1 What Are Data and Data Science?

Figure 1.1 Petroglyphs are one of the earliest types of data generated by humanity, providing vital information about the daily life of the people who created them. (credit: modification of work "Indian petroglyphs (~100 B.C. to ~1540 A.D.) (Newspaper Rock, southeastern Utah, USA) 24" by James St. John/Flickr, CC BY 2.0)

Chapter Outline

Introduction

Many of us use the terms "data" and "data science," but not necessarily with a lot of precision. This chapter will define data science terminology and apply the terms in multiple fields. The chapter will also briefly introduce the types of technology (such as statistical software, spreadsheets, and programming languages) that data scientists use to perform their work and will then take a deeper dive into the use of Python for data analysis. The chapter should help you build a technical foundation so that you can practice the more advanced data science concepts covered in future chapters.

1.1 What Is Data Science?

Learning Outcomes

By the end of this section, you should be able to:

- 1.1.1 Describe the goals of data science.
- 1.1.2 Explain the data science cycle and goals of each step in the cycle.
- 1.1.3 Explain the role of data management in the data science process.

Data science is a field of study that investigates how to collect, manage, and analyze data of all types in order to retrieve meaningful information. Although we will describe data in more detail in Data and Datasets, you can consider *data* to be any pieces of evidence or observations that can be analyzed to provide some insights.

In its earliest days, the work of data science was spread across multiple disciplines, including statistics, mathematics, computer science, and social science. It was commonly believed that the job of data collection, management, and analysis would be carried out by different types of experts, with each job independent of one another. To be more specific, *data collection* was considered to be the province of so-called domain experts (e.g., doctors for medical data, psychologists for psychological data, business analysts for sales, logistic, and marketing data, etc.) as they had a full context of the data; *data management* was for computer scientists/engineers as they knew how to store and process data in computing systems (e.g., a single computer, a server, a data warehouse); and *data analysis* was for statisticians and mathematicians as they knew how to derive some meaningful insights from data. Technological advancement brought about the proliferation of data, muddying the boundaries between these jobs, as shown in Figure 1.2. Now, it is expected that a data scientist or data science team will have some expertise in all three domains.

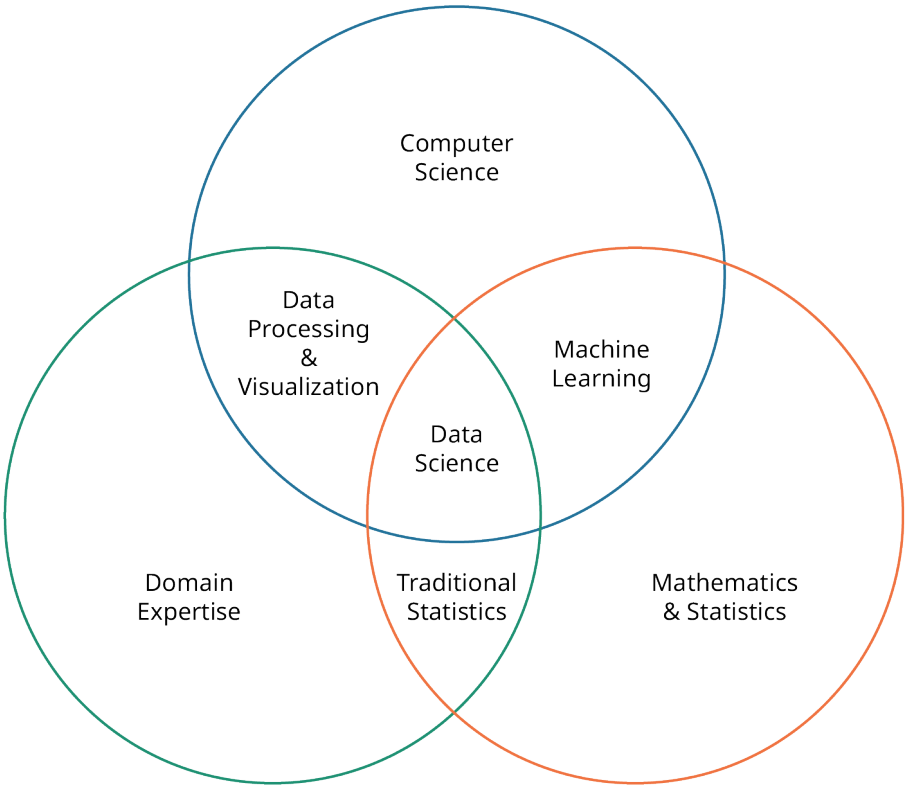


Figure 1.2 The Field of Data Science

One good example of this is the development of personal cell phones. In the past, households typically had only one landline telephone, and the only data that was generated with the telephone was the list of phone numbers called by the household members. Today the majority of consumers own a smartphone, which contains a tremendous amount of data: photos, social media contacts, videos, locations (usually), and perhaps health data (with the consumers' consent), among many other things.

