IBM Analytics

Practical data science tips for data engineers

IBM

Introduction

In the domain of data science, solving problems and answering questions through data analysis is standard practice. Data scientists experiment continuously by constructing models to predict outcomes or discover underlying patterns, with the goal of gaining new insights. But data scientists can only go so far without support.

Enter the data engineer. Together with business leaders, IT leaders and other skilled professionals, data scientists and data engineers make up a larger organizational team, with each member playing an essential role in leveraging internal and external data to increase competitive advantage and make better decisions.

As data analytic capabilities become more accessible and prevalent, several new practices are emerging, including the use of very large data sets, the incorporation of text analytics into predictive models and increased automation. To help guide the processes and activities within a given domain, data scientists and data engineers need to adopt a data-centric view of a data science foundational methodology.

This ebook describes the components and processes that comprise this foundational methodology for data science and discusses some of the integral tools and techniques being used by today's data engineers to collect, process, analyze and deploy data.

Processes, models and more: Why a data-centric methodology matters

A methodology is a general strategy that guides the processes and activities within a given domain. It does not depend on particular technologies or tools, nor is it a set of techniques or recipes. Rather, a methodology provides the data engineer with a framework for how to proceed with whatever methods, processes and heuristics he or she will use to obtain answers or results.

A data-centric view of the foundational data science methodology illustrates the iterative nature of the problem-solving process (Figure 1). As data engineers learn more about the data and how data scientists are using it for modeling, they frequently return to a previous stage to make adjustments. Models are not created once, deployed and left in place; instead, through feedback, refinement and redeployment, models are continually improved and adapted to evolving conditions. In this way, both the model and the work behind it can provide ongoing value to the organization for as long as the solution is needed.

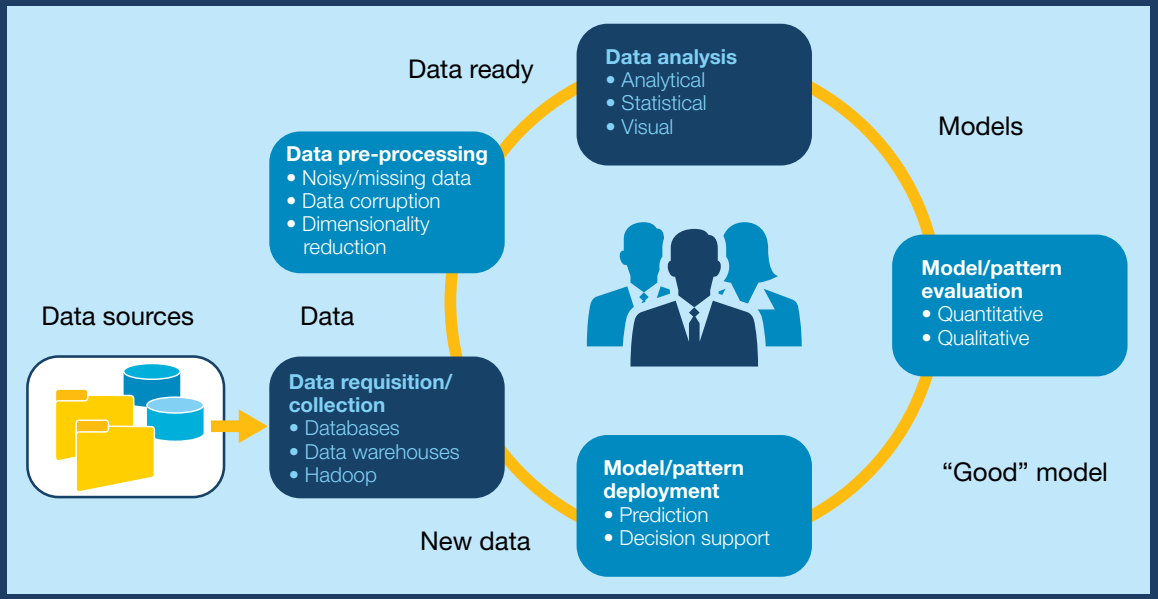


Figure 1. Data-centric view of methodology.

Data collection: Gather data relevant to the problem domain

In the initial data collection stage, being clear about why data is being assembled and what it will be used for is critical. Relevant data sets rarely reside in the same place. You could have data coming from relational database management systems where you could use tools such as SQL, Apache Sqoop and Java Database Connectivity to collect the data. In other cases, data may be coming from Hadoop environments such as Hadoop Distributed File System (HDFS), Hive, HBase and others—or from real-time streams such as Apache Kafka, IBM® Streams and RSS feeds. Either way, you'll need to make a strategic decision as outlined in Figure 2: Do you want to centralize the data in a single location or use a federated approach?

