SCSB4013 INTRODUCTION TO DATA SCIENCE

UNIT 1 INTRODUCTION

Introduction - Need for data science – benefits and uses – facets of data – data science process –
setting the research goal – retrieving data – cleansing, integrating, and transforming data – exploratory
data analysis – build the models – presenting and building applications.

1.1 Need for Data Science

Today lots of data are generated via various electronic gadgets. Earlier lots of data were stored in unstructured as well as structured format and lots of data flow happen. If you look today, data is created faster than it is imagined. While travelling by road, we can recognize lots of data being created, for example, vehicles speed, traffic light switching, Google map, etc. which get captured through satellites and transmitted to handheld devices in real time. It guides by showing several alternative paths with traffic intensity and a favorable path to take. We also get a suggestion on which route to take and which route to avoid. So there is a lot of data being created as we do our day-to-day activity. The problem is that we are not doing anything with the data. We are not able to analyze due to inefficient scientific insights.

We are creating data, but we are not utilizing it for behavioral analysis or predictions. If we look at the corporate world, we find that lots of data reports are generated. But what about utilization? How much time are we spending on preparation? We are not drawing any scientific insights; e.g., we are not using data for forecasting. It can be forecasted for sale in the coming months, seasonal time, or years. We can conclude from available data for decision making and prediction. From the data, we can visualize what the actual status of the data is and what decision we have to take. It is possible for shop owner to predict that if person wearing a red shirt and have age between 25 and 30, will be interested in the product. We can also see pattern discovery. There are lots of sales happening in November and December every year, that is, in the festival season, and also the changes in weather increase the sale, so actually we need to draw lots of pattern from data and do discovery. It needs to recognize the process of data analysis, data patterns and behaviors patterns using data science for increasing sale.

The upcoming automobile technology is an autonomous car. It is exciting to have a car driving automatically which will take you from home to office and office to home, and data science uses the coordinates to take a lot of decisions in this whole process. Decisions of whether to speed up, whether to apply the brake, whether to take left or right turn, and whether to slow down are a part

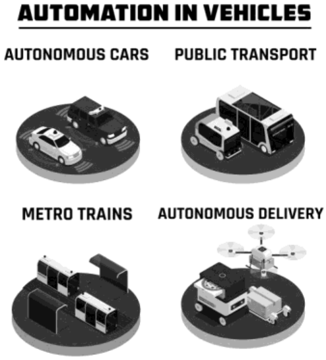


FIGURE 1.1
Autonomous vehicles.

of data science. The case study predicts that self-driving cars will minimize accidents, and in fact, it will save more than two million deaths caused by car accidents annually. Self-driving cars have a lot of scope for research design and testing in force to update knowledge of cars. Every automotive company is investing in self-driving cars. Studies say that in about 10–15 years, most of the cars will be autonomous self-driving cars [1]. The possible environment for autonomous vehicles is shown in Figure 1.1.

Another example, while booking travel plan, passenger are unaware about the weather condition as per decided travel schedule, hence need data science for predicting and alerting passenger about weather condition for taking decision of travel. Using data science, proper route planning can be done if in advance weather condition is predicted. Also air traffic managements system can plan the alternate route by rescheduling the flights as per environmental conditions. If the route is not planned properly, then you might end up in a situation where the flight is not available. Planning alternate flights between cities are challenging in last minutes due to preschedule air traffics. The roles of data science in airline industries are shown in Figures 1.2 and 1.3. If we use data science properly, most of these problems can be avoided and it will help in reducing the pain both for the airlines and for the passengers.

Few more applications can be possible in the airline industry such as better route planning so that there will be fewer cancellations and avoid the frustration of passengers. Predictive analytics can be used for predictions of any delays so that some flights can be rescheduled ahead of time and there will be no last-minute change. Data science can also be used to make promotional offers, select flights, and explain different classes of flight and different routes for better performance, as shown in Figure 1.3.

The logistics industry is another example where data science can be used. Companies such as FedEx use data science models to increase their efficiency by optimizing the routes and save cost. For shipping the materials, company

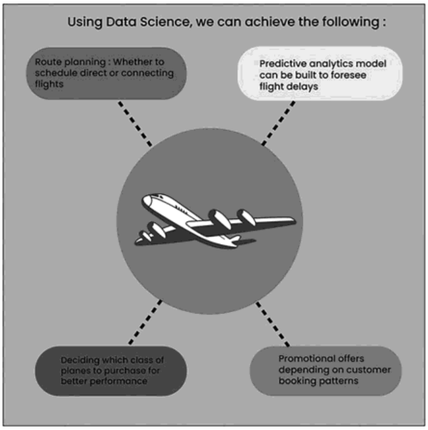

Using Data Science, we can achieve the following :
Route planning : Whether to schedule direct or connecting flights
Predictive analytics model can be built to foresee flight delays
Deciding which class of planes to purchase for better performance
Promotional offers depending on customer booking patterns

FIGURE 1.2
Data science and airline industries.

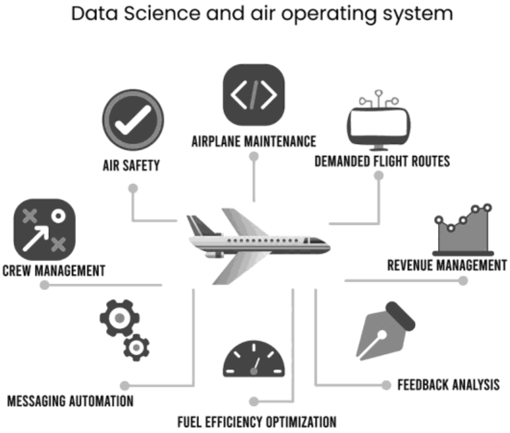


FIGURE 1.3
Use of data science in an airline operating system.

can use data science for prediction of possible routes, suitable time and best mode of transport for the delivery [4, 5].

The main purpose of data science is to compute better decision making. In some situations, there are always tricky decisions to be made, for example, on which prediction techniques are suitable for specific conditions, as shown in Figure 1.4. For example, can we predict delays in the case of an airline? Can we protect demand for a certain product? Another area is business in wearable sales. The apparels sales can be possible to schedule appropriately, if wearable pattern and demand recognized in time to increase the business. Also, selection patterns of apparels as per data of buying apparels in certain months,

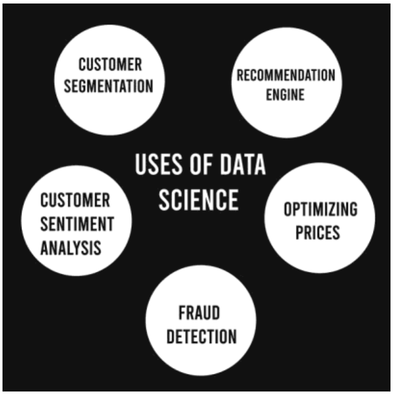


FIGURE 1.4
Uses of data science.

seasons, and so on help to decide sale time [2]. Another area of data science is predictive maintenance. For example, we need to find out whether our car or refrigerator will break down in the next year or two if we prepare to buy a new car or refrigerator. We can potentially apply data science here as well.

Currently, we find the use of data science effectively in politics. Most of the advertisements about election candidatures we see on television may be about elections in the USA or in the UK or even in India. Nowadays everybody is applying data science in elections and trying to capture the votes, or rather the voters influence the voters by personalizing messages or providing personalized messages. The voters use data science to predict who is going to win the elections and with what margin. Probably not all predictions come out to be true, as it depends on the quality of data and prediction techniques used. There are other domains where data science plays an active role as shown in Figure 1.5.

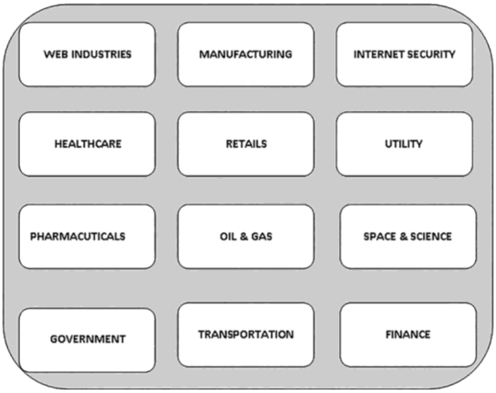


FIGURE 1.5
Data science domains.

1.2 What Is Data Science?

Jonathan Goldman joined LinkedIn in 2006. When he joined, LinkedIn was not as popular as it is today. There was no fancy activity, and people were started joining, and by adding their profiles and sharing other stuff. There was no much activity to do. But it was growing, that is, from several thousand to several million accounts, and they were having the problems of scaling. Jonathan Goldman found that the data gathered is very abnormal and there is something wrong in the structure. The LinkedIn management was just trying to get more people in to connect by searching relative data of peoples. There were no activities, discussion in network as per the interest of members, hence they had started leaving LinkedIn. He concluded something is missing in their data. The engineers of LinkedIn did not take his suggestion seriously. Then, the engineers gave Jonathan a free hand and asked to implement any idea if he has in mind. Jonathan Goldman introduced the idea known as LinkedIn recommendation system. He applied recommendations, and then, actually traffic started growing. Through the recommendation system, they target advertisements for various products. He had several thoughts in his mind for applications to LinkedIn such as co-relation with people. There is a chance that if A knows B and B knows C, then there is a good chance for A to know C. He had many such theories and applied his several ideas to LinkedIn. He is a good statistician and also a good programmer. He was sure that if he applies such concepts, traffic will increase definitely. By adding multiple facilities and interest domains, the LinkedIn management system can realize that traffic got increased since people started surfing, finding interest partners, jobs and business through this system and then LinkedIn started flourishing. So we can guess the power of data. The job executed by Jonathan Goldman in the way he utilized the data is the job of a data scientist. It is about an increase in revenue and about get- ting insights into the data, so the data can recognize the type of data and domains of expertize required. The LinkedIn story is the best example for the role of a data scientist [6]. Today we find the status of LinkedIn. For searching prospective candidate as per emergent requirements, the best possible option is LinkedIn recommendation system

You have the algorithm to deal with data, but it doesn't mean that the algorithm is perfect to analyze the data. If you have more and more data, there is a chance of more and more information and more kinds of scenarios. One of the simple and popular examples of a recommendation system is movies on your past references. If you try to search on the Internet, you will find lots of search spirits to give more information about movies. The increased data if not managed perfectly, it loose the performance of the algorithm, no matter your algorithm is strongly efficient. They can often be beaten simply by having more or exploded data [7].

So what is data science? We take number of decisions in day-to-day life, for example, buying furniture for setting new office. So we start scrutinizing possibilities with different options for searching materials through websites or other survey. The activity is to search for a proper and reliable website for furniture selection from several websites. So, which website to use is the first decision you need to take. We may surf multiple websites, and we can’t discard the websites that don’t sell furniture and stick to those websites that sell furniture. Now within that, we try to find out what are the ratings of the selected website. If the rating is more, that means they are reliable and the quality is probably good, and only then you decide to buy from that website as shown in Figure 1.6 [1].

While surfing website pages for choosing furniture as per requirements, feedback ratings, discounts, promotions, etc. get checked. Some of them will probably not be providing any discount, and some websites will probably be providing discounts. We may go ahead and select the furniture and purchase it [3]. Another example can be booking a CAB. For travelling from location A to location B, need to find best route and optimum travel time it will take. There could be several factors while choosing routes, such as traffic, road conditions, and weather.

Another example is TV shows. Netflix and maybe other TV channels have to perform an analysis to find out what kinds of shows people are watching, what kinds of shows people like, and so on. The data is analyzed and then this information is sent to advertisers because their main source of revenue is advertising. So this is again a major application of data science as shown in Figure 1.7.

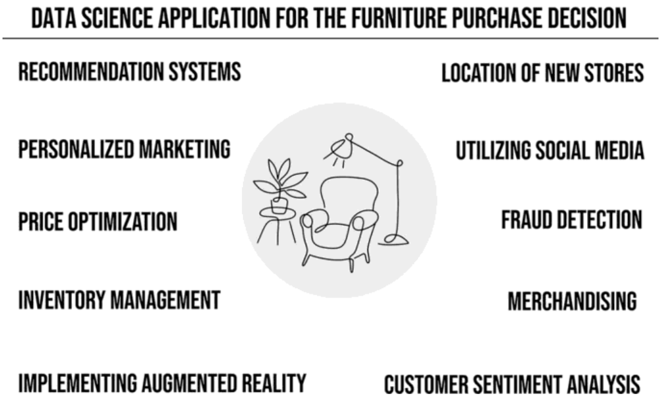


FIGURE 1.6
The furniture purchase decision using data science.

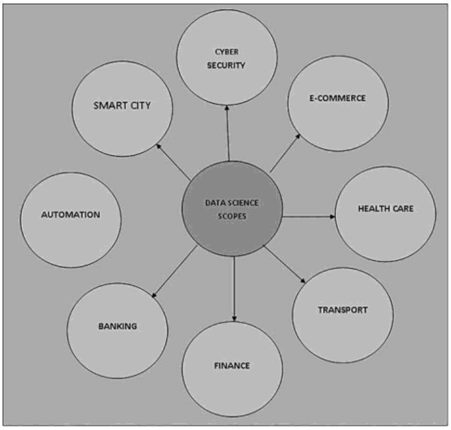


FIGURE 1.7
Scopes of data science.

1.3 Data Science Process

There are various steps in data science. The first step is asking the right question and exploring the data. You must know exactly the problem that needs to be solved. The problem can be understood by asking the right questions. As per the available data for input, the data may be incomplete for which, exploratory analysis can be done. Or data cleansing can be done on raw data for accurate data input which is a part of exploratory analysis. And then, you need to do modeling as shown in Figure 1.8.

Let us say that if you have to perform machine learning, you need to decide which algorithm to use, and through which model, and then you need to train the model accordingly. This is the modeling process, which then runs your data through the model. Then through this process, you come out to the final

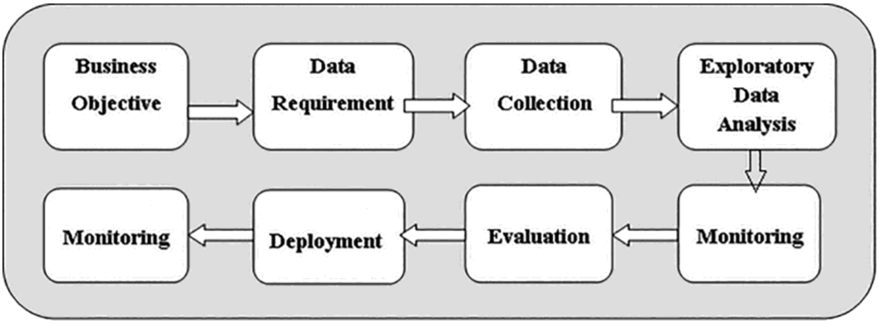


FIGURE 1.8
Data science process.

results of this exercise which includes visualizing the results and preparing a way to communicate the results to the people concerned. So it could be in the form of PowerPoint slides, or it could be in the form of dashboard for proper visualization, so appropriate communication for easily understanding can be possible for intended users.

1.4 Business Intelligence and Data Science

Business intelligence is one of the initial phases where people want to make some sense out of data. Earlier the manufacturing and selling was managed through enterprise resource planning ERP system or customer relation- ship management (CRM) systems. But later people started taking interest in understanding data for information and that's how business intelligence started affecting.

In view of data source, we can compare criteria concerns to methods, skills and focus of the data. For creating business intelligence through data science, structured data were used the ERP or CRM system works on relational database system, RDBMS such as Oracle, SQL and MySQL which use structured data inputs. The ERP or CRM system works on relational database system, RDBMS such as Oracle, SQL and MySQL which use structured data inputs.

All these data are structured in good form of a database table, rows and columns, and they were all brought into a centralized place because it remembers these were still different applications, so they were working on a different database in silos. So, if you want to get a combined view, you need to create what is known as a data warehouse and bring all the data together and then look at it in a uniform way for the required application. The business intelligence uses structured data that reports on dashboards. Data science in addition to structured data also uses a lot of unstructured data, for example, Web or comments. If we are talking about customer feedback, it may be structured if gathered in formatted rubrics or may be unstructured if scripted for performance analysis. That is, data sources may be in different formats.

