Standing up a Data Science Group

Introduction

Many large corporations have recognized that data science creates significant revenue-enhancement or cost-reduction opportunities. Historically, data science work has mostly been the purview of experts such as industrial engineers, clinical researchers, market researchers, advanced business intelligence (BI) analysts, or stock market modelers. Recently corporations have started anointing chief data officers to champion the strategic, scientific use of data. Likewise, many are creating internal, general-purpose data science teams.

Standing up a successful data science team is a complex and error-prone undertaking. Data science differs substantially from ordinary IT or software development work, or from historically more common heuristic approaches to business management. Data science is, well, *science*. Its most effective practitioners are trained *applied* scientists—scientists with the training and inclination to achieve practical outcomes using scientific methods. Building a successful data science team takes these apparently bland observations as its points of departure, with surprising and challenging consequences.

So What is Data Science, Really?

In common business parlance, the phrase 'data science' refers to analytical methods that one can apply to business problems, especially problems involving large quantities of data generated by human activity.[1] The same methods appear in many scientific domains, and arise wherever there is data to be analyzed formally.[2] In recent years the academic and business communities have recognized that many sciences share three classes of modeling techniques:

- classification
- prediction
- optimization.

Data science is the art of developing and applying these modeling techniques.

[1] The phrase 'big data' has a surprisingly narrow technical definition: It means data that is too voluminous to manage economically in a traditional relational database management system (RDBMS). (Note that 'economically' is in the eye of the beholder. There is no industry standard.) Data science is frequently applied to big data problems, but data science also applies to "small data" problems—those whose data can be stored economically in an RDBMS. In fact, data science is applied far more often to small data.

[2] In this sense the phrase 'data science' is almost redundant. While there are those who believe "science" includes informal methods such as historical or ethnographic analysis, empirical science at least combines observation with *formal* analysis, so that any formal method used to model empirical data is "data science."

Will the Real Data Scientists Please Stand Up?

Historically the development and application of data science techniques to business problems (and more generally to problems of all human organizations) has been the purview of **management science**: operations research, industrial engineering, decision science, econometrics, etc. All of these disciplines borrow computational methods liberally from applied mathematics, computer science, and statistics. Over the last two decades, the business world's near universal adoption of computer and networking technologies has broadened the immediate practical application of many mathematical, computational, and statistical techniques to business problems. But other sciences and engineering disciplines also make increasingly frequent use of the same methods. Consequently, data scientists can be trained as clinical researchers, demographers, market analysts, or aircraft engineers. Whatever the original field of application (if any), the data scientist has learned to classify, predict, and/or optimize using state-of-the-art formal methods.

Does it Matter What Kind of Data Scientists we Hire?

Different organizations have very different data science requirements. The differences have several dimensions.

> **Problem types.** *What mixture of classification, prediction, and optimization problems does the organization face or expect?* Few, if any, data scientists are expert in every modeling technique. The team's data scientists should have a mix of skills that mirror the types of problems the organization faces.
>
> Notice, however, that many optimization problems masquerade as classification or prediction problems. For example, the best classification models are optimized to minimize the total cost of misclassification (that is, to maximize the total benefit of proper classification), rather than merely trying to minimize the frequency of misclassification. And the best prediction models are likewise optimized to maximize the benefit of their predictions (that is, to penalize strongly the most costly incorrect predictions), rather than merely trying to make the most numerically accurate predictions.
>
> The foregoing observation has deep implications. Many data scientists trained in statistically oriented classification and prediction techniques are at most only dimly aware that the problems they face frequently require deep optimization methods. They also may not realize that the economic benefit that the organization derives from data science may depend strongly on how carefully the data scientists approach a problem's optimization requirements. They may lack adequate insight, experience, or humility to judge when optimization is needed. Likewise, management's failure to recognize optimization opportunities may result in building a data science team capable only of solving classification and

prediction problems. Such a team may fail to deliver much of the economic benefit that data science might otherwise offer.

Problem complexity. *How formally complex are the organization's problems?* Data science problems range in complexity from schoolbook problems that one can solve with traditional statistical techniques (such as linear regression or analysis of variance) to problems that mix several layers and kinds of optimization modeling with multiple sources of highly random inputs (such as weather or market events). Hiring the right data scientists depends on gauging accurately the complexity of the organization's data science problems and, thus, the depth of expertise that solving those problems will require. Errors on either side of this dilemma can be costly. Overwhelmed data scientists may do their well-intended best but fail to deliver the high return on investment (ROI) that many data science projects offer. Underwhelmed data scientists will quickly leave the team for more interesting problems, leaving the team understaffed and/or overwhelmed.

