O'REILLY®

Compliments of GitHub

Development Workflows for Data Scientists

Enabling Fast, Efficient, and Reproducible Results for Data Science Teams

Ciara Byrne

Development Workflows for Data Scientists

Ciara Byrne

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

Development Workflows for Data Scientists
by Ciara Byrne

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://oreilly.com/safari*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Editor: Marie Beaugureau
Production Editor: Shiny Kalapurakkel
Copyeditor: Octal Publishing, Inc.
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Rebecca Demarest

March 2017: First Edition

Revision History for the First Edition
2017-03-08: First Release

978-1-491-98330-0

[LSI]

Table of Contents

Foreword

The field of data science has taken all industries by storm. Data scientist positions are consistently in the top-ranked best job listings, and new job opportunities with titles like data engineer and data analyst are opening faster than they can be filled. The explosion of data collection and subsequent backlog of big data projects in every industry has lead to the situation in which "*we're drowning in data and starved for insight*."

To anyone who lived through the growth of software engineering in the previous two decades, this is a familiar scene. The imperative to maintain a competitive edge in software by rapidly delivering higher-quality products to market, led to a revolution in software development methods and tooling; it is the manifesto for Agile software development, Agile operations, DevOps, Continuous Integration, Continuous Delivery, and so on.

Much of the analysis performed by scientists in this fast-growing field occurs as software experimentation in languages like R and Python. This raises the question: *what can data science learn from software development?*

Ciara Byrne takes us on a journey through the data science and analytics teams of many different companies to answer this question. She leads us through their practices and priorities, their tools and techniques, and their capabilities and concerns. It's an illuminating journey that shows that even though the pace of change is rapid and the desire for the knowledge and insight from data is ever growing, the dual disciplines of software engineering and data science are up for the task.

— Compliments of GitHub

Development Workflows for Data Scientists

Engineers learn in order to build, whereas scientists build in order to learn, according to Fred Brooks, author of the software development classic *The Mythical Man Month*. It's no mistake that the term "data science" includes the word "science." In contrast with the work of engineers or software developers, the product of a data science project is not code; the product is useful insight.

"A data scientist has a very different relationship with code than a developer does," says Drew Conway, CEO of Alluvium and a coauthor of *Machine Learning for Hackers*. Conway continues:

> I look at code as a tool to go from the question I am interested in answering to having some insight. That code is more or less disposable. For developers, they are thinking about writing code to build into a larger system. They are thinking about how can I write something that can be reused?

However, data scientists often need to write code to arrive at useful insight, and that insight might be wrapped in code to make it easily consumable. As a result, data science teams have borrowed from software best practices to improve their own work. But which of those best practices are most relevant to data science? In what areas do data scientists need to develop new best practices? How have data science teams improved their workflows and what benefits have they seen? These are the questions this report addresses.

Many of the data scientists with whom I spoke said that software development best practices really become useful when you already have a good idea of what to build. At the beginning of a project, a

data scientist doesn't always know what that is. "Planning a data science project can be difficult because the scope of a project can be difficult to know ex ante," says Conway. "There is often a zero-step of exploratory data analysis or experimentation that must be done in order to know how to define the end of a project."

What's a Good Data Science Workflow?

A workflow is the definition, execution, and automation of business processes toward the goal of coordinating tasks and information between people and systems. In software development, standard processes like planning, development, testing, integration, and deployment, as well as the workflows that link them have evolved over decades. Data science is a young field so its processes are still in flux.

A good workflow for a particular team depends on the tasks, goals, and values of that team, whether they want to make their work faster, more efficient, correct, compliant, agile, transparent, or reproducible. A tradeoff often exists between different goals and values—do I want to get something done quickly or do I want to invest time now to make sure that it can be done quickly next time? I quizzed multiple data science teams about their reasons for defining, enforcing, and automating a workflow.

Produce Results Fast

The data science team at BinaryEdge, a Swiss cybersecurity firm that provides threat intelligence feeds or security reports based on internet data, wanted to create a rigorous, objective, and reproducible data science process. The team works with data that has an expiration date, so it wanted its workflow to produce initial results fast, and then allow a subsequent thorough analysis of the data while avoiding common pitfalls. Finally, the team is tasked with transmitting the resulting knowledge in the most useful ways possible.

The team also wanted to record all the steps taken to reach a particular result, even those that did not lead anywhere. Otherwise, time will be lost in future exploring avenues that have already proved to be dead ends. During the exploratory stage of a project, data scientists at the company create a codebook, recording all steps taken, tools used, data sources, and conclusions reached.

