O'REILLY®

What Is Data Science?

The future belongs to the companies and people that turn data into products



Mike Loukides

www.dbooks.org











Strata+ Hadoop WORLD

Make Data Work

strataconf.com

Presented by O'Reilly and Cloudera, Strata + Hadoop World helps you put big data, cutting-edge data science, and new business fundamentals to work.

- Learn new business applications of data technologies
- Develop new skills through trainings and in-depth tutorials
- Connect with an international community of thousands who work with data

Job # D2044

What Is Data Science?

Mike Loukides

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®

www.dbooks.org

What Is Data Science?
by Mike Loukides

Copyright © 2011 O'Reilly Media, Inc. All rights reserved.

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://safaribooksonline.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Interior Designer: David Futato

Cover Designer: Karen Montgomery
Illustrator: Robert Romano

April 2011: First Edition

Revision History for the First Edition
2011-04-15: First Release

The O'Reilly logo is a registered trademark of O'Reilly Media, Inc. *What Is Data Science?,* the cover image, and related trade dress are trademarks of O'Reilly Media, Inc.

While the publisher and the author have used good faith efforts to ensure that the information and instructions contained in this work are accurate, the publisher and the author disclaim all responsibility for errors or omissions, including without limitation responsibility for damages resulting from the use of or reliance on this work. Use of the information and instructions contained in this work is at your own risk. If any code samples or other technology this work contains or describes is subject to open source licenses or the intellectual property rights of others, it is your responsibility to ensure that your use thereof complies with such licenses and/or rights.

978-1-491-91186-0

[LSI]

Table of Contents

1. What is data science? **1**
The future belongs to the companies and people that turn data into products 1
What is data science? 1
Where data comes from 4
Working with data at scale 8
Making data tell its story 13
Data scientists 14

www.dbooks.org

CHAPTER 1

What is data science?

The future belongs to the companies and people that turn data into products

We've all heard it: according to Hal Varian, statistics is the next sexy job (*http://www.nytimes.com/2009/08/06/technology/06stats.html*). Five years ago, in What is Web 2.0 (*http://oreilly.com/web2/archive/what-is-web-20.html*), Tim O'Reilly said that "data is the next Intel Inside." But what does that statement mean? Why do we suddenly care about statistics and about data?

In this post, I examine the many sides of data science—the technologies, the companies and the unique skill sets.

What is data science?

The web is full of "data-driven apps." Almost any e-commerce application is a data-driven application. There's a database behind a web front end, and middleware that talks to a number of other databases and data services (credit card processing companies, banks, and so on). But merely using data isn't really what we mean by "data science." A data application acquires its value from the data itself, and creates more data as a result. It's not just an application with data; it's a data product. Data science enables the creation of data products.

One of the earlier data products on the Web was the CDDB database (*http://en.wikipedia.org/wiki/CDDB*). The developers of CDDB realized that any CD had a unique signature, based on the exact length

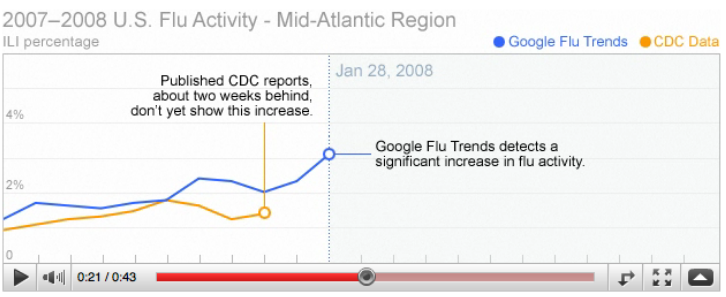
www.dbooks.org

(in samples) of each track on the CD. Gracenote built a database of track lengths, and coupled it to a database of album metadata (track titles, artists, album titles). If you've ever used iTunes to rip a CD, you've taken advantage of this database. Before it does anything else, iTunes reads the length of every track, sends it to CDDB, and gets back the track titles. If you have a CD that's not in the database (including a CD you've made yourself), you can create an entry for an unknown album. While this sounds simple enough, it's revolutionary: CDDB views music as data, not as audio, and creates new value in doing so. Their business is fundamentally different from selling music, sharing music, or analyzing musical tastes (though these can also be "data products"). CDDB arises entirely from viewing a musical problem as a data problem.