Business intelligence is analytical in the sense that you have some data, and you are just trying to present the truth out of it. In the case of data science, we go beyond that. Business intelligence uses many statistics such as correlations and regressions to predict what will be the sales maybe in the future. In the case of data science, skills are many more as compared to business intelligence.

The focus of business intelligence is pretty much on historical data. In data science, you take historical data, but you also combine it with some other required information and you also try to predict the future.

1.5 Prerequisites for a Data Scientist

Now moving on to the life of a data scientist, what does a data scientist do during his work? Typically a data scientist is given a problem, i.e., business problem, and he needs to solve it. To do that, he asks the question as to what is the problem that needs to be solved, which is the first thing he gets from a problem. The next thing is to gather the data that is required to solve the problem, so he searches for data from anywhere. In the enterprise, very often the data is not provided in the structured format that he would like to have. The first step is to get possible raw data in any format. The data may be enterprise data or public data. The raw data collected and converted into required format for analysis. The cleaned and formatted data used as an input to statistical model through machine learning mechanisms for communicating analytics results to stakeholders. This is the very high-level view of a day in the life of a data scientist. The role of data scientist is to gather data, process data in required format, manipulates as per requirements and feed for analysis. The analysis helps to build mathematical models or machine learning models which are used to present system behavior to concern stakeholders. There are three essential things required to be a data scientist. The first one is *curiosity*. You should be able to ask the right questions: The first step in data science is asking the question what is the problem you are trying to solve; if you ask the right question, then you will get the right answer. This is very crucial step, since it directly impact on project success. Asking incorrect questions will be lead to incorrect response and may impact on correct analysis. You must ask the right question. A second essential thing that a data scientist should have is the *common sense*. The data scientist needs to be creative in solving problems at hand. In many cases, you may not have sufficient data to conclude; hence, you need to fulfill this gap intelligently, finally, a data scientist should have *communication skills*. After doing all the analysis, if you are unable to communicate the results in the right way, the whole exercise will fail. Communication is the key trait for data scientists. You may be technically brilliant, but if you are unable to communicate the result in the proper way, it will not help. So, these are the three main traits, i.e., curiosity, common sense, and communication skills, as shown in Figure 1.9.

1.6 Components of Data Science

The different components for data science are as follows:

1. **Machine Learning** – Machine learning is the backbone of data science. Data science involves quite a bit of learning the basics of statistics to design a learning algorithm.

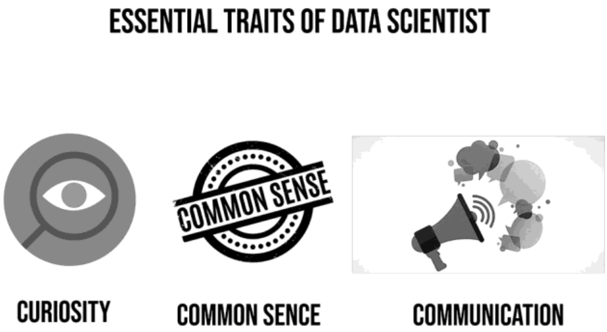


FIGURE 1.9
Essential traits of a data scientist.

2. **Modeling –** Modeling is also a part of machine learning in a way, but you need to be good at identifying what are the algorithms that are more suitable to solve the given problems, which model can be used, and how to train this model.
3. **Statistics and Probabilities –** The statistics and probabilities are like the core foundation of data science. The data scientist must be prominent in statistics and probability theories, so that he can for- mulate the problems for processing the required results. For the data science projects, basic programming skills and some fundamental knowledge of databases is required. The most common programming languages are Python, R, MATLAB, and Octave. In particular, Python is becoming a very popular programming language in data science because of its ease of learning and the multiple libraries that it supports. Python is the most popular language in data science. To begin for data science career, can start python programming since its simplest language and easy to understand for beginners. Then fundamentals of database is another important aspect to learn for handling databases and extract required information from raw data. So, these are some of the key components of data science.

1.7 Benefits and uses of data science and big data

Data science and big data are used almost everywhere in both commercial and non- commercial settings. The number of use cases is vast, and the examples we'll provide throughout this book only scratch the surface of the possibilities.

Commercial companies in almost every industry use data science and big data to gain insights into their customers, processes, staff, completion, and products. Many companies use data science to offer customers a better user experience, as well as to cross-sell, up-sell, and personalize their offerings. A good example of this is Google AdSense, which collects data from internet users so relevant commercial messages can be matched to the person browsing the internet. MaxPoint (http://maxpoint.com/us) is another example of real-time personalized advertising.

Human resource profession- als use people analytics and text mining to screen candidates, monitor the mood of employees, and study informal networks among coworkers. People analytics is the cen- tral theme in the book *Moneyball: The Art of Winning an Unfair Game*. In the book (and movie) we saw that the traditional scouting process for American baseball was ran- dom, and replacing it with correlated signals changed everything. Relying on statistics allowed them to hire the right players and pit them against the opponents where they would have the biggest advantage. Financial institutions use data science to predict stock markets, determine the risk of lending money, and learn how to attract new clients for their services. At the time of writing this book, at least 50% of trades world- wide are performed automatically by machines based on algorithms developed by *quants*, as data scientists who work on trading algorithms are often called, with the help of big data and data science techniques.

Governmental organizations are also aware of data's value. Many governmental organizations not only rely on internal data scientists to discover valuable informa- tion, but also share their data with the public. You can use this data to gain insights or build data-driven applications. *Data.gov* is but one example; it's the home of the US Government's open data. A data scientist in a governmental organization gets to work on diverse projects such as detecting fraud and other criminal activity or optimizing project funding. A well-known example was provided by Edward Snowden, who leaked internal documents of the American National Security Agency and the British Government Communications Headquarters that show clearly how they used data science and big data to monitor millions of individuals. Those organizations collected 5 bil- lion data records from widespread applications such as Google Maps, Angry Birds, email, and text messages, among many other data sources. Then they applied data science techniques to distill information.

Nongovernmental organizations (NGOs) are also no strangers to using data. They use it to raise money and defend their causes. The World Wildlife Fund (WWF), for instance, employs data scientists to increase the effectiveness of their fundraising efforts. Many data scientists devote part of their time to helping NGOs, because NGOs often lack the resources to collect data and employ data scientists. DataKind is one such data scientist group that devotes its time to the benefit of mankind.

Universities use data science in their research but also to enhance the study experience of their students. The rise of massive open online courses (MOOC) produces a lot of data, which allows universities to study how this type of learning can complement traditional classes. MOOCs are an invaluable asset if you want to become a data scientist and big data professional, so definitely look at a few of the better-known ones: Coursera, Udacity, and edX. The big data and data science landscape changes quickly, and MOOCs allow you to stay up to date by following courses from top universities.

1.8 Facets of data

In data science and big data you'll come across many different types of data, and each of them tends to require different tools and techniques. The main categories of data are these:

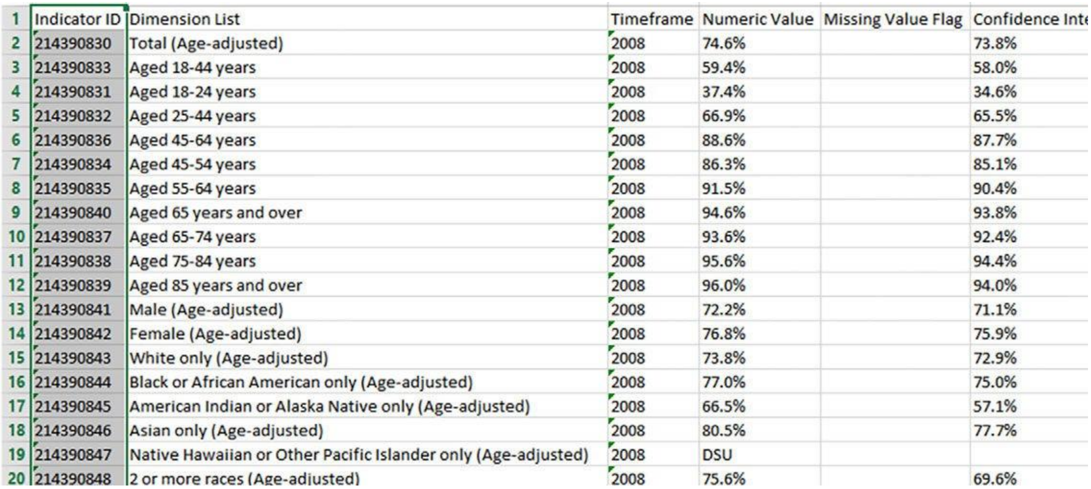
- Structured
- Unstructured
- Natural language
- Machine-generated
- Graph-based
- Audio, video, and images
- Streaming

Let's explore all these interesting data types.

1. Structured data

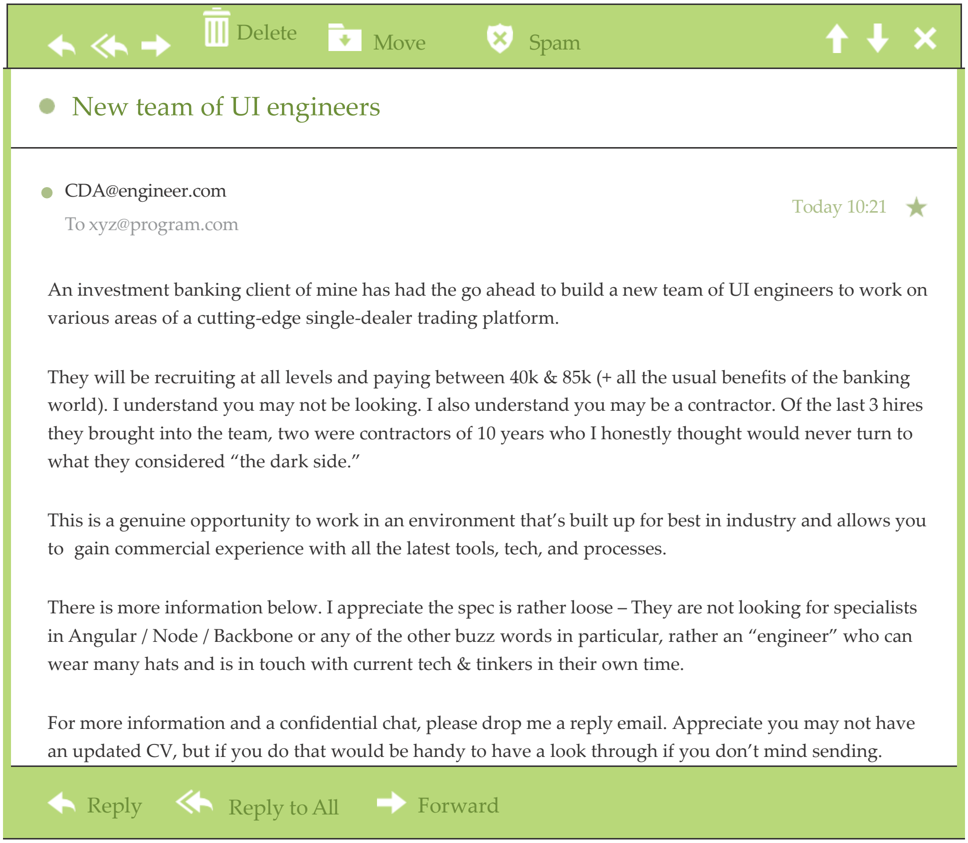
Structured data is data that depends on a data model and resides in a fixed field within a record. As such, it's often easy to store structured data in tables within data- bases or Excel files (figure 1.10). SQL, or Structured Query Language, is the preferred way to manage and query data that resides in databases. You may also come across structured data that might give you a hard time storing it in a traditional relational database. Hierarchical data such as a family tree is one such example.

The world isn't made up of structured data, though; it's imposed upon it by humans and machines. More often, data comes unstructured.



1	Indicator ID	Dimension List	Timeframe	Numeric Value	Missing Value Flag	Confidence Inte
2	214390830	Total (Age-adjusted)	2008	74.6%		73.8%
3	214390833	Aged 18-44 years	2008	59.4%		58.0%
4	214390831	Aged 18-24 years	2008	37.4%		34.6%
5	214390832	Aged 25-44 years	2008	66.9%		65.5%
6	214390836	Aged 45-64 years	2008	88.6%		87.7%
7	214390834	Aged 45-54 years	2008	86.3%		85.1%
8	214390835	Aged 55-64 years	2008	91.5%		90.4%
9	214390840	Aged 65 years and over	2008	94.6%		93.8%
10	214390837	Aged 65-74 years	2008	93.6%		92.4%
11	214390838	Aged 75-84 years	2008	95.6%		94.4%
12	214390839	Aged 85 years and over	2008	96.0%		94.0%
13	214390841	Male (Age-adjusted)	2008	72.2%		71.1%
14	214390842	Female (Age-adjusted)	2008	76.8%		75.9%
15	214390843	White only (Age-adjusted)	2008	73.8%		72.9%
16	214390844	Black or African American only (Age-adjusted)	2008	77.0%		75.0%
17	214390845	American Indian or Alaska Native only (Age-adjusted)	2008	66.5%		57.1%
18	214390846	Asian only (Age-adjusted)	2008	80.5%		77.7%
19	214390847	Native Hawaiian or Other Pacific Islander only (Age-adjusted)	2008	DSU		
20	214390848	2 or more races (Age-adjusted)	2008	75.6%		69.6%

Figure 1.10 An Excel table is an example of structured data.



New team of UI engineers

CDA@engineer.com
To xyz@program.com

Today 10:21

An investment banking client of mine has had the go ahead to build a new team of UI engineers to work on various areas of a cutting-edge single-dealer trading platform.

They will be recruiting at all levels and paying between 40k & 85k (+ all the usual benefits of the banking world). I understand you may not be looking. I also understand you may be a contractor. Of the last 3 hires they brought into the team, two were contractors of 10 years who I honestly thought would never turn to what they considered "the dark side."

This is a genuine opportunity to work in an environment that's built up for best in industry and allows you to gain commercial experience with all the latest tools, tech, and processes.

There is more information below. I appreciate the spec is rather loose – They are not looking for specialists in Angular / Node / Backbone or any of the other buzz words in particular, rather an "engineer" who can wear many hats and is in touch with current tech & tinkers in their own time.

For more information and a confidential chat, please drop me a reply email. Appreciate you may not have an updated CV, but if you do that would be handy to have a look through if you don't mind sending.

Figure 1.11 Email is simultaneously an example of unstructured data and natural language data.

2. Unstructured data

Unstructured data is data that isn't easy to fit into a data model because the content is context-specific or varying. One example of unstructured data is your regular email (figure 1.11). Although email contains structured elements such as the sender, title, and body text, it's a challenge to find the number of people who have written an email complaint about a specific employee because so many ways exist to refer to a person, for example. The thousands of different languages and dialects out there further complicate this.

A human-written email, as shown in figure 1.11, is also a perfect example of natural language data.

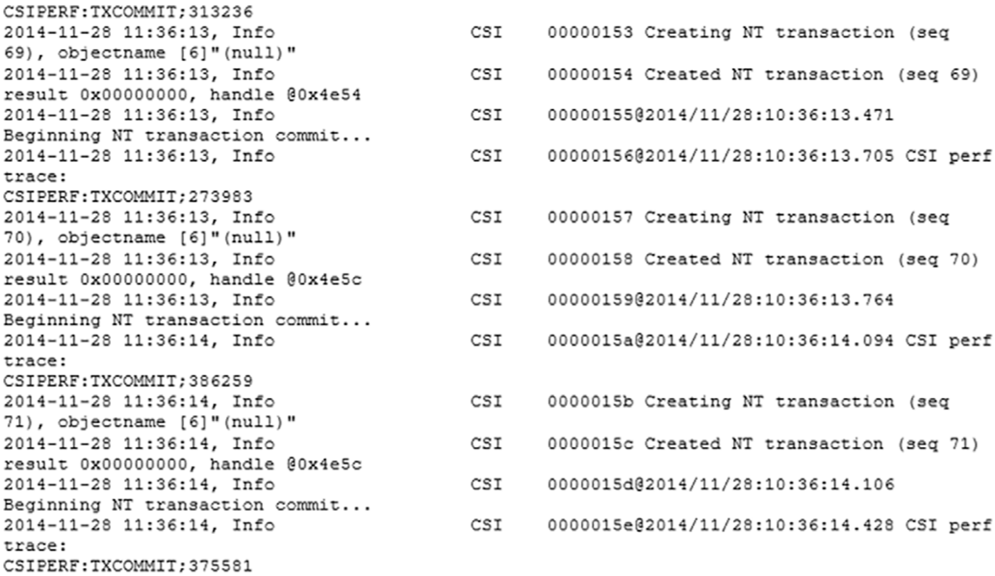
3. Natural language

Natural language is a special type of unstructured data; it's challenging to process because it requires knowledge of specific data science techniques and linguistics.

