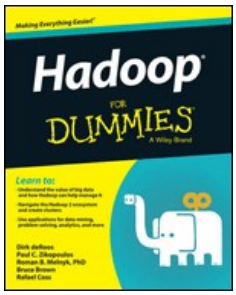

Hadoop for Dummies

by Dirk deRoos et al.

Chapter 2: Common Use Cases for Big Data in Hadoop

In This Chapter

- Extracting business value from Hadoop
- Digging into log data
- Moving the (data) warehouse into the 21st century
- Taking a bite out of fraud
- Modeling risk
- Seeing what's causing a social media stir
- Classifying images on a massive scale
- Using graphs effectively
- Looking toward the future

By writing this book, we want to help our readers answer the questions "What is Hadoop?" and "How do I use Hadoop?" Before we delve too deeply into the answers to these questions, though, we want to get you excited about some of the tasks that Hadoop excels at. In other words, we want to provide answers to the eternal question "What should I use Hadoop for?" In this chapter, we cover some of the most popular use cases we've seen in the Hadoop space, but first we have a couple thoughts on how you can make your Hadoop project successful.

The Keys to Successfully Adopting Hadoop (Or, "Please, Can We Keep Him?")

We strongly encourage you *not* to go looking for a "science project" when you're getting started with Hadoop. By that, we mean that you shouldn't try to find an open-ended problem that, despite being interesting, has neither clearly defined milestones nor measurable business value. We've seen some shops set up nifty, 100-node Hadoop clusters, but all that effort did little or nothing to add value to their businesses (though its implementers still seemed proud of themselves). Businesses want to see value from their IT investments, and with Hadoop it may come in a variety of ways. For example, you may pursue a project whose goal is to create lower licensing and storage costs for warehouse data or to find insight from large-scale data analysis. The best way to request resources to fund interesting Hadoop projects is by working with your business's leaders. In any serious Hadoop project, you should start by teaming IT with business leaders from VPs on down to help solve your business's *pain points* — those problems (real or perceived) that loom large in everyone's mind.

Also examine the perspectives of people and processes that are adopting Hadoop in your organization. Hadoop deployments tend to be most successful when adopters make the effort to create a culture that's supportive of data science by fostering experimentation and data exploration. Quite simply, after you've created a Hadoop cluster, you still have work to do — you still need to enable people to experiment in a hands-on manner. Practically speaking, you should keep an eye on these three important goals:

- **Ensure that your business users and analysts have access to as much data as possible**. Of course, you still have to respect regulatory requirements for criteria such as data privacy.
- **Mandate that your Hadoop developers expose their logic so that results are accessible through standard tools in your organization**. The logic and any results must remain easily consumed and reusable.
- **Recognize the governance requirements for the data you plan to store in Hadoop**. Any data under governance control in a relational database management system (RDBMS) also needs to be under the same controls in Hadoop. After all, personally identifiable information has the same privacy requirements no matter where it's stored. Quite simply, you should ensure that you can pass a data audit for both RDBMS and Hadoop!

All the uses cases we cover in this chapter have Hadoop at their core, but it's when you combine it with the broader business and its repositories like databases and document stores that you can build a more complete picture of what's happening in your business. For example, social sentiment analysis performed in Hadoop might alert you to *what* people are saying, but do you know *why* they're saying it? This concept requires thinking beyond Hadoop and linking your company's systems of record (sales, for example) with its systems of engagement (like call center records — the data where you may draw the sentiment from).

Log Data Analysis

Log analysis is a common use case for an inaugural Hadoop project. Indeed, the earliest uses of Hadoop were for the large-scale analysis of *clickstream* logs — logs that record data about the web pages that people visit and in which order they visit them. We often refer to all the logs of data generated by your IT infrastructure as *data exhaust*. A log is a by-product of a functioning server, much like smoke coming from a working engine's exhaust pipe. Data exhaust has the connotation of pollution or waste, and many enterprises undoubtedly approach this kind of data with that thought in mind. Log data often grows quickly, and because of the high volumes produced, it can be tedious to analyze. And, the potential value of this data is often unclear. So the temptation in IT departments is to store this log data for as little time as reasonably possible. (After all, it costs money to retain data, and if there's no perceived business value, why store it?) But Hadoop changes the math: The cost of storing data is comparatively inexpensive, and Hadoop was originally developed especially for the large-scale batch processing of log

data.

TIP The log data analysis use case is a useful place to start your Hadoop journey because the chances are good that the data you work with is being deleted, or "dropped to the floor." We've worked with companies that consistently record a terabyte (TB) or more of customer web activity per week, only to discard the data with no analysis (which makes you wonder why they bothered to collect it). For getting started quickly, the data in this use case is likely easy to get and generally doesn't encompass the same issues you'll encounter if you start your Hadoop journey with other (governed) data.

When industry analysts discuss the rapidly increasing volumes of data that exist (4.1 exabytes as of 2014 — more than 4 million 1TB hard drives), log data accounts for much of this growth. And no wonder: Almost every aspect of life now results in the generation of data. A smartphone can generate hundreds of log entries per day for an active user, tracking not only voice, text, and data transfer but also geolocation data. Most households now have smart meters that log their electricity use. Newer cars have thousands of sensors that record aspects of their condition and use. Every click and mouse movement we make while browsing the Internet causes a cascade of log entries to be generated. Every time we buy something — even without using a credit card or debit card — systems record the activity in databases — and in logs. You can see some of the more common sources of log data: IT servers, web clickstreams, sensors, and transaction systems.