Google is a master at creating data products. Here's a few examples:

- Google's breakthrough was realizing that a search engine could use input other than the text on the page. Google's PageRank (*http://en.wikipedia.org/wiki/PageRank*) algorithm was among the first to use data outside of the page itself, in particular, the number of links pointing to a page. Tracking links made Google searches much more useful, and PageRank has been a key ingredient to the company's success.
- Spell checking isn't a terribly difficult problem, but by suggesting corrections to misspelled searches, and observing what the user clicks in response, Google made it much more accurate. They've built a dictionary of common misspellings, their corrections, and the contexts in which they occur.
- Speech recognition has always been a hard problem, and it remains difficult. But Google has made huge strides by using the voice data they've collected, and has been able to integrate voice search (*http://gdgt.com/discuss/voice-recognition-is-amazing-ive-only-68e/*) into their core search engine.

- During the Swine Flu epidemic of 2009, Google was able to track the progress of the epidemic by following searches for flu-related topics (*http://www.google.org/flutrends/about/how.html*).



Google was able to spot trends in the Swine Flu epidemic roughly two weeks before the Center for Disease Control by analyzing searches that people were making in different regions of the country.

Google isn't the only company that knows how to use data. Facebook (*http://www.facebook.com*) and LinkedIn (*http://www.linkedin.com*) use patterns of friendship relationships to suggest other people you may know, or should know, with sometimes frightening accuracy. Amazon (*http://www.amazon.com*) saves your searches, correlates what you search for with what other users search for, and uses it to create surprisingly appropriate recommendations. These recommendations are "data products" that help to drive Amazon's more traditional retail business. They come about because Amazon understands that a book isn't just a book, a camera isn't just a camera, and a customer isn't just a customer; customers generate a trail of "data exhaust" that can be mined and put to use, and a camera is a cloud of data that can be correlated with the customers' behavior, the data they leave every time they visit the site.

The thread that ties most of these applications together is that data collected from users provides added value. Whether that data is search terms, voice samples, or product reviews, the users are in a feedback loop in which they contribute to the products they use. That's the beginning of data science.

www.dbooks.org

In the last few years, there has been an explosion in the amount of data that's available. Whether we're talking about web server logs, tweet streams, online transaction records, "citizen science," data from sensors, government data, or some other source, the problem isn't finding data, it's figuring out what to do with it. And it's not just companies using their own data, or the data contributed by their users. It's increasingly common to mashup data from a number of sources. "Data Mashups in R (*http://oreilly.com/catalog/9780596804787*)" analyzes mortgage foreclosures in Philadelphia County by taking a public report from the county sheriff's office, extracting addresses and using Yahoo to convert the addresses to latitude and longitude, then using the geographical data to place the foreclosures on a map (another data source), and group them by neighborhood, valuation, neighborhood per-capita income, and other socio-economic factors.

The question facing every company today, every startup, every nonprofit, every project site that wants to attract a community, is how to use data effectively and not just their own data, but all the data that's available and relevant. Using data effectively requires something different from traditional statistics, where actuaries in business suits perform arcane but fairly well-defined kinds of analysis. What differentiates data science from statistics is that data science is a holistic approach. We're increasingly finding data in the wild, and data scientists are involved with gathering data, massaging it into a tractable form, making it tell its story, and presenting that story to others.

To get a sense for what skills are required, let's look at the data lifecycle: where it comes from, how you use it, and where it goes.

