of the same host the day before. This can also be combined with windowing data, known as "grouping over time". More information can be found in Tyler Akidau's QCon San Francisco 2016 talk, where he discussed this concept in the context of Apache Beam.

Handling missing data is also important. Before you can apply any machine learning, you have to know how you want to handle missing values. Putting constant values, like zeros or infinities, in place of missing values probably leads to unexpected results. However, not putting anything there will likely yield runtime exceptions later in the loop. This can be prevented by using the pandas Python data-analysis library, a Swiss army knife for data manipulation. You can use the fillna() method, which has some sane and sensible default values. Note that there are a lot of interesting ways to fill gaps in your data, and a lot of research and methods you can use. Some fields call it "predicting" missing data, other fields call it "imputation", "inference", or "sampling". You could forward-fill points, back-fill them, or interpolate them.

## Acting on the data

A simple thing to think about when setting up a logging system is fixed-threshold alerting. The goal of alerts is to wake somebody when your website goes down or another unexpected event occurs. Many people start the development of alerts by hiring a domain expert who can set sensible thresholds for various aspects of your system. For example, you could set an alert to fire as soon as 5% of the requests take longer than two seconds, at which point it noti-




fies the engineer who is on call at that moment.

However, human experts don't scale well. You might want to automatically compare the behaviour of machines to that of other machines, especially when you have many machines outputting many time series. You can't analyse and compare all these time series yourself, and a large number of machines could prevent you from comparing time series among them. This is the point at which you could try to apply machine learning.

Predictive models and outliers

One approach is outlier detection by using predictive modelling. By predicting what your machines' normal behaviour is, you can also detect when your machines act outside the predicted output. However, you do need to take a lot into account before you are able to do this. There are four key questions you have to ask:

- Is the behaviour actually regular?
- How can you model the behaviour?
- How do you define a major deviation from your expectation?
- Is it actually valuable to detect surprises and deviations from your expectation?

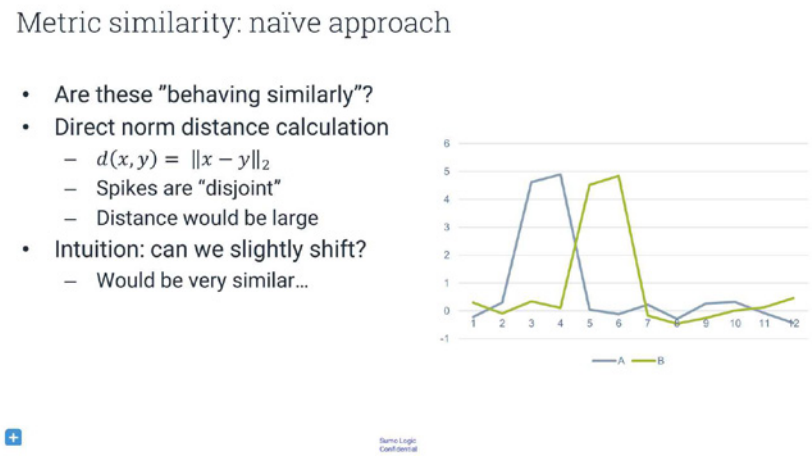
An important thing to consider when doing predictive modelling is seasonality or rhythm of your data. Any service that has humans in the loop has potential for rhythm. For example, most people use Sumo Logic at work, which means that Sumo Logic usage data for any given country will show a lot of activity during normal working hours but not so much outside these hours. However, the usage data for Netflix likely shows the reverse trend. You might be able to model this by manually adjusting your data or by using Fourier transformations. Another option many people use are hidden Markov models.

## Distance-based data mining of time series

When you have multiple machines, you probably want to compare the behaviour of machines to each other. If you see weird behaviour by one machine, you want to find out if other machines are behaving the same way. Perhaps they are all running different versions of software, perhaps they are in the same data centre, or perhaps something else is happening. To analyse this, you have to compare the distance between time series.

What metric should you use to determine the similarity between two time series? Simply differencing them time-wise by subtracting them from each other is bound to give wrong results. In the image above, although the time series are actually quite similar, this metric will tell you that they are completely different.

There is a whole universe of metrics you can use. One popular technique is dynamic time warping, which basically asks you how you can mutate, warp, or mangle your time series to get them into the best alignment, and what penalty you have to pay for this modification. With this metric, you can either find the N most-similarly behaving hosts, or you can build a graph of host



similarity. Using spectral clustering could provide an image that informs you about any structure in your hosts.

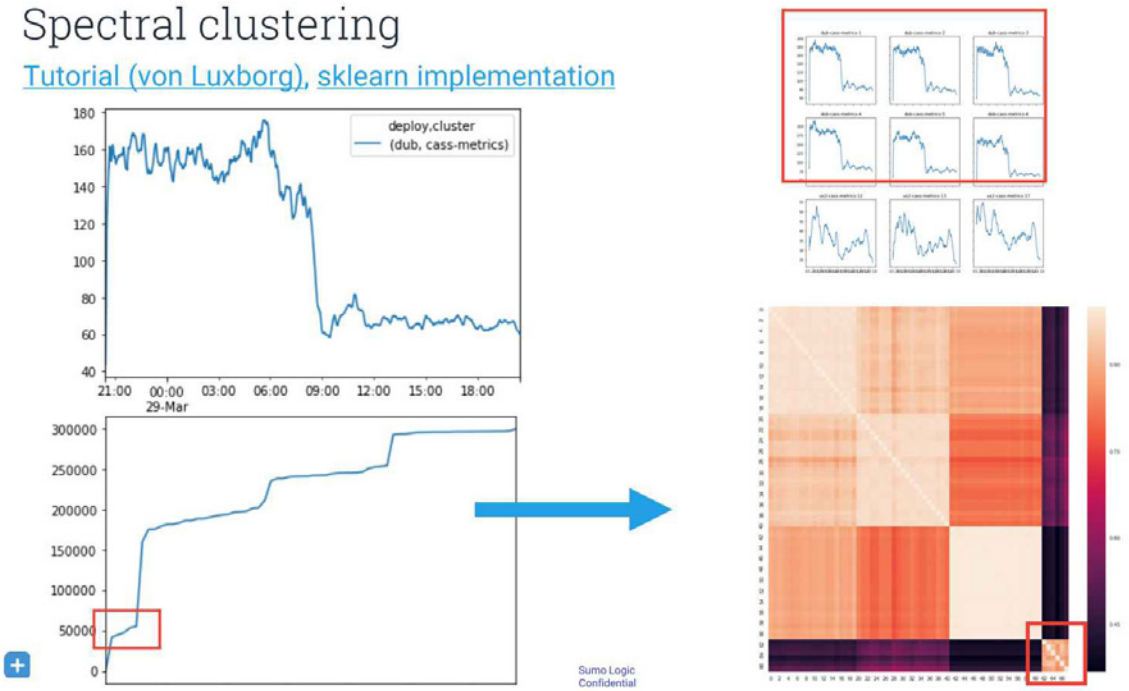
## Anomaly detection and event classification with log data