The natural language processing community has had success in entity recognition, topic recognition, summarization, text completion, and sentiment analysis, but mod- els trained in one domain don't generalize well to other domains. Even state-of-the-art techniques aren't able to decipher the meaning of every piece of text. This shouldn't be a surprise though: humans struggle with natural language as well. It's ambiguous by nature. The concept of meaning itself is questionable here. Have two people listen to the same conversation. Will they get the same meaning? The meaning of the same words can vary when coming from someone upset or joyous.

4. Machine-generated data

Machine-generated data is information that's automatically created by a computer, process, application, or other machine without human intervention. Machine-generated data is becoming a major data resource and will continue to do so. Wikibon has fore- cast that the market value of the *industrial Internet* (a term coined by Frost & Sullivan to refer to the integration of complex physical machinery with networked sensors and software) will be approximately $540 billion in 2020. IDC (International Data Corpo- ration) has estimated there will be 26 times more connected things than people in 2020. This network is commonly referred to as *the internet of things*. The analysis of machine data relies on highly scalable tools, due to its high volume and speed. Examples of machine data are web server logs, call detail records, network event logs, and telemetry (figure 1.12).



```
CSIPERF:TXCOMMIT;313236
2014-11-28 11:36:13, Info                  CSI    00000153 Creating NT transaction (seq
69), objectname [6]"(null)"
2014-11-28 11:36:13, Info                  CSI    00000154 Created NT transaction (seq 69)
result 0x00000000, handle @0x4e54
2014-11-28 11:36:13, Info                  CSI    00000155@2014/11/28:10:36:13.471
Beginning NT transaction commit...
2014-11-28 11:36:13, Info                  CSI    00000156@2014/11/28:10:36:13.705 CSI perf
trace:
CSIPERF:TXCOMMIT;273983
2014-11-28 11:36:13, Info                  CSI    00000157 Creating NT transaction (seq
70), objectname [6]"(null)"
2014-11-28 11:36:13, Info                  CSI    00000158 Created NT transaction (seq 70)
result 0x00000000, handle @0x4e5c
2014-11-28 11:36:13, Info                  CSI    00000159@2014/11/28:10:36:13.764
Beginning NT transaction commit...
2014-11-28 11:36:14, Info                  CSI    0000015a@2014/11/28:10:36:14.094 CSI perf
trace:
CSIPERF:TXCOMMIT;386259
2014-11-28 11:36:14, Info                  CSI    0000015b Creating NT transaction (seq
71), objectname [6]"(null)"
2014-11-28 11:36:14, Info                  CSI    0000015c Created NT transaction (seq 71)
result 0x00000000, handle @0x4e5c
2014-11-28 11:36:14, Info                  CSI    0000015d@2014/11/28:10:36:14.106
Beginning NT transaction commit...
2014-11-28 11:36:14, Info                  CSI    0000015e@2014/11/28:10:36:14.428 CSI perf
trace:
CSIPERF:TXCOMMIT;375581
```

Figure 1.12 Example of machine-generated data

The machine data shown in figure 1.3 would fit nicely in a classic table-structured database. This isn't the best approach for highly interconnected or "networked" data, where the relationships between entities have a valuable role to play.

5. Graph-based or network data

"Graph data" can be a confusing term because any data can be shown in a graph. "Graph" in this case points to mathematical *graph theory*. In graph theory, a graph is a mathematical structure to model pair-wise relationships between objects. Graph or network data is, in short, data that focuses on the relationship or adjacency of objects. The graph structures use nodes, edges, and properties to represent and store graphical data. Graph-based data is a natural way to represent social networks, and its structure allows you to calculate specific metrics such as the influence of a person and the shortest path between two people. Examples of graph-based data can be found on many social media websites (figure 1.13). For instance, on LinkedIn you can see who you know at which company. Your follower list on Twitter is another example of graph-based data. The power and sophistication comes from multiple, overlapping graphs of the same nodes. For example, imagine the connecting edges here to show "friends" on Facebook. Imagine another graph with the same people which connects business colleagues via LinkedIn. Imagine a third graph based on movie interests on Netflix. Overlapping the three different-looking graphs makes more interesting questions possible.

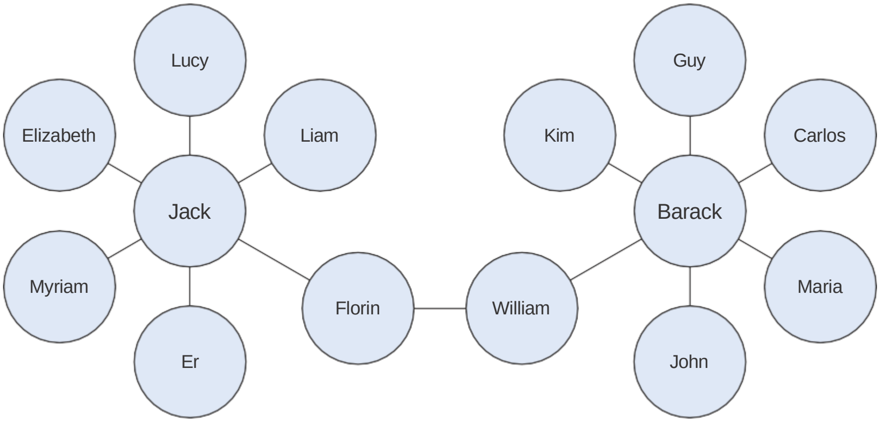


Figure 1.13 Friends in a social network are an example of graph-based data.

Graph databases are used to store graph-based data and are queried with specialized query languages such as SPARQL. Graph data poses its challenges, but for a computer interpreting additive and image data, it can be even more difficult.

6. Audio, image, and video

Audio, image, and video are data types that pose specific challenges to a data scientist. Tasks that are trivial for humans, such as recognizing objects in pictures, turn out to be challenging for computers. MLBAM (Major League Baseball Advanced Media) announced in 2014 that they'll increase video capture to approximately 7 TB per game for the purpose of live, in-game analytics. High-speed cameras at stadiums will capture ball and athlete movements to calculate in real time, for example, the path taken by a defender relative to two baselines.

Recently a company called DeepMind succeeded at creating an algorithm that's capable of learning how to play video games. This algorithm takes the video screen as input and learns to interpret everything via a complex process of deep learning. It's a remarkable feat that prompted Google to buy the company for their own Artificial Intelligence (AI) development plans. The learning algorithm takes in data as it's pro- duced by the computer game; it's streaming data.

7. Streaming data

While streaming data can take almost any of the previous forms, it has an extra property. The data flows into the system when an event happens instead of being loaded into a data store in a batch. Although this isn't really a different type of data, we treat it here as such because you need to adapt your process to deal with this type of information.

Examples are the "What's trending" on Twitter, live sporting or music events, and the stock market.

1.9 The data science process

The data science process typically consists of six steps, as you can see in the mind map in figure 1.14. We will introduce them briefly here.

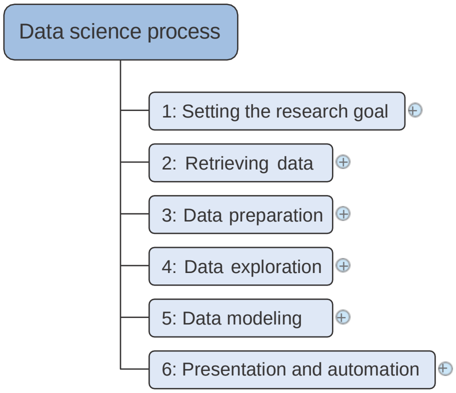


Figure 1.14 The data science process

1. Setting the research goal

Data science is mostly applied in the context of an organization. When the business asks you to perform a data science project, you'll first prepare a project charter. This charter contains information such as what you're going to research, how the company benefits from that, what data and resources you need, a timetable, and deliverables.

2. Retrieving data

   The second step is to collect data. You've stated in the project charter which data you need and where you can find it. In this step you ensure that you can use the data in your program, which means checking the existence of, quality, and access to the data. Data can also be delivered by third-party companies and takes many forms ranging from Excel spreadsheets to different types of databases.

3. Data preparation

   Data collection is an error-prone process; in this phase you enhance the quality of the data and prepare it for use in subsequent steps. This phase consists of three sub- phases: *data cleansing* removes false values from a data source and inconsistencies across data sources, *data integration* enriches data sources by combining information from multiple data sources, and *data transformation* ensures that the data is in a suit- able format for use in your models.

4. Data exploration

   Data exploration is concerned with building a deeper understanding of your data. You try to understand how variables interact with each other, the distribution of the data, and whether there are outliers. To achieve this you mainly use descriptive statis- tics, visual techniques, and simple modeling. This step often goes by the abbreviation EDA, for Exploratory Data Analysis.

5. Data modeling or model building

   In this phase you use models, domain knowledge, and insights about the data you found in the previous steps to answer the research question. You select a technique from the fields of statistics, machine learning, operations research, and so on. Building a model is an iterative process that involves selecting the variables for the model, executing the model, and model diagnostics.

6. Presentation and automation

   Finally, you present the results to your business. These results can take many forms, ranging from presentations to research reports. Sometimes you'll need to automate the execution of the process because the business will want to use the insights you gained in another project or enable an operational process to use the outcome from your model.

2 The data science process

2.1 Overview of the data science process

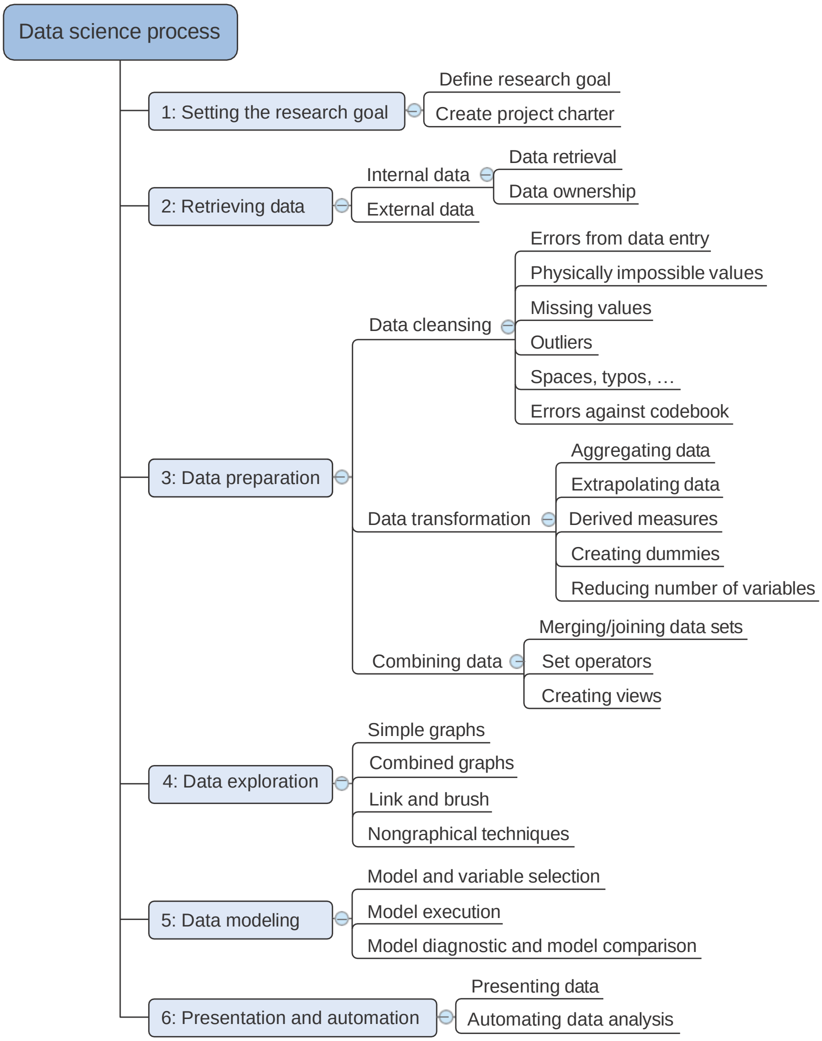


Figure 2.1 The six steps of the data science process

Figure 2.1 summarizes the data science process and shows the main steps and actions you'll take during a project. The following list is a short introduction; each of the steps will be discussed in greater depth throughout this chapter.

1 The first step of this process is setting a *research goal*. The main purpose here is making sure all the stakeholders understand the *what*, *how*, and *why* of the project. In every serious project this will result in a project charter.

2The second phase is *data retrieval*. You want to have data available for analysis, so this step includes finding suitable data and getting access to the data from the data owner. The result is data in its raw form, which probably needs polishing and transformation before it becomes usable.

3 Now that you have the raw data, it's time to *prepare* it. This includes transforming the data from a raw form into data that's directly usable in your models. To achieve this, you'll detect and correct different kinds of errors in the data, com- bine data from different data sources, and transform it. If you have successfully completed this step, you can progress to data visualization and modeling.

4 The fourth step is *data exploration*. The goal of this step is to gain a deep under- standing of the data. You'll look for patterns, correlations, and deviations based on visual and descriptive techniques. The insights you gain from this phase will enable you to start modeling.

5 Finally, we get to the sexiest part: *model building* (often referred to as "data modeling" throughout this book). It is now that you attempt to gain the insights or make the predictions stated in your project charter. Now is the time to bring out the heavy guns, but remember research has taught us that often (but not always) a combination of simple models tends to outperform one complicated model. If you've done this phase right, you're almost done.

6 The last step of the data science model is *presenting your results and automating the analysis,* if needed. One goal of a project is to change a process and/or make better decisions. You may still need to convince the business that your findings will indeed change the business process as expected. This is where you can shine in your influencer role. The importance of this step is more apparent in projects on a strategic and tactical level. Certain projects require you to per- form the business process over and over again, so automating the project will save time.

In reality you won't progress in a linear way from step 1 to step 6. Often you'll regress and iterate between the different phases.

Following these six steps pays off in terms of a higher project success ratio and increased impact of research results. This process ensures you have a well-defined research plan, a good understanding of the business question, and clear deliverables before you even start looking at data. The first steps of your process focus on getting high-quality data as input for your models. This way your models will perform better later on. In data science there's a well-known saying: *Garbage in equals garbage out*.

Another benefit of following a structured approach is that you work more in *prototype mode* while you search for the best model. When building a *prototype,* you'll probably try multiple models and won't focus heavily on issues such as program speed or writing code against standards. This allows you to focus on bringing business value instead.

Not every project is initiated by the business itself. Insights learned during analysis or the arrival of new data can spawn new projects. When the data science team generates an idea, work has already been done to make a proposition and find a business sponsor.

Dividing a project into smaller stages also allows employees to work together as a team. It's impossible to be a specialist in everything. You'd need to know how to upload all the data to all the different databases, find an optimal data scheme that works not only for your application but also for other projects inside your company, and then keep track of all the statistical and data-mining techniques, while also being an expert in presentation tools and business politics. That's a hard task, and it's why more and more companies rely on a team of specialists rather than trying to find one person who can do it all.

The process we described in this section is best suited for a data science project that contains only a few models. It's not suited for every type of project. For instance, a project that contains millions of real-time models would need a different approach than the flow we describe here. A beginning data scientist should get a long way fol- lowing this manner of working, though.