Every industry (as well as all the log types just described) have the huge potential for valuable analysis — especially when you can zero in on a specific kind of activity and then correlate your findings with another data set to provide context.

As an example, consider this typical web-based browsing and buying experience:

1. You surf the site, looking for items to buy.
2. You click to read descriptions of a product that catches your eye.
3. Eventually, you add an item to your shopping cart and proceed to the checkout (the buying action).

After seeing the cost of shipping, however, you decide that the item isn't worth the price and you close the browser window. Every click you've made — and then stopped making — has the potential to offer valuable insight to the company behind this e-commerce site.

In this example, assume that this business collects clickstream data (data about every mouse click and page view that a visitor "touches") with the aim of understanding how to better serve its customers. One common challenge among e-commerce businesses is to recognize the key factors behind abandoned shopping carts. When you perform deeper analysis on the clickstream data and examine user behavior on the site, patterns are bound to emerge.

Does your company know the answer to the seemingly simple question, "Are certain products abandoned more than others?" Or the answer to the question, "How much revenue can be recaptured if you decrease cart abandonment by 10 percent?" Figure 2-1 gives an example of the kind of reports you can show to your business leaders to seek their investment in your Hadoop cause.

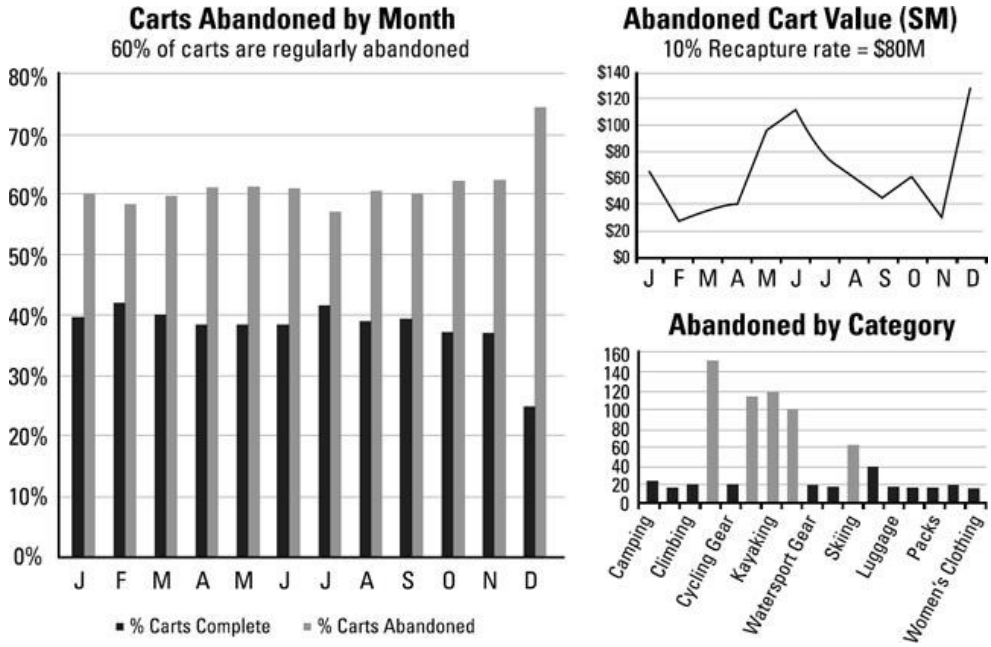

Figure 2-1: Reporting on abandoned carts

To get to the point where you can generate the data to build the graphs shown in Figure 2-1, you isolate the web browsing sessions of individual users (a process known as *sessionization)*, identify the contents of their shopping carts, and then establish the state of the transaction at the end of the session — all by examining the clickstream data.

In Figure 2-2, we give you an example of how to assemble users' web browsing sessions by grouping all clicks and URL addresses by IP address. (The example is a simple one in order to illustrate the point.) Remember: In a Hadoop context, you're always working with keys and values — each phase of MapReduce inputs and outputs data in sets of keys and values. (We discuss this in greater detail in Chapter 6.) In

Figure 2-2, the key is the IP address, and the value consists of the timestamp and the URL. During the map phase, user sessions are assembled in parallel for all file blocks of the clickstream data set that's stored in your Hadoop cluster.

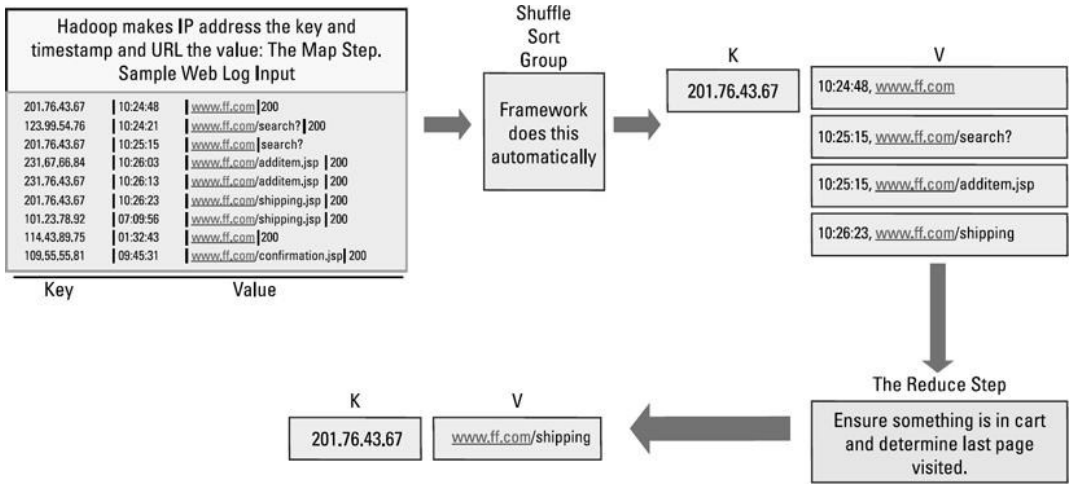

Figure 2-2: Building user sessions from clickstream log data and calculating the last page visited for sessions where a shopping cart is abandoned