Where data comes from

Data is everywhere: your government, your web server, your business partners, even your body (*http://www.nytimes.com/2010/05/02/magazine/02self-measurement-t.html?ref=magazine*). While we aren't drowning in a sea of data, we're finding that almost everything can (or has) been instrumented. At O'Reilly, we frequently combine publishing industry data from Nielsen BookScan (*http://en.wikipedia.org/wiki/Nielsen_BookScan*) with our own sales data, publicly available Amazon data, and even job data to see what's happening in the publishing industry. Sites like Infochimps (*http://infochimps.org/*) and Factual (*http://www.factual.com/*) provide access to

many large datasets, including climate data, MySpace activity streams, and game logs from sporting events. Factual enlists users to update and improve its datasets, which cover topics as diverse as endocrinologists to hiking trails.

1956 disk drive



One of the first commercial disk drives from IBM. It has a 5 MB capacity and it's stored in a cabinet roughly the size of a luxury refrigerator. In contrast, a 32 GB microSD card measures around 5/8 x 3/8 inch and weighs about 0.5 gram.

Photo: Mike Loukides. Disk drive on display at IBM Almaden Research (*http://www.almaden.ibm.com/*)

Much of the data we currently work with is the direct consequence of Web 2.0, and of Moore's Law applied to data. The web has people

www.dbooks.org

spending more time online, and leaving a trail of data wherever they go. Mobile applications leave an even richer data trail, since many of them are annotated with geolocation, or involve video or audio, all of which can be mined. Point-of-sale devices and frequent-shopper's cards make it possible to capture all of your retail transactions, not just the ones you make online. All of this data would be useless if we couldn't store it, and that's where Moore's Law comes in. Since the early '80s, processor speed has increased from 10 MHz (*http://en.wikipedia.org/wiki/Motorola_68000*) to 3.6GHz—an increase of 360 (not counting increases in word length and number of cores). But we've seen much bigger increases in storage capacity, on every level. RAM has moved from $1,000/MB to roughly $25/GB—a price reduction of about 40000, to say nothing of the reduction insize and increase in speed. Hitachi made the first gigabyte disk drives (*http://news.cnet.com/2300-1010_3-6031405-6.html*) in 1982, weighing in at roughly 250 pounds; now terabyte drives are consumer equipment, and a 32 GB microSD card weighs about half a gram. Whether you look at bits per gram, bits per dollar, or raw capacity, storage has more than kept pace with the increase of CPU speed.

The importance of Moore's law as applied to data isn't just geek pyrotechnics. Data expands to fill the space you have to store it. The more storage is available, the more data you will find to put into it. The data exhaust you leave behind whenever you surf the web, friend someone on Facebook, or make a purchase in your local supermarket, is all carefully collected and analyzed. Increased storage capacity demands increased sophistication in the analysis and use of that data. That's the foundation of data science.

So, how do we make that data useful? The first step of any data analysis project is "data conditioning," or getting data into a state where it's usable. We are seeing more data in formats that are easier to consume: Atom data feeds, web services, microformats, and other newer technologies provide data in formats that's directly machine-consumable. But old-style screen scraping (*http://en.wikipedia.org/wiki/Data_scraping#Screen_scraping*) hasn't died, and isn't going to die. Many sources of "wild data" are extremely messy. They aren't well-behaved XML files with all the metadata nicely in place. The foreclosure data used in "Data Mashups in R (*http://oreilly.com/catalog/9780596804787id=hni2*)" was posted on a public website by the Philadelphia county sheriff's office. This data was presented as an HTML file that was probably generated automatically from a

spreadsheet. If you've ever seen the HTML that's generated by Excel, you know that's going to be fun to process.