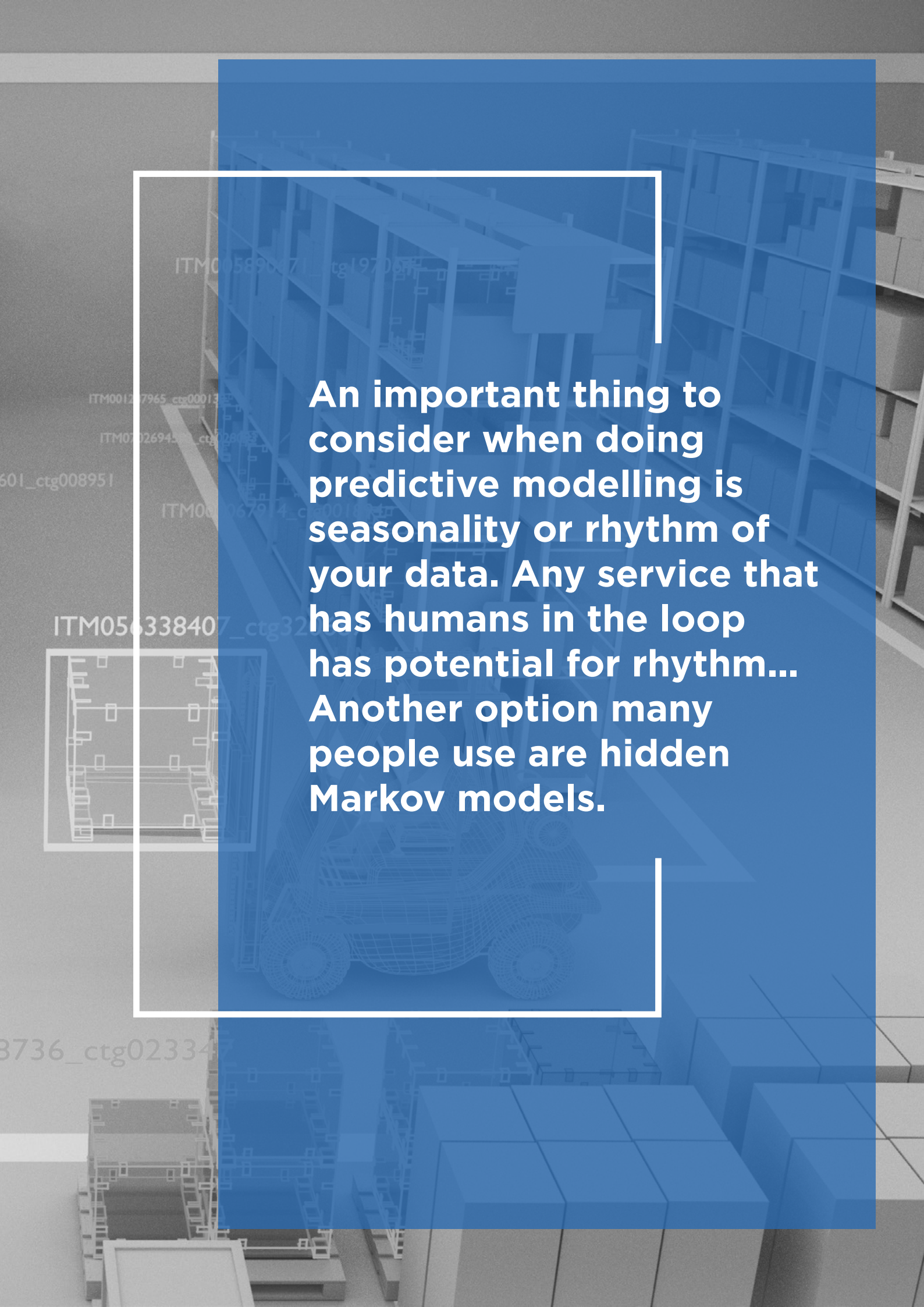
There are ways to transform your log data into a time series. When you have a high volume of semi-structured strings, you can either count the messages or extract information from them. These logs are an approximate program-execution trace. As you cannot enter a debugger for your machines once they are in production, you can only infer the behaviour of your software through these log messages. If your program prints a string every time there a request times out, you can count the number of timeouts every hour. This gives you a time series, which you just learned how to analyse!

You might be tempted to set a threshold on the values of a certain time series. However, you don't want to fool yourself into thinking that you've found an interesting event when that event was not actually interesting. Imagine that you have a super-accurate model, and you want to send an alert any time there is only a 0.01% chance for a pattern to occur. When you have a service with a million time series, you can expect about a hundred false positives. Baron Schwartz goes into more detail about what techniques you should use to determine a threshold in his talk "Why nobody cares about your anomaly detection".

With all the recent advancements in deep learning, you might want to use it to help with predictions and anomaly detection but deep learning is still not able to free you from understanding a problem domain. You still have to find a way to frame your problems. One possible approach is the use of recurrent neural networks to forecast. This is a great idea if you happen to have access to a lot of training data. If not, however, your first priority should be aggregating data before trying to do something with it.

In conclusion, the rabbit hole of inspecting data goes very deep. We have machines that are running our lives. These machines produce data, but analysing the data is complicated so we go to machine learning tools, which are delicate. It is of great importance to prevent noise and false positives, and to do this, you have to make sure you understand what you are trying to do. Know why you are not using deterministic SQL-like analysis, and understand the methods you use on a math scale. Lastly, know if you are either automating the interpretation or transforming data into predictive residuals and using that for anomaly prediction.





**An important thing to consider when doing predictive modelling is seasonality or rhythm of your data. Any service that has humans in the loop has potential for rhythm... Another option many people use are hidden Markov models.**


Watch online on InfoQ

# ANALYZING AND PREVENTING UNCONSCIOUS BIAS IN MACHINE LEARNING

by **Srini Penchikala**

This article is based on Rachel Thomas's keynote presentation, "Analyzing & Preventing Unconscious Bias in Machine Learning" at QCon.ai 2018.