The map phase returns these elements:

- The final page that's visited
- A list of items in the shopping cart
- The state of the transaction for each user session (indexed by the IP address key)

The reducer picks up these records and performs aggregations to total the number and value of carts abandoned per month and to provide totals of the most common final pages that someone viewed before ending the user session.

This single example illustrates why Hadoop is a great fit for analyzing log data. The range of possibilities is limitless, and by leveraging some of the simpler interfaces such as Pig and Hive, basic log analysis makes for a simple initial Hadoop project.

Data Warehouse Modernization

Data warehouses are now under stress, trying to cope with increased demands on their finite resources. The rapid rise in the amount of data generated in the world has also affected data warehouses because the volumes of data they manage are increasing — partly because more *structured* data — the kind of data that is strongly typed and slotted into rows and columns — is generated but also because you often have to deal with regulatory requirements designed to maintain queryable access to historical data. In addition, the processing power in data warehouses is often used to perform transformations of the relational data as it either enters the warehouse itself or is loaded into a *child data mart* (a separate subset of the data warehouse) for a specific analytics application. In addition, the need is increasing for analysts to issue new queries against the structured data stored in warehouses, and these ad hoc queries can often use significant data processing resources. Sometimes a one-time report may suffice, and sometimes an exploratory analysis is necessary to find questions that haven't been asked yet that may yield significant business value. The bottom line is that data warehouses are often being used for purposes beyond their original design.

Hadoop can provide significant relief in this situation. Figure 2-3 shows, using high-level architecture, how Hadoop can live alongside data warehouses and fulfill some of the purposes that they aren't designed for.

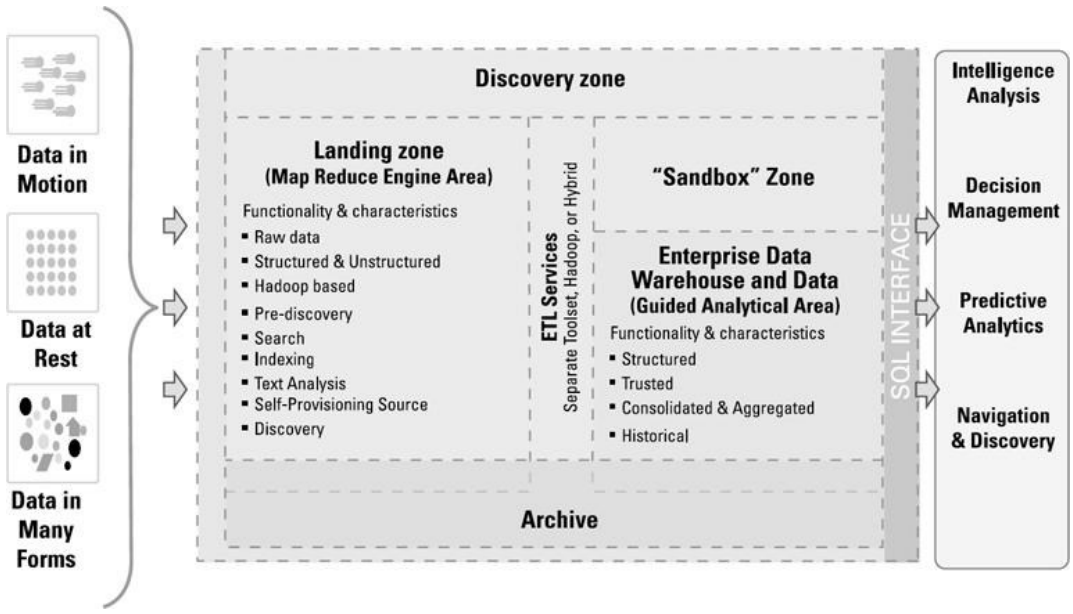

Figure 2-3: Using Hadoop to modernize a traditional relational data warehouse

Our view is that Hadoop is a warehouse *helper*, not a warehouse replacement. Later, in Chapter 11, we describe four ways that Hadoop can modernize a data warehousing ecosystem, here they are in summary:

- Provide a landing zone for all data.
- Persist the data to provide a queryable archive of cold data.
- Leverage Hadoop's large-scale batch processing efficiencies to preprocess and transform data for the warehouse.
- Enable an environment for ad hoc data discovery.

Fraud Detection

Fraud is a major concern across all industries. You name the industry (banking, insurance, government, health care, or retail, for example) and you'll find fraud. At the same time, you'll find folks who are willing to invest an incredible amount of time and money to try to prevent fraud. After all, if fraud were easy to detect, there wouldn't be so much investment around it. In today's interconnected world, the sheer volume and complexity of transactions makes it harder than ever to find fraud. What used to be called "finding a needle in a haystack" has become the task of "finding a specific needle in stacks of needles." Though the sheer volume of transactions makes it harder to spot fraud because of the volume of data, ironically, this same challenge can help create better fraud predictive models — an area where Hadoop shines. (We tell you more about statistical analysis in Chapter 9.)