2.1.1 Don't be a slave to the process

Not every project will follow this blueprint, because your process is subject to the preferences of the data scientist, the company, and the nature of the project you work on. Some companies may require you to follow a strict protocol, whereas others have a more informal manner of working. In general, you'll need a structured approach when you work on a complex project or when many people or resources are involved.

The *agile* project model is an alternative to a sequential process with iterations. As this methodology wins more ground in the IT department and throughout the company, it's also being adopted by the data science community. Although the agile methodology is suitable for a data science project, many company policies will favor a more rigid approach toward data science.

Planning every detail of the data science process upfront isn't always possible, and more often than not you'll iterate between the different steps of the process. For instance, after the briefing you start your normal flow until you're in the exploratory data analysis phase. Your graphs show a distinction in the behavior between two groups—men and women maybe? You aren't sure because you don't have a variable that indicates whether the customer is male or female. You need to retrieve an extra data set to confirm this. For this you need to go through the approval process, which indicates that you (or the business) need to provide a kind of project char- ter. In big companies, getting all the data you need to finish your project can be an ordeal.

2.2 Step 1: Defining research goals and creating a project charter

A project starts by understanding the *what*, the *why*, and the *how* of your project (figure 2.2). What does the company expect you to do? And why does management place such a value on your research? Is it part of a bigger strategic picture or a "lone wolf" project originating from an opportunity someone detected? Answering these three questions (what, why, how) is the goal of the first phase, so that everybody knows what to do and can agree on the best course of action.

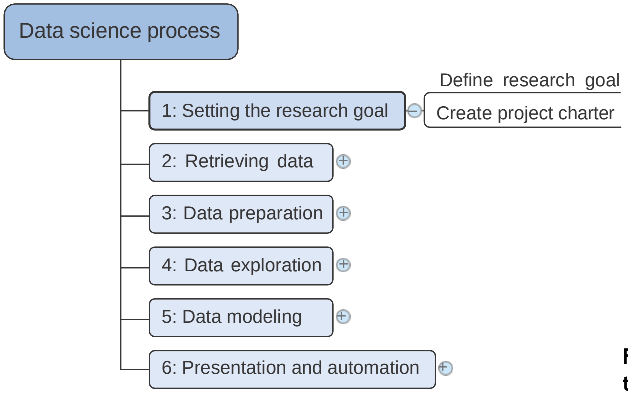


Figure 2.2 Step 1: Setting the research goal

The outcome should be a clear research goal, a good understanding of the con- text, well-defined deliverables, and a plan of action with a timetable. This information is then best placed in a project charter. The length and formality can, of course, differ between projects and companies. In this early phase of the project, people skills and business acumen are more important than great technical prowess, which is why this part will often be guided by more senior personnel.

2.2.1 Spend time understanding the goals and context of your research

An essential outcome is the research goal that states the purpose of your assignment in a clear and focused manner. Understanding the business goals and context is critical for project success. Continue asking questions and devising examples until you grasp the exact business expectations, identify how your project fits in the bigger picture, appreciate how your research is going to change the business, and understand how they'll use your results. Nothing is more frustrating than spending months researching something until you have that one moment of brilliance and solve the problem, but when you report your findings back to the organization, everyone immediately realizes that you misunderstood their question. Don't skim over this phase lightly. Many data scientists fail here:

despite their mathematical wit and scientific brilliance, they never seem to grasp the business goals and context.

2.2.2 Create a project charter

Clients like to know upfront what they're paying for, so after you have a good under- standing of the business problem, try to get a formal agreement on the deliverables. All this information is best collected in a project charter. For any significant project this would be mandatory.

A project charter requires teamwork, and your input covers at least the following:

- A clear research goal
- The project mission and context
- How you're going to perform your analysis
- What resources you expect to use
- Proof that it's an achievable project, or proof of concepts
- Deliverables and a measure of success
- A timeline

Your client can use this information to make an estimation of the project costs and the data and people required for your project to become a success.

2.3 Step 2: Retrieving data

The next step in data science is to retrieve the required data (figure 2.3). Sometimes you need to go into the field and design a data collection process yourself, but most of the time you won't be involved in this step. Many companies will have already collected and stored the data for you, and what they don't have can often be bought from third parties. Don't be afraid to look outside your organization for data, because more and more organizations are making even high-quality data freely available for public and commercial use.

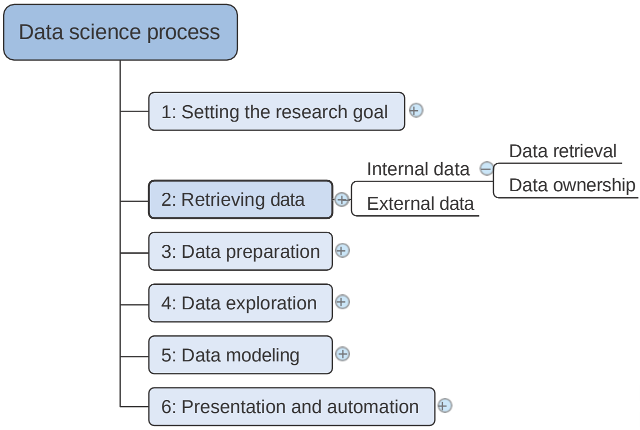


Figure 2.3 Step 2: Retrieving data

Data can be stored in many forms, ranging from simple text files to tables in a data- base. The objective now is acquiring all the data you need. This may be difficult, and even if you succeed, data is often like a diamond in the rough: it needs polishing to be of any use to you.

2.3.1 Start with data stored within the company

Your first act should be to assess the relevance and quality of the data that's readily available within your company. Most companies have a program for maintaining key data, so much of the cleaning work may already be done. This data can be stored in official data repositories such as *databases*, *data marts*, *data warehouses*, and *data lakes* maintained by a team of IT professionals. The primary goal of a database is data storage, while a data warehouse is designed for reading and analyzing that data. A data mart is a subset of the data warehouse and geared toward serving a specific business unit. While data warehouses and data marts are home to preprocessed data, data lakes contains data in its natural or raw format. But the possibility exists that your data still resides in Excel files on the desktop of a domain expert.

Finding data even within your own company can sometimes be a challenge. As companies grow, their data becomes scattered around many places. Knowledge of the data may be dispersed as people change positions and leave the company. Documentation and metadata aren't always the top priority of a delivery manager, so it's possible you'll need to develop some Sherlock Holmes–like skills to find all the lost bits.

Getting access to data is another difficult task. Organizations understand the value and sensitivity of data and often have policies in place so everyone has access to what they need and nothing more. These policies translate into physical and digital barriers called *Chinese walls*. These "walls" are mandatory and well-regulated for customer data in most countries. This is for good reasons, too; imagine everybody in a credit card company having access to your spending habits. Getting access to the data may take time and involve company politics.

2.3.2 Don't be afraid to shop around

If data isn't available inside your organization, look outside your organization's walls. Many companies specialize in collecting valuable information. For instance, Nielsen and GFK are well known for this in the retail industry. Other companies provide data so that you, in turn, can enrich their services and ecosystem. Such is the case with Twitter, LinkedIn, and Facebook.

Although data is considered an asset more valuable than oil by certain companies, more and more governments and organizations share their data for free with the world. This data can be of excellent quality; it depends on the institution that creates and manages it. The information they share covers a broad range of topics such as the number of accidents or amount of drug abuse in a certain region and its demograph- ics. This data is helpful when you want to enrich proprietary data but also convenient when training your data science skills at home. Table 2.1 shows only a small selection from the growing number of open-data providers.

Table 2.1 A list of open-data providers that should get you started

Open data site	Description
Data.gov	The home of the US Government's open data
https://open-data.europa.eu/	The home of the European Commission's open data
Freebase.org	An open database that retrieves its information from sites like Wikipedia, MusicBrains, and the SEC archive
Data.worldbank.org	Open data initiative from the World Bank
Aiddata.org	Open data for international development
Open.fda.gov	Open data from the US Food and Drug Administration

2.3.3 Do data quality checks now to prevent problems later

Expect to spend a good portion of your project time doing data correction and cleans- ing, sometimes up to 80%. The retrieval of data is the first time you'll inspect the data in the data science process. Most of the errors you'll encounter during the data- gathering phase are easy to spot, but being too careless will make you spend many hours solving data issues that could have been prevented during data import.

You'll investigate the data during the import, data preparation, and exploratory phases. The difference is in the goal and the depth of the investigation. During *data retrieval*, you check to see if the data is equal to the data in the source document and look to see if you have the right data types. This shouldn't take too long; when you have enough evidence that the data is similar to the data you find in the source document, you stop. With *data preparation*, you do a more elaborate check. If you did a good job during the previous phase, the errors you find now are also present in the source document. The focus is on the content of the variables: you want to get rid of typos and other data entry errors and bring the data to a common standard among the data sets. For example, you might correct USQ to USA and United Kingdom to UK. During the *exploratory phase* your focus shifts to what you can learn from the data. Now you assume the data to be clean and look at the statistical properties such as distributions, correlations, and outliers. You'll often iterate over these phases. For instance, when you discover outliers in the exploratory phase, they can point to a data entry error.

2.4 Step 3: Cleansing, integrating, and transforming data

The data received from the data retrieval phase is likely to be "a diamond in the rough." Your task now is to sanitize and prepare it for use in the modeling and reporting phase. Doing so is tremendously important because your models will perform better and you'll lose less time trying to fix strange output. It can't be mentioned nearly enough times: garbage in equals garbage out. Your model needs the data in a specific format, so data transformation will always come into play. It's a good habit to correct data errors as early on in the process as possible. However, this isn't always possible in a realistic setting, so you'll need to take corrective actions in your program.

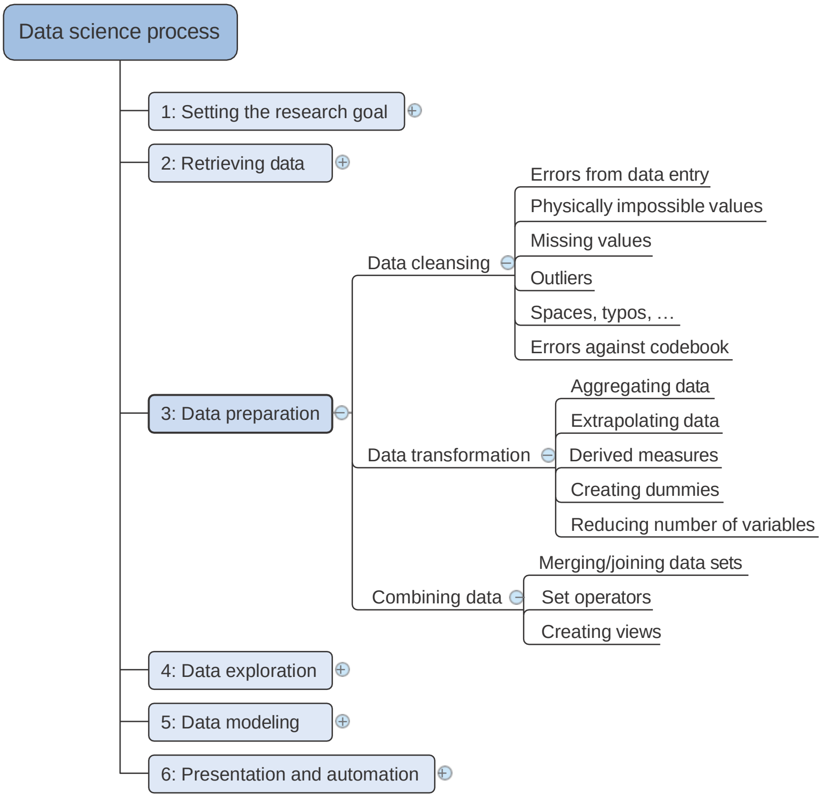


Figure 2.4 Step 3: Data preparation

Figure 2.4 shows the most common actions to take during the data cleansing, inte- gration, and transformation phase. This mind map may look a bit abstract for now, but we'll handle all of these points in more detail in the next sections. You'll see a great commonality among all of these actions.

2.4.1 Cleansing data

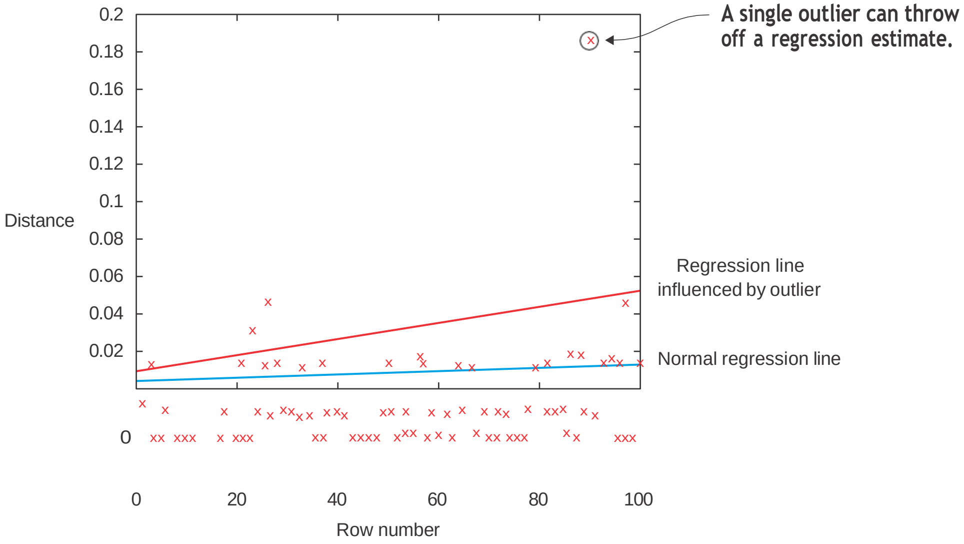
Data cleansing is a subprocess of the data science process that focuses on removing errors in your data so your data becomes a true and consistent representation of the processes it originates from.

By “true and consistent representation” we imply that at least two types of errors exist. The first type is the *interpretation error*, such as when you take the value in your data for granted, like saying that a person’s age is greater than 300 years. The second type of error points to *inconsistencies* between data sources or against your company’s standardized values. An example of this class of errors is putting “Female” in one table and “F” in another when they represent the same thing: that the person is female. Another example is that you use Pounds in one table and Dollars in another. Too many possible errors exist for this list to be exhaustive, but table 2.2 shows an overview of the types of errors that can be detected with easy checks—the “low hanging fruit,” as it were.

Table 2.2 An overview of common errors

General solution	
Try to fix the problem early in the data acquisition chain or else fix it in the program.	
Error description	**Possible solution**
Errors pointing to false values within one data set	
Mistakes during data entry	Manual overrules
Redundant white space	Use string functions
Impossible values	Manual overrules
Missing values	Remove observation or value
Outliers	Validate and, if erroneous, treat as missing value (remove or insert)
Errors pointing to inconsistencies between data sets	
Deviations from a code book	Match on keys or else use manual overrules
Different units of measurement	Recalculate
Different levels of aggregation	Bring to same level of measurement by aggregation or extrapolation

Sometimes you’ll use more advanced methods, such as simple modeling, to find and identify data errors; diagnostic plots can be especially insightful. For example, in fig- ure 2.5 we use a measure to identify data points that seem out of place. We do a regression to get acquainted with the data and detect the influence of individual observations on the regression line. When a single observation has too much influ- ence, this can point to an error in the data, but it can also be a valid point. At the data cleansing stage, these advanced methods are, however, rarely applied and often regarded by certain data scientists as overkill.



A single outlier can throw off a regression estimate.

Figure 2.5 The encircled point influences the model heavily and is worth investigating because it can point to a region where you don't have enough data or might indicate an error in the data, but it also can be a valid data point.

Data entry errors

Data collection and data entry are error-prone processes. They often require human intervention, and because humans are only human, they make typos or lose their con- centration for a second and introduce an error into the chain. But data collected by machines or computers isn't free from errors either. Errors can arise from human sloppiness, whereas others are due to machine or hardware failure. Examples of errors originating from machines are transmission errors or bugs in the extract, trans- form, and load phase (ETL).