Data volumes and data architecture. *How much data do the problems involve? (Do they require using big data technologies to store and analyze the data? More generally, what data architectures will best support appropriate analysis?)* There is a non-trivial interaction between data architecture and data science. Appropriate representation of data is often vital, not merely convenient, to support efficient or effective data analysis. Staffing a data science team thus involves understanding the variety and depth of data architecture problems that the team will face to ensure that the team chooses, deploys, and exploits the right data storage technologies.[3]

Modeling prowess. *How important will it be to have the data scientist use the best possible models, perhaps even improving the state of the art while solving the organization's data science problems?* In many cases, standard solutions are good enough. But in some cases a state-of-the-art modeling technique, or an extra person-year of modeling effort, will deliver an extra margin of benefit that can translate into additional millions of dollars in saved costs (or some similarly compelling benefit). Judging when state-of-the-art modeling is crucial frequently requires someone who is capable of state-of-the-art modeling.

Domain knowledge. *How significant is domain expertise in correctly modeling the organization's data science problems?* Business analysis—that is, understanding and describing succinctly a business' processes, value propositions, strategic differentiators, etc.—is an important skill for a data science team.

[3] The right data storage technologies may include the usual relational database fare: data marts, operational data stores, enterprise data warehouses, and online transaction processing systems. But a data science team may also need to use a variety of **noSQL** data stores: a key value store such as Berkeley DB, a MapReduce implementation such as Hadoop, a text database such as MongoDB, or a graph database such as Neo4J. **NewSQL** data stores such as NuoDB and in-memory databases such as Oracle TimesTen can also be useful in operational BI contexts involving high transaction speeds or volumes.

Having said that, some business problems require fairly deep understanding of arcane science or technology. In these cases it may be important to have a domain expert on the team, or at least to have domain experts regularly available to make sure that the team's problem formulations are faithful to the domain.

How do we Organize our Data science Work?

Data science has its own development lifecycle. The widely cited Cross-Industry Standard Process for Data Mining (CRISP-DM)[4] describes much of the data science lifecycle (DSLC), but there's more to it.

Problem definition. Defining a data science problem involves discovering a set of informal business requirements and translating them into a formal problem definition. The problem definition specifies possible independent, dependent, and decision variables. It also specifies stopping criteria. **Stopping criteria** tell the data scientists when to stop the data science lifecycle, either because they have solved the problem to the business' satisfaction or because they have exhausted the resources the business is willing to devote to the DSLC. Stopping criteria thus combine success and failure criteria.

Proper problem identification helps to reveal hidden optimization opportunities. Informal success criteria usually involve a specific set of improvements. Translated into formal terms, these improvements become maximization or minimization problems: maximize a given product's revenues, minimize a given line of business' operating expenses, etc. These are optimization problems.

Concept development. Usually the business wants to develop a business case to justify each data science development project. To do that, the data science team must outline a technical approach to the problem. The team must also conduct a gap analysis of the approach's data and infrastructure requirements. The approach and gap analyses must have sufficient detail to estimate development expense and the potential project ROI. The team should express the resulting business case using an accounting model (such as internal rate of return) favored by the project's executive sponsors. The team should also translate the concept into a project roadmap. Each phase of the roadmap should have stopping criteria consistent with the overall stopping criteria defined during problem definition. For example, stopping criteria for the data collection/preparation phase may require that the team gather and prepare a sample of 20,000 records of a given set of input variables within two months, using internally available or open-source tools.

[4] CRISP-DM divides the lifecycle into six phases: business understanding, data understanding, data preparation, modeling, evaluation, and deployment. See en.wikipedia.org/wiki/Cross_Industry_Standard_Process_for_Data_Mining for an overview. For more details see Nisbet, Elder, and Miner, "Chapter 3: The Data Mining Process," *Handbook of Statistical Analysis and Data Mining Applications* (Elsevier, 2009).