Thomas works at fast.ai, a non-profit research lab that partners with the University of San Francisco's Data Institute to provide training in deep learning to the developer community. The lab offers a free course called "Practical Deep Learning for Coders".

Thomas discussed three case studies of bias in machine learning, its sources, and how to avoid it.

## Case study 1: Software for hiring, firing, and criminal-justice systems

Deep-learning algorithms are increasingly being used to make life-impacting decisions, such as in hiring and firing employees and in the criminal justice system. Coding bias brings pitfalls and risk to the decision-making process.

Pro Publica in 2016 investigated the COMPAS recidivism algorithm that is used to predict the likelihood that a prisoner or accused criminal would commit further crimes if released. The algorithm is used in granting bail, sentencing, and determining parole. Pro Publica found that the false-positive rate (labeled as "high-risk" but did not re-offend) was nearly twice as high for black defendants (error rate of 45%) as for white defendants (24%).

Race was not an explicit variable put into this algorithm, but race and gender are latently encoded in a lot of other variables, like where we live, our social networks, and our education. Even a conscious effort to not look at race or gender does not guarantee a lack of bias — assuming blindness doesn't work. Despite the doubts about the accuracy of COMPAS, the Wisconsin Supreme Court upheld the use of it last year. Thomas argued that it is horrifying that it is still in use.

It's important to have a good baseline to know what good performance is and to help indicate that a simpler model might be more efficient. Just because something's complicated, it doesn't mean that it works. The use of artificial intelligence (AI) for predictive policing is a concern.

Taser acquired two AI companies last year and is marketing predictive software to police departments. The company owns 80% of the police body-camera market in the US, so they have a lot of video data. Additionally, the Verge revealed in February that New Orleans police had been using predictive policing software from Palantir for the last six years in a top-secret program that even city council members didn't know about. Applications like these are of concern because there's no transparency. Because these are private companies, they're not subject to state/public record laws in the same way that police departments are. Often, they're protected in court from having to reveal what they're doing.

Also, there's a lot of racial bias in existing police data so the datasets that these algorithms are going to be learning from are biased from the start.

Finally, there's been a repeated failure of computer vision to work on people of color. Thomas said this is a scary combination of things to go wrong.

## Case study 2: Computer vision

Computer vision is often bad at recognizing people of color. One of the most infamous examples comes from 2015. Google Photos, which automatically labels photos, usefully categorized graduation photos and images of buildings. It also labeled black people as gorillas.

In 2016, the Beauty.AI website was using AI robots as judges for beauty contests. It found that people with light skin were judged much more attractive than people with dark skin. And in 2017, FaceApp, which uses neural networks to create filters for photographs, created a hotness filter that lightened people's skin and gave them more European features. Rachel showed a tweet of a user's actual face and a hotter version of him that the app created.

Thomas spoke about a research paper by Joy Buolamwini and Timnit Gebru, who evaluated several commercial computer-vision classifiers from Microsoft, IBM, and Face++ (a Chinese company). They found that the classifiers worked better on men than on women, and better on people with light skin than people with dark skin. There's a pretty noticeable gap: the error rate for light-skinned males is essentially 0% but ranges between 20% and 35% for dark-skinned females. Buolamwini and Gebru also broke down the error rates for women by skin shade. Errors increased with darkness of skin. The category of the darkest skin had error rates of 25% and 47%.

## Case study 3: Word embeddings

Thomas's third case study is the word embeddings in products like Google Translate.

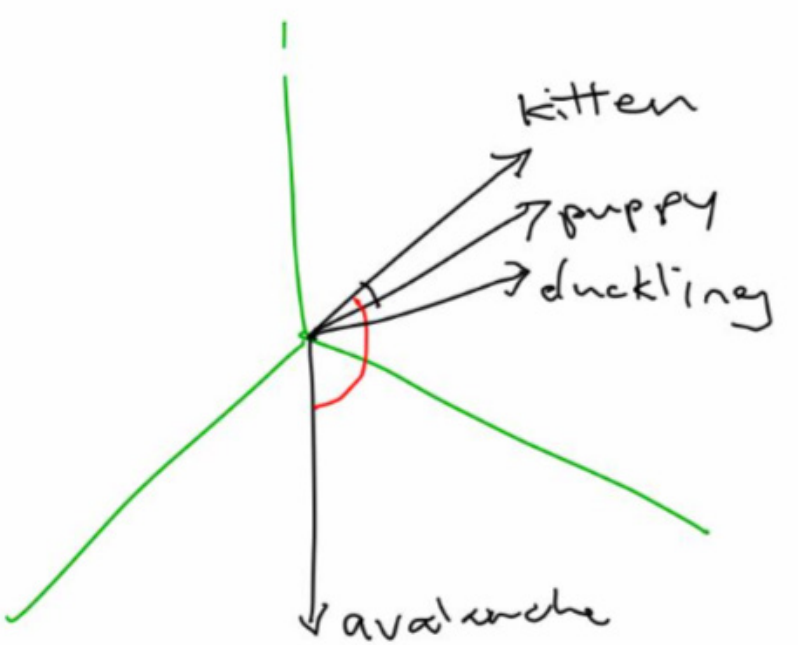
Take a pair of sentences like "She is a doctor. He is a nurse." Use Google Translate to translate them into Turkish and then translate them back into English. The genders become flipped to so

that the sentences now say, "He is a doctor. She is a nurse." Turkish has a gender-neutral singular pronoun that becomes translated into a stereotype in English. This happens with other languages that have gender-neutral singular pronouns. It's been documented for a variety of words that translation stereotypes hold that women are lazy, women are unhappy, and many more characterizations.

Thomas explained why this is happening. Computers and machine learning treat pictures and words as numbers. The same approach is used in speech recognition and image captioning. The way these algorithms work is that they take a supplied image and output something like "man in black shirt is playing guitar," or "construction worker in orange vest is working on the road." The same mechanism automatically suggests responses to emails in products like Google Smart Reply — if someone asks about your vacation plans, Smart Reply suggests that you might want to say, "No plans yet," or "I just sent them to you."

Thomas talked about an example in the fast.ai course "Practical Deep Learning for Coders". In this example, we can supply words and get back a picture. Give it the words "tench" (a type of fish) and "net" and it returns a picture of a tench in a net. This approach goes through a bunch of words and doesn't give us any notion of what it means for words to be similar. So "cat" and "catastrophe" might be sequential number wise but there's not any sort of semantic relationship between them.