Traditional approaches to fraud prevention aren't particularly efficient. For example, the management of improper payments is often managed by analysts auditing what amounts to a very small sample of claims paired with requesting medical documentation from targeted submitters. The industry term for this model is pay and chase: Claims are accepted and paid out and processes look for intentional or unintentional overpayments by way of post-payment review of those claims. (The U.S. Internal Revenue Service (IRS) operation uses the pay-and-chase approach on tax returns.)

Of course, you may wonder why businesses don't simply apply extra due diligence to every transaction proactively. They don't do so because it's a balancing act. Fraud detection can't focus only on stopping fraud when it happens, or on detecting it quickly, because of the customer satisfaction component. For example, traveling outside your home country and finding that your credit card has been invalidated because the transactions originated from a geographical location that doesn't match your purchase patterns can place you in a bad position, so vendors try to avoid false-positive results. They don't want to anger clients by stopping transactions that seem suspicious but turn out to be legitimate.

So how is fraud detection done now? Because of the limitations of traditional technologies, fraud models are built by sampling data and using the sample to build a set of fraud-prediction and -detection models. When you contrast this model with a Hadoop-anchored fraud department that uses the full data set — no sampling — to build out the models, you can see the difference.

The most common recurring theme you see across most Hadoop use cases is that it assists business in breaking through the glass ceiling on the volume and variety of data that can be incorporated into decision analytics. The more data you have (and the more history you store), the better your models can be.

Mixing nontraditional forms of data with your set of historical transactions can make your fraud models even more robust. For example, if a worker makes a worker's compensation claim for a bad back from a slip-and-fall incident, having a pool of millions of patient outcome cases that detail treatment and length of recovery helps create a detection pattern for fraud.

As an example of how this model can work, imagine trying to find out whether patients in rural areas recover more slowly than those in urban areas. You can start by examining the proximity to physiotherapy services. Is there a pattern correlation between recovery times and

geographical location? If your fraud department determines that a certain injury takes three weeks of recovery but that a farmer with the same diagnosis lives one hour from a physiotherapist and the office worker has a practitioner in her office, that's another variable to add to the fraud-detection pattern. When you harvest social network data for claimants and find a patient who claims to be suffering from whiplash is boasting about completing the rugged series of endurance events known as Tough Mudder, it's an example of mixing new kinds of data with traditional data forms to spot fraud.

If you want to kick your fraud-detection efforts into a higher gear, your organization can work to move away from market segment modeling and move toward at-transaction or at-person level modeling. Quite simply, making a forecast based on a segment is helpful, but making a decision based on particular information about an individual transaction is (obviously) better. To do this, you work up a larger set of data than is conventionally possible in the traditional approach. In our experiences with customers, we estimate that only (a maximum of) 30 percent of the available information that may be useful for fraud modeling is being used.

For creating fraud-detection models, Hadoop is well suited to

- **Handle volume**: That means processing the full data set — no data sampling.
- **Manage new varieties of data**: Examples are the inclusion of proximity-to-care-services and social circles to decorate the fraud model.
- **Maintain an agile environment**: Enable different kinds of analysis and changes to existing models.

Fraud modelers can add and test new variables to the model without having to make a proposal to your database administrator team and then wait a couple of weeks to approve a schema change and place it into their environment. This process is critical to fraud detection because dynamic environments commonly have cyclical fraud patterns that come and go in hours, days, or weeks. If the data used to identify or bolster new fraud-detection models isn't available at a moment's notice, by the time you discover these new patterns, it could be too late to prevent damage. Evaluate the benefit to your business of not only building out more comprehensive models with more types of data but also being able to refresh and enhance those models faster than ever. We'd bet that the company that can refresh and enhance models daily will fare better than those that do it quarterly.

You may believe that this problem has a simple answer — just ask your CIO for operational expenditure (OPEX) and capital expenditure (CAPEX) approvals to accommodate more data to make better models and load the other 70 percent of the data into your decision models. You may even believe that this investment will pay for itself with better fraud detection; however, the problem with this approach is the high up-front costs that need to be sunk into *unknown* data, where you don't know whether it contains any truly valuable insight. Sure, tripling the size of your data warehouse, for example, will give you more access to structured historical data to fine-tune your models, but they can't accommodate social media bursts. As we mention earlier in this chapter, traditional technologies aren't as agile, either. Hadoop makes it easy to introduce new variables into the model, and if they turn out not to yield improvements to the model, you can simply discard the data and move on.

Risk Modeling

Risk modeling is another major use case that's energized by Hadoop. We think you'll find that it closely matches the use case of fraud detection in that it's a model-based discipline. The more data you have and the more you can "connect the dots," the more often your results will yield better risk-prediction models.

The all-encompassing word *risk* can take on a lot of meanings. For example, customer churn prediction is the risk of a client moving to a competitor; the risk of a loan book relates to the risk of default; risk in health care spans the gamut from outbreak containment to food safety to the probability of reinfection and more.

The financial services sector (FSS) is now investing heavily in Hadoop-based risk modeling. This sector seeks to increase the automation and accuracy of its risk assessment and exposure modeling. Hadoop offers participants the opportunity to extend the data sets that are used in their risk models to include underutilized sources (or sources that are never utilized), such as e-mail, instant messaging, social media, and interactions with customer service representatives, among other data sources. Risk models in FSS pop up everywhere. They're used for customer churn prevention, trade manipulation modeling, corporate risk and exposure analytics, and more.