Is the data from a smartphone solely collected by domain experts who are specialized in photos, videos, and such? Probably not. They are automatically logged and collected by the smartphone system itself, which is designed by computer scientists/engineers. For a health care scientist to collect data from many individuals in the "traditional" way, bringing patients into a laboratory and taking vital signs regularly over a period of time takes a lot of time and effort. A smartphone application is a more efficient and productive method, from a data collection perspective.

Data science tasks are often described as a process, and this section provides an overview for each step of that process.

The Data Science Cycle

Data science tasks follow a process, called the **data science cycle**, which includes problem definition, then data collection, preparation, analysis, and reporting, as illustrated in Figure 1.3. See this animation about data science (https://openstax.org/r/youtube), which describes the data science cycle in much the same way.

Figure 1.3 The Data Science Cycle

Although data collection and preparation may sound like simple tasks compared to the more important work of analysis, they are actually the most time- and effort-consuming steps in the data science cycle. According to a survey conducted by Anaconda (2020), data scientists spend about half of the entire process in data collection and cleaning, while data analysis and communication take about a quarter to a third of the time each, depending on the job.

Problem Definition, Data Collection, and Data Preparation

The first step in the data science cycle is a precise definition of the problem statement to establish clear objectives for the goal and scope of the data analysis project. Once the problem has been well defined, the data must be generated and collected. **Data collection** is the systematic process of gathering information on variables of interest. Data is often collected purposefully by domain experts to find answers to predefined problems. One example is data on customer responses to a product satisfaction survey. These survey questions will likely be crafted by the product sales and marketing representatives, who likely have a specific plan for how they wish to use the response data once it is collected.

Not all data is generated this purposefully, though. A lot of data around our daily life is simply a by-product of our activity. These by-products are kept as data because they could be used by others to find some helpful insights later. One example is our web search histories. We use a web search engine like Google to search for information about our interests, and such activity leaves a history of the search text we used on the Google server. Google employees can utilize the records of numerous Google users in order to analyze common search patterns, present accurate search results, and potentially, to display relatable advertisements back to the searchers.

Often the collected data is not in an optimal form for analysis. It needs to be processed somehow so that it can be analyzed, in the phase called **data preparation** or *data processing*. Suppose you work for Google and want to know what kind of food people around the globe search about the most during the night. You have users' search history from around the globe, but you probably cannot use the search history data as is. The search keywords will probably be in different languages, and users live all around the Earth, so nighttime will vary by each user's time zone. Even then, some search keywords would have some typos, simply not make sense, or even remain blank if the Google server somehow failed to store that specific search history record. Note that all these scenarios are possible, and therefore data preparation should address these issues so that the actual analysis can draw more accurate results. There are many different ways to manage these issues, which we will discuss more fully in Collecting and Preparing Data.

Data Analysis

Once the data is collected and prepared, it must be analyzed in order to discover meaningful insights, a process called **data analysis**. There are a variety of data analysis methods to choose from, ranging from simple ones like checking minimum and maximum values, to more advanced ones such as modelling a dependent variable. Most of the time data scientists start with simple methods and then move into more

advanced ones, based on what they want to investigate further. Descriptive Statistics: Statistical Measurements and Probability Distributions and Inferential Statistics and Regression Analysis discuss when and how to use different analysis methods. Time Series and Forecasting and Decision-Making Using Machine Learning Basics discuss forecasting and decision-making.

Data Reporting

Data reporting involves the presentation of data in a way that will best convey the information learned from data analysis. The importance of data reporting cannot be overemphasized. Without it, data scientists cannot effectively communicate to their audience the insights they discovered in the data. Data scientists work with domain experts from different fields, and it is their responsibility to communicate the results of their analysis in a way that those domain experts can understand. **Data visualization** is a graphical way of presenting and reporting data to point out the patterns, trends, and hidden insights; it involves the use of visual elements such as charts, graphs, and maps to present data in a way that is easy to comprehend and analyze. The goal of data visualization is to communicate information effectively and facilitate better decision-making. Data visualization and basic statistical graphing, including how to create graphical presentations of data using Python, are explored in depth in Visualizing Data. Further details on reporting results are discussed in Reporting Results.