A better approach is to represent words as vectors. Word embeddings are represented as high-dimensional vectors. She gave an




example of "kitten", and "puppy", and "duckling", which might all be close to each other in space because they're all baby animals. But the vector for "avalanche" might be far away since there's no real connection.

For more on word vectors, refer to "The amazing power of word vectors" by Adrian Colyer.

## Word2Vec

Word2Vec is a library of word embeddings released by Google. There are other similar libraries like Facebook's fastText and GloVe from the Natural Language Processing Group at Stanford University. It takes a lot of data, time, and computational power to train these libraries so it's handy that these groups have already done this and released their libraries for public use. It's much easier to use and this is an already trained version. The code for all three projects is available at GitHub, as is Thomas's own word-embeddings workshop. You can run her program using Jupyter Notebook and try out different words.

The word vectors for similar words like "puppy" and "dog" or "queen" and "princess" are closer in distance. And, of course, unrelated words like "celebrity" and "dusty" or "kitten" and "airplane" are more distant. The program uses a co-sign similarity, not Euclidian distance, since you don't want to use Euclidian distance in high dimensions.

You can use this solution to capture something about language. You can also find the 10 closest words to a specific target word. For example, if you look for the words closest to "swimming", you get words like "swim", "rowing", "diving", "volleyball", "gymnastics", and "pool". Word analogies are also useful. They capture things like "Spain is to Madrid as Italy is to Rome". However, there's a lot of opportunity for bias here. For example, the distance between "man" and "genius" is much smaller than the distance between "woman" and "genius".

The researchers studied baskets of words more systematically. They would take a basket or group of words like all flowers: clover, poppy, marigold, iris, etc. Another basket was insects: locust, spider, bedbug, maggot, etc. They had a basket of pleasant words (health, love, peace, cheer, etc.) and a basket of unpleasant words (abuse, filth, murder, death, etc.). The researchers looked at the distances between these different word baskets and found that flowers were closer to pleasant words and insects were closer to unpleasant words.

This all seems reasonable so far, but then the researchers looked at stereotypically black names and stereotypically white names. They found that the black names were closer to unpleasant words and the white names were closer to pleasant words, which is a bias. They found a number of racial and gender biases among entire groups of words, which produced analogies like "father is to doctor as mother is to nurse", "man is to computer programmer as woman is to homemaker". These are all analogies found in Word2Vec and in GloVe.

Thomas talked about another example of bias in a system for restaurant reviews that ranked Mexican restaurants lower because the word embeddings for "Mexican" had negative connotations. These word embeddings are trained with a giant corpus of texts. These texts contain a lot of racial and gender biases, which is how the word embeddings learn this associations at the same time as they learn the semantic meanings that we want them to know.

## Machine learning can amplify bias

Machine learning can actually amplify bias. An example of this is discussed in "Men also like shopping: Reducing gender bias amplification using corpus-level constraints", which looked at visual semantic role labeling of images in a dataset. The researchers found that 67% of images of people cooking were women but the algorithm labeled 84% of the cooks as women. There is a risk of machine learning algorithms amplifying what we see in the real world.

Thomas mentioned research by Zeynep Tufekci, who has provided insights into the intersection of technology and society. Tufekci has tweeted that "the number of people telling me that YouTube autoplay ends up with white supremacist videos from all sorts of starting points is pretty staggering." Examples include:

- "I was watching a leaf blower video and three videos later, it was white supremacy."
- "I was watching an academic discussion of the origins of plantation slavery and the next video was from holocaust deniers."
- "I was watching a video with my daughters on Nelson Mandela and the next video was something saying that the black people in South Africa are the true racist and criminals."

It's scary.

Renée DiResta, an expert in disinformation and how propaganda spreads, noticed a few years ago that if you join an anti-vaccine group on Facebook, the

Machine learning can actually amplify bias. An example of this is discussed in the white-paper called "Men also like shopping" where they looked at a dataset. They found that 67% of people cooking are women and the algorithm predicted that 84% of the people cooking would be women."

site would also recommend to you groups about natural cancer cures, chemtrails, flat Earth, and of all sorts of anti-science groups. These networks are doing a lot to promote this kind of propaganda.

Thomas mentioned a research paper on how runaway feedback loops can work on predictive policing. If software or analysis predicts that there will be more crime in an area, the police might send more officers there — but because there are more police there, they might make more arrests, which might cause us to think that there's more crime there, which might cause us to send even more police there. We can easily enter this runaway feedback loop.

Thomas suggested that we really need to think about the ethics of including certain variables in our models. Although we may have access to data, and even if that data improves our model's performance, is it ethical to use? Is it in keeping with our values as a society? Even engineers need to be asking ethical questions about the work they do and should be able to answer ethical questions about it. We're going to see less and less tolerance from society for this.

Angela Bassa, the director of data science at iRobot said, "It's not that data can be biased. Data is biased. If you want to use data, you need to understand how it was generated."

## Addressing bias in word embeddings

Even if we remove bias early in model development, there are so many places that bias can seep in that we need to continue to look out for it.

More representative datasets can be one solution. Buolamwini and Gebru identified the bias failures in computer-vision products mentioned above and put together a much more representative dataset of men and women with all different skin shades. This data set is available at Gender Shades. The website also offers their academic paper and a short video about their work.

Gebru with others recently released a paper called "Datasheets for Datasets". The paper provides prototype datasheets for recording characteristics and metadata that reveal how a dataset was created, how it was composed, what sort of preprocessing was done, what sort of work is needed to maintain it, and any legal or ethical considerations. It's really important to understand the datasets that go into building our models.

Thomas emphasized that it's our job to think about unintended consequences in advance. Think about how trolls or harassers or authoritarian governments could use a platform that we build. How could our platform be used for propaganda or disinformation? When Facebook announced that it was going to start threat modelling, many people asked why it hadn't been doing that for the last 14 years.

There's also an argument for not storing data we don't need so that nobody can ever take that data.

Our job is to think about how our software could be misused before it happens. The culture of the field of information security is based on that. We need to start doing more of thinking about how things could go wrong.