When a company issues an insurance policy against natural disasters at home, one challenge is clearly seeing how much money is potentially at risk. If the insurer fails to reserve money for possible payouts, regulators will intervene (the insurer doesn't want that); if the insurer puts too much money into its reserves to pay out future policy claims, they can't then invest your premium money and make a profit (the insurer doesn't want that, either). We know of companies that are "blind" to the risk they face because they have been unable to run an adequate amount of catastrophic simulations pertaining to variance in wind speed or precipitation rates (among other variables) as they relate to their exposure. Quite simply, these companies have difficulty stress-testing their risk models. The ability to fold in more data — for example, weather patterns or the ever-changing socioeconomic distribution of their client base — gives them a lot more insight and capability when it comes to building better risk models.

Building and stress-testing risk models like the one just described is an ideal task for Hadoop. These operations are often computationally expensive and, when you're building a risk model, likely impractical to run against a data warehouse, for these reasons:

- The warehouse probably isn't optimized for the kinds of queries issued by the risk model. (Hadoop isn't bound by the data models used in data warehouses.)
- A large, ad hoc batch job such as an evolving risk model would add load to the warehouse, influencing existing analytic applications. (Hadoop can assume this workload, freeing up the warehouse for regular business reporting.)
- More advanced risk models may need to factor in unstructured data, such as raw text. (Hadoop can handle that task efficiently.)

Social Sentiment Analysis

Social sentiment analysis is easily the most overhyped of the Hadoop use cases we present, which should be no surprise, given that we live in a world with a constantly connected and expressive population. This use case leverages content from forums, blogs, and other social media resources to develop a sense of what people are doing (for example, life events) and how they're reacting to the world around them (sentiment). Because text-based data doesn't naturally fit into a relational database, Hadoop is a practical place to explore and run analytics on this data.

Language is difficult to interpret, even for human beings at times — especially if you're reading text written by people in a social group that's different from your own. This group of people may be speaking your language, but their expressions and style are completely foreign, so you have no idea whether they're talking about a good experience or a bad one. For example, if you hear the word *bomb* in reference to a movie, it might mean that the movie was bad (or good, if you're part of the youth movement that interprets "It's da bomb" as a compliment); of course, if you're in the airline security business, the word *bomb* has quite a different meaning. The point is that language is used in many variable ways and is constantly evolving.

Social sentiment analysis is, in reality, text analysis

Though this section focuses on the "fun" aspects of using social media, the ability to extract understanding and meaning from unstructured text is an important use case. For example, corporate earnings are published to the web, and the same techniques that you use to build social sentiment extractors may be used to try to extract meaning from financial disclosures or to auto-assemble intrasegment earnings reports that compare the services revenue in a specific sector. In fact, some hedge fund management teams are now doing this to try to get a leg up on their competition.

Perhaps your entertainment company wants to crack down on violations of intellectual property on your event's video footage. You can use the same techniques outlined in this use case to extract textual clues from various web postings and teasers such as *Watch for free* or *Free on your PC*. You can use a library of custom-built text extractors (built and refined on data stored in Hadoop) to crawl the web to generate a list of links to pirated video feeds of your company's content.

These two examples don't demonstrate sentiment analysis; however, they do a good job of illustrating how social text analytics doesn't focus only on sentiment, despite the fun in illustrating the text analytics domain using sentiment analysis.

TECHNICAL STUFF When you analyze sentiment on social media, you can choose from multiple approaches. The basic method programmatically parses the text, extracts strings, and applies rules. In simple situations, this approach is reasonable. But as requirements evolve and rules become more complex, manually coding text-extractions quickly becomes no longer feasible from the perspective of code maintenance, especially for performance optimization. Grammar- and rules-based approaches to text processing are computationally expensive, which is an important consideration in large-scale extraction in Hadoop. The more involved the rules (which is inevitable for complex purposes such as sentiment extraction), the more processing that's needed.

Alternatively, a statistics-based approach is becoming increasingly common for sentiment analysis. Rather than manually write complex rules, you can use the classification-oriented machine-learning models in Apache Mahout. (See Chapter 9 for more on these models.) The catch here is that you'll need to train your models with examples of positive and negative sentiment. The more training data you provide (for example, text from tweets and your classification), the more accurate your results.

Like the other use cases in this chapter, the one for social sentiment analysis can be applied across a wide range of industries. For example, consider food safety: Trying to predict or identify the outbreak of foodborne illnesses as quickly as possible is extremely important to health officials. Figure 2-4 shows a Hadoop-anchored application that ingests tweets using extractors based on the potential illness: FLU or FOOD POISONING. (We've anonymized the tweets so that you don't send a message asking how they're doing; we didn't clean up the grammar, either.)