For small data sets you can check every value by hand. Detecting data errors when the variables you study don't have many classes can be done by tabulating the data with counts. When you have a variable that can take only two values: "Good" and "Bad", you can create a frequency table and see if those are truly the only two values present. In table 2.3, the values "Godo" and "Bade" point out something went wrong in at least 16 cases.

Table 2.3 Detecting outliers on simple variables with a frequency table

Value	Count
Good	1598647
Bad	1354468
Godo	15
Bade	1

Most errors of this type are easy to fix with simple assignment statements and if-then- else rules:

```
if x == "Godo":
    x = "Good"
if x == "Bade":
    x = "Bad"
```

Redundant whitespace

Whitespaces tend to be hard to detect but cause errors like other redundant characters would. Who hasn't lost a few days in a project because of a bug that was caused by whitespaces at the end of a string? You ask the program to join two keys and notice that observations are missing from the output file. After looking for days through the code, you finally find the bug. Then comes the hardest part: explaining the delay to the project stakeholders. The cleaning during the ETL phase wasn't well executed, and keys in one table contained a whitespace at the end of a string. This caused a mismatch of keys such as "FR " – "FR", dropping the observations that couldn't be matched.

If you know to watch out for them, fixing redundant whitespaces is luckily easy enough in most programming languages. They all provide string functions that will remove the leading and trailing whitespaces. For instance, in Python you can use the `strip()` function to remove leading and trailing spaces.

> **FIXING CAPITAL LETTER MISMATCHES** Capital letter mismatches are common. Most programming languages make a distinction between "Brazil" and "bra- zil". In this case you can solve the problem by applying a function that returns both strings in lowercase, such as `.lower()` in Python. `"Brazil".lower() == "brazil".lower()` should result in `true`.

Impossible values and sanity checks

Sanity checks are another valuable type of data check. Here you check the value against physically or theoretically impossible values such as people taller than 3 meters or someone with an age of 299 years. Sanity checks can be directly expressed with rules:

```
check = 0 <= age <= 120
```

Outliers

An outlier is an observation that seems to be distant from other observations or, more specifically, one observation that follows a different logic or generative process than the other observations. The easiest way to find outliers is to use a plot or a table with the minimum and maximum values. An example is shown in figure 2.6.

The plot on the top shows no outliers, whereas the plot on the bottom shows possible outliers on the upper side when a normal distribution is expected. The normal distribution, or Gaussian distribution, is the most common distribution in natural sciences.

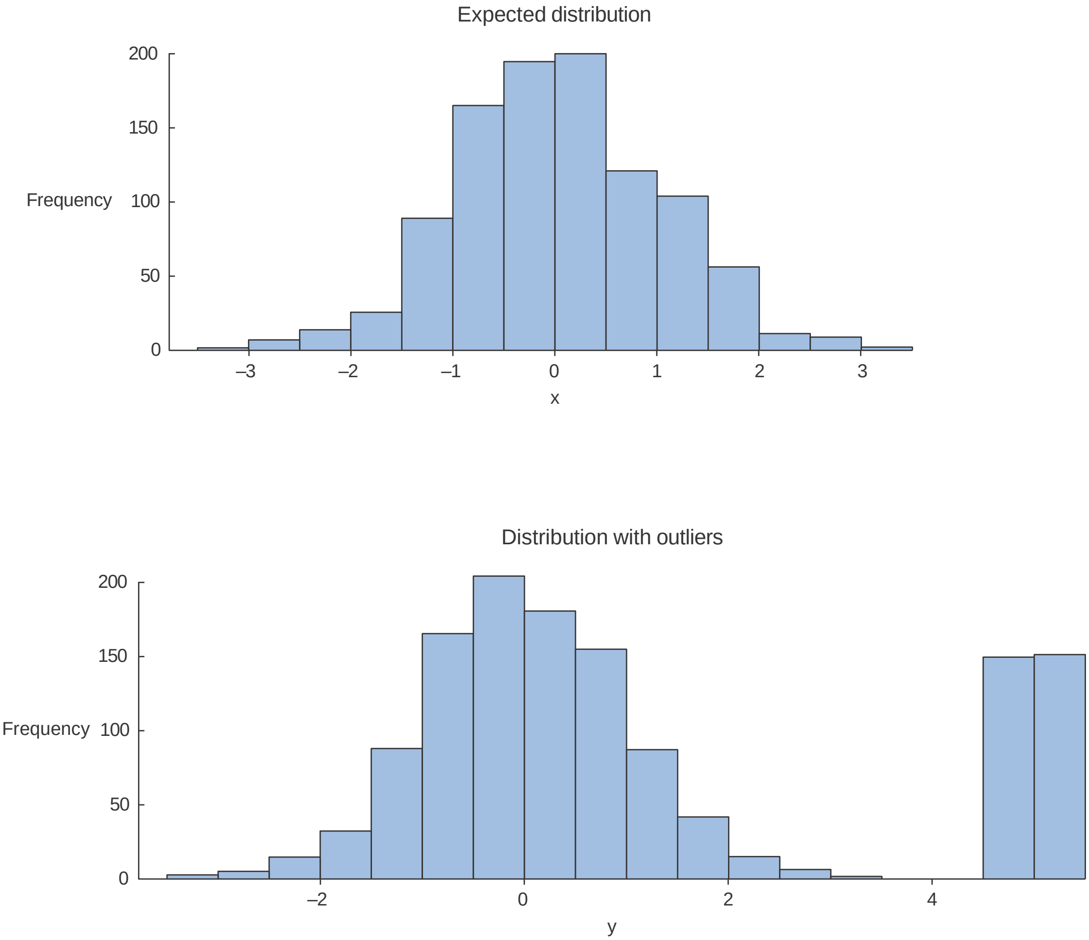


Figure 2.6 Distribution plots are helpful in detecting outliers and helping you understand the variable.

It shows most cases occurring around the average of the distribution and the occurrences decrease when further away from it. The high values in the bottom graph can point to outliers when assuming a normal distribution. As we saw earlier with the regression example, outliers can gravely influence your data modeling, so investigate them first.

Dealing with missing values

Missing values aren't necessarily wrong, but you still need to handle them separately; certain modeling techniques can't handle missing values. They might be an indicator that something went wrong in your data collection or that an error happened in the ETL process. Common techniques data scientists use are listed in table 2.4.

Table 2.4 An overview of techniques to handle missing data

Technique	Advantage	Disadvantage
Omit the values	Easy to perform	You lose the information from an observation
Set value to `null`	Easy to perform	Not every modeling technique and/or implementation can handle `null` values
Impute a static value such as 0 or the mean	Easy to perform You don't lose information from the other variables in the observation	Can lead to false estimations from a model
Impute a value from an estimated or theoretical distribution	Does not disturb the model as much	Harder to execute You make data assumptions
Modeling the value (nondepen- dent)	Does not disturb the model too much	Can lead to too much confidence in the model Can artificially raise dependence among the variables Harder to execute You make data assumptions

Which technique to use at what time is dependent on your particular case. If, for instance, you don't have observations to spare, omitting an observation is probably not an option. If the variable can be described by a stable distribution, you could impute based on this. However, maybe a missing value actually means "zero"? This can be the case in sales for instance: if no promotion is applied on a customer basket, that customer's promo is missing, but most likely it's also 0, no price cut.

Deviations from a code book

Detecting errors in larger data sets against a code book or against standardized values can be done with the help of set operations. A code book is a description of your data, a form of metadata. It contains things such as the number of variables per observa- tion, the number of observations, and what each encoding within a variable means. (For instance "0" equals "negative", "5" stands for "very positive".) A code book also tells the type of data you're looking at: is it hierarchical, graph, something else?

You look at those values that are present in set A but not in set B. These are values that should be corrected. It's no coincidence that *sets* are the data structure that we'll use when we're working in code. It's a good habit to give your data structures addi- tional thought; it can save work and improve the performance of your program.

If you have multiple values to check, it's better to put them from the code book into a table and use a difference operator to check the discrepancy between both tables.

DIFFERENT UNITS OF MEASUREMENT

When integrating two data sets, you have to pay attention to their respective units of measurement. An example of this would be when you study the prices of gasoline in the world. To do this you gather data from different data providers. Data sets can contain prices per gallon and others can contain prices per liter. A simple conversion will do the trick in this case.

DIFFERENT LEVELS OF AGGREGATION

Having different levels of aggregation is similar to having different types of measurement. An example of this would be a data set containing data per week versus one containing data per work week. This type of error is generally easy to detect, and *summarizing* (or the inverse, *expanding*) the data sets will fix it.

After cleaning the data errors, you combine information from different data sources. But before we tackle this topic we'll take a little detour and stress the importance of cleaning data as early as possible.

2.4.2 Correct errors as early as possible

A good practice is to mediate data errors as early as possible in the data collection chain and to fix as little as possible inside your program while fixing the origin of the problem. Retrieving data is a difficult task, and organizations spend millions of dollars on it in the hope of making better decisions. The data collection process is error- prone, and in a big organization it involves many steps and teams.

Data should be cleansed when acquired for many reasons:

- Not everyone spots the data anomalies. Decision-makers may make costly mis- takes on information based on incorrect data from applications that fail to correct for the faulty data.
- If errors are not corrected early on in the process, the cleansing will have to be done for every project that uses that data.
- Data errors may point to a business process that isn't working as designed. For instance, both authors worked at a retailer in the past, and they designed a couponing system to attract more people and make a higher profit. During a data science project, we discovered clients who abused the couponing system and earned money while purchasing groceries. The goal of the couponing system was to stimulate cross-selling, not to give products away for free. This flaw cost the company money and nobody in the company was aware of it. In this case the data wasn't technically wrong but came with unexpected results.
- Data errors may point to defective equipment, such as broken transmission lines and defective sensors.
- Data errors can point to bugs in software or in the integration of software

> that may be critical to the company. While doing a small project at a bank we discovered that two software applications used different local settings. This caused problems with numbers greater than 1,000. For one app the number 1.000 meant one, and for the other it meant one thousand.

Fixing the data as soon as it's captured is nice in a perfect world. Sadly, a data scientist doesn't always have a say in the data collection and simply telling the IT department to fix certain things may not make it so. If you can't correct the data at the source, you'll need to handle it inside your code. Data manipulation doesn't end with correcting mistakes; you still need to combine your incoming data.

As a final remark: always keep a copy of your original data (if possible). Sometimes you start cleaning data but you'll make mistakes: impute variables in the wrong way, delete outliers that had interesting additional information, or alter data as the result of an initial misinterpretation. If you keep a copy you get to try again. For "flowing data" that's manipulated at the time of arrival, this isn't always possible and you'll have accepted a period of tweaking before you get to use the data you are capturing. One of the more difficult things isn't the data cleansing of individual data sets however, it's combining different sources into a whole that makes more sense.

2.4.3 Combining data from different data sources

Your data comes from several different places, and in this substep we focus on integrating these different sources. Data varies in size, type, and structure, ranging from databases and Excel files to text documents.

We focus on data in table structures in this chapter for the sake of brevity. It's easy to fill entire books on this topic alone, and we choose to focus on the data science process instead of presenting scenarios for every type of data. But keep in mind that other types of data sources exist, such as key-value stores, document stores, and so on, which we'll handle in more appropriate places in the book.

THE DIFFERENT WAYS OF COMBINING DATA

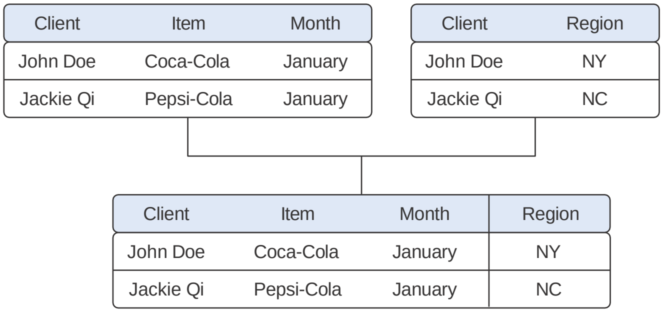
You can perform two operations to combine information from different data sets. The first operation is *joining*: enriching an observation from one table with information from another table. The second operation is *appending* or *stacking*: adding the observations of one table to those of another table.

When you combine data, you have the option to create a new physical table or a virtual table by creating a view. The advantage of a view is that it doesn't consume more disk space. Let's elaborate a bit on these methods.

JOINING TABLES

Joining tables allows you to combine the information of one observation found in one table with the information that you find in another table. The focus is on enriching a single observation. Let's say that the first table contains information about the purchases of a customer and the other table contains information about the region where your customer lives. Joining the tables allows you to combine the information so that you can use it for your model, as shown in figure 2.7.

To join tables, you use variables that represent the same object in both tables, such as a date, a country name, or a Social Security number. These common fields are known as keys. When these keys also uniquely define the records in the table they are called *primary keys*. One table may have buying behavior and the other table may have demographic information on a person. In figure 2.7 both tables contain the client name, and this makes it easy to enrich the client expenditures with the region of the client. People who are acquainted with Excel will notice the similarity with using a lookup function. The number of resulting rows in the output table depends on the exact join type that you use.



Client	Item	Month
John Doe	Coca-Cola	January
Jackie Qi	Pepsi-Cola	January

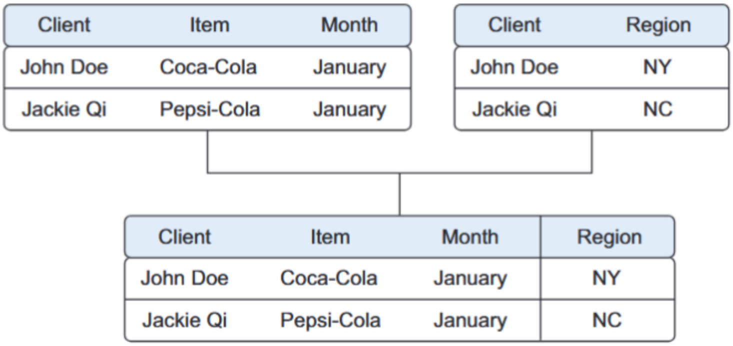
Client	Region
John Doe	NY
Jackie Qi	NC

Client	Item	Month	Region
John Doe	Coca-Cola	January	NY
Jackie Qi	Pepsi-Cola	January	NC

Figure 2.7 Joining two tables on the Item and Region keys

Appending tables

Appending or stacking tables is effectively adding observations from one table to another table. Figure 2.8 shows an example of appending tables. One table contains the observations from the month January and the second table contains observations from the month February. The result of appending these tables is a larger one with the observations from January as well as February. The equivalent operation in set theory would be the union, and this is also the command in SQL, the common language of relational databases. Other set operators are also used in data science, such as set difference and intersection.