## Questions to ask about AI

Thomas listed some questions to ask about AI:

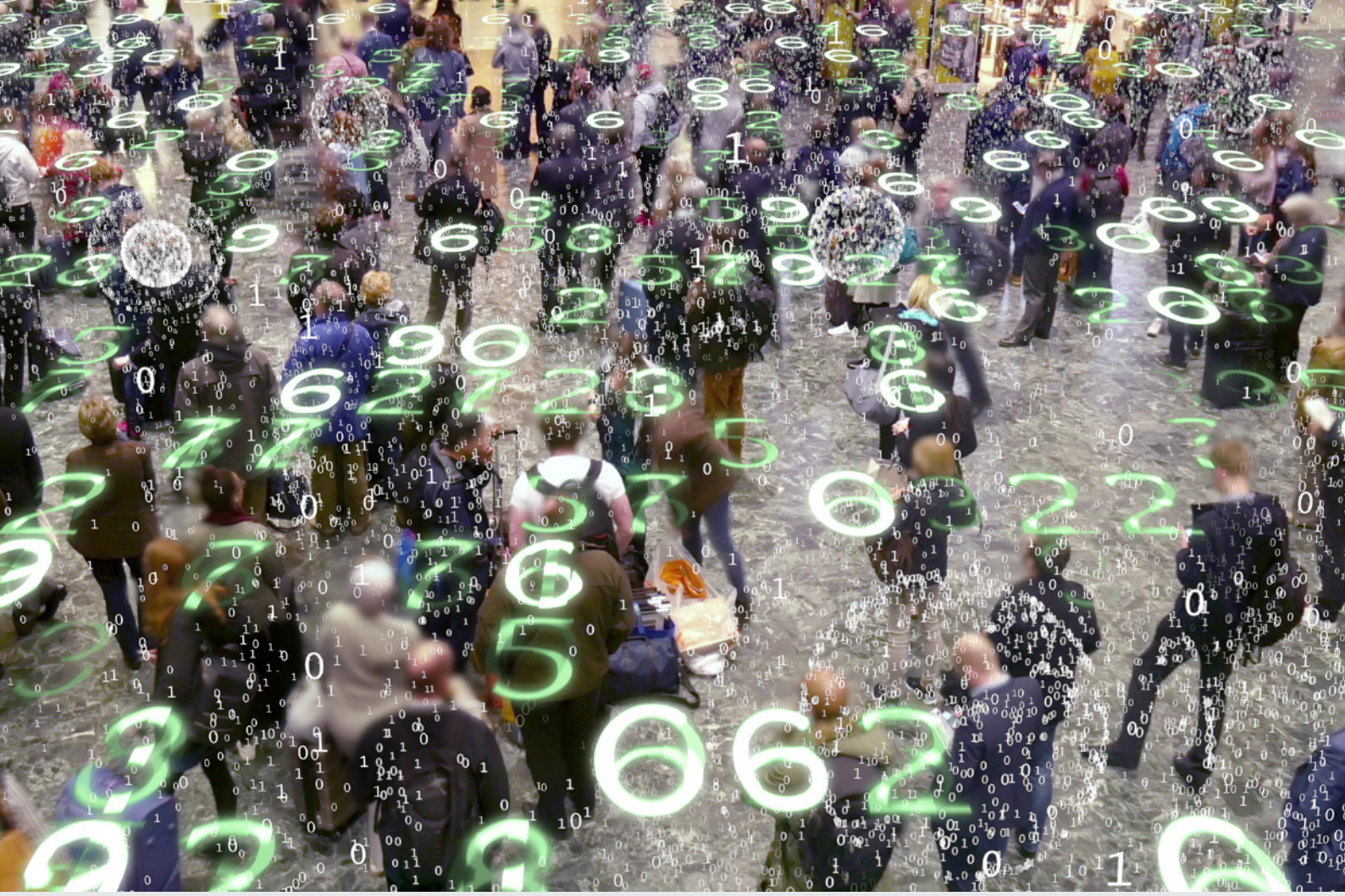
- What bias is in the data? There's some bias in all data and we need to understand what it is and how the data was created.
- Can the code and data be audited? Are they open source? There's a risk when closed-source proprietary algorithms are used to decide things in healthcare and criminal justice and who gets hired or fired.
- What are error rates for different subgroups? If we don't have a representative dataset, we may not notice that our algorithm is performing poorly on some subgroup. Are our sample sizes large enough for all subgroups in your dataset? It's important to check this, just like Pro Publica did with the recidivism algorithm that looked at race.
- What is the accuracy of a simple rule-based alternative? It's really important to have a good baseline, and that should be the first step whenever we're working on a problem because if someone asks if 95% accuracy is good, we need to have an answer. The correct answer depends on the context. This came up with the recidivism algorithm, which was no more effective than a linear classifier of two variables. It's good to know what that simple alternative is.
- What processes are in place to handle appeals or mistakes? We need a human appeals process for things that affect people's lives. We, as engineers, have relatively more power in asking these questions within our companies.
- How diverse is the team that built it? The teams building our technology should be representative of the people that are going to be affected by it, which increasingly is all of us.

Research shows that diverse teams perform better, and believing we're meritocratic actually increases bias. It takes time and effort to do interviews consistently. A good reference for this is the blog post titled "Making small culture changes" by Julia Evans.

Advanced technology is not a substitute for good policy. Thomas talked about fast.ai students all over the world who are applying deep learning to social problems like saving rainforests or improving the care of patients with Parkinson's disease.

There are AI regulations like the Age Discrimination and Employment Act from 1967 and Equal Credit Opportunity Act that are relevant. These are not perfect but are better than not having any protection since we really need to think about what rights we as a society want to protect.

Thomas concluded her talk by saying you can never be done checking for bias. We can follow some steps towards the solutions but bias could seep back in from so many places. There's no checklist that assures us that bias is gone and we no longer have to worry. It's something that we always have to continue to look out for.



Read online on InfoQ

## KEY TAKEAWAYS

There is no consensus on how to define, avoid, or even make explicit the bias in the algorithms that are used in executing public policy or in scientific research.

The seamless and convenient nature of many technologies, such as personalized homes, makes it difficult to understand where data comes from, how it is used by algorithms, and where it goes.

Companies and individuals, especially when working in the public sector, should assume that the results of algorithmic decisions will have to be explained to people who are adversely affected by them in a timely fashion so they can appeal or challenge these decisions

It also seems reasonable to assume that organizations will also have to explain how they are using an individual's data.

# CAN PEOPLE TRUST THE AUTOMATED DECISIONS MADE BY ALGORITHMS?

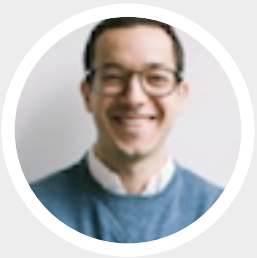
by **Michael Stiefel**

The use of automated decision making is increasing. The algorithms underlying these systems can produce results that are incomprehensible or socially undesirable. How can regulators determine the safety or effectiveness of algorithms embedded in devices or machines if they cannot understand them? How can scientists understand a relationship based on an algorithmic discovery?