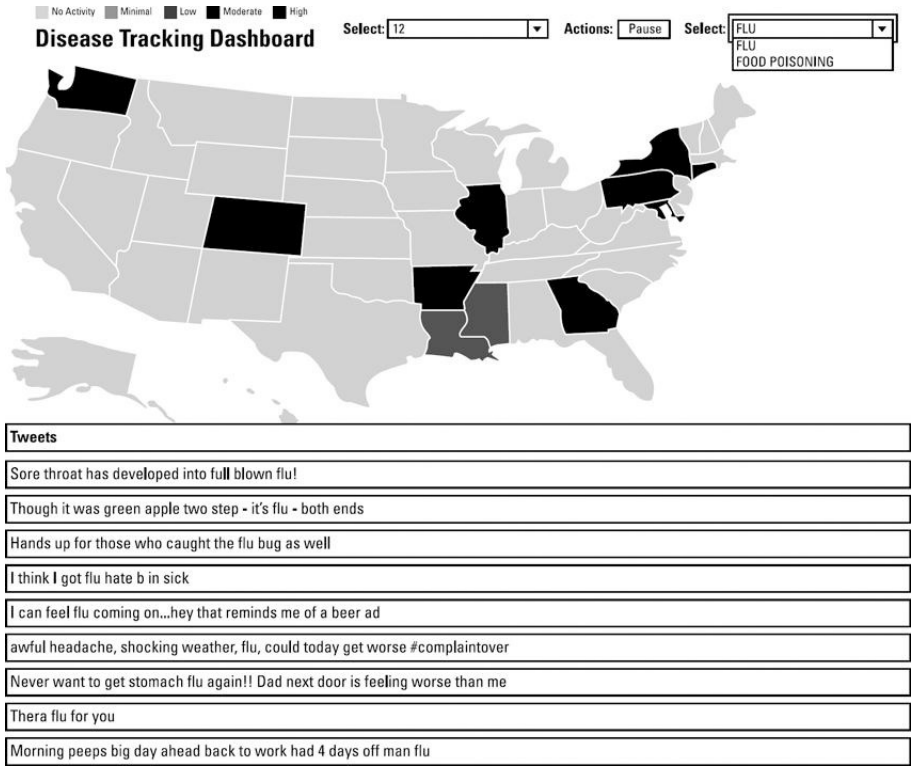

Figure 2-4: Using Hadoop to analyze and classify tweets in an attempt to classify a potential outbreak of the flu or food poisoning

Do you see the generated heat map that shows the geographical location of the tweets? One characteristic of data in a world of big data is that most of it is *spatially enriched*: It has locality information (and temporal attributes, too). In this case, we reverse-engineered the Twitter profile by looking up the published location. As it turns out, lots of Twitter accounts have geographic locations as part of their public profiles (as well as disclaimers clearly stating that their thoughts are their own as opposed to speaking for their employers).

How good of a prediction engine can social media be for the outbreak of the flu or a food poisoning incident? Consider the anonymized sample data shown in Figure 2-5.

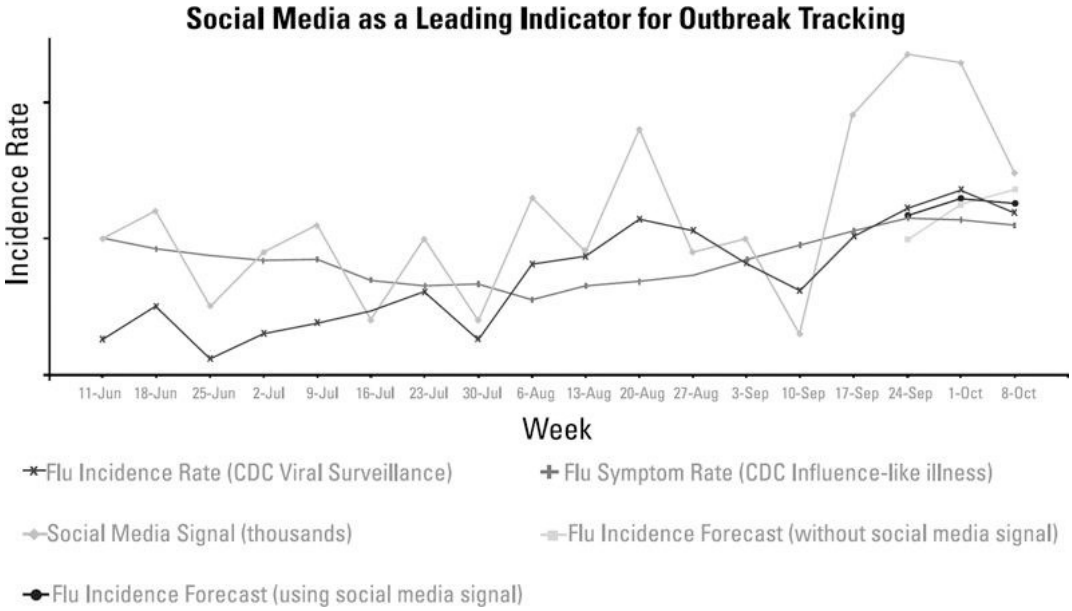

Figure 2-5: Chances are good that social media can tell you about a flu outbreak before traditional indicators can

You can see that social media signals trumped all other indicators for predicting a flu outbreak in a specific U.S. county during the late summer and into early fall.

This example shows another benefit that accrues from analyzing social media: It gives you an unprecedented opportunity to look at attribute information in posters' profiles. Granted, what people say about themselves in their Twitter profiles is often incomplete (for example, the location code isn't filled in) or not meaningful (the location code might say *cloud nine*). But you can learn a lot about people over time, based

on what they say. For example, a client may have *tweeted* (posted on Twitter) the announcement of the birth of her baby, an Instagram picture of her latest painting, or a Facebook posting stating that she can't believe Walter White's behavior in last night's *Breaking Bad* finale. (Now that many people watch TV series in their entirety, even long after they've ended, we wouldn't want to spoil the ending for you.) In this ubiquitous example, your company can extract a life event that populates a family-graph (a new child is a valuable update for a person-based Master Data Management profile), a hobby (painting), and an interest attribute (you love the show *Breaking Bad*). By analyzing social data in this way, you have the opportunity to flesh out personal attributes with information such as hobbies, birthdays, life events, geographical locations (country, state, and city, for example), employer, gender, marital status, and more.