Data conditioning can involve cleaning up messy HTML with tools like Beautiful Soup (*http://www.crummy.com/software/BeautifulSoup/*), natural language processing to parse plain text in English and other languages, or even getting humans to do the dirty work. You're likely to be dealing with an array of data sources, all in different forms. It would be nice if there was a standard set of tools to do the job, but there isn't. To do data conditioning, you have to be ready for whatever comes, and be willing to use anything from ancient Unix utilities such as awk (*http://oreilly.com/catalog/9780596000707*) to XML parsers and machine learning libraries. Scripting languages, such as Perl (*http://oreilly.com/perl/*) and Python (*http://oreilly.com/python/*), are essential.

Once you've parsed the data, you can start thinking about the quality of your data. Data is frequently missing or incongruous. If data is missing, do you simply ignore the missing points? That isn't always possible. If data is incongruous, do you decide that something is wrong with badly behaved data (after all, equipment fails), or that the incongruous data is telling its own story, which may be more interesting? It's reported that the discovery of ozone layer depletion was delayed because automated data collection tools discarded readings that were too low (*http://www.nas.nasa.gov/About/Education/Ozone/history.html*).[1] In data science, what you have is frequently all you're going to get. It's usually impossible to get "better" data, and you have no alternative but to work with the data at hand.

If the problem involves human language, understanding the data adds another dimension to the problem. Roger Magoulas, who runs the data analysis group at O'Reilly, was recently searching a database for Apple job listings requiring geolocation skills. While that sounds like a simple task, the trick was disambiguating "Apple" from many job postings in the growing Apple industry. To do it well you need to understand the grammatical structure of a job posting; you need to be able to parse the English. And that problem is showing up more and more frequently. Try using Google Trends (*http://google.com/trends*) to figure out what's happening with the Cassandra (*http://*

1 The NASA article denies this, but also says that in 1984, they decided that the low values (which went back to the 70s) were "real." Whether humans or software decided to ignore anomalous data, it appears that data was ignored.

www.dbooks.org

google.com/trends?q=Cassandra) database or the Python (*http://google.com/trends?q=Python*) language, and you'll get a sense of the problem. Google has indexed many, many websites about large snakes. Disambiguation is never an easy task, but tools like the Natural Language Toolkit (*http://www.nltk.org/*) library can make it simpler.

When natural language processing fails, you can replace artificial intelligence with human intelligence. That's where services like Amazon's Mechanical Turk (*https://www.mturk.com/mturk/welcomeid=k3la*) come in. If you can split your task up into a large number of subtasks that are easily described, you can use Mechanical Turk's marketplace for cheap labor. For example, if you're looking at job listings, and want to know which originated with Apple, you can have real people do the classification for roughly $0.01 each. If you have already reduced the set to 10,000 postings with the word "Apple," paying humans $0.01 to classify them only costs $100.

Working with data at scale

We've all heard a lot about "big data," but "big" is really a red herring. Oil companies, telecommunications companies, and other data-centric industries have had huge datasets for a long time. And as storage capacity continues to expand, today's "big" is certainly tomorrow's "medium" and next week's "small." The most meaningful definition I've heard: *"big data" is when the size of the data itself becomes part of the problem*. We're discussing data problems ranging from gigabytes to petabytes of data. At some point, traditional techniques for working with data run out of steam.

What are we trying to do with data that's different? According to Jeff Hammerbacher[2] (@hackingdata (*http://twitter.com/hackingdata*)), we're trying to build information platforms or dataspaces. Information platforms are similar to traditional data warehouses, but different. They expose rich APIs, and are designed for exploring and understanding the data rather than for traditional analysis and reporting. They accept all data formats, including the most messy, and their schemas evolve as the understanding of the data changes.