# THE PANELISTS

## Michael Veale

is a doctoral researcher in responsible public-sector machine learning at University College London, specialising in the fairness and accountability of data-driven tools in the public sector as well as the interplay between advanced technologies and data-protection law. His research has been cited by international bodies and regulators, in the media, as well as debated in the UK Parliament. He has acted as consultant on machine learning and society for the World Bank, the Royal Society, and the British Academy, and previously worked on IoT, health and ageing at the European Commission. Veale tweets at @mikarv.



## Andrew Burt

is chief privacy officer and legal engineer at Immuta, the world's leading data-management platform for data science. He is also a visiting fellow at Yale Law School's Information Society Project. Previously, Burt was a special advisor for policy to the head of the FBI Cyber Division, where he served as lead author on the FBI's after-action report on the 2014 attack on Sony. Burt has published articles on technology, history, and law in the New York Times, the Financial Times, the Los Angeles Times, Slate, and the Yale Journal of International Affairs, among others. Burt holds a JD from Yale Law School and a BA from McGill University.

## Rebecca Williams

is professor of public law and criminal law at the University of Oxford. Her work includes examining optimum methods of decision making and the use of criminal law as a form of regulation. Increasingly, her work also focuses on the relationship of law and technology and the ways in which the law will need to develop in order to keep pace with technological developments.

Automated decision making is at work in areas such as determining who is let out on bail, who gets financial credit, predicting where crime will take place, ascertaining violations of anti-discrimination laws, or adjudicating fault in an accident with a self-driving car.

It is unclear if the algorithms can detect their own flaws any more than a human being can determine if they are truly mentally ill. There is no line of code in these algorithms that says "do a bad thing to someone."

What can we do to solve this problem?

InfoQ asked these three experts for their thoughts:

- Rebecca Williams is a professor of public law and criminal law at Oxford University.
- Andrew Burt is chief privacy officer and legal engineer at Immuta.
- Michael Veale is a PhD candidate in the Department of Science, Technology, Engineering, and Public Policy at University College London.

**InfoQ: People are often unaware of the role of algorithms in society. What is the best way to educate people about the benefits and problems associated with the growing pervasive use of algorithms?**

**Andrew Burt**: What we need most is history and context — about how this type of technology has been used before and about what's different now, especially when it comes to what's commonly referred to as AI. We have, on the one hand, people like Elon Musk declaring that AI is an existential threat to life on Earth, which is having a real im-

> Most useful evidence is causal in nature. We want to know what causes what, and how the world works. Machine learning algorithms aren't so good at that, and their results and predictive power can be quite brittle as a result.

pact on the way the public thinks about AI. And we have, on the other hand, some diehard proponents of AI suggesting it will solve every problem we have. The truth is, of course, on neither extreme. What's more, not every challenge AI poses is new. We've already developed tools and practices to confront some of these challenges in other areas. So, I think everyone would benefit from a broader discussion that places the challenges of AI in perspective, and lets us build off of past successes and correct mistakes in how we adopted earlier technologies. There's a lot of good we can do if we get this right. Conversely, there's a lot of harm that could come about if we get this wrong — discriminatory harms, missed opportunities, and more. The stakes are high.

**Rebecca Williams**: Articles 13(2)(f), 14(2)(g), and 15(1)(h) of the GDPR state that data subjects have "the right to know the existence of automated decision-making including profiling". So whatever else they are entitled to by way of information about the process, at the very least people will have to be told when a particular decision about them or concerning them is being made using an automated process. The hope is that that will raise some awareness of when and where these systems are being used.

In terms of education, obviously the earlier we start with these issues the better. Schools increasingly teach coding to students as well as ethical issues such as citizenship or personal and social education, so the more that can be done to raise awareness and discussion in those contexts, the better prepared future generations will be when they come to design, operate, and interact with these systems. This is definitely something that universities can also help to facilitate. There are already contexts in which academics visit schools to support learning and it would be great if this could happen on this subject too.

That of course leaves the question of how we can reach those who went through their school education before these kinds of concerns had arisen. The same challenges arise here as arise in relation to the dissemination of any kind of information: people will tend to rely on certain sources rather than others, giving rise to the risk of echo chambers and misinformation. There will certainly be a role for the mainstream media here, and balanced, scientifically-based reporting by those media will be vital, as always, but the less reliance the public place on such sources of information, the less effective this will be. There will certainly be a role for institutions like the Information Commissioner's Office to provide advice and information for citizens through its website, and again as an academic I would be keen to see universities assisting in this context too, either by supporting these other outlets or through direct public engagement.

**Michael Veale**: In technology design, there has been a big trend towards making systems "seamless". In short, this means that people can focus on what they want to do, not how they want to do it, which is usually really great for helping individuals to achieve what they want. Smart homes are an example of this, although many are a bit too clunky to have totally earned that title. Yet with a range of algorithmic systems today, too much seamlessness means that individuals don't get a chance to question

whether this system works the way they want it to. Your smart home might be personalised, but you can't see where, and to whom, it is sending the data. Your Facebook news feed might seem compelling, but you can't see who is being excluded and why.

We could run courses about algorithms in society, but that's unlikely to solve deeper problems. Technologies move fast. My young cousin told me the other day that at school, they'd been learning about cybersecurity. "They told us not to click on pop-ups," she said. "But how will I know what a pop-up looks like?" Browsers have moved so quickly to block them, and on mobile devices it's simply not the paradigm at all anymore. So that one-off education, unless it is building general critical skills, usually is a bit too much of a moving target.

Consequently, we need to imbue education into the products and services we use every day. These services should explain themselves, not necessarily with a passage of text or a manual, but by virtue of clever design that makes it clear when data flows, automated decisions, and other behaviours are happening. Where that's the case, individuals should be able to drill down further to see and learn more if interested — and then they'll no doubt get more of a feel for what is happening around them even when the options to perceive and drill down are not there.

**InfoQ: Algorithms will often be used in executing public policy or in scientific research that will affect public policy. Legal requirements, value judgements, and bias are almost unavoidable. How can social values be made explicitly visible and bias be avoided in algorithmic programming and in interpreting the results?**

**Burt:** On the technology side, there are all sorts of important tools that are being developed to help minimize many of these downsides. A tool called LIME, which helps explain so-called black-box algorithms, is one great example. A data scientist named Patrick Hall really deserves a shout out for doing some great work on interpretability in machine learning. And there are many more examples to cite. Our legal-engineering and data-science teams are staying on top of all these developments at Immuta.