Assume for a minute that you're the CIO of an airline. You can use the postings of happy or angry frequent travelers to not only ascertain sentiment but also round out customer profiles for your loyalty program using social media information. Imagine how much better you could target potential customers with the information that was just shared — for example, an e-mail telling the client that Season 5 of *Breaking Bad* is now available on the plane's media system or announcing that children under the age of two fly for free. It's also a good example of how systems of record (say, sales or subscription databases) can meet systems of engagement (say, support channels). Though the loyalty members' redemption and travel history is in a relational database, the system of engagement can update records (for example, a `HAS_KIDS` column).

Image Classification

Image classification starts with the notion that you build a training set and that computers learn to identify and classify what they're looking at. In the same way that having more data helps build better fraud detection and risk models, it also helps systems to better classify images. This requires a significant amount of data processing resources, however, which has limited the scale of deployments. Image classification is a hot topic in the Hadoop world because no mainstream technology was capable — until Hadoop came along — of opening doors for this kind of expensive processing on such a massive and efficient scale.

In this use case, the data is referred to as the training set as well as the models are classifiers. *Classifiers* recognize features or patterns within sound, image, or video and classify them appropriately. Classifiers are built and iteratively refined from training sets so that their precision scores (a measure of exactness) and recall scores (a measure of coverage) are high. Hadoop is well suited for image classification because it provides a massively parallel processing environment to not only create classifier models (iterating over training sets) but also provide nearly limitless scalability to process and run those classifiers across massive sets of unstructured data volumes. Consider multimedia sources such as YouTube, Facebook, Instagram, and Flickr — all are sources of unstructured binary data. Figure 2-6 shows one way you can use Hadoop to scale the processing of large volumes of stored images and video for multimedia semantic classification.

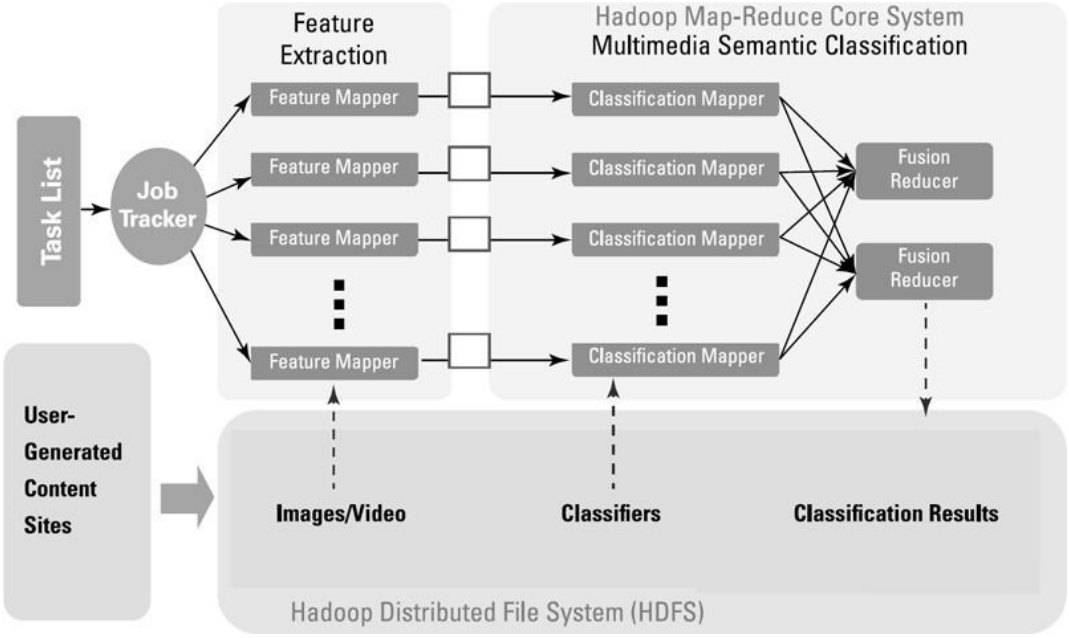

Figure 2-6: Using Hadoop to semantically classify video and images from social media sites

In Figure 2-6, you can see how all the concepts relating to the Hadoop processing framework that are outlined in this book are applied to this data. Notice how images are loaded into HDFS. The classifier models, built over time, are now applied to the extra image-feature components in the Map phase of this solution. As you can see in the lower-right corner of Figure 2-6, the output of this processing consists of image classifications that range from cartoons to sports and locations, among others.

TIP Though this section focuses on image analysis, Hadoop can be used for audio or voice analytics, too. One security industry client we work with creates an audio classification system to classify sounds that are heard via acoustic-enriched fiber optic cables laid around the perimeter of nuclear reactors. For example, this system knows how to nearly instantaneously classify the whisper of the wind as compared to the whisper of a human voice or to distinguish the sound of human footsteps running in the perimeter parklands from that of wildlife.

We realize that this description may have sort of a *Star Trek* feel to it, but you can now see live examples. In fact, IBM makes public one of the

largest image-classification systems in the world, via the IBM Multimedia Analysis and Retrieval System (IMARS). Try it for yourself at

http://researcher.watson.ibm.com/researcher/view_project. php?id=877

Figure 2-7 shows the result of an IMARS search for the term *alpine skiing*. At the top of the figure, you can see the results of the classifiers mapped to the image set that was processed by Hadoop, along with an associated tag cloud. Note the more coarsely defined parent classifier `Wintersports`, as opposed to the more granular `Sailing`. In fact, notice the multiple classification tiers: `Alpine_Skiing` rolls into `Snow_Sports`, which rolls into `Wintersports` — all generated automatically by the classifier model, built and scored using Hadoop.