Client	Item	Month
John Doe	Coca-Cola	January
Jackie Qi	Pepsi-Cola	January

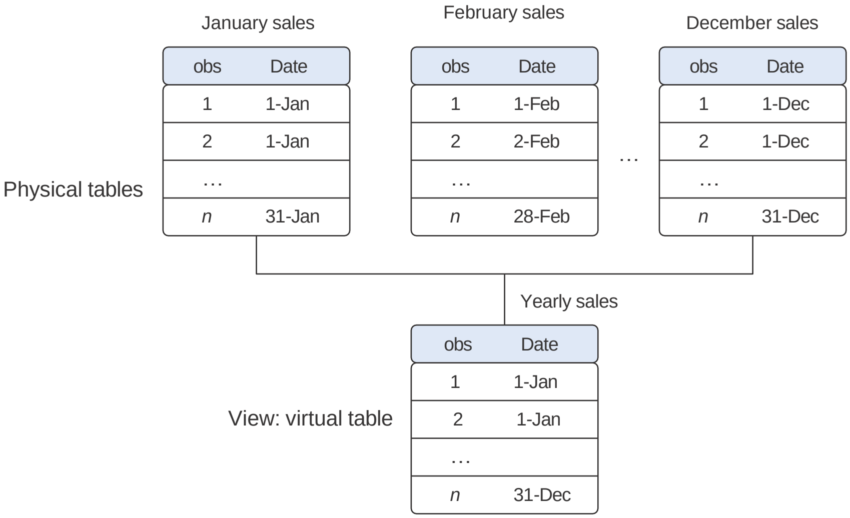
Client	Region
John Doe	NY
Jackie Qi	NC

Client	Item	Month	Region
John Doe	Coca-Cola	January	NY
Jackie Qi	Pepsi-Cola	January	NC

Figure 2.7 Joining two tables on the Item and Region keys

USING VIEWS TO SIMULATE DATA JOINS AND APPENDS

To avoid duplication of data, you virtually combine data with views. In the previous example we took the monthly data and combined it in a new physical table. The prob- lem is that we duplicated the data and therefore needed more storage space. In the example we're working with, that may not cause problems, but imagine that every table consists of terabytes of data; then it becomes problematic to duplicate the data. For this reason, the concept of a view was invented. A view behaves as if you're work- ing on a table, but this table is nothing but a virtual layer that combines the tables for you.



obs	Date
1	1-Jan
2	1-Jan
…	
n	31-Jan

obs	Date
1	1-Feb
2	2-Feb
…	
n	28-Feb

obs	Date
1	1-Dec
2	1-Dec
…	
n	31-Dec

obs	Date
1	1-Jan
2	1-Jan
…	
n	31-Dec

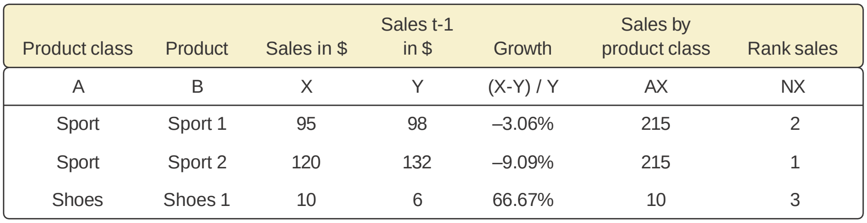
Figure 2.9 A view helps you combine data without replication.

Figure 2.9 shows how the sales data from the different months is combined virtually into a yearly sales table instead of duplicating the data. Views do come with a draw- back, however. While a table join is only performed once, the join that creates the view is recreated every time it's queried, using more processing power than a pre-calculated table would have.

Enriching aggregated measures

Data enrichment can also be done by adding calculated information to the table, such as the total number of sales or what percentage of total stock has been sold in a certain region (figure 2.10).

Extra measures such as these can add perspective. Looking at figure 2.10, we now have an aggregated data set, which in turn can be used to calculate the participation of each product within its category. This could be useful during data exploration but more so when creating data models. As always this depends on the exact case, but from our experience models with "relative measures" such as % sales (quantity of product sold/total quantity sold) tend to outperform models that use the raw numbers (quantity sold) as input.



Product class	Product	Sales in $	Sales t-1 in $	Growth	Sales by product class	Rank sales
A	B	X	Y	(X-Y) / Y	AX	NX
Sport	Sport 1	95	98	–3.06%	215	2
Sport	Sport 2	120	132	–9.09%	215	1
Shoes	Shoes 1	10	6	66.67%	10	3

Figure 2.10 Growth, sales by product class, and rank sales are examples of derived and aggregate measures.

2.4.4 Transforming data

Certain models require their data to be in a certain shape. Now that you've cleansed and integrated the data, this is the next task you'll perform: transforming your data so it takes a suitable form for data modeling.

Transforming data

Relationships between an input variable and an output variable aren't always linear. Take, for instance, a relationship of the form $y = ae^{bx}$. Taking the log of the independent variables simplifies the estimation problem dramatically. Figure 2.11 shows how transforming the input variables greatly simplifies the estimation problem. Other times you might want to combine two variables into a new variable.

x	1	2	3	4	5	6	7	8	9	10
log(*x*)	0.00	0.43	0.68	0.86	1.00	1.11	1.21	1.29	1.37	1.43
y	0.00	0.44	0.69	0.87	1.02	1.11	1.24	1.32	1.38	1.46

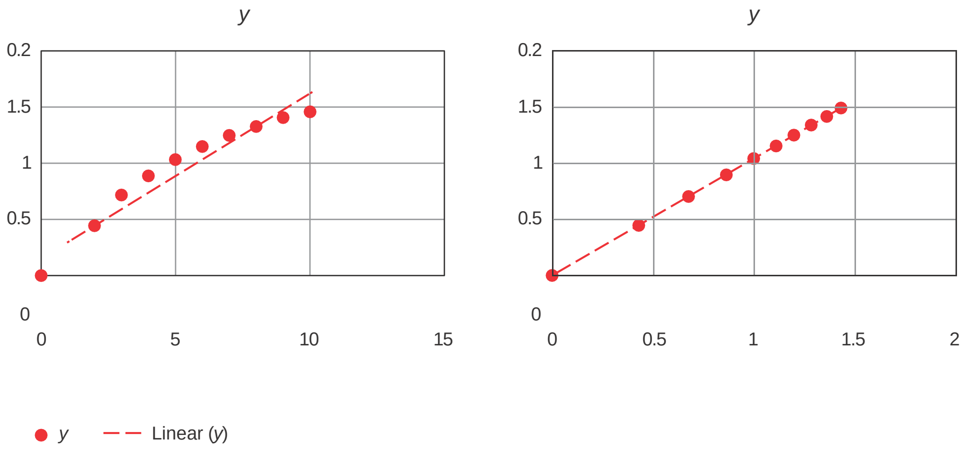


Figure 2.11 Transforming x to log x makes the relationship between x and y linear (right), compared with the non-log x (left).

Reducing the number of variables

Sometimes you have too many variables and need to reduce the number because they don't add new information to the model. Having too many variables in your model makes the model difficult to handle, and certain techniques don't perform well when you overload them with too many input variables. For instance, all the techniques based on a Euclidean distance perform well only up to 10 variables.

Euclidean distance

Euclidean distance or "ordinary" distance is an extension to one of the first things anyone learns in mathematics about triangles (trigonometry): Pythagoras's leg theorem. If you know the length of the two sides next to the 90° angle of a right-angled triangle you can easily derive the length of the remaining side (hypotenuse). The formula for this is hypotenuse = $\sqrt{(\text{side1} + \text{side2})^2}$. The Euclidean distance between two points in a two-dimensional plane is calculated using a similar formula: distance = $\sqrt{((x1 - x2)^2 + (y1 - y2)^2)}$. If you want to expand this distance calculation to more dimensions, add the coordinates of the point within those higher dimensions to the formula. For three dimensions we get distance = $\sqrt{((x1 - x2)^2 + (y1 - y2)^2 + (z1 - z2)^2)}$.

Data scientists use special methods to reduce the number of variables but retain the maximum amount of data. We'll discuss several of these methods in chapter 3. Figure 2.12 shows how reducing the number of variables makes it easier to understand the key values. It also shows how two variables account for 50.6% of the variation within the data set (component1 = 27.8% + component2 = 22.8%). These variables, called "component1" and "component2," are both combinations of the original variables. They're the *principal components* of the underlying data structure. If it isn't all that clear at this point, don't worry, principal components analysis (PCA) will be explained more thoroughly in chapter 3. What you can also see is the presence of a third (unknown) variable that splits the group of observations into two.

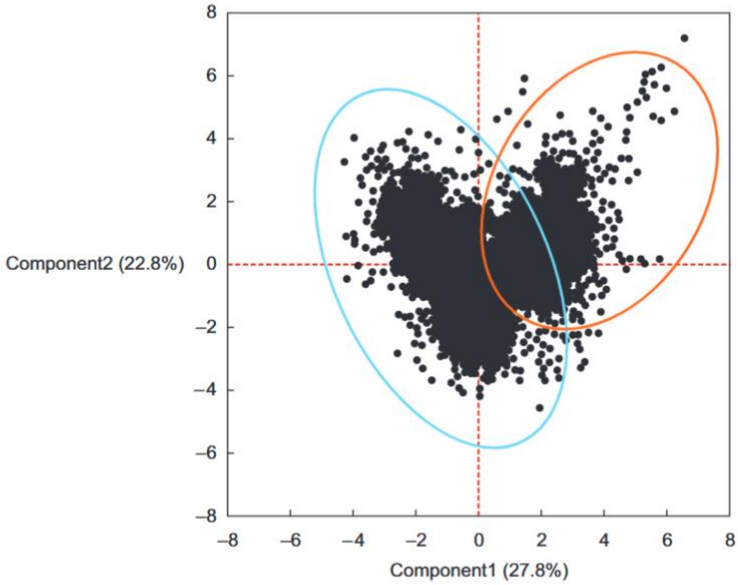
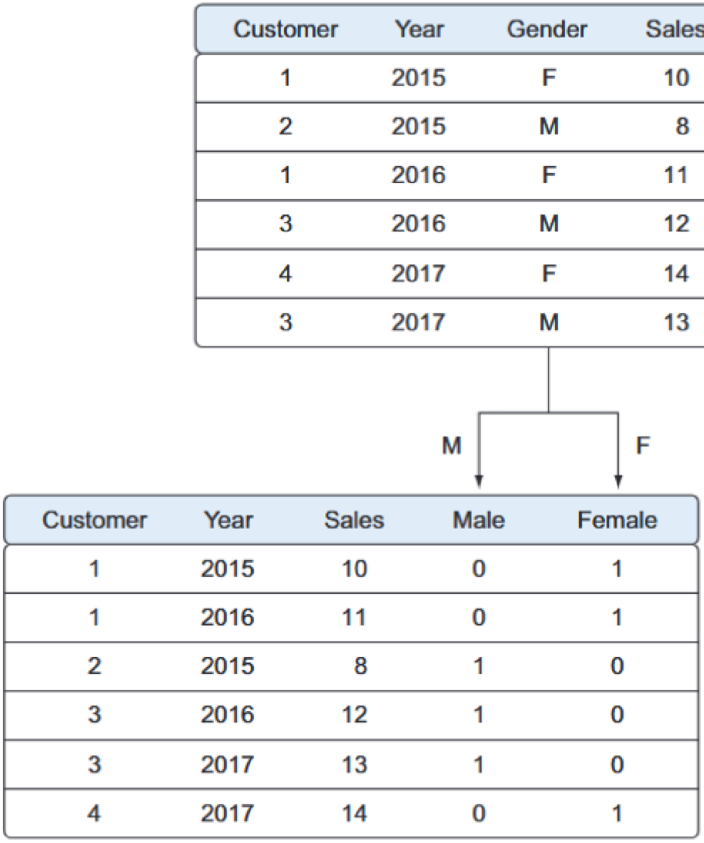


Figure 2.12 Variable reduction allows you to reduce the number of variables while maintaining as much information as possible.

TURNING VARIABLES INTO DUMMIES

Variables can be turned into dummy variables (figure 2.13). *Dummy variables* can only take two values: true(1) or false(0). They're used to indicate the absence of a categorical effect that may explain the observation. In this case you'll make separate columns for the classes stored in one variable and indicate it with 1 if the class is present and 0 otherwise. An example is turning one column named Weekdays into the columns Monday through Sunday. You use an indicator to show if the observation was on a Monday; you put 1 on Monday and 0 elsewhere. Turning variables into dummies is a technique that's used in modeling and is popular with, but not exclusive to, economists.

In this section we introduced the third step in the data science process—cleaning, transforming, and integrating data—which changes your raw data into usable input for the modeling phase. The next step in the data science process is to get a better understanding of the content of the data and the relationships between the variables and observations; we explore this in the next section.



Customer	Year	Gender	Sales
1	2015	F	10
2	2015	M	8
1	2016	F	11
3	2016	M	12
4	2017	F	14
3	2017	M	13

Customer	Year	Sales	Male	Female
1	2015	10	0	1
1	2016	11	0	1
2	2015	8	1	0
3	2016	12	1	0
3	2017	13	1	0
4	2017	14	0	1

Figure 2.13 Turning variables into dummies is a data transformation that breaks a variable that has multiple classes into multiple variables, each having only two possible values: 0 or 1.

2.5 Step 4: Exploratory data analysis

During exploratory data analysis you take a deep dive into the data (see figure 2.14). Information becomes much easier to grasp when shown in a picture, therefore you mainly use graphical techniques to gain an understanding of your data and the inter- actions between variables. This phase is about exploring data, so keeping your mind open and your eyes peeled is essential during the exploratory data analysis phase. The goal isn't to cleanse the data, but it's common that you'll still discover anomalies you missed before, forcing you to take a step back and fix them.

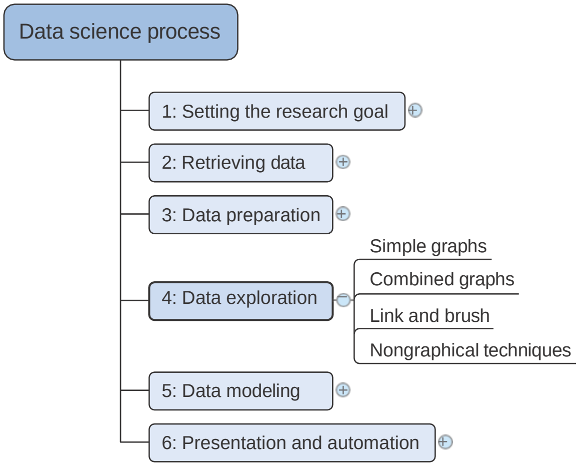


Figure 2.14 Step 4: Data exploration

The visualization techniques you use in this phase range from simple line graphs or histograms, as shown in figure 2.15, to more complex diagrams such as Sankey and network graphs. Sometimes it's useful to compose a composite graph from simple graphs to get even more insight into the data. Other times the graphs can be ani- mated or made interactive to make it easier and, let's admit it, way more fun. An example of an interactive Sankey diagram can be found at http://bost.ocks.org/ mike/sankey/.

Mike Bostock has interactive examples of almost any type of graph. It's worth spending time on his website, though most of his examples are more useful for data presentation than data exploration. These plots can be combined to provide even more insight, as shown in figure 2.16.

Overlaying several plots is common practice. In figure 2.17 we combine simple graphs into a Pareto diagram, or 80-20 diagram.

Figure 2.18 shows another technique: *brushing and linking*. With brushing and linking you combine and link different graphs and tables (or views) so changes in one graph are automatically transferred to the other graphs. An elaborate example of this can be found in chapter 9. This interactive exploration of data facilitates the discovery of new insights.

Figure 2.18 shows the average score per country for questions. Not only does this indicate a high correlation between the answers, but it's easy to see that when you select several points on a subplot, the points will correspond to similar points on the other graphs. In this case the selected points on the left graph correspond to points on the middle and right graphs, although they correspond better in the middle and right graphs.