But I think what's often overlooked is the procedural side. The processes used to develop and deploy machine learning are incredibly important, and model risk-management frameworks like the Federal Reserve Board's SR 11-7 have long recognized this fact. That regulation applies to the use of algorithms within financial institutions in the US. The folks at the AI Now Institute have also come forward with what they're calling "algorithmic impact assessments", which offer another framework for this type of approach.

There's a lot out there, frankly, and we'll be releasing a white paper shortly that will sum up some of these best practices — both technical and procedural — to help our customers and others manage the risks of deploying machine-learning models in practice. We're hard at work finalizing that whitepaper and are excited to release it in the next few months.

**Williams:** There are various different ways we can approach this issue. First, it is vital to examine carefully the data used to train and operate automated decision-making systems. If the data itself is biased, the outcome will be too. There has been a lot of discussion of the risk-prediction systems used in the criminal-justice context in a number of US states and the difficulty with such systems is that they tend to over-predict recidivism by black defendants while under-predicting it for white defendants. But just to take an example, one potential predictor of risk used might be prior arrests for more minor possession offences. And yet such offences are most likely to be detected by stop and search, and stop-and-search tactics tend to skew in the same direction: over-predicting a reason to stop and search black people while under-predicting the need to stop and search white people. So, because stop and search is skewed against black people in favour of white, more black people are found to be in possession than white and thus black people are calculated to have a higher risk of recidivism than white people. The initial discrimination in data collection thus feeds through the whole system into the output. So, if we think our initial data is likely to produce this kind of skewed effect, we should think carefully about whether or not it is appropriate to use it, and we may need to think about imposing duties to gather counterbalancing data.

Second, there are important policy choices to be made in the process of coding the system. Krishna Gummadi's work has shown that it is not always possible to have one's cake and eat it too. Usually, it will be necessary to choose between different measures of accuracy. For example, a system that has the most accurate method of prediction on

aggregate, taken across all cases, might also have the biggest problem with producing skewed results in relation to particular categories of case, as above. Or, conversely, a system that has maximum accuracy in relation to any particular category (such as ethnic status or gender) might not have such a high degree of accuracy across all categories on aggregate. It is vital that any such policy choices between different systems are understood as being just that: they are policy choices that must be made openly and transparently and by an entity that can then be held accountable for making them, not unconsciously by anonymous coders.

Third, even if we are confident that we have done all we can ex ante to gather balanced data and make responsible coding choices, it will also be necessary ex post to ensure that such systems are subject to regular audit to ensure that they are not spontaneously generating forms of discrimination that we had not predicted. It will be necessary to do this even if we are not sure why it is happening, but, fourth, it is also vital that we do everything we can to make the algorithms themselves transparent and accountable, so that if an audit of this kind does pick up a problem we can see where and how it has arisen. There are a number of people working on this and a group of us in Aberdeen (Pete Edwards), Oxford, and Cambridge (Jat Singh) have just received an EPSRC grant to work further on this issue.

In terms of the sources of regulation for each of these four issues, the systems will be used by both public and private entities. Where they are operated by public or governmental entities I think there is definitely a role for our existing public law to play in holding such entities to account and imposing further duties of transparency, fairness etc., which are already inherent in public law. For private entities the challenge will be to think which of these duties of transparency, accountability, and fairness should be carried across into the private sector as the price for the increased power offered by such systems.

**Veale:** Most useful evidence is causal in nature. We want to know what causes what, and how the world works. Machine-learning algorithms aren't so good at that, and their results and predictive power can be quite brittle as a result. The main way to make social values explicitly visible is to slow down and recognise that our aims are often not just prediction, but understanding. We are in huge danger of training a generation of people who can do the former but not the latter. When you build causal models, you have got a greater opportunity to discuss if this is how you want the world to work and behave. Perhaps it is, perhaps it isn't; but it's a conversation that's more visible, and much easier to have and to communicate.

**InfoQ: In May of this year, the EU GDPR came into effect. Among its provisions is Article 22, which deals with automated individual decision making. Many people argue this rule requires not only that the privacy rights for data must be respected, but that decisions made by algorithms be explainable.**

**Do you agree with this interpretation of the regulation? Does this regulation require that data be removed from use by algorithms? If so, could this reduce the effectiveness of the algorithm? In general, is the EU's approach a valid one or is the law of unintended consequences going to make it worse?**

**Burt:** There's a huge debate going on in the legal community over how, exactly, the GDPR will impact the deployment of machine learning. And given that the GDPR only came into effect within the last month, there's a lot that's still up in the air. But my take is that Article 22 needs to be read alongside Articles 13-15, which state that data subjects have a right to "meaningful information about the logic involved" in cases of automated decision making. In practice, I think this is going to mean that data subjects are going to have the right to be educated about when, why, and, most importantly, how something like a machine-learning model is using their data. As with any legal analysis, there's a ton of nuance here. So I'd encourage readers to check out an earlier article I put together on the subject for the International Association of Privacy Professionals. It's also worth mentioning that a group called the Working Party 29, which has a huge influence on how EU privacy laws are enforced, has come out with its own guidance on this subject, flatly stating that automated decision making is prohibited by default under the GDPR, with certain exemptions.

**Williams:** You'll already know that there has been an intense debate between Goodman and Flaxman, who argue that the GDPR gives a full "right to explanation", while Wachter, Mittelstadt, and Floridi, in my view, more plausibly argue that it will be sufficient for the data subject to be told of the existence of a

machine-learning component and what measures of accuracy are being used to check it. I agree with them that the data subject should be told more than just which data points are being used, but also how they are weighed in the circumstances. As I mentioned above, where the system is being operated by a public entity, I think there is significant potential for an analogy to be drawn with our current approach to closed material procedure decisions, where if the impact on the individual is significant, (s)he has the right to know at least the "gist" of the case against him/her so that (s)he can make "meaningful" use of the right to reply. That might just involve ex ante explanation, as Wachter, Mittelstadt, and Floridi suggest, but it might include ex post explanation too. In relation to private entities, the situation is more difficult as they are generally subject to fewer duties, although our existing law on discrimination will do some work and there is also the potential for public-style duties to be attached to the use of such systems even in a private context.