When BinaryEdge's team works with data in a familiar format (where the data structure is known *a priori*), most steps in its workflow are automated. When dealing with a new type of data, all of the steps of the workflow are initially followed manually to derive maximum knowledge from that data.

Reproduce and Reuse Results

Reproducibility is as basic a tenet of science as reuse is of software development, but both are often still an afterthought in data science projects. Airbnb has made a concerted effort to make previous data science work discoverable so that it can be reproduced and reused. The company defined a process to contribute and review data science work and created a tool called the *Knowledge Repo* to share that work well beyond the data science team.

Airbnb introduced a workflow specifically for data scientists to add new work to the Knowledge Repo and make it searchable. "We basically had to balance out short-term costs and long-term costs," says Nikki Ray, the core maintainer of the Knowledge Repo. Ray elaborates:

> Short-term, you're being asked to go through a specified format and going through an actual review cycle which is a little bit longer, but long-term you're answering less questions and your research is in one place where other people can find it in the future.

GitHub's machine learning team builds user-facing features using machine learning and big data techniques. Reuse is also one of the team's priorities. "We try to build on each other's work," says Ho-Hsiang Wu, a data scientist in the data product team. "We can go back and iterate on each model separately to improve that model."

Tools created to improve your data science workflow can also be reused. "It's easy to turn your existing code—whether it's written in Python, R, or Java—into a command-line tool so that you can reuse it and combine it with other tools," says Jeroen Janssens, founder of Data Science Workshops and author of *Data Science at the Command Line*. "Thanks to GitHub, it's easier than ever to share your tools with the rest of the world or find ones created by others."

Audit Results

In regulated industries like banking or healthcare, data scientists must also consider the compliance and auditability of models when designing a workflow. The Data Science and Model Innovation team at Canadian bank Scotiabank, for example, built a deep-learning model to discover patterns in credit card payment collection. The model identifies potentially delinquent customers as well as those who might have simply forgotten to pay, and suggests the best way to approach them about payment.

In the future, the bank's internal auditors will need to evaluate new models, whether they comply with regulations and are of sufficient quality to help make decisions about real-life customers. A reproducible and, as far as possible, automated workflow makes auditing much easier.

"Then, you no longer need to believe somebody's word or make sure that you've taken the manual steps," says Suhail Shergill, director of data science and model innovation at Scotiabank. "The automation actually gives us more confidence. Whenever somebody needs to review, it's right there."

Team Structure and Roles

There are as many data science workflows as there are data scientists because their tasks, goals, and skills vary so much. Research scientists performing exploratory work or more ad hoc analyses might never need to write production code. Data scientists, like those in GitHub's machine learning team, write code that ends up in a software product that must perform at scale.

To define an effective team workflow, you first need to clearly define the roles within your team. According to Alluvium's Conway, the three roles that work best in data science teams are the data scientist, machine learning engineer, and data engineer.

The data scientist explores the data and can build a minimum viable product (MVP) version of a function, feature, or product. Machine learning engineers are concerned with performance and scale. How is this feature actually going to work on a website with millions of people interacting with it in a day? The data engineer designs tool-

ing and infrastructure to serve both the product and the data scientists.

Scotiabank's Shergill sees these roles as more fluid, defined on a continuum rather than via a hard divide, especially as a team evolves over time:

> We got to a state where a lot of the engineers, either software or data engineers, were really providing excellent things from a data science perspective. The data scientists were also coming up with suggestions to make software better.

Friederike Schuur, a data scientist at Fast Forward Labs, a machine learning research firm, says a proliferation of roles are emerging, including specialists in particular types of algorithms like natural language processing (NLP) or deep learning. "Data science is opening up these new specialized positions," says Schuur. "Every time that happens, you're creating touch points, and those touch points are becoming potential friction points."

Eliminating friction points is one of the reasons why you need a good team workflow, or sometimes even an entirely new role. Andrew Ng, the chief scientist at Baidu, advocates for the role of the AI product manager, who translates all the business requirements into a test set. Over the full lifecycle of their projects, data science teams also might work with Scrum masters, designers, software developers, DevOps, and even auditors.

The Data Science Process

There is no universally agreed upon data science process. The introductory data science course at Harvard uses the following basic process, which I will use as reference when discussing workflow and best practices at each stage (see Figure 1-1). (Hilary Mason's OSEMN taxonomy of data science, although it dates from 2010, is also still an excellent overview of the stages in a data science project.)