2 "Information Platforms as Dataspaces," by Jeff Hammerbacher (in Beautiful Data)

Most of the organizations that have built data platforms have found it necessary to go beyond the relational database model. Traditional relational database systems stop being effective at this scale. Managing sharding and replication across a horde of database servers is difficult and slow. The need to define a schema in advance conflicts with reality of multiple, unstructured data sources, in which you may not know what's important until after you've analyzed the data. Relational databases are designed for consistency, to support complex transactions that can easily be rolled back if any one of a complex set of operations fails. While rock-solid consistency is crucial to many applications, it's not really necessary for the kind of analysis we're discussing here. Do you really care if you have 1,010 or 1,012 Twitter followers? Precision has an allure, but in most data-driven applications outside of finance, that allure is deceptive. Most data analysis is comparative: if you're asking whether sales to Northern Europe are increasing faster than sales to Southern Europe, you aren't concerned about the difference between 5.92 percent annual growth and 5.93 percent.

To store huge datasets effectively, we've seen a new breed of databases appear. These are frequently called NoSQL databases, or Non-Relational databases, though neither term is very useful. They group together fundamentally dissimilar products by telling you what they aren't. Many of these databases are the logical descendants of Google's BigTable (*http://labs.google.com/papers/bigtable.html*) and Amazon's Dynamo (*http://www.allthingsdistributed.com/2007/10/amazons_dynamo.html*), and are designed to be distributed across many nodes, to provide "eventual consistency" but not absolute consistency, and to have very flexible schema. While there are two dozen or so products available (almost all of them open source), a few leaders have established themselves:

1. Cassandra (*http://cassandra.apache.org/*): Developed at Facebook, in production use at Twitter, Rackspace, Reddit, and other large sites. Cassandra is designed for high performance, reliability, and automatic replication. It has a very flexible data model. A new startup, Riptano (*http://www.riptano.com/*), provides commercial support.
2. HBase (*http://hadoop.apache.org/hbase/*): Part of the Apache Hadoop project, and modelled on Google's BigTable. Suitable for extremely large databases (billions of rows, millions of columns), distributed across thousands of nodes. Along with

www.dbooks.org

Hadoop, commercial support is provided by Cloudera (*http://www.cloudera.com/*).

Storing data is only part of building a data platform, though. Data is only useful if you can do something with it, and enormous datasets present computational problems. Google popularized the MapReduce (*http://labs.google.com/papers/mapreduce.html*) approach, which is basically a divide-and-conquer strategy for distributing an extremely large problem across an extremely large computing cluster. In the "map" stage, a programming task is divided into a number of identical subtasks, which are then distributed across many processors; the intermediate results are then combined by a single reduce task. In hindsight, MapReduce seems like an obvious solution to Google's biggest problem, creating large searches. It's easy to distribute a search across thousands of processors, and then combine the results into a single set of answers. What's less obvious is that MapReduce has proven to be widely applicable to many large data problems, ranging from search to machine learning.

The most popular open source implementation of MapReduce is the Hadoop project (*http://hadoop.apache.org/*). Yahoo's claim that they had built the world's largest production Hadoop application (*http://developer.yahoo.net/blogs/hadoop/2008/02/yahoo-worlds-largest-production-hadoop.html*), with 10,000 cores running Linux, brought it onto center stage. Many of the key Hadoop developers have found a home at Cloudera (*http://www.cloudera.com/*), which provides commercial support. Amazon's Elastic MapReduce (*http://aws.amazon.com/elasticmapreduce/*) makes it much easier to put Hadoop to work without investing in racks of Linux machines, by providing preconfigured Hadoop images for its EC2 clusters. You can allocate and de-allocate processors as needed, paying only for the time you use them.

Hadoop (*http://oreilly.com/catalog/9780596521981*) goes far beyond a simple MapReduce implementation (of which there are several); it's the key component of a data platform. It incorporates HDFS (*http://hadoop.apache.org/hdfs/*), a distributed filesystem designed for the performance and reliability requirements of huge datasets; the HBase database; Hive (*http://hadoop.apache.org/hive/*), which lets developers explore Hadoop datasets using SQL-like queries; a high-level dataflow language called Pig (*http://hadoop.apache.org/*

pig/); and other components. If anything can be called a one-stop information platform, Hadoop is it.