Data collection/preparation. In this phase the team gathers a sample of the input data that the scientific model may require. The purpose of the sampling is to support model development only. The data gathering techniques may differ from those used later to develop the production solution. The data preparation techniques often reach far beyond the cleansing and transformation operations required for traditional BI. For example, it is almost always necessary to replace null values with appropriate surrogates. Likewise, data must sometimes be smoothed or discretized using techniques appropriate for the specific types of scientific model that will consume the data.[5]

Scientific model development. Model development explores the sample data gathered in the data collection/preparation phase to determine which raw variables are most important for the modeling techniques specified during concept development, and to define new variables constructed as functions of raw variables. Then the scientific model proper is developed and validated, and its efficacy assessed.

In many cases developing, validating, and assessing a scientific model involves building a simulation of the flow of business events that will eventually be inputs for the production solution. The model developers use the simulation to generate a more robust sample of input data than was gathered during data collection/ preparation. This synthetic sample must be statistically realistic. For example, a variable having a certain distribution in real life must have the same distribution in the synthetic sample, and variable values must correlate in a realistic way. Industrial-strength simulations must also account for (and can help to discover) possible deadlock conditions in contemporaneous business processes.[6]

The team uses the simulated input data to exercise aspects of the scientific model that would not be exercised during testing that only used less variable real sample data. This lets the team tune the model thoroughly and later will support development, verification, and validation of the production solution.[7] Simulation is thus often a key step in model development.

Solution development. In some cases a data science project only needs to execute the validated model once and report the results to management. (This may be true of a market segmentation, for example.) In many cases, however, the team must implement the model in a decision-support system (DSS) or decision-

[5] See Chapter 4 of Nisbet et/al for a more comprehensive list of possible data preparation activities.
[6] These are basic formal requirements for simulation. There are many practical requirements, such as performance, scalability, logging, and playback. Mosaic has developed a proprietary fast-time simulation framework with distributed computing support to enable very fast, highly scalable simulations. See http://www.mosaicatm.com/Commercial/CTOPCACR.aspx for a sample application.
[7] In this context solution *verification* means confirming that the model correctly implements its design, while *validation* means confirming that the model accurately represents its real-world domain. See http://en.wikipedia.org/wiki/Verification_and_Validation_of_Computer_Simulation_Models.

automation system (DAS) (hereafter both termed a decision system, or DS). Developing a DS requires an entire software development lifecycle (SDLC).

There are several well-known SDLC models in industry. Describing and comparing them is beyond the scope of this document. We recommend the agile/scrum/lean paradigm, in part because a highly iterative SDLC is consistent with the iterative nature of scientific-model development.

Data science projects place unusually stringent requirements on software developers in several areas.

Algorithm complexity: DSs require proper implementation of complex algorithms, often using third-party libraries of statistical, machine learning, or optimization techniques. The developers responsible for coding these algorithms should either be data scientists with strong coding skills or very senior developers with strong backgrounds in algorithms, data structures, machine precision, etc., collaborating closely with data scientists. This collaborative development process differs significantly from traditional top-down development in which specifications are "thrown over the fence" and assumed to be 100% correct. Again, the agile/lean approach is most effective due to the need for iteration.

Algorithm performance: One motivation for a DAS is to handle high transaction volumes. (DSSs usually don't have that problem because humans make decisions more slowly than computers recommend them.) Where a DAS must handle high transaction volumes, the developers must be able not only to implement correct algorithms and data structures, but also to optimize the same code. Code optimization may involve advanced distributed programming techniques such as GPU programming.

Quality control: The data science SDLC requires very strong quality control (QC) to ensure that the DS generates proper decisions in all cases without unacceptable performance degradation. These features generally include rigorous, automated unit, functional, integration, and performance/scalability testing. Depending on the development tools the team uses, advanced software testing techniques such as static and dynamic analysis may be appropriate. The DS QC process is more complicated than traditional software QC because it is not always possible *a priori* to specify correct results for a software module. This does not mean that automated testing cannot be performed. On the contrary, this makes automated testing even more important. However, many software development organizations lack the skills and discipline to create automated test suites in such situations.

Data-storage and ETL capabilities. We have already mentioned the importance of choosing the right data storage technologies, and using

them correctly. The same caveats apply to the team's ability properly to select and deploy data integration tools, especially extract, transform, and load (ETL) tools. For example, a DS may require real-time ETL, and the team should be sufficiently conversant in ETL technologies to evaluate, select, and deploy an ETL tool that can handle the application's throughput requirements in the most cost-effective way possible.