Data Management

In the early days of data analysis (when generated data was mostly structured and not quite so “big”), it was possible to keep data in local storage (e.g., on a single computer or a portable hard drive). With this setup, data processing and analysis was all done locally as well.

When so much more data began to be collected—much of it unstructured as well as structured—cloud-based management systems were developed to store all the data on a designated server, outside a local computer. At the same time, data scientists began to see that most of their time was being spent on data processing rather than analysis itself. To address this concern, modern data management systems not only store the data itself but also perform some basic processing on a cloud. These systems, referred to as **data warehousing**, store and manage large volumes of data from various sources in a central location, enabling efficient retrieval and analysis for business intelligence and decision-making. (Data warehousing is covered in more detail in Handling Large Datasets.)

Today, enterprises simply subscribe to a cloud-warehouse service such as Amazon RedShift (which runs on the Amazon Web Services) or Google BigQuery (which runs on the Google Cloud) instead of buying physical storage and configuring data management/processing systems on their own. These services ensure the data is safely stored and processed on the cloud, all without spending money on purchasing/maintaining physical storage.

1.2 Data Science in Practice

Learning Outcomes

By the end of this section, you should be able to:

- 1.2.1 Explain the interdisciplinary nature of data science in various fields.
- 1.2.2 Identify examples of data science applications in various fields.
- 1.2.3 Identify current issues and challenges in the field of data science

While data science has adopted techniques and theories from fields such as math, statistics, and computer science, its applications concern an expanding number of fields. In this section we introduce some examples of how data science is used in business and finance, public policy, health care and medicine, engineering and sciences, and sports and entertainment.

Data Science in Business

Data science plays a key role in many business operations. A variety of data related to customers, products, and sales can be collected and generated within a business. These include customer names and lists of products they have purchased as well as daily revenue. Business analytics investigate these data to launch new products and to maximize the business revenue/profit.

Retail giant Walmart is known for using business analytics to improve the company's bottom line. Walmart collects multiple *petabytes* (1 petabyte = 1,024 terabytes) of unstructured data every hour from millions of customers (commonly referred to as "big data"); as of 2024, Walmart's customer data included roughly 255 million weekly customer visits (Statista, 2024). Walmart uses this big data to investigate consumer patterns and adjust its inventories. Such analysis helps the company avoid overstocking or understocking and resulted in an estimated online sales increase of between 10% and 15%, translating to an extra $1 billion in revenue (ProjectPro, 2015). One often-reported example includes the predictive technology Walmart used to prepare for Hurricane Frances in 2004. A week before the hurricane's arrival, staff were asked to look back at their data on sales during Hurricane Charley, which hit several weeks earlier, and then come up with some forecasts about product demand ahead of Frances (Hays, 2004). Among other insights, the executives discovered that strawberry Pop-Tart sales increased by about sevenfold during that time. As a result, in the days before Hurricane Frances, Walmart shipped extra supplies of strawberry Pop-Tarts to stores in the storm's path (Hays, 2004). The analysis also provided insights on how many checkout associates to assign at different times of the day, where to place popular products, and many other sales and product details. In addition, the company has launched social media analytics efforts to investigate hot keywords on social media and promptly make related products available (ProjectPro, 2015).

Amazon provides another good example. Ever since it launched its Prime membership service, Amazon has focused on how to minimize delivery time and cost. Like Walmart, it started by analyzing consumer patterns and was able to place products close to customers. To do so, Amazon first divided the United States into eight geographic regions and ensured that most items were warehoused and shipped within the same region; this allowed the company to reduce the shipping time and cost. As of 2023, more than 76% of orders were shipped from within the customer's region, and items in same-day shipping facilities could be made ready to put on a delivery truck in just 11 minutes (Herrington, 2023). Amazon also utilizes machine learning algorithms to predict the demand for items in each region and have the highest-demand items available in advance at the fulfillment center of the corresponding region. This predictive strategy has helped Amazon reduce the delivery time for each product and extend the item selections for two-day shipping (Herrington, 2023).