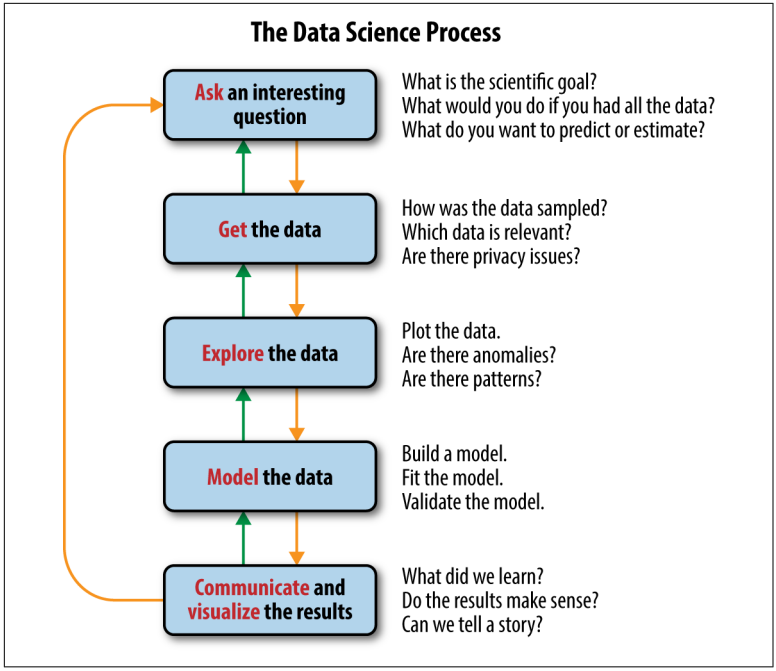

Figure 1-1. One representation of the data science process (courtesy of Joe Blitzstein and Hanspeter Pfister, created for the Harvard data science course)

Development workflows come in many different flavors, but they generally include steps to define specifications, design, write code, test and review that code, document, integrate your code with the rest of the software system, and ultimately deploy the system to a production environment where it can serve some business purpose. Because we are discussing development workflows for data scientists, the sections that follow refer to a mix of steps from the data science process and the relevant steps in a typical software development process.

Ask an Interesting Question

Asking good questions is both a science and an art. It has been described as the hardest task in data science. Understanding both the goals of your business (or client) and the limitations of your data seem to be key prerequisites to asking interesting questions.

Dean Malmgren is a cofounder of the data science consultancy Datascope. "Almost always during the course of our engagements, our clients have already tried something similar to the thing that we're doing for them," he says. "Just having them talk to us about it in a way that is comprehensible for people who haven't been staring at it for two years is really hard."

You can't understand a question or problem by looking at the data alone. "You have to become familiar with the subject," says Fast Forward Labs' Schuur. She goes on to say:

> There's one specific client which wants to automate part of customer service. The technical team has already been thinking about it for maybe three or four months. I asked them 'did you ever talk to someone who does that job, who is a customer service representative?' It hadn't even occurred to them.

At the beginning of every project, GitHub's machine learning team defines not just the problem or question it addresses, but what the success metrics should be (see Figure 1-2). "What does user success mean?" says Ho-Hsiang Wu. "We work with product managers and application engineers to define the metrics and have all the instruments ready."