Solution deployment. Data science models often effect dramatic changes in work organization. In extreme cases a DAS may eliminate jobs. For example, one expert system architected by a Mosaic data scientist reduced two person years of custom software development by an IT organization to a single person day of data input by a business analyst. The data science team and its management sponsors must anticipate these sorts of changes in work organization and proactively effect changes to the formal and informal reward systems so that the business employees who must use or support the DS do so out of enlightened self-interest. Sometimes this involves calculating and publishing metrics reflecting the gap between potential and actual DS benefit due to decision makers' continuing to make suboptimal decisions manually.[8]

Solution maintenance. The external realities that generate input data for a DS (such as market conditions) can change over time enough to require that the DS' model be re-tuned. As with any software application, end users may submit enhancement requests and bug reports, and the development team must evaluate and respond to these requests. Significant changes to the system may require corresponding adjustments to the simulation and then validation and verification of the system with the adjusted simulation.

Post-implementation evaluation. An enlightened data science team reviews significant projects dispassionately to learn as much as possible from the experience, especially about improving the discovery, development, and deployment processes.

Who Else Should be on the Team?

The overall DSLC outlined above has broad and far-reaching implications for how a data science team is staffed and how its work is organized. In many cases the data scientists are a distinct minority, and most of the team consists of business and technical analysts, ETL developers, database architects, software developers, simulation developers, and change management specialists. Many of these roles already exist in traditional IT organizations, which raises the difficult question of whether and when existing staff can adequately fill these roles, possibly "renting" appropriately skills (especially simulation development and data science) as needed for specific projects. This question has no general, easy answer. It is important, however, to recognize that significant data science

[8] A great case study highlighting this issue is the 2009 Gartner publication ID G00166979, "Effective Metrics Drive Business Results at CitationShares." https://www.gartner.com/doc/946612.

projects require unusually strong skills all around. Complex, high-impact projects especially are more likely to succeed if the team has strong, temporary staff than if it has less skilled, permanent staff.

Here are two hypothetical examples of the importance of strong data science skills on complex projects.

> **Example 1: Repair/Replace DSS.** A prestigious Fortune 500 engineering company assembled a software development team to develop and market a DSS. The DSS' goal was to guide large utilities in optimizing their physical-asset repair/replace policies and decisions. The revenue opportunity was substantial; reducing a large utility's asset-management expenses by a few percentage points translates to millions of dollars in annual savings. The technical problem had significant geospatial features, and the development team was very strong in geospatial programming. In fact the team's object-oriented programming skills were generally mature, and the team was committed to an agile/scrum SDLC.
>
> Unfortunately, the team's architects had very limited exposure to optimization methods, data architecture, and the specialized problems of DSS design. The team foundered on these issues for more than two calendar years (about 15 person years). Eventually, it hired a consultant, nominally to help them solve some data and DSS architecture problems. Over the next year, the consultant led the team through three re-architecture exercises:
>
> 1. designing an appropriate object-relational data-storage model to couple a relational database to the team's object-oriented code base,
>
> 2. evaluating several productized and open-source production rule engines and incorporating the best suited of them into the design to automate certain decisions, and
>
> 3. expressing the optimization problem in terms of continuous (rather than discrete) time so the optimization model fit real-world data.
>
> These changes let the development team finish its work without further obstacles, and the product went into beta testing within a year of hiring the consultant. At the same time, the third exercise led management to assert a stopping condition: The team had exhausted its development resources without solving the optimization problem. The optimization functionality was de-scoped, and the product's potential impact on a large utility's bottom line was dramatically reduced. Had the team recognized its architectural deficits near the start of the project, the team could have completed the optimization model and marketed a much more powerful, valuable product within two years of project onset.

Example 2: Retail Sales DSS. A consultant was hired to help a DSS development team building a software-as-a-service DSS designed to guide retail customer service personnel in up-sell and cross-sell decisions while serving customers. The DSS incorporated a large, multi-tenant data warehouse, and a traditional (*not* state-of-the-art) statistical model identifying sales opportunities. During the first year of development, the team's management (*not* its bright but inexperienced architects) recognized that the team was struggling to design a dimensional data warehouse and the related ETL and to implement the product's statistical model in the team's programming paradigm. The consultant's responsibility was to help solve these problems.

Over the next year, the consultant led the team in designing a dimensional data architecture and in evaluating several ETL architectures. The team quickly accepted dimensional modeling; but for several months resisted vigorously the notion that its ETL problems were well understood, and productized solutions readily available. Eventually the team accepted that it did not need to re-invent that particular wheel.