	Centralized	Federated
Performance	Data extraction is usually faster	Subject to longer delays in data delivery
Data availability	Better availability	Could be an issue if data cannot be queried all the time
Implementation cost	How much does it cost to build the centralization?	How much does it cost to leverage federation?
Time constraints	Is there time to build a centralized repository?	
Data quality	Easier to ensure data quality if quality is low	

Figure 2. Comparing centralized and federated data collection approaches.

A **centralized system** may help resolve data duplications, uncover inconsistent master data and improve data quality. However, implementing this model can present challenges when it comes to determining the geographical locations of the applications, absorbing the cost of the implementation and maintaining compliance with business rules and regulations.

A **federated approach** allows you to extend data and business services to acquire data from multiple sources. The goal is to make data available to all departments and partners of an organization. Implementing a federated model comes with a few hurdles of its own, such as synchronizing data between transactional and master data, potential data delivery delays, and identifying roles and responsibilities for managing the data.

Finding the right tools for the job

No matter what platform or tool you are using, effective data mining hinges on creating a suitable data model and structure that can be used to process, identify and build the information you need.

As you explore different data collection and data mining tools, determine which ones can be used in combination with your existing software and infrastructure. You can perform data mining with comparatively modest database systems and simple tools, including creating and writing your own, or using off-the-shelf software packages. You can mine data with various data sets, including traditional SQL databases, raw text data, key/value stores and document databases.

HDFS data

The MapReduce framework is the powerhouse behind most of today's big data processing. Creating MapReduce jobs involves writing functions to encapsulate the map and reduce stages of the computation. The data to be processed must be loaded into the HDFS. This integrated design scales easily, provides cost-efficient storage options and improves the performance of your overall big data environment. Although you can configure and manually execute data migration, there are tools available to do this for you. One such tool is Sqoop, which takes data from a relational source to HDFS, and vice versa.

Streaming data

Streaming data sources, such as web server log files or sensor output, are a growing source of input to big data systems. Apache Kafka has emerged as an effective tool for real-time data handling. In the simplest case, Kafka can provide a buffer for storing application logs. Combined with a technology such as Spark Streaming, you can use Kafka to track data changes and take action on that data before saving it to a final destination. Kafka's predictive mode makes it highly effective for detecting fraud, such as checking the validity of a credit card transaction when it happens.

Apache Spark

One of the fastest-growing open source tools is Apache Spark, which has several important advantages. First, with capabilities such as in-memory storage and near-real-time processing, its performance can be several times faster than other big data technologies such as MapReduce and Apache Storm. Spark also supports lazy evaluation of big data queries, which helps with optimization of the steps in data processing workflows.

Spark allows data scientists and data engineers to work together in a unified platform. Because Spark is not tied to a particular storage type or format, it can handle a wide variety of data sources, including batch, streaming and interactive applications. As a result, you can use Spark to support a federated approach to data collection and as the backbone for compiling and centralizing data. It also provides a higher-level application programming interface to help improve developer productivity and offer a consistent architecture model for big data solutions.

Data collection and data mining best practices

Wherever your data comes from, and whatever form and structure it's in, you must organize the information in a format that allows the data mining to take place in as efficient a model as possible. Consider the combination of the business requirements for data mining, the identification of the existing variables (customer, values and country) and the necessity to create new variables you might use to analyze the data in the preparation step.

Data mining is more than running some complex queries on the data stored in your database. You must work with your data, reformat it or restructure it, regardless of whether you are using SQL, document-based databases such as Hadoop or simple flat files. Identifying the format of the information you need is based on the technique and the analysis you want to do. Once you have the information in the appropriate format, you can apply the various techniques (individually or together) regardless of the required underlying data structure or data set you employ.