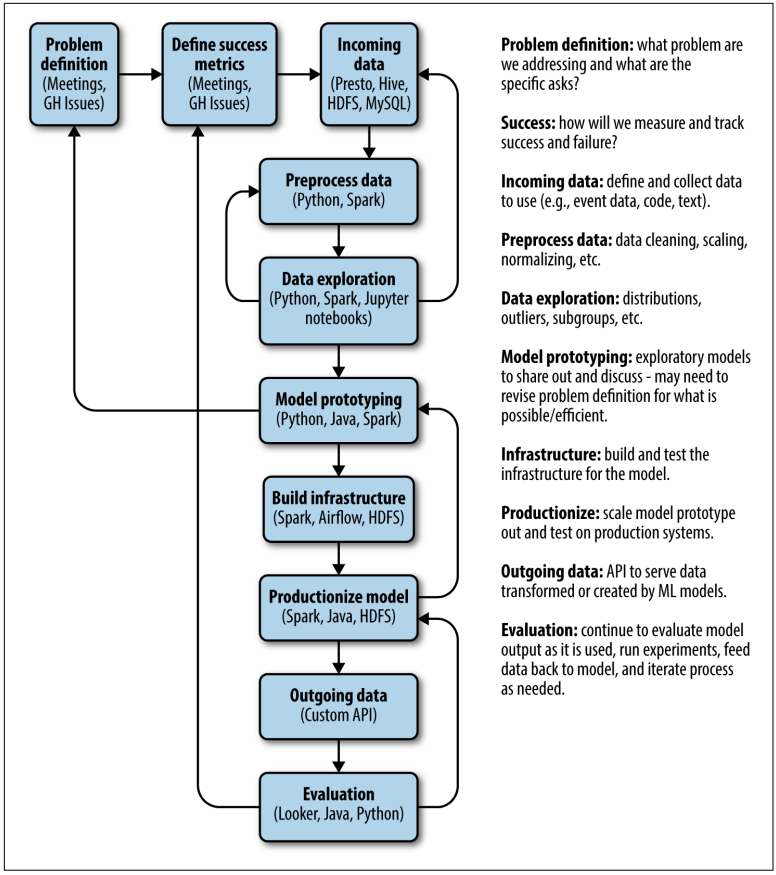

Figure 1-2. The data science workflow of GitHub's machine learning team

Defining a success measure that makes sense to both the business and the data science team can be a challenge. "The success measure that you give to your technical team is the thing that they optimize," says Schuur. "What often happens is that the success measure is something like accuracy that is easily measurable and that machine learning engineers or data scientists are used to." If that metric decouples from the business objective, the business feels that the data science team doesn't deliver, or the technical team creates models and reaches conclusions that might not be valid.

Examine Previous Work

As data science teams expand and mature, knowledge-sharing within data science teams and across the entire organization becomes a growing challenge. In scientific research, examining previous relevant work on a topic is a basic prerequisite to doing new work. Frequently, however, no formal processes or tools exist within organizations to discover earlier data science work.

Airbnb's Knowledge Repo is the company's attempt to make data science work discoverable, not just by the data science team, but by the entire company. Posts written in Jupyter notebooks, R markdown files, or in plain markdown are committed to a Git repository. A web app renders the repository's contents as an internal blog (Figure 1-3). Research work in a different format, such as a slide deck or a Google doc, can also be automatically imported and rendered in the Knowledge Repo.

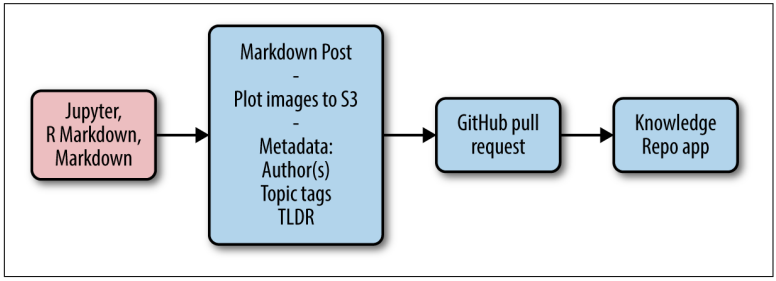

Figure 1-3. Airbnb's Knowledge Repo

Every file begins with a small amount of structured metadata, including author(s), tags, and a synopsis (also known as a TLDR). A Python script validates the content and transforms the post into plain text with markdown syntax. GitHub's pull request mechanism is used for the review process. Airbnb has implemented a checklist for reviewers including questions like does the synopsis accurately summarize the file? Does the title make sense?

Data science work at Airbnb is now discoverable via a full-text search over the synopsis, title, author, and tags. Since the Knowledge Repo started two years ago, the number of weekly users has risen from 50 to 250 people, most of whom are not data scientists. Product managers at Airbnb often subscribe to tags or authors and get an email every time a new post on a topic appears.

"The anecdotal evidence is that the number of times we have to redo or recreate a piece of research has gone down," says Knowledge Repo's Ray. She explains:

> On Slack, we've seen that when people ask questions, people will direct them to the Knowledge Repo. More people are getting feedback on research. Before, the first time you shared research was when you were presenting to your product manager or your engineers. Now there's a formal review cycle, a lot more people are willing to chime in and give their feedback.

Get the Data

GitHub's insights data science team provides data and analyses on user behavior to the rest of the company. According to data scientist Frannie Zlotnick, who works in the team, the insights teams regularly acts as a data intermediary, pulling CSV or JSON files for other teams that might not have the access permissions for the data or the skills to transform it.

If a project requires new data, the insights team will collaborate with GitHub's engineering teams to put instrumentation in place to gather that data. Often the data is already available but in a format that is not usable; for example, an unstructured flow or JSON that is not schematized. The insights team then transforms the data into a regular format that can be converted to SQL tables. Restructuring of the data is automated by using scripts.

It's not enough to gather relevant data; you need to understand how it was generated. According to Fast Forward Labs' Schuur:

> If you don't look at that process and if you don't witness it, then I think it will be very hard to really understand the data that you're working with. There is too quickly a jump to 'how are we going to model that?'