The consultant failed entirely to persuade the team to use a packaged statistical library to compute the statistical model. The team insisted on building entire generalized regression computations into single SQL statements, making the application's SQL needlessly complex, slow, and unmaintainable. Hard-coding the statistical model into the same SQL statement that fetched data from the database also made the application's architecture less modular. If the team ever decides to replace its traditional statistical model with a more powerful state-of-the-art prediction paradigm, doing so will require significantly more work.[9,10]

The consultant also failed to persuade the team or its product management to consider a more powerful, state-of-the-art statistical model. Nobody in product management, development management, or the engineering team had enough data science depth to recognize that a current generation model might produce substantially more accurate predictions or identify real sales opportunities far more often.

In this case the project went to market successfully, suffering modest delays, some architectural blemishes, and limited predictive power. The team did not start its project with enough technical depth but basically succeeded because management recognized some of the consequences of the team's inexperience.

[9] This is a good example of data science technical debt. See Chris Sterling, *Managing Software Debt: Building for Inevitable Change* (Pearson Education, 2011) for a fine survey of the more general issue.
[10] The team also did not attend to the numerical computation issues (e.g., rounding error) that should be addressed when coding a statistical computation. Possibly their naïve SQL implementation of the regression model sometimes resulted in significant numerical error affecting the model's predictions. Using a packaged library could have avoided this problem as well. The consultant experienced enough resistance around the more basic design issues that he did not even raise this one!

How do we Motivate and Retain our Data Scientists?

Problem depth, complexity, and variety strongly motivate most data scientists. In fact, a significant issue in managing data science is making sure the data scientists don't do science for its own sake, beyond the point of diminishing marginal returns—hence our earlier advice about defining clear stopping conditions for each project milestone and for the overall project. This makes it important to hire data scientists who will feel appropriately challenged by the work the employer can offer, balancing the organization's need for technical competence with the scientist's need for interesting challenges. (See "Does it Matter What Kind of Data Scientists we Hire?" above.)

Many data scientists are introverts who prefer working very independently with data and computers to working with people. (In extreme cases, the opportunity to work in isolation is a job-selection criterion.) This makes it important to determine realistically how much and what types of interpersonal work each data science role requires. A mismatch between a data scientist's interpersonal inclinations and a data science job's requirements may result in turnover. In most cases it is prudent to have a lead data scientist who is able and willing to communicate technical results to executives and other business people, and who can provide overall project vision and leadership. Larger organizations should consider a senior executive role, such as a chief data officer or chief data scientist, to own and evangelize the organization's data strategy.[11]

How do we Measure Success?

A successful data science practice does much more than solve technical problems:

- It works effectively with executive and managerial sponsors to identify, evaluate, prioritize, and execute projects having the highest potential ROI.
- It applies appropriate (often mathematically complex) optimization techniques (not merely classification and/or prediction techniques) to each problem, to maximize the project's actual ROI.
- It manages organizational change throughout each project's lifecycle to ensure that the organization embraces the increased reliance on quantitative analysis (rather than traditional business heuristics) that the project represents.
- It operates a highly disciplined, quality-oriented approach to model and application development and operation.
- It retains and rewards key staff in ways that are consistent with their personal needs for technical opportunities, and with their interpersonal effectiveness.

[11] In this context 'data strategy' refers to an organization's approach to leveraging its data assets to achieve competitive advantage.

This is a daunting spectrum of challenges. Often the appropriate response is nuanced, relying initially (at least) to some degree on outside experts to help a data science team to develop its capability while also executing some high-ROI projects quickly. Partnering with outside experts also creates mentorship opportunities that can help a young data science team to learn quickly, gain credibility, and produce significant early results, while avoiding costly mistakes that the team might otherwise fail to anticipate.

How do we get Started?

If your organization contemplates building a significant data science capability, or is partway through that process, it may accelerate the process substantially by hiring external experts to complete the following tasks:

- inventory the organization's current and foreseeable data science opportunities,
- assess its current data science capabilities,
- analyze the gap between opportunities and capabilities,
- coach management through the process of closing that gap systematically, and
- help execute some early high-ROI/low-risk projects.

Mosaic Data Science's executives have over a half-century of collective experience building and managing data science organizations that have successfully solved very diverse data science problems, including highly complex optimization problems producing project ROIs measured in the tens of millions of dollars. We would be delighted to work with your organization to help develop its data science capability.