Article 17 allows for the right to erasure of personal data, but not where the processing is necessary to comply with a legal obligation. The key distinction here is between individual and general data. For removal of individual data, there are some limited rights, like those in Article 17, but for any duty or obligation to remove general data (i.e., data affecting a whole category of people such as the stop-and-search data above), you might have to look either to more general provisions in the regulation like "suitable measures to safeguard the data subject's rights and freedoms and legitimate interests", or general duties in, for example, public law (where the processor is public/governmental) or law prohibiting discrimination.

Again, that depends on whether what is being used is individual or general data. Removing skewed general data might make the algorithm more accurate, whereas removing accurate individual data in relation to particular kinds of applicants might make it more inaccurate and give rise to the skewing effect.

I don't think anyone knows the answer to that for certain at this point! I do think it will be necessary to remember the ex post audits I discussed above, so that if in practice we do see unintended consequences, there is an opportunity to catch those and remedy them.

**Veale:** Article 22 in the GDPR is a really old provision. It dates back to French law in 1978, and much of it is unchanged from Article 15 of the Data Protection Directive in 1995 (the UK Data Protection Act 1998). Yet it hasn't been used much, and some scholars have called it a "second-class right" as a result.

The fundamental purpose of Article 22 is to ensure that if an organisation wants to take a fully automated, potentially significant decision about someone, they need to have a legal basis to do it (freely given consent, necessity for performing a contract, or legal obligation). If the organisation doesn't have one of these, they can't take the decision. If they do secure one, they have to put safeguards into place to ensure the decision is taken fairly, including allowing an individual to challenge the decision. It's unclear in many cases how that challenge will work: many significant decisions are taken very quickly. If a video of a topical, political event is automatically removed from YouTube, how quickly can it be brought back up? If its time of relevance has past, a human review is of little use.

Another one of these safeguards, beyond human challenge, is described in Recital 71 of the GDPR. Recitals, which begin a European law, are meant to illustrate its spirit and context, but in highly fought-over laws like the GDPR, they have become, frustratingly for lawyers, a place to put things that really should be in the main, binding articles. This explanation safeguard, unlike others like the right to human intervention, was placed there, and so we will see if and when the European Court of Justice thinks it is binding on data controllers.

Yet let's not forget the actual meaning of Article 22, which isn't just about explanations. It definitely restricts some uses of algorithmic systems people think are unfair. Automated hiring and CV filtering, for example, are techniques that are highly suspect under Article 22. When you are deciding to interview someone automatically, using one of the analytic products on the market today, you are likely making a solely automated, significant decision. What is your legal basis? You don't have a contract and probably don't have a legal obligation, so that leaves consent. Consent in any employment context is highly problematic due to the power imbalances, and can rarely be seen as freely given. Personally, I think that Article 22 renders a lot of large-scale, automatic hiring practices very legally suspect.

**Our legal and regulatory structures need to provide and incentivise transparency and accountability, working closely with the computer scientists who generate the systems.**

**InfoQ: What do you think is the critical issue facing societies with widespread use of algorithms instead of humans to make critical decisions?**

**Burt:** In two words: silent failures. As we begin to rely more and more on complex algorithms, especially various forms of neural networks, our ability to explain their inner workings is going to get progressively harder. This isn't simply because these models are hard to interpret, but because the networks we're connecting them to are becoming more and more complex. Every day, the world of IT gets harder to manage — we have more endpoints, more data, more databases, and more storage technologies than ever before. And so, I believe our biggest challenge lies in being able to understand the data environments we are relying on. Because if we don't understand them, there's a very real possibility that we'll be constantly confronting silent failures, where something has gone wrong that we simply don't know about, with very real — and potentially devastating — consequences.

**Williams:** I think most people would encapsulate this in the word "fairness". But that really boils down into transparency and accountability: (1) We need to know as much as possible about what these systems are doing, how, and why; (2) There needs to be an appropriate entity to hold accountable and an appropriate and accessible system for holding that entity accountable.

Our legal and regulatory structures need to provide and incentivise these two things, working closely with the computer scientists who generate the systems.

**Veale:** The biggest issue here is that algorithms take maintenance and oversight, which can be hard to do at a small scale. They theoretically allow a huge volume and speed of automated decisions, many more than a human can do. Small organisations can really benefit from that. Previously, if organisations wanted a lot of decision making to happen, they needed a lot of people. Those people could provide oversight and feedback, even if they brought their own biases. Now, a few individuals can deploy and manage huge decision-making infrastructures, but they don't bring the human capacity to look over them and maintain them. This creates a huge imbalance, particularly for low-capacity organisations who might be tempted by relying on automation and on machine learning. In these cases, external oversight is needed — but who provides that? Who pays for it? And how does it really get to grips with some hidden challenges that algorithmic decision making might cause, challenges which are often buried deep within organisations and their work politics?

## Conclusion

Failing to take into consideration what the public fears or the inability to foresee adverse consequences have impeded technologies such as nuclear energy and genetically modified crops.

New York City is establishing a task force to propose recommendations for obtaining explanation and mitigations for people affected by the use of algorithms by city agencies. The European Union's GDPR is another attempt to start to deal with the issue.

Carl Jung is reputed to have said that within every human being hides a lunatic. If algorithms model human behavior, what does that mean for society?

**More on this**


**CATHY O'NEIL ON PERNICIOUS MACHINE LEARNING ALGORITHMS AND HOW TO AUDIT THEM**

# PREVIOUS ISSUES


BOOK

## Big Data Processing with Apache Spark

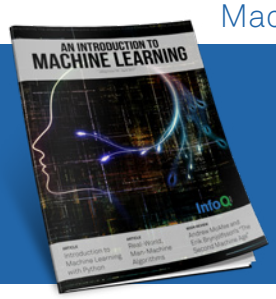
In this mini-book, the reader will learn about the Apache Spark framework and will develop Spark programs for use cases in big-data analysis. The book covers all the libraries that are part of Spark ecosystem, which includes Spark Core, Spark SQL, Spark Streaming, Spark MLlib, and Spark GraphX.

## Getting a Handle on Data Science


49

This eMag looks at data science from the ground up, across technology selection, assembling raw and unstructured data, statistical thinking, machine learning basics, and the ethics of applying these new weapons.

## Introduction to Machine Learning




50

InfoQ has curated a series of articles for this introduction to machine learning eMagazine, covering everything from the very basics of machine learning (what are typical classifiers and how do you measure their performance?) and production considerations (how do you deal with changing patterns in data after you've deployed your model?), to newer techniques in deep learning

## Streaming Architecture


57

This InfoQ emag aims to introduce you to core stream processing concepts like the log, the dataflow model, and implementing fault-tolerant streaming systems.