Some data is more sensitive, and special procedures must be followed to gather and use it. "We had to go through extensive audit and security review to make sure that what we've done is not compromising any client information," says Scotiabank's Shergill. He follows up:

> Doing that while being able to learn from that data, it becomes a bit challenging. There's security, compliance, anonymization. What are the security procedures that you're following? Do you have a security design document of the workflow? We made heavy use of containerization, provisioning tools, and automation.

Explore the Data

In addition to being Data Science Workshops' founder, Jeroen Janssens is the author of *Data Science at the Command Line*. He says that the command line is most useful when you want to explore a new dataset, create some quick visualizations, or compute some aggregate statistics:

> When you start exploring, you don't know anything about the data. What files are in here? What file type are they? Are they compressed or not? Does the CSV have a header? When you start working with a language like R or Python, it would really help if you know these kinds of things.

Janssens also says that having a good data directory structure helps:

> A tool called cookiecutter can take care of all the setup and boilerplate for data science projects. Inside the *data* directory, there are subdirectories called *external* or *interim process* and *raw*. When I started working for a startup, I learned that they were throwing away all the raw streaming data and only temporarily stored aggregates. I had to convince the developers that you should leave the raw data untouched.

Datascope's Malmgren says that his team frequently downsamples data in order to quickly prototype, a process that is largely automated. Says Malmgren:

> If the original data is a terabyte, we'll downsample it to a couple hundred megabytes that still represents something significant, but it's small enough that we can iterate quickly. If you have to wait for an hour for your model to train before it computes a result, that's kind of an unnatural amount of time to let your brain linger on other things.

Model the Data

The optimal choice of model depends both on the goal of the analysis and the characteristics of the data. "It's not okay to simply have the best model (in terms of predictive power)" says Suhail Shergill. He expands on the point:

> What is the speed with which you can train it? That may be different from the speed of inference where you actually define the prediction tasks, which may be different from how rich a pattern you can express in it, which may be different from how well or not you can explain this model.

The needs of the business users of the final models are also important when it comes to deciding what to optimize. Says Shergill:

> Various areas in business have different appetites for explainability. When we look on the collections side (credit card payments) the black box approach is okay. In approval, if you decline somebody (for a credit card), you should have a reason, you should be able to explain that decision.

There have been some attempts to automate model selection and building. Alluvium's Conway, mentions a company called DataRobot that has a product for selecting the best model given a particular dataset and prediction target.

Scotiabank has an entirely separate team to independently validate models built by the global risk team. "The way to go is have a team do something, then have a completely independent team create a different model for the same problem, and see if the results are similar," says Shergill. "And they do it here. That to me, is awesome, and something that I've not seen in other industries."

Test

Testing is one area where data science projects often deviate from standard software development practices. Alluvium's Drew Conway explains:

> The rules get a bit broken because the tests are not the same kinds of tests that you might think about in regular software where you say "does the function always give the expected value? Is the pixel always where it needs to be?" With data science, you may have some stochastic process that you're measuring, and therefore testing is going to be a function of that process.

However, Conway thinks that testing is just as important for data science as it is for development. "Part of the value of writing tests is that you should be able to interpret the results," he says. "You have some expectation of the answer coming out of a model given some set of inputs, and that improves interpretability." Conway points out that tooling specifically for data science testing has also improved

over the past couple of years, and mentions a package called testthat in R.

At data science consultancy Datascope, there are two main types of testing: a prediction accuracy test suite, and testing to improve the user experience of reports. When testing for accuracy, the team often downsamples the original test set. A first small downsample is used to iterate quickly on models, and then a second downsample consisting of 1 to 10 percent of the dataset is used to test with some statistical accuracy the quality of those models. Only when a model is already working well is it tested on the full dataset.

"We often do ad hoc tests just to make sure that the data that we're inputting into our systems is what we expect, and that the output from our algorithms are actually working properly," says Datascope's Dean Malmgren. "The value of the output is when it makes sense. This is hard to do algorithmically. I think I just made software engineers all over the world die a little bit inside, but it's kind of how we roll."

Test-driven software development, as applied to data science, still seems to be a contentious issue. "I don't want to kill innovation so I don't want to say, 'Before we write this predictive model let's make sure we have all these test cases ahead of time,'" says Aaron Wolf, a data scientist at Fiat Chrysler Automobiles. "We've got to understand the data. There's a lot of trial and error that occurs there, which I don't like putting in the test-driven framework."