Data science is utilized extensively in finance as well. Detecting and managing fraudulent transactions is now done by automated machine learning algorithms (IABAC, 2023). Based on the customer data and the patterns of past fraudulent activities, an algorithm can determine whether a transaction is fraudulent in real time. Multiple tech companies, such as IBM and Amazon Web Services, offer their own fraud detection solutions to their corporate clients. (For more information, see this online resource on Fraud Detection through Data Analytics (https://openstax.org/r/iabac).)

Data Science in Engineering and Science

Various fields of engineering and science also benefit from data science. Internet of Things (IoT) is a good example of a new technology paradigm that has benefited from data science. **Internet of Things (IoT)** describes the network of multiple objects interacting with each other through the Internet. Data science plays a crucial role in these interactions since behaviors of the objects in a network are often triggered by data collected by another object in the network. For example, a smart doorbell or camera allows us to see a live stream on our phone and alerts us to any unusual activity.

In addition, weather forecasting has always been a data-driven task. Weather analysts collect different measures of the weather such as temperature and humidity and then make their best estimate for the

weather in the future. Data science has made weather forecasting more reliable by adopting more sophisticated prediction methods such as time-series analysis, artificial intelligence (AI), and machine learning (covered in Time Series and Forecasting, Decision-Making Using Machine Learning Basics, and Deep Learning and AI Basics). Such advancement in weather forecasting has also enabled engineers and scientists to predict some natural disasters such as flood or wildfire and has enabled precision farming, with which agricultural engineers can identify an optimal time window to plant, water, and harvest crops. For example, an agronomy consultant, Ag Automation, has partnered with Hitachi to offer a solution that both automates data collection and remotely monitors and controls irrigation for the best efficiency (Hitachi, 2023).

EXPLORING FURTHER

Using AI for Irrigation Control

See this Ag Automation video (https://openstax.org/r/youtube128) demonstrating the use of data collection for controlling irrigation.

Data Science in Public Policy

Smart cities are among the most representative examples of the use of data science in public policy. Multiple cities around the world, including Masdar City in the United Arab Emirates and Songdo in South Korea, have installed thousands of data-collecting sensors used to optimize their energy consumption. The technology is not perfect, and smart cities may not yet live up to their full potential, but many corporations and companies are pursuing the goal of developing smart cities more broadly (Clewlow, 2024). The notion of a smart city has also been applied on a smaller scale, such as to a parking lot, a building, or a street of lights. For example, the city of San Diego installed thousands of sensors on the city streets to control the streetlights using data and smart technology. The sensors measure traffic, parking occupancy, humidity, and temperature and are able to turn on the lights only when necessary (Van Bocxlaer, 2020). New York City has adopted smart garbage bins that monitor the amount of garbage in a bin, allowing garbage collection companies to route their collection efforts more efficiently (Van Bocxlaer, 2020).

EXPLORING FURTHER

Sensor Networks to Monitor Energy Consumption

See how Songdo (https://openstax.org/r/youtube4) monitors energy consumption and safety with sensor networks.

Data Science in Education

Data science also influences education. Traditional instruction, especially in higher education, has been provided in a one-size-fits-all form, such as many students listening to a single instructor's lecture in a classroom. This makes it difficult for an instructor to keep track of each individual student's learning progress. However, many educational platforms these days are online and can produce an enormous amount of student data, allowing instructors to investigate everyone's learning based on these collected data. For example, online learning management systems such as Canvas (https://openstax.org/r/instructure) compile a grade book in one place, and online textbooks such as zyBooks (https://openstax.org/r/zybooks) collect students' mastery level on each topic through their performance on exercises. All these data can be used to capture each student's progress and offer personalized learning experiences such as intelligent tutoring systems or adaptive learning. ALEKS (https://openstax.org/r/aleks), an online adaptive learning application, offers personalized material for each learner based on their past performance.

Data Science in Health Care and Medicine

The fields of health care and medicine also use data science. Often their goal is to offer more accurate diagnosis and treatment using **predictive analytics**—statistical techniques, algorithms, and machine learning that analyze historical data and make predictions about future events. Medical diagnosis and prescription practices have traditionally been based on a patient's verbal description of symptoms and a doctor's or a group of doctors' experience and intuition; this new movement allows health care professionals to make decisions that are more data-driven. Data-driven decisions became more feasible thanks to all the personal data collected through personal gadgets—smartphones, smart watches, and smart bands. Such devices collect daily health/activity records, and this in turn helps health care professionals better capture each patient's situation. All this work will enable patients to receive more accurate diagnoses along with more personalized treatment regimens in the long run.