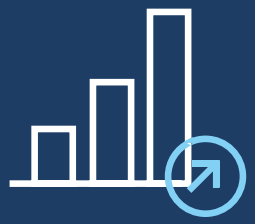
Data preparation: Extract and transform data into usable formats

Data engineers are often looking to deliver actionable insights through the marriage of structured and unstructured data. Unfortunately, you can't assume that the data—even structured data—is perfect and ready to use. It may need some work:

- **Data may be incomplete,** with missing or incorrect values
- **Data can be corrupted or incoherent,** with broken lines or fields in the wrong place
- **Data may need to be transformed or standardized** into an algorithm- or model-friendly format
- **Data may have too much "noise":** random, irrelevant data that throws off analytics

First, you have to capture and preserve the data structure as you move it into your Hadoop environment. Next, you'll need to bring along your metadata, particularly business metadata around critical terms, semantic meaning and business context.

The goal is to relate different data sets to find insights. One approach is data tagging. Tagging data before handing it off to Hadoop, Spark or another implementation medium for analysis can save an enormous amount of time. The trick is to process the data "just enough" so that it is useful when loaded into a big data environment—able to be discovered, accessed and joined with other data. Later, if you're looking to operationalize a useful analysis, it may be worthwhile to do more data preparation work at that time.



As the big data open source landscape continues to grow, new techniques are emerging that can help you process massive volumes of data quickly and efficiently. Hadoop tools such as Pig, Hive, Oozie, Sqoop and Flume are designed to help configure and handle big data. After you import your data into Hadoop, you can use tools such as Jaql and R to analyze that data.

Reducing data dimensionality

The recent explosion of data set size, in number of records and attributes, has led to an increase in the use of data dimensionality reduction procedures. Dimensionality reduction not only helps speed up algorithm execution, but also helps improve model performance. Removing uninformative—or even worse, misleading—input attributes can help engineers build a model with more extensive data regions, more general classification rules and overall better performance on new data.

The idea behind reducing data dimensionality is that raw data tends to have two subcomponents: useful features and noise (random and irrelevant structures). The goal is to take out salient and informative features from input data so they can be used further in predictive algorithms. There are many examples of highly dimensional data sets that are difficult to process at once, such as handwritten digits, facial features, user logs and even human gene expressions.

Feature extraction methods seek to transform the vector space representation of the document into one of a lower dimensionality. You could apply stemming with the objective of extracting the "word stem" to eliminate the variations arising from multiple occurrences of the same word in several grammatical forms; for example, "fishing," "fished" and "fisher" all map to the same word stem: "fish."

Examples of applications of dimensionality reduction

Extracting the important features in face/pattern recognition

Removing stop words such as "the" when working on text classification

Stemming: fishing, fished, fisher, _fish

A classic algorithm for reducing dimension is **principal component analysis (PCA).** The goal of PCA is to project input data onto a lower dimensional subspace, preserving as much variance within the data as possible. The results of a PCA are usually discussed in terms of component scores, sometimes called factor scores (the transformed variable values corresponding to a particular data point) and loadings (the weight by which each standardized original variable should be multiplied to get the component score).

Another method of dimensionality reduction is **singular value decomposition (SVD),** which seeks to identify and order the dimensions along the data points that exhibit the most variation. Once you have identified where the most variation is, you can find the best approximation of the original data points using fewer dimensions. SVD is ideal for natural-language processing applications because you can ignore variation below a particular threshold to massively reduce your data, yet remain assured that the main relationships of interest have been preserved.

Choosing your approach to dimensionality reduction depends heavily on the requirements of the application in which text classification will be used. For example, some applications require accurate performance regardless of the computational time; others may sacrifice some accuracy in favor of the execution time and storage needed.

Transforming to match algorithm requirements

Successful data mining relies on building a suitable data model and structure that can be used to process, identify and build the information you need. Regardless of the source data's form, you must structure and organize the information in a format that supports the most efficient machine learning model possible. In some cases, data may need to be transformed to match algorithm requirements. There are several techniques to accomplish this task.

Tokenization involves breaking up text into words, phrases, symbols or other meaningful elements called tokens. The list of tokens becomes input for further processing, such as parsing or text mining. Tokenization often relies on simple heuristics such as:

- Punctuation and white space may or may not be included in the resulting list of tokens.
- All contiguous strings of alphabetic or numeric characters are part of one token.
- Tokens are separated by white space characters, such as a space or line break, or by punctuation characters.

Bucketization is the process of defining several record groupings based on their sensitive values. The apparent sensitive values of the records' attributes are identified and sorted in ascending order based on the attributes' frequencies. After sorting, the values are grouped into similar buckets. Only those containing distinct sensitive values are kept.

Feature standardization helps you compensate for differences in continuous attribute distributions that could mislead some modeling algorithms (Figure 3).