Document the Code

In software development, documenting code always means documenting a working solution. That's not necessarily sufficient for data science projects says Fast Forward Labs' Friederike Schuur:

> In data science and machine learning you're doing so many things before you know what actually works. You can't just document the working solution. It's equally valuable to know the dead ends. Otherwise, someone else will take the same approach.

BinaryEdge includes a step in its data science workflow for exactly this kind of documentation. During the exploratory stage, data scientists at the company create a codebook, recording all steps taken, tools used, data sources, where the results were saved, conclusions reached (even when the conclusion is that the data does not contain

any information relevant to the question at hand), and every detail required to reproduce the analysis.

Deploy to Production

Scotiabank's global risk data science team built a sophisticated, automated deployment system for new risk models (see Figure 1-4). "We want to develop models almost as quickly as you can think about an idea," says data science director Shergill. "Executing it and getting an answer should be as quick and easy as possible without compromising quality, compliance, and security. To enable that, lots of infrastructure has to be put into place and a lot of restructuring of teams needs to happen."

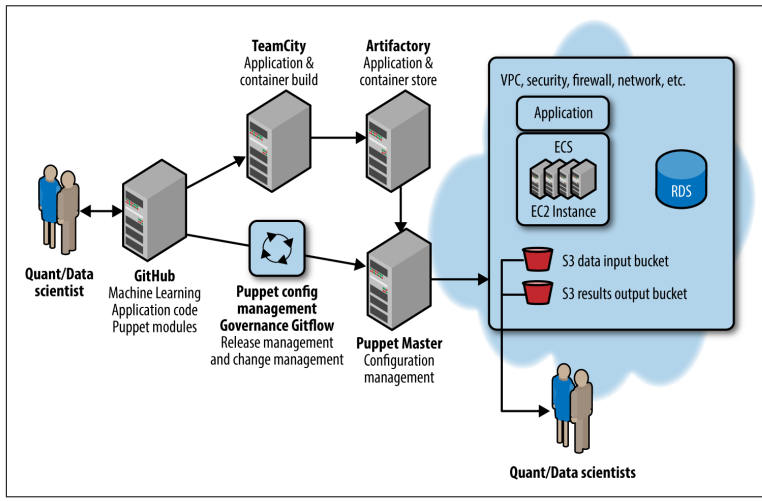

Figure 1-4. Scotiabank's automated deployment system; this deployment model is based on the model proposed by Seamus Birch

Banking is a regulated industry, so Scotiabank's entire development and deployment process must conform to strict compliance and security requirements and must be fully auditable. At the same time, the process has to be extremely efficient and easy to use for Scotiabank's data scientists.

Models, application code, and Puppet modules to configure infrastructure in the production environment are stored in GitHub. A standard gitflow is used to version all changes to the models and to the environment so that they are fully auditable. Changes are approved using GitHub's pull request mechanism.

At GitHub, the machine learning team experiments with new models in a development environment, but it is also responsible for writing production-ready code and plugging the most promising models into the production data pipeline. New models and features are always launched to GitHub employees first, followed by a small percentage of GitHub users.

Communicate the Results

Data scientists often cited the first and final steps of data science projects—asking a question and communicating results—as the most problematic. They are also the steps that are the least amenable to automation.

"It is difficult for people to wrap their heads around what it means to work within a world where everything is just a probability," says Fast Forward Labs' Schuur. "People want exact numbers and definite results. Even just two days ago, I was in a meeting and we had built a proof-of-concept prototype for the client, and they just went, 'Does it work? Yes or no.'"

Another issue mentioned several times was the tendency of data science teams to build over-complicated models that the target users do not use. As Fiat Chrysler Automobiles' Wolf puts it:

> Either they don't understand it because they don't understand the black box, or the output is not something that's consumable. Focus more on usability as opposed to complexity.

GitHub's machine learning team provides its models via an API, which is easy for the company's frontend developers to use. Because every department at GitHub, including sales and marketing, uses GitHub as a standard part of its workflow, pull requests provide a mechanism for all GitHub employees to follow and comment on the work of the data science team. Rowan Wing, a data scientist on GitHub's machine learning team explains it this way:

> I think it helps with that global brainstorming as a company when anyone can come in and see what you are working on. It helps give direction to the project. I might be working on something for one problem and then three people come in and say, "Oh, this is solving a bigger problem."

Datascope's Dean Malmgren also believes in involving clients throughout the data science process. "We use GitHub in that way," he says. "We invite our clients into the repositories. From coming up

with ideas about how you can approach a problem, to building those first couple prototypes, to collecting feedback and then iterating on the whole thing, they can add comments to issues as they arise."