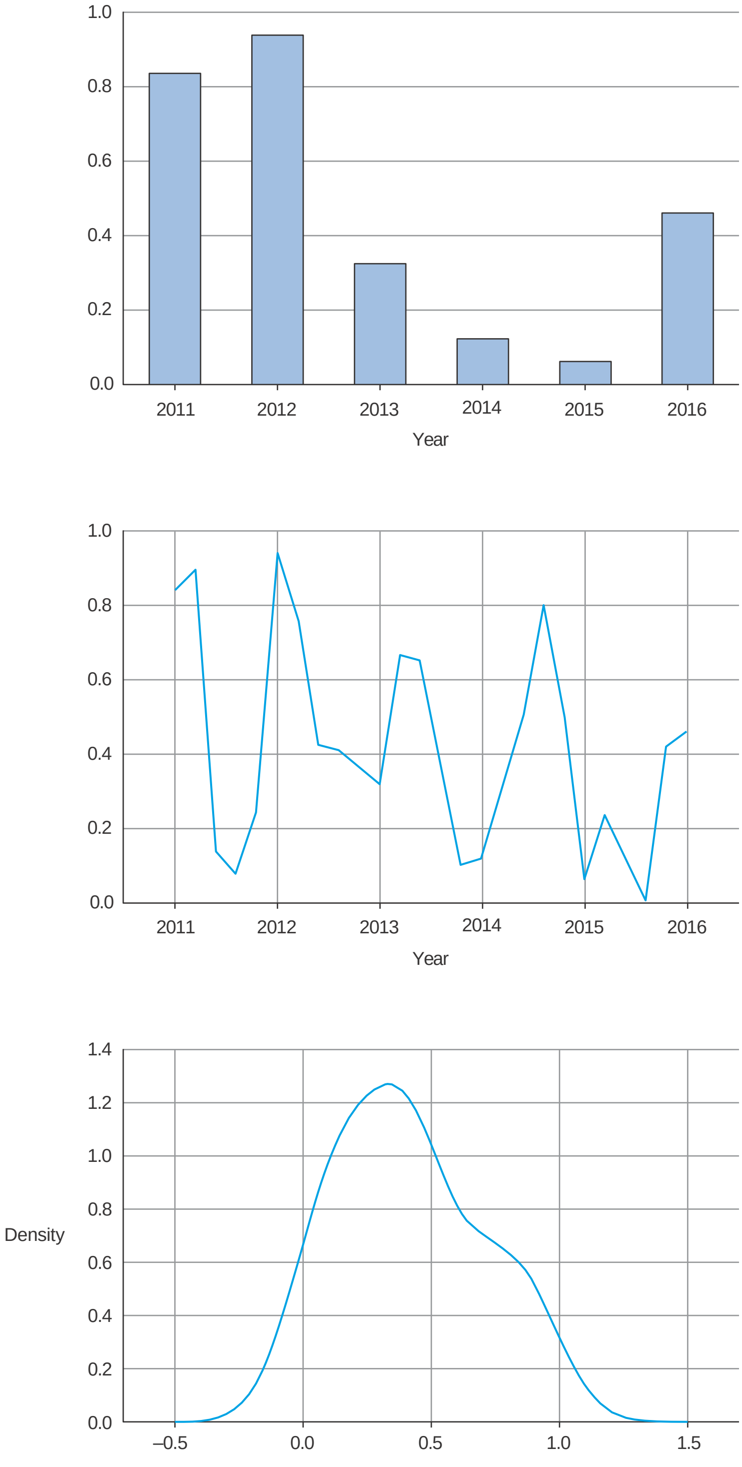


Figure 2.15 From top to bottom, a bar chart, a line plot, and a distribution are some of the graphs used in exploratory analysis.

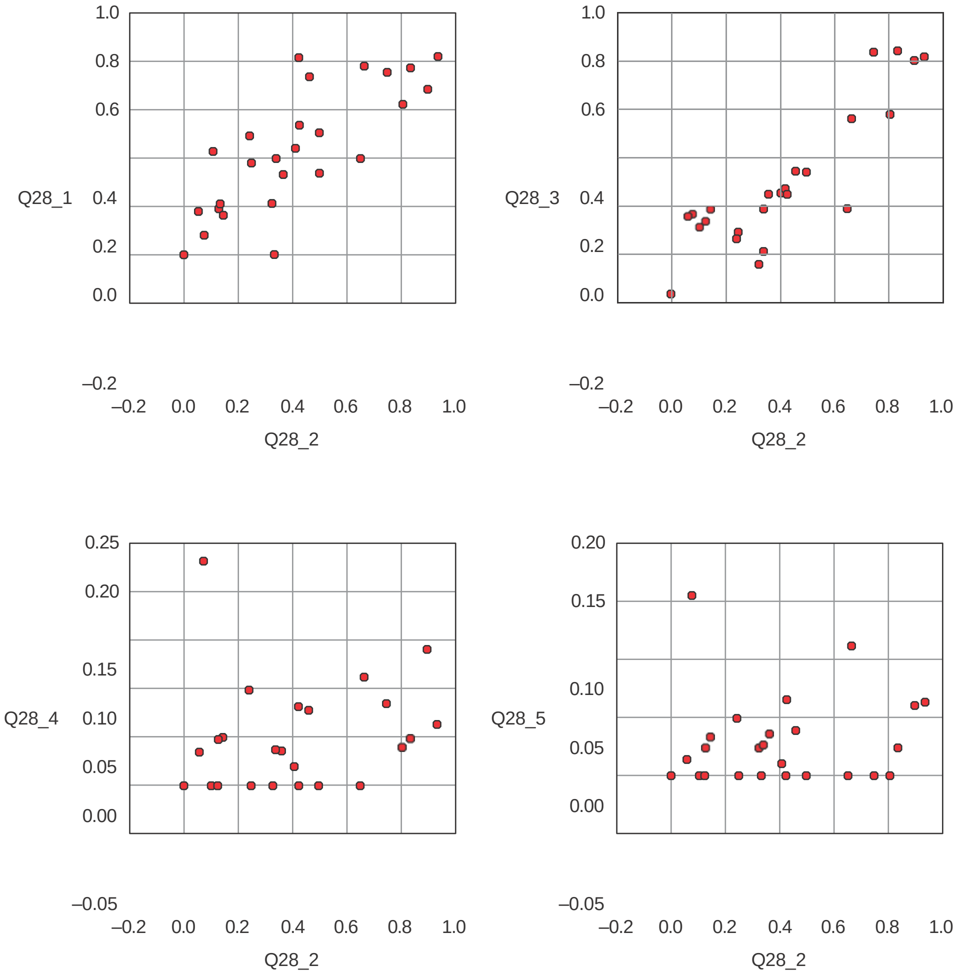


Figure 2.16 Drawing multiple plots together can help you understand the structure of your data over multiple variables.

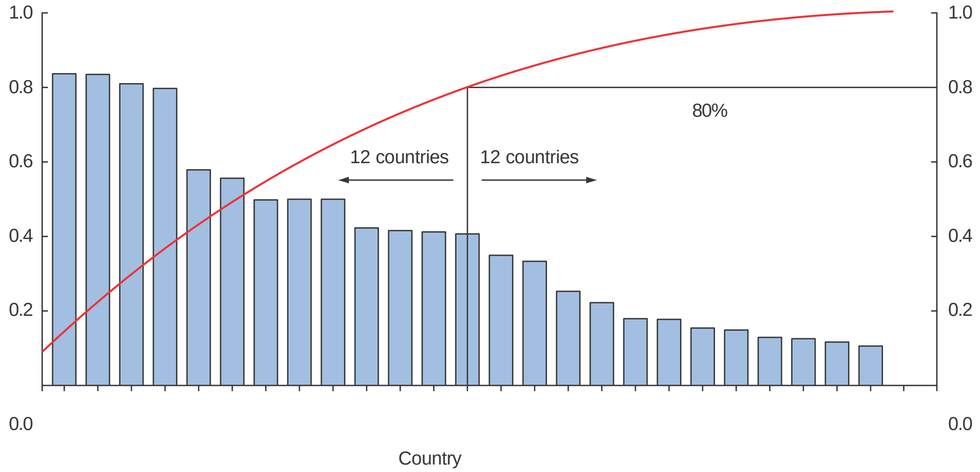


Figure 2.17 A Pareto diagram is a combination of the values and a cumulative distribution. It's easy to see from this diagram that the first 50% of the countries contain slightly less than 80% of the total amount. If this graph represented customer buying power and we sell expensive products, we probably don't need to spend our marketing budget in every country; we could start with the first 50%.

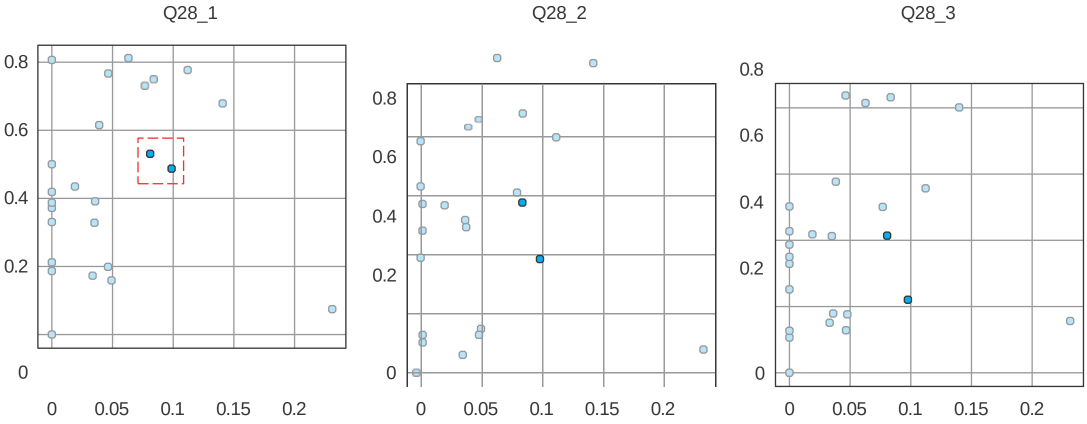


Figure 2.18 Link and brush allows you to select observations in one plot and highlight the same observations in the other plots.

Two other important graphs are the histogram shown in figure 2.19 and the boxplot shown in figure 2.20. In a histogram a variable is cut into discrete categories and the number of occurrences in each category are summed up and shown in the graph. The boxplot, on the other hand, doesn't show how many observations are present but does offer an impression of the distribution within categories. It can show the maximum, minimum, median, and other characterizing measures at the same time.

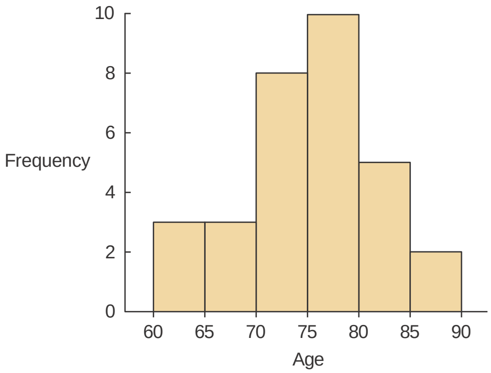


Figure 2.19 Example histogram: the number of people in the age-groups of 5-year intervals

The techniques we described in this phase are mainly visual, but in practice they're certainly not limited to visualization techniques. Tabulation, clustering, and other modeling techniques can also be a part of exploratory analysis. Even building simple models can be a part of this step.

Now that you've finished the data exploration phase and you've gained a good grasp of your data, it's time to move on to the next phase: building models.

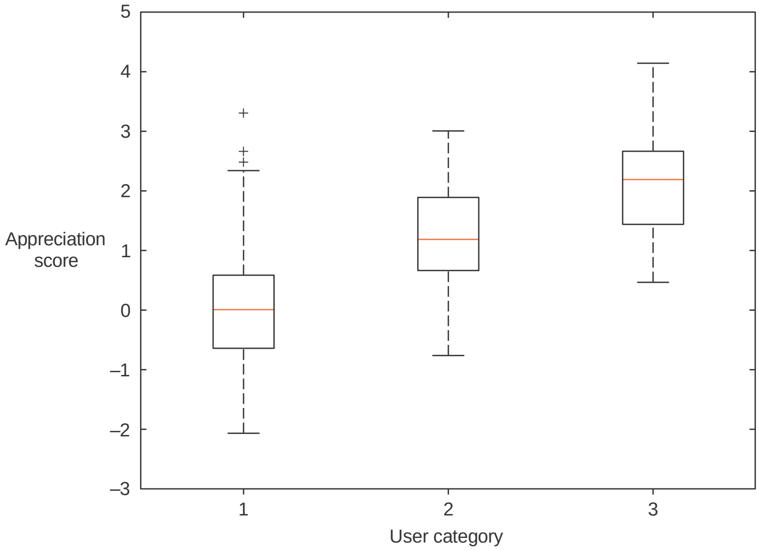


Figure 2.20 Example boxplot: each user category has a distribution of the appreciation each has for a certain picture on a photography website.

2.6 Step 5: Build the models

With clean data in place and a good understanding of the content, you're ready to build models with the goal of making better predictions, classifying objects, or gaining an understanding of the system that you're modeling. This phase is much more focused than the exploratory analysis step, because you know what you're looking for and what you want the outcome to be. Figure 2.21 shows the components of model building.

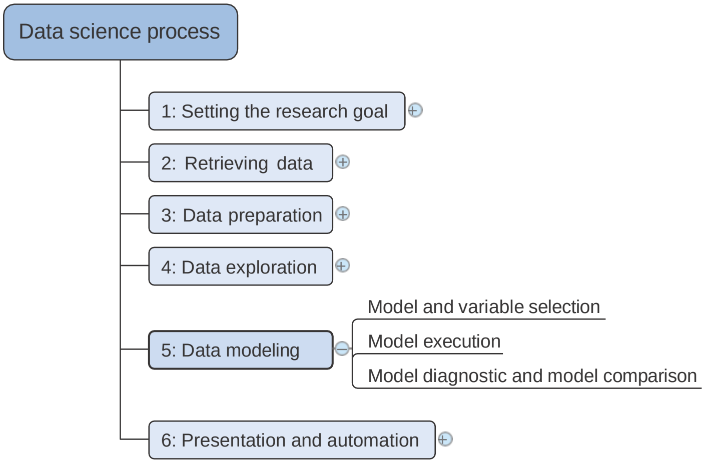

Data science process
1: Setting the research goal
2: Retrieving data
3: Data preparation
4: Data exploration
5: Data modeling
Model and variable selection
Model execution
Model diagnostic and model comparison
6: Presentation and automation

Figure 2.21 Step 5: Data modeling

The techniques you'll use now are borrowed from the field of machine learning, data mining, and/or statistics. In this chapter we only explore the tip of the iceberg of existing techniques, while chapter 3 introduces them properly. It's beyond the scope of this book to give you more than a conceptual introduction, but it's enough to get you started; 20% of the techniques will help you in 80% of the cases because tech- niques overlap in what they try to accomplish. They often achieve their goals in similar but slightly different ways.

Building a model is an iterative process. The way you build your model depends on whether you go with classic statistics or the somewhat more recent machine learning school, and the type of technique you want to use. Either way, most models consist of the following main steps:

1. Selection of a modeling technique and variables to enter in the model
2. Execution of the model
3. Diagnosis and model comparison

2.6.1 Model and variable selection

You’ll need to select the variables you want to include in your model and a modeling technique. Your findings from the exploratory analysis should already give a fair idea of what variables will help you construct a good model. Many modeling techniques are available, and choosing the right model for a problem requires judgment on your part. You’ll need to consider model performance and whether your project meets all the requirements to use your model, as well as other factors:

- Must the model be moved to a production environment and, if so, would it be easy to implement?
- How difficult is the maintenance on the model: how long will it remain relevant if left untouched?
- Does the model need to be easy to

explain? When the thinking is done, it’s time

for action.

2.6.2 Model execution

Once you’ve chosen a model you’ll need to implement it in code.

> **REMARK** This is the first time we’ll go into actual Python code execution so make sure you have a virtual env up and running. Knowing how to set this up is required knowledge, but if it’s your first time, check out appendix D.
>
> All code from this chapter can be downloaded from https://www.manning.com/books/introducing-data-science. This chapter comes with an ipython (.ipynb) notebook and Python (.py) file.

Luckily, most programming languages, such as Python, already have libraries such as StatsModels or Scikit-learn. These packages use several of the most popular techniques. Coding a model is a nontrivial task in most cases, so having these libraries available can speed up the process. As you can see in the following code, it’s fairly easy to use linear regression (figure 2.22) with StatsModels or Scikit-learn. Doing this your- self would require much more effort even for the simple techniques. The following listing shows the execution of a linear prediction model.

Listing 2.1 Executing a linear prediction model on semi-random data

```
import statsmodels.api as sm
import numpy as np
predictors = np.random.random(1000).reshape(500,2)
target = predictors.dot(np.array([0.4, 0.6])) + np.random.random(500)
lmRegModel = sm.OLS(target,predictors)
result = lmRegModel.fit()
result.summary()
```

Imports required Python modules.

Creates random data for predictors (x-values) and semi-random data for the target (y-values) of the model. We use predictors as input to create the target so we infer a correlation here.

Fits linear regression on data.

Shows model fit statistics.