Hadoop has been instrumental in enabling "agile" data analysis. In software development, "agile practices" are associated with faster product cycles, closer interaction between developers and consumers, and testing. Traditional data analysis has been hampered by extremely long turn-around times. If you start a calculation, it might not finish for hours, or even days. But Hadoop (and particularly Elastic MapReduce) make it easy to build clusters that can perform computations on long datasets quickly. Faster computations make it easier to test different assumptions, different datasets, and different algorithms. It's easer to consult with clients to figure out whether you're asking the right questions, and it's possible to pursue intriguing possibilities that you'd otherwise have to drop for lack of time.

Hadoop is essentially a batch system, but Hadoop Online Prototype (HOP) (*http://code.google.com/p/hop/*) is an experimental project that enables stream processing. Hadoop processes data as it arrives, and delivers intermediate results in (near) real-time. Near real-time data analysis enables features like trending topics (*http://search.twitter.com/*) on sites like Twitter (*http://twitter.com*). These features only require soft real-time; reports on trending topics don't require millisecond accuracy. As with the number of followers on Twitter, a "trending topics" report only needs to be current to within five minutes—or even an hour. According to Hilary Mason (@hmason (*http://twitter.com/hmason*)), data scientist at bit.ly, it's possible to precompute much of the calculation, then use one of the experiments in real-time MapReduce to get presentable results.

Machine learning is another essential tool for the data scientist. We now expect web and mobile applications to incorporate recommendation engines, and building a recommendation engine is a quintessential artificial intelligence problem. You don't have to look at many modern web applications to see classification, error detection, image matching (behind Google Goggles (*http://www.google.com/mobile/goggles/*) and SnapTell (*http://www.snaptell.com/*)) and even face detection—an ill-advised mobile application lets you take someone's picture with a cell phone, and look up that person's identity using photos available online. Andrew Ng's Machine Learning course (*http://www.stanford.edu/class/cs229/*) is one of the most popular courses in computer science at Stanford,

www.dbooks.org

with hundreds of students (this video is highly recommended (*http://www.youtube.com/watch?v=UzxYlbK2c7Eid=j0ha*)).

There are many libraries available for machine learning: PyBrain (*http://pybrain.org/*) in Python, Elefant (*http://elefant.developer.nicta.com.au/*), Weka (*http://www.cs.waikato.ac.nz/ml/weka/*) in Java, and Mahout (*http://lucene.apache.org/mahout/*) (coupled to Hadoop). Google has just announced their Prediction API (*http://code.google.com/apis/predict/*), which exposes their machine learning algorithms for public use via a RESTful interface. For computer vision, the OpenCV (*http://opencv.willowgarage.com/wiki/*) library is a de-facto standard.

Mechanical Turk (*https://www.mturk.com/mturk/welcomeid=k3la*) is also an important part of the toolbox. Machine learning almost always requires a "training set," or a significant body of known data with which to develop and tune the application. The Turk is an excellent way to develop training sets. Once you've collected your training data (perhaps a large collection of public photos from Twitter), you can have humans classify them inexpensively—possibly sorting them into categories, possibly drawing circles around faces, cars, or whatever interests you. It's an excellent way to classify a few thousand data points at a cost of a few cents each. Even a relatively large job only costs a few hundred dollars.

While I haven't stressed traditional statistics, building statistical models plays an important role in any data analysis. According to Mike Driscoll (*http://www.dataspora.com/*) (@dataspora (*http://twitter.com/dataspora*)), statistics is the "grammar of data science." It is crucial to "making data speak coherently." We've all heard the joke that eating pickles causes death, because everyone who dies has eaten pickles. That joke doesn't work if you understand what correlation means. More to the point, it's easy to notice that one advertisement for *R in a Nutshell (http://oreilly.com/catalog/9780596801717/)* generated 2 percent more conversions than another. But it takes statistics to know whether this difference is significant, or just a random fluctuation. Data science isn't just about the existence of data, or making guesses about what that data might mean; it's about testing hypotheses and making sure that the conclusions you're drawing from the data are valid. Statistics plays a role in everything from traditional business intelligence (BI) to understanding how Google's ad auctions work. Statistics has become a

basic skill. It isn't superseded by newer techniques from machine learning and other disciplines; it complements them.