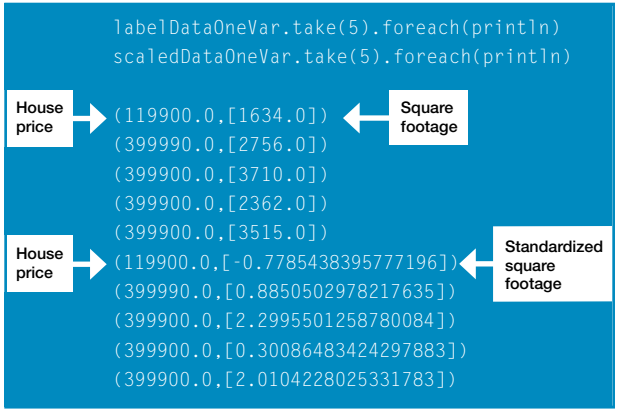


Figure 3. Example of feature standardization.

Normalization of data traditionally involves transforming data so that each vector has a unit norm. Since some models collapse at the value of zero, consider choosing an arbitrary range of 0.1 to 0.9 instead.

Dummy encoding is used when categorical values cannot be converted to numerical values. It provides one way of using categorical predictor variables in various kinds of estimation models, such as linear regression (Figure 4). Dummy variables are "proxy" variables or numeric stand-ins for qualitative facts in a regression model. They are used as devices to sort data into mutually exclusive categories. As such, they can be thought of as a truth value represented as a numerical value 0 or 1.

Marital status: {"Single", "Married", "Divorced", "Widowed"}
Marital status: {"0001", "0010", "0100", "1000"}

This is necessary if the algorithm could make some wrong inference from the numerical-based categorical encoding:

- Single = 3
- Married = 2
- Divorced = 1
- Widowed = 0

```
Single = Married + Divorced
Single = Divorced x 3
```

Figure 4. Dummy variables are stand-ins for qualitative facts used to sort data into mutually exclusive categories (such as married, single and so on).

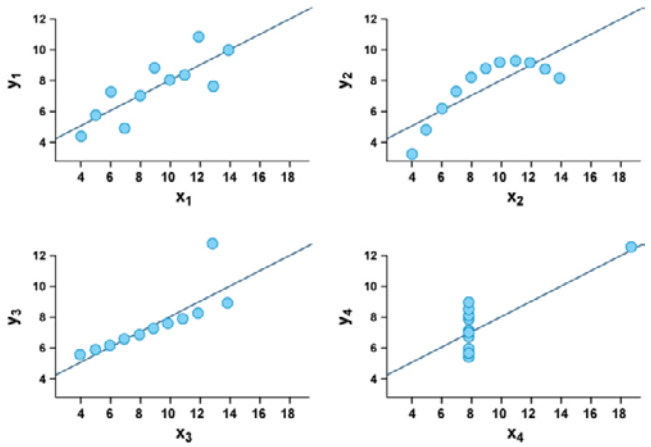
Data analysis: Drive insights through effective modeling

The data analysis phase guides you toward better and more relevant modeling. Modeling focuses on developing predictive or descriptive models according to the previously defined analytic approach. The modeling process is typically highly iterative as you start with the first version of the prepared data set, gain intermediate insights and use them to refine data preparation and model specifications.

The data analysis phase is likely to be intertwined with the data preparation phase. In fact, it is not uncommon to use data analysis results to refine or drive the data preparation.

The data analysis phase should leverage mathematical tools such as statistics, correlations and chi-squared tests, as well as visualizations, which can be used to communicate important trends across the organization.

Summary statistics allow you to describe a vast, complex data set using just a few key numbers. But there's a danger in relying only on summary statistics and ignoring the overall distribution. Anscombe's quartet offers a classic example of this risk (Figure 5). It comprises four data sets of 11 x,y points that have nearly identical simple statistical properties, yet appear very different when graphed.



The four data sets have similar statistical properties:

- The mean of x is 9
- The variance of x is 11
- The mean of y is approx. 7.50
- The variance of y is approx. 4.12
- The correlation is 0.816

As shown the linear regression lines are approx. y=3.00+0.500x

Figure 5. Example of Anscombe's quartet.

Because summary statistics can be misleading on their own, it's important to use them as just one tool in a larger data analysis process. Visualizing the data enables you to revisit the summary statistics and re-contextualize them as needed.