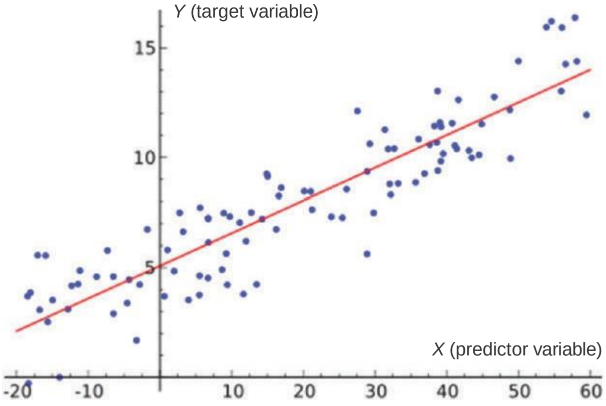
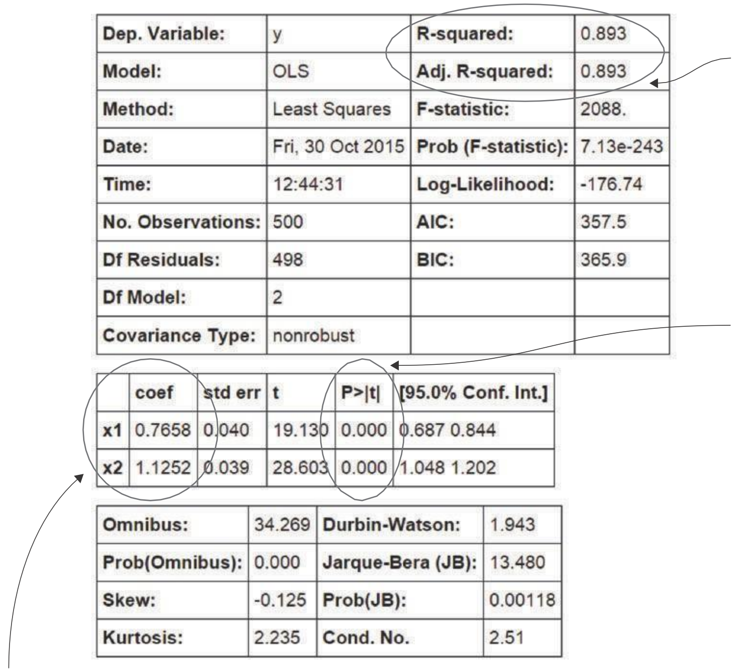


Figure 2.22 Linear regression tries to fit a line while minimizing the distance to each point

Okay, we cheated here, quite heavily so. We created predictor values that are meant to pre- dict how the target variables behave. For a linear regression, a "linear relation" between each x (predictor) and the y (target) variable is assumed, as shown in figure 2.22.

We, however, created the target variable, based on the predictor by adding a bit of randomness. It shouldn't come as a surprise that this gives us a well-fitting model. The `results.summary()` outputs the table in figure 2.23. Mind you, the exact outcome depends on the random variables you got.



Dep. Variable:	y	R-squared:	0.893
Model:	OLS	Adj. R-squared:	0.893
Method:	Least Squares	F-statistic:	2088.
Date:	Fri, 30 Oct 2015	Prob (F-statistic):	7.13e-243
Time:	12:44:31	Log-Likelihood:	-176.74
No. Observations:	500	AIC:	357.5
Df Residuals:	498	BIC:	365.9
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf. Int.]
x1	0.7658	0.040	19.130	0.000	0.687 0.844
x2	1.1252	0.039	28.603	0.000	1.048 1.202

Omnibus:	34.269	Durbin-Watson:	1.943
Prob(Omnibus):	0.000	Jarque-Bera (JB):	13.480
Skew:	-0.125	Prob(JB):	0.00118
Kurtosis:	2.235	Cond. No.	2.51

Model fit: higher is better but too high is suspicious.

p-value to show whether a predictor variable has a significant influence on the target. Lower is better and <0.05 is often considered "significant."

Linear equation coefficients. y = 0.7658xl + 1.1252x2.

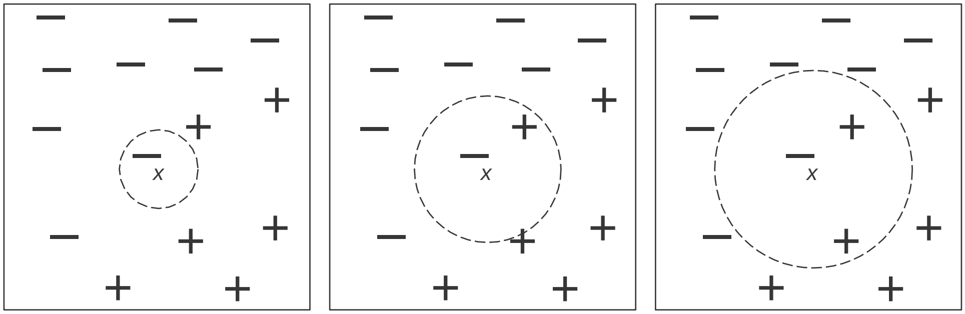
Figure 2.23 Linear regression model information output

Let's ignore most of the output we got here and focus on the most important parts:

- *Model fit* —For this the R-squared or adjusted R-squared is used. This measure is an indication of the amount of variation in the data that gets captured by the model. The difference between the adjusted R-squared and the R-squared is minimal here because the adjusted one is the normal one + a penalty for model complexity. A model gets complex when many variables (or features) are introduced. You don't need a complex model if a simple model is available, so the adjusted R-squared punishes you for overcomplicating. At any rate, 0.893 is high, and it should be because we cheated. Rules of thumb exist, but for models in businesses, models above 0.85 are often considered good. If you want to win a competition you need in the high 90s. For research however, often very low model fits (<0.2 even) are found. What's more important there is the influence of the introduced predictor variables.
- *Predictor variables have a coefficient*—For a linear model this is easy to interpret. In our example if you add "1" to x1, it will change y by "0.7658". It's easy to see how finding a good predictor can be your route to a Nobel Prize even though your model as a whole is rubbish. If, for instance, you determine that a certain gene is significant as a cause for cancer, this is important knowledge, even if that gene in itself doesn't determine whether a person will get cancer. The example here is classification, not regression, but the point remains the same: detecting influences is more important in scientific studies than perfectly fitting models (not to mention more realistic). But when do we know a gene has that impact? This is called significance.
- *Predictor significance* —Coefficients are great, but sometimes not enough evidence exists to show that the influence is there. This is what the p-value is about. A long explanation about type 1 and type 2 mistakes is possible here but the short explanations would be: if the p-value is lower than 0.05, the variable is considered significant for most people. In truth, this is an arbitrary number. It means there's a 5% chance the predictor doesn't have any influence. Do you accept this 5% chance to be wrong? That's up to you. Several people introduced the extremely significant (p<0.01) and marginally significant thresholds (p<0.1).

Linear regression works if you want to predict a value, but what if you want to classify something? Then you go to classification models, the best known among them being k-nearest neighbors.

As shown in figure 2.24, k-nearest neighbors looks at labeled points nearby an unlabeled point and, based on this, makes a prediction of what the label should be.



(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

Figure 2.24 K-nearest neighbor techniques look at the k-nearest point to make a prediction.

Let's try it in Python code using the Scikit learn library, as in this next listing.

Listing 2.2 Executing k-nearest neighbor classification on semi-random data

```
from sklearn import neighbors

predictors = np.random.random(1000).reshape(500,2)
target = np.around(predictors.dot(np.array([0.4, 0.6])) +
         np.random.random(500))

clf = neighbors.KNeighborsClassifier(n_neighbors=10)
knn = clf.fit(predictors,target)
knn.score(predictors, target)
```

Imports modules.

Creates random predictor data and semi-random target data based on predictor data.

Fits 10-nearest neighbors model.

Gets model fit score: what percent of the classification was correct?

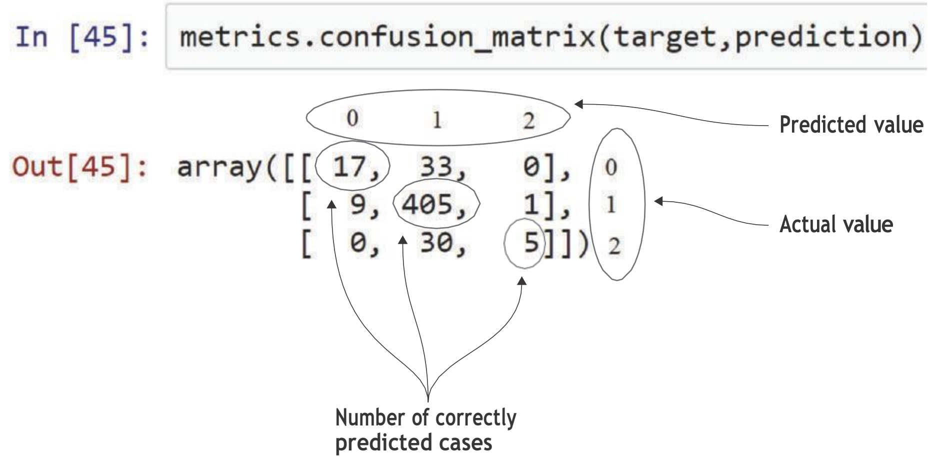
As before, we construct random correlated data and surprise, surprise we get 85% of cases correctly classified. If we want to look in depth, we need to score the model. Don't let `knn.score()` fool you; it returns the model accuracy, but by "scoring a model" we often mean applying it on data to make a prediction.

```
prediction = knn.predict(predictors)
```

Now we can use the prediction and compare it to the real thing using a confusion matrix.

```
metrics.confusion_matrix(target,prediction)
```

We get a 3-by-3 matrix as shown in figure 2.25.



```
In [45]: metrics.confusion_matrix(target,prediction)

Out[45]: array([[ 17,  33,   0],
               [  9, 405,   1],
               [  0,  30,   5]])
```

Figure 2.25 Confusion matrix: it shows how many cases were correctly classified and incorrectly classified by comparing the prediction with the real values. Remark: the classes (0,1,2) were added in the figure for clarification.

The confusion matrix shows we have correctly predicted 17+405+5 cases, so that's good. But is it really a surprise? No, for the following reasons:

- For one, the classifier had but three options; marking the difference with last time `np.around()` will round the data to its nearest integer. In this case that's either 0, 1, or 2. With only 3 options, you can't do much worse than 33% correct on 500 guesses, even for a real random distribution like flipping a coin.
- Second, we cheated again, correlating the response variable with the predictors. Because of the way we did this, we get most observations being a "1". By guessing "1" for every case we'd already have a similar result.
- We compared the prediction with the real values, true, but we never predicted based on fresh data. The prediction was done using the same data as the data used to build the model. This is all fine and dandy to make yourself feel good, but it gives you no indication of whether your model will work when it encounters truly new data. For this we need a holdout sample, as will be discussed in the next section.

Don't be fooled. Typing this code won't work miracles by itself. It might take a while to get the modeling part and all its parameters right.

To be honest, only a handful of techniques have industry-ready implementations in Python. But it's fairly easy to use models that are available in R within Python with the help of the RPy library. RPy provides an interface from Python to R. *R* is a free software environment, widely used for statistical computing.

(if not the most popular) programming languages for data science. For more information, see http://www.kdnuggets.com/polls/2014/languages-analytics-data-mining- data-science.html.

2.6.3 Model diagnostics and model comparison

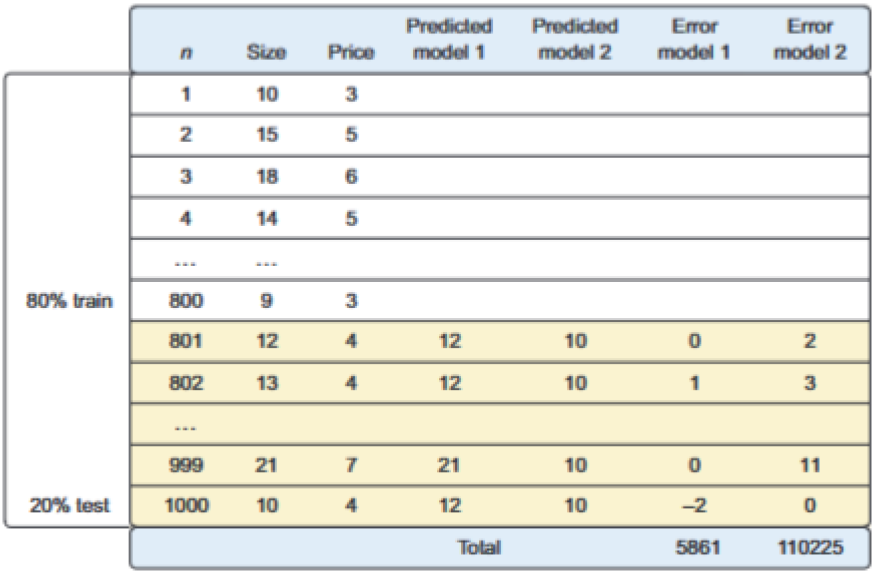
You'll be building multiple models from which you then choose the best one based on multiple criteria. Working with a holdout sample helps you pick the best-performing model. A holdout sample is a part of the data you leave out of the model building so it can be used to evaluate the model afterward. The principle here is simple: the model should work on unseen data. You use only a fraction of your data to estimate the model and the other part, the holdout sample, is kept out of the equation. The model is then unleashed on the unseen data and error measures are calculated to evaluate it. Multiple error measures are available, and in figure 2.26 we show the general idea on comparing models. The error measure used in the example is the mean square error.

Mean square error is a simple measure: check for every prediction how far it was from the truth, square this error, and add up the error of every prediction. Figure 2.27 compares the performance of two models to predict the order size from the price. The first model is *size = 3 * price* and the second model is *size = 10.* To estimate the models, we use 800 randomly chosen observations out of 1,000 (or 80%), without showing the other 20% of data to the model. Once the model is trained, we predict the values for the other 20% of the variables based on those for which we already know the true value, and calculate the model error with an error measure. Then we choose the model with the lowest error. In this example we chose model 1 because it has the lowest total error. Many models make strong assumptions, such as independence of the inputs, and you have to verify that these assumptions are indeed met. This is called *model diagnostics*. This section gave a short introduction to the steps required to build a valid model. Once you have a working model you're ready to go to the last step

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

Figure 2.26 Formula for mean square error

.



	n	Size	Price	Predicted model 1	Predicted model 2	Error model 1	Error model 2
	1	10	3				
	2	15	5				
	3	18	6				
	4	14	5				
	...	...					
80% train	800	9	3				
	801	12	4	12	10	0	2
	802	13	4	12	10	1	3
	...						
	999	21	7	21	10	0	11
20% test	1000	10	4	12	10	–2	0
				Total		5861	110225

Figure 2.27 A holdout sample helps you compare models and ensures that you can generalize results to data that the model has not yet seen.

2.7 Step 6: Presenting findings and building applications on top of them

After you've successfully analyzed the data and built a well-performing model, you're ready to present your findings to the world (figure 2.28). This is an exciting part; all your hours of hard work have paid off and you can explain what you found to the stakeholders.

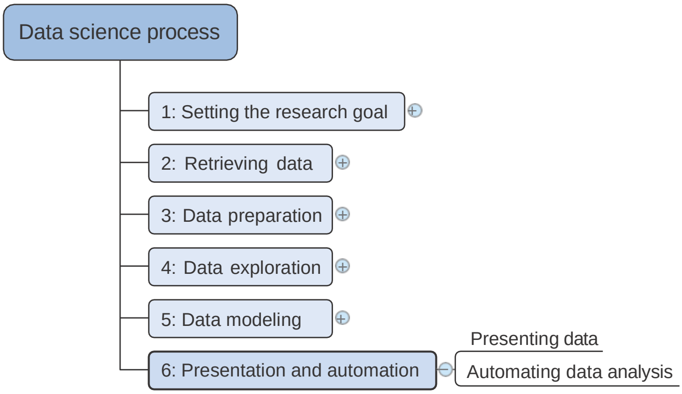


Figure 2.28 Step 6: Presentation and automation

Sometimes people get so excited about your work that you'll need to repeat it over and over again because they value the predictions of your models or the insights that you produced. For this reason, you need to automate your models. This doesn't always mean that you have to redo all of your analysis all the time. Sometimes it's sufficient that you implement only the model scoring; other times you might build an application that automatically updates reports, Excel spreadsheets, or PowerPoint presentations. The last stage of the data science process is where your *soft skills* will be most useful, and yes, they're extremely important.