While there are many commercial statistical packages, the open source R language (*http://www.r-project.org/*) — and its comprehensive package library, CRAN (*http://cran.r-project.org/*) — is an essential tool. Although R is an odd and quirky language, particularly to someone with a background in computer science, it comes close to providing "one stop shopping" for most statistical work. It has excellent graphics facilities; CRAN includes parsers for many kinds of data; and newer extensions extend R into distributed computing. If there's a single tool that provides an end-to-end solution for statistics work, R is it.

Making data tell its story

A picture may or may not be worth a thousand words, but a picture is certainly worth a thousand numbers. The problem with most data analysis algorithms is that they generate a set of numbers. To understand what the numbers mean, the stories they are really telling, you need to generate a graph. Edward Tufte's Visual Display of Quantitative Information (*http://www.amazon.com/Visual-Display-Quantitative-Information-2nd/dp/0961392142/*) is the classic for data visualization, and a foundational text for anyone practicing data science. But that's not really what concerns us here. Visualization is crucial to each stage of the data scientist. According to Martin Wattenberg (@wattenberg (*http://twitter.com/wattenberg*), founder of Flowing Media) (*http://flowingmedia.com/id=iu13*), visualization is key to data conditioning: if you want to find out just how bad your data is, try plotting it. Visualization is also frequently the first step in analysis. Hilary Mason says that when she gets a new data set, she starts by making a dozen or more scatter plots, trying to get a sense of what might be interesting. Once you've gotten some hints at what the data might be saying, you can follow it up with more detailed analysis.

There are many packages for plotting and presenting data. GnuPlot (*http://www.gnuplot.info/*) is very effective; R incorporates a fairly comprehensive graphics package; Casey Reas' and Ben Fry's Processing (*http://processing.org/*) is the state of the art, particularly if you need to create animations that show how things change over time. At IBM's Many Eyes (*http://manyeyes.alphaworks.ibm.com/*

www.dbooks.org

manyeyes/), many of the visualizations are full-fledged interactive applications.

Nathan Yau's FlowingData (*http://flowingdata.com/*) blog is a great place to look for creative visualizations. One of my favorites is this animation of the growth of Walmart (*http://flowingdata.com/2010/04/07/watching-the-growth-of-walmart-now-with-100-more-sams-club/*) over time. And this is one place where "art" comes in: not just the aesthetics of the visualization itself, but how you understand it. Does it look like the spread of cancer throughout a body? Or the spread of a flu virus through a population? Making data tell its story isn't just a matter of presenting results; it involves making connections, then going back to other data sources to verify them. Does a successful retail chain spread like an epidemic, and if so, does that give us new insights into how economies work? That's not a question we could even have asked a few years ago. There was insufficient computing power, the data was all locked up in proprietary sources, and the tools for working with the data were insufficient. It's the kind of question we now ask routinely.

Data scientists

Data science requires skills ranging from traditional computer science to mathematics to art. Describing the data science group he put together at Facebook (possibly the first data science group at a consumer-oriented web property), Jeff Hammerbacher said:

> ... on any given day, a team member could author a multistage processing pipeline in Python, design a hypothesis test, perform a regression analysis over data samples with R, design and implement an algorithm for some data-intensive product or service in Hadoop, or communicate the results of our analyses to other members of the organization.[3]

Where do you find the people this versatile? According to DJ Patil, chief scientist at LinkedIn (*http://www.linkedin.com/*) (@dpatil (*http://twitter.com/dpatil*)), the best data scientists tend to be "hard scientists," particularly physicists, rather than computer science majors. Physicists have a strong mathematical background, computing skills, and come from a discipline in which survival depends on getting the most from the data. They have to think about the big pic-

3 "Information Platforms as Dataspaces," by Jeff Hammerbacher (in Beautiful Data)

ture, the big problem. When you've just spent a lot of grant money generating data, you can't just throw the data out if it isn't as clean as you'd like. You have to make it tell its story. You need some creativity for when the story the data is telling isn't what you think it's telling.