A Real-Life Data Science Development Workflow

Swiss cybersecurity firm BinaryEdge provides threat intelligence feeds and security reports based on internet data. The data science team works mainly with data obtained via the BinaryEdge scanning platform. One of the responsibilities of the team is to ensure data quality as well as enhancing the data by cleaning or supplementing it with other data sources.

The data science team wanted to create a data science workflow that was rigorous, objective, and reproducible (see Figure 1-5). The team cited reproducibility as the biggest problem it sees in data science workflows.

The team works with data that quickly becomes out-of-date, so it wanted its workflow to produce initial results fast, allow a subsequent thorough analysis of the data while avoiding common pitfalls, and transmit the resulting knowledge in the most useful ways possible.

BinaryEdge's team admitted that the workflow is not always rigorously followed. Sometimes a few steps are skipped (for example, the feature-engineering step) in order to quickly deliver a first functional version of the product. The team later executes all the steps that were skipped in order to create a new, better version of the product.

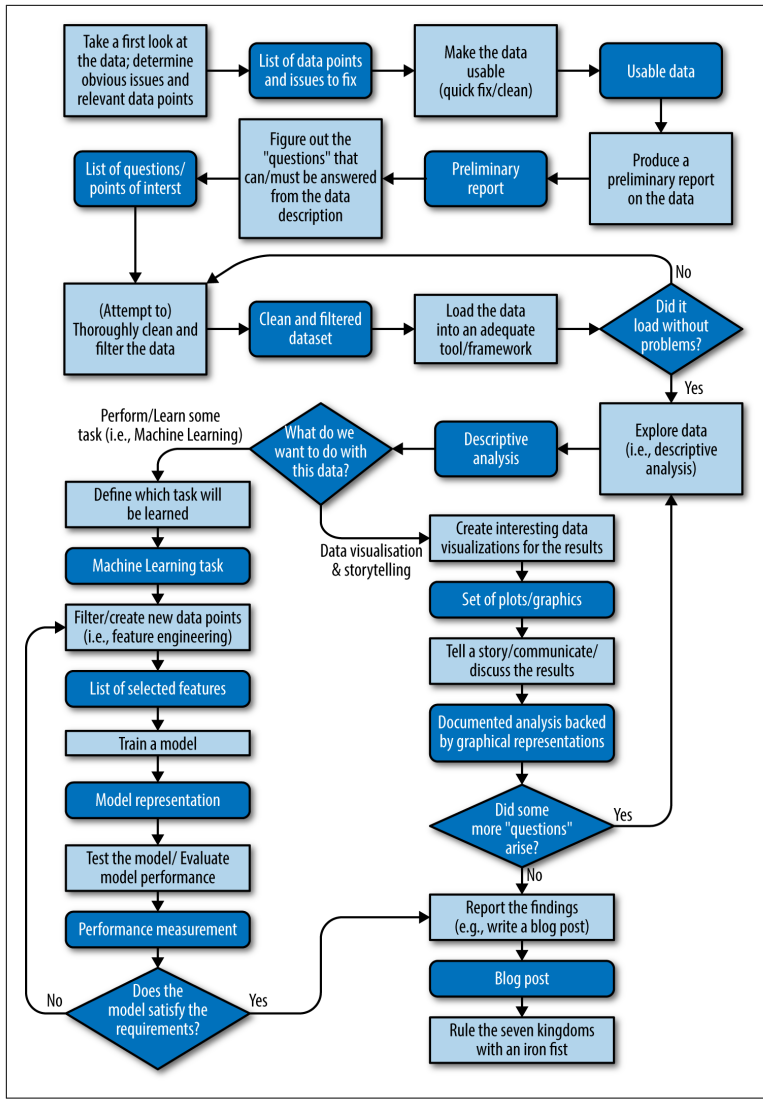

Figure 1-5. BinaryEdge's data science workflow

Preliminary Data Analysis

The process of gathering data from the BinaryEdge platform is automated, as are the first steps of the analysis, such as cleaning the data and generating reports. The main tools used at this stage are Bash, Python, and a Jupyter notebook. The notebook contains a quick

overview of the data and some statistics. All of these documents are stored in a GitHub repository.

Exploratory Data Analysis

Usually the output of this phase is also a Jupyter notebook, documenting everything done with the data: all exploratory analysis steps executed, results, conclusions drawn, problems found, notes for reference, and so on. BinaryEdge mainly uses Python. However, according to the volume of the data (specifically whether it can be fully loaded to memory or not), the company sometimes also uses another framework, typically pandas and/or Spark.