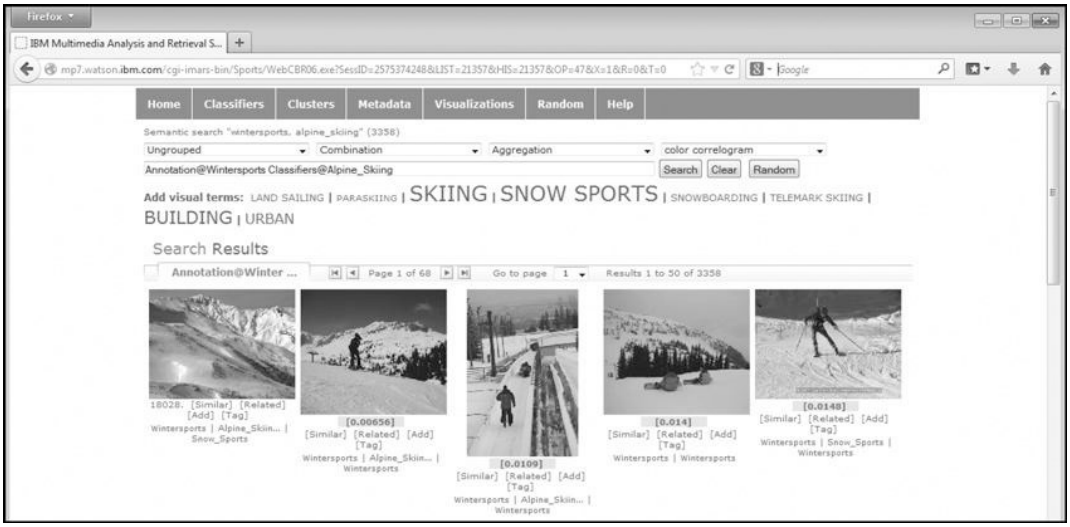

Figure 2-7: The result of an IMARS search

REMEMBER None of these pictures has any added metadata. No one has opened iPhoto and tagged an image as a winter sport to make it show up in this classification. It's the winter sport classifier that was built to recognize image attributes and characteristics of sports that are played in a winter setting.

Image classification has many applications, and being able to perform this classification at a massive scale using Hadoop opens up more possibilities for analysis as other applications can use the classification information generated for the images. To see what we mean, look at this example from the health industry. We worked with a large health agency in Asia that was focused on delivering health care via mobile clinics to a rural population distributed across a large land mass. A significant problem that the agency faced was the logistical challenge of analyzing the medical imaging data that was generated in its mobile clinics. A radiologist is a scarce resource in this part of the world, so it made sense to electronically transmit the medical images to a central point and have an army of doctors examine them. The doctors examining the images were quickly overloaded, however. The agency is now working on a classification system to help identify possible conditions to effectively provide suggestions for the doctors to verify. Early testing has shown this strategy to help reduce the number of missed or inaccurate diagnoses, saving time, money, and — most of all — lives.

Graph Analysis

Elsewhere in this chapter, we talk about log data, relational data, text data, and binary data, but you'll soon hear about another form of information: graph data. In its simplest form, a graph is simply a collection of nodes (an entity, for example — a person, a department, or a company), and the lines connecting them are edges (this represents a relationship between two entities, for example two people who know each other). What makes graphs interesting is that they can be used to represent concepts such as relationships in a much more efficient way than, say, a relational database. Social media is an application that immediately comes to mind — indeed, today's leading social networks (Facebook, Twitter, LinkedIn, and Pinterest) are all making heavy use of graph stores and processing engines to map the connections and relationships between their subscribers.

In Chapter 11, we discuss the NoSQL movement, and the graph database is one major category of alternative data-storage technologies. Initially, the predominant graph store was Neo4j, an open source graph database. But now the use of Apache Giraph, a graph processing engine designed to work in Hadoop, is increasing rapidly. Using YARN, we expect Giraph adoption to increase even more because graph processing is no longer tied to the traditional MapReduce model, which was inefficient for this purpose. Facebook is reportedly the world's largest Giraph shop, with a massive *trillion-edge* graph. (It's the Six Degrees of Kevin Bacon game on steroids.)

Graphs can represent any kind of relationship — not just people. One of the most common applications for graph processing now is mapping the Internet. When you think about it, a graph is the perfect way to store this kind of data, because the web itself is essentially a graph, where its websites are nodes and the hyperlinks between them are edges. Most PageRank algorithms use a form of graph processing to calculate the weightings of each page, which is a function of how many other pages point to it.

To Infinity and beyond

This chapter easily could have been expanded into an entire book — there are that many places where Hadoop is a game changer. Before you apply one of the use cases from this chapter to your own first project and start seeing how to use Hadoop in Chapter 3, we want to

reiterate some repeating patterns that we've noticed when organizations start taking advantage of the potential value of Hadoop:

- When you use more data, you can make better decisions and predictions and guide better outcomes.
- In cases where you need to retain data for regulatory purposes and provide a level of query access, Hadoop is a cost-effective solution.
- The more a business depends on new and valuable analytics that are discovered in Hadoop, the more it wants. When you initiate successful Hadoop projects, your clusters will find new purposes and grow!