Scientists also know how to break large problems up into smaller problems. Patil described the process of creating the group recommendation feature at LinkedIn. It would have been easy to turn this into a high-ceremony development project that would take thousands of hours of developer time, plus thousands of hours of computing time to do massive correlations across LinkedIn's membership. But the process worked quite differently: it started out with a relatively small, simple program that looked at members' profiles and made recommendations accordingly. Asking things like, did you go to Cornell? Then you might like to join the Cornell Alumni group. It then branched out incrementally. In addition to looking at profiles, LinkedIn's data scientists started looking at events that members attended. Then at books members had in their libraries. The result was a valuable data product that analyzed a huge database—but it was never conceived as such. It started small, and added value iteratively. It was an agile, flexible process that built toward its goal incrementally, rather than tackling a huge mountain of data all at once.

Hiring trends for data science



It's not easy to get a handle on jobs in data science. However, data from O'Reilly Research (http://radar.oreilly.com/research/) shows a steady year-over-year increase in Hadoop and Cassandra job listings, which are good proxies for the "data science" market as a whole. This

www.dbooks.org

graph shows the increase in Cassandra jobs, and the companies listing Cassandra positions, over time.

This is the heart of what Patil calls "data jiujitsu"—using smaller auxiliary problems to solve a large, difficult problem that appears intractable. CDDB is a great example of data jiujitsu: identifying music by analyzing an audio stream directly is a very difficult problem (though not unsolvable—see midomi (*http://www.midomi.com/*), for example). But the CDDB staff used data creatively to solve a much more tractable problem that gave them the same result. Computing a signature based on track lengths, and then looking up that signature in a database, is trivially simple.

Entrepreneurship is another piece of the puzzle. Patil's first flippant answer to "what kind of person are you looking for when you hire a data scientist?" was "someone you would start a company with." That's an important insight: we're entering the era of products that are built on data. We don't yet know what those products are, but we do know that the winners will be the people, and the companies, that find those products. Hilary Mason came to the same conclusion. Her job as scientist at bit.ly is really to investigate the data that bit.ly is generating, and find out how to build interesting products from it. No one in the nascent data industry is trying to build the 2012 Nissan Stanza or Office 2015; they're all trying to find new products. In addition to being physicists, mathematicians, programmers, and artists, they're entrepreneurs.

Data scientists combine entrepreneurship with patience, the willingness to build data products incrementally, the ability to explore, and the ability to iterate over a solution. They are inherently interdiscplinary. They can tackle all aspects of a problem, from initial data collection and data conditioning to drawing conclusions. They can think outside the box to come up with new ways to view the problem, or to work with very broadly defined problems: "here's a lot of data, what can you make from it?"

The future belongs to the companies who figure out how to collect and use data successfully. Google, Amazon, Facebook, and LinkedIn have all tapped into their datastreams and made that the core of their success. They were the vanguard, but newer companies like bit.ly are following their path. Whether it's mining your personal biology, building maps from the shared experience of millions of

travellers, or studying the URLs that people pass to others, the next generation of successful businesses will be built around data. The part of Hal Varian's quote that nobody remembers says it all (*http://www.mckinseyquarterly.com/Hal_Varian_on_how_the_Web_challenges_managers_2286*):

> **The ability to take data—to be able to understand it, to process it, to extract value from it, to visualize it, to communicate it—that's going to be a hugely important skill in the next decades.**

Data is indeed the new Intel Inside.

www.dbooks.org