Knowledge Discovery

The data BinaryEdge works with is constantly changing, and as such, so does the model. To model, the team uses frameworks such as scikit-learn, OpenCV for image data, and NLTK for textual data.

When building a new model, all of the steps, including choosing the most relevant features, the best algorithm, and parameter values, are performed manually. At this stage, the process involves a lot of research, experimentation, and general trial and error.

When feeding an existing model with new data, the entire process (cleaning, retrieving the best features, normalizing) can be automated. However, if the results substantially change when new data is injected, the model will be retuned, leading back to the manual work of experimentation.

Visualization

The main outputs of this step in the workflow are reports such as BinaryEdge's Internet Security Exposure 2016 Report, blog posts on security issues, dashboards for internal data quality, and infographics.

Data visualizations are created using one of three tools: Plotly for dashboards and interactive plots, Matplotlib when the output is a Jupyter notebook report, and Illustrator when more sophisticated design is needed.

How to Improve Your Workflow

Improving your workflow can improve your working life, but a good workflow for you or your team is one that is tailored to your tasks, goals, and values.

Enable Collaboration and Knowledge Sharing

Collaboration on, and sharing of, data science work emerged as an ongoing problem for many teams. "I am seeing some questions being answered again and again every year, just with a new group of people," says GitHub's Ho-Hsiang Wu. "Can we find a way to propagate that knowledge to the later people or to a larger audience? I think because we are at GitHub, we try to use GitHub to do all this." Wu's team uses iPython and Spark notebooks in the exploratory phase, which are saved in GitHub and can be rendered there for all to easily access and view.

At Airbnb, one unexpected side effect of the introduction of the Knowledge Repo was the publication of many posts on data science methodology and best practices. Says Nikki Ray, who maintains the Knowledge Repo:

> Before this existed, this was all either in people's heads or in emails where you couldn't find them. I think this is probably one of the biggest improvements. With the Knowledge Repo being introduced as a normal part of the workflow, more people have taken it upon themselves to document these best practices.

Agreeing on standard formats like notebooks, setting up collaboration tools, and creating new workflows to make previous work discoverable are all helpful but don't yet solve the problem entirely. "The notebook paradigm is useful because it creates a kind of sharable entity," says Drew Conway. "Sharing still becomes nontrivial because sharing data and images and reports, and all these things, can be cumbersome."

Learn from Developers

Data Science Workshop's Jeroen Janssens offers the following advice:

> I have learned so much from developers at the companies I've worked with because they have been developing these best practices for much longer. If the development environment is easy to set up and consistent for every member of the team, then this will greatly

> improve everybody's workflow. The creation of these environments can easily be put into version control, as well.

Nowadays, when Janssens creates a Jupyter Notebook, for example, he also creates a Docker image to go with it. "The barrier becomes much lower for someone else to try out my analysis," he says, "It's just a couple of commands to set up exactly the same environment and to run the analysis again and get the same results."

Tweak Development Tools for Data Science

Airbnb noticed that data scientists would often push one pull request to GitHub with the contents of a new Knowledge Repo post, merge, and then follow up with lots of little pull requests because they didn't realize what their posts looked like in production. Nikki Ray explains:

> So, we wrapped all the Git commands in a wrapper. We also added preview functionality, which meant that you could see what your post looked like in production. A lot of the normal formatting in R Markdown or an iPython notebook is not what it ends up looking like on the web app side. We added a lot of our own CSS and styling to it to make it look and feel more like an Airbnb product.

Define Workflows Specifically for Data Science

Ultimately, data science doesn't fit neatly into a pure software development workflow and will, over time, need to create its own best practices and not just borrow them from other fields.

"How will things from traditional software development need to be modified (for data science)?" asks Drew Conway. "I think one of them is time. Time is so core to how all success is measured inside an agile development process. That's a little bit harder to do with data science because there's a lot more unknowns in a world where you're running an experiment."

Software development processes and workflows have evolved over decades to integrate roles like design, support methodologies like Agile development, and accommodate the fact that software is now created by teams, not by solo coders. Data science will undergo a similar evolution before it reaches maturity. In the meantime, data scientists will continue to build in order to learn.

About the Author

Ciara Byrne is a lapsed software developer, current tech journalist, wannabe data scientist, and started her career in Machine Learning research. She went on to manage teams of software developers, suites of products, and built her own products as well.

Her writing has appeared in *Fast Company*, *Forbes*, *MIT Technology Review*, VentureBeat, O' Reilly Radar, Techcrunch, and *The New York Times* Digital. In 2014, she was shortlisted for the Knight Science Journalism Fellowship at MIT and was a Significance Labs Fellow.