The Precision Medicine Initiative (https://openstax.org/r/obamawhitehouse) is a long-term research endeavor carried out by the National Institutes of Health (NIH) and other research centers, with the goal of better understanding how a person's genetics, environment, and lifestyle can help determine the best approach to prevent or treat disease. The initiative aims to look at as much data as possible to care for a patient's health more proactively. For example, the initiative includes genome sequencing to look for certain mutations that indicate a higher risk of cancer or other diseases.

Another application of data science in health focuses on lowering the cost of health care services. Using historical records of patients' symptoms and prescription, a chatbot that is powered by artificial intelligence can provide automated health care advice. This will reduce the need for patients to see a pharmacist or doctor, which in turn improves health care accessibility for those who are in greater need.

EXPLORING FURTHER

Big Data In Health Care

The 2015 launch of the National Institutes of Health Precision Medicine Initiative was documented in One Woman's Quest to Cure Her Fatal Brain Disease (https://openstax.org/r/youtubevk). "Promise of Precision Medicine" signaled a new approach to health care in the United States—one heavily reliant on big data.

Data Science in Sports and Entertainment

Data science is prevalent in the sports and the entertainment industry as well. Sports naturally produce much data—about the player, positions, teams, seasons, and so on. Therefore, just as there is the concept of business analytics, the analysis of such data in sports is called **sports analytics**. For example, the Oakland Athletics baseball team famously analyzed player recruitment for the 2002 season. The team's management adapted a statistical approach referred to as **sabermetrics** to recruit and position players. With sabermetrics, the team was able to identify critical yet traditionally overlooked metrics such as on-base percentage and slugging percentage. The team, with its small budget compared to other teams, recruited a number of undervalued players with strong scores on these metrics, and in the process, they became one of the most exciting teams in baseball that year, breaking the American League record for 20 wins in a row. Does this story sound familiar? This story was so dramatic that Michael Lewis wrote a book about it, which was also made into a movie: *Moneyball*.

EXPLORING FURTHER

The Sabermetrics YouTube Channel

Sabermetrics is so popular that there is a YouTube channel devoted to it: Simple Sabermetrics (https://openstax.org/r/simplesabermetrics), with baseball animations and tutorials explaining how data impacts the way today's game is played behind the scenes.

In the entertainment industry, data science is commonly used to make data-driven, personalized suggestions that satisfy consumers known as **recommendation systems**. One example of a recommendation system is on video streaming services such as Netflix. Netflix Research considers subscribers' watch histories, satisfaction with the content, and interaction records (e.g., search history). Their goal is to make perfect personalized recommendations despite some challenges, including the fact that subscribers themselves often not do not know what they want to see.

EXPLORING FURTHER

Careers in Data Science

As you advance in your data science training, consider the many professional options in this evolving field. This helpful graphic from edX (https://openstax.org/r/edx) distinguishes data analyst vs. data science paths. This Coursera article (https://openstax.org/r/coursera) lists typical skill sets by role. Current practitioner discussions are available on forums such as Reddit's r/data science (https://openstax.org/r/reddit).

Trends and Issues in Data Science

Technology has made it possible to collect abundant amounts of data, which has led to challenges in the processing and analyzing of that data. But technology comes to the rescue again! Data scientists now use machine learning to better understand the data, and artificial intelligence can make an automated, data-driven decision on a task. Decision-Making Using Machine Learning Basics and Deep Learning and AI Basics will cover the details of machine learning and artificial intelligence.

With these advances, many people have raised concerns about ethics and privacy. Who is allowed to collect these data, and who has access to them? None of us want someone else to use our personal data (e.g., contact information, health records, location, photos, web search history) without our consent or without knowing the risk of sharing our data. Machine learning algorithms and artificial intelligence are trained to make a decision based on the past data, and when the past data itself inherits some bias, the trained machine learning algorithms and artificial intelligence will make biased decisions as well. Thus, carefully attending to the process of collecting data and evaluating the bias of a trained results is critical. Ethics Throughout the Data Science Cycle will discuss these and other ethical concerns and privacy issues in more depth.

1.3 Data and Datasets

Learning Outcomes

By the end of this section, you should be able to:

- 1.3.1 Define data and dataset.
- 1.3.2 Differentiate among the various data types used in data science.
- 1.3.3 Identify the type of data used in a dataset.
- 1.3.4 Discuss an item and attribute of a dataset.
- 1.3.5 Identify the different data formats and structures used in data science.

What Is Data Science? and Data Science in Practice introduced the many varieties