Making complex data accessible

A primary goal of data visualization is to communicate information clearly and efficiently through statistical graphics, plots and information graphics. Numerical data may be encoded using dots, lines or bars to visually communicate a quantitative message.

Effective visualization helps users analyze and reason about data and evidence. It makes complex data more accessible, understandable and usable. The design principle of the graphic (such as showing comparisons or showing causality) should follow the analytical task. Mapping in particular is an extremely useful tool for making data accessible. If you can visualize diverse data through mapping, you can often do your analysis in half the time. Even users who don't have high-level mathematical skills can do this process and achieve their goals faster.

One form of powerful visualization, especially when dealing with geographical data such as latitude and longitude, is to plot a sample of the data on a map. This can help you see trends or sometimes blatant mistakes, such as latitude or longitude values carrying the wrong sign (data points in the southern hemisphere instead of the northern hemisphere or vice versa). Line and bar plots can also be useful in visually detecting trends such as positive or negative correlations.

Effective visualization helps users analyze and reason about data and evidence. It makes complex data more accessible, understandable and usable.

For a given technique, you can try multiple algorithms with their respective parameters to find the best model for the available variables. You can also assign statistical significance tests to the model as further proof of its quality. This additional proof can be instrumental in justifying model implementation or taking actions when the stakes are high, such as an expensive supplemental medical protocol or a critical airplane flight system.

Ready for a test-drive?
Join us as we build and test a text classification model

Text analysis is often used to extract meaning from unstructured text attributes, which are valuable in tasks such as behavioral profiling. Using the Naive Bayes classification algorithm, for example, you can quickly create probabilistic classification models. Naive Bayes is not a single algorithm, but a family of classification algorithms that share one common assumption: every feature of the data being classified is independent of all other features, given the class. In other words, the value of one feature has no effect on the value of another feature.

Want more? Check out our deep-dive presentation, in which we prepare the data and then train a Naive Bayes classification model, and learn how to associate the "features" of a text to the class to which it is assigned. Using the test set, we then verify the accuracy of the trained model.

Learn more in our slideshare presentation, "Text Classification & Analytics Deep Dive: Naive Bayes classifier."

Creating value, not just insight

While extracting insight from your data is important, insight alone does not create value. You want insight that allows you to understand the likely result of specific actions. Then you can select the action with the best potential for producing the desired result.

In many cases, the data analysis phase can guide you toward better and more relevant modeling. For example, when you are building a classifier, it is important to understand the prevalence of the condition you are building a model for—such as how common or uncommon this condition effectively is.

Imagine you are building a classifier for a rare medical condition. Your training and testing of data sets yield the following model:

	Test positive	Test negative
Disease (100)	**95 (true positive)**	**5 (false positive)**
Normal (100)	5 (false negative)	95 (true negative)

With 95 percent sensitivity and specificity, this sounds like a great test. But what truly matters to the users of your new model (doctors, patients, practitioners) is the predictive value of the test: if the test is positive, what is the actual chance of being sick?

Let's run the test on a population of 1,000,000 where 1 percent of individuals (10,000) are suffering from this condition:

	Test positive	Test negative
Disease (10,000)	**9,500 (95% true positive)**	**500 (5% false negative)**
Normal (990,000)	49,500 (5% false positive)	940,500 (95% true negative)

So what is the probability of being sick if the test is positive?

- (# of people truly sick)/(# positive test results)
- 10,000/(49,500 + 9,500) = **16.9% probability of being sick**

What is happening here? The condition is rare, and the 5 percent of false positives is far higher in actual numbers than the true positives. Data analysis coupled with the prevalence of the condition indicates that you need a test with a sensitivity/specificity level of 99 percent or higher.

Cultivating a data science culture

Adopting a data-centric approach to a data science methodology helps data engineers bring consumability to data science. Now more than ever, success lies in the details—and data science scrutinizes those data-centric details to help leaders make good decisions quickly and confidently.

Today's data engineers hold a foundational role in integrating data science into business processes. Leading-edge organizations understand the need to cultivate a healthy data science culture that includes enterprising data engineers who are skilled at gathering, organizing and preparing data—thereby laying the groundwork for data analysts or scientists to easily retrieve and use data in their business-critical evaluations.

Learn more about data science best practices and tips at **ibm.com**/datascience

IBM Analytics
Route 100
Somers, NY 10589

Produced in the United States of America
August 2016

CDM12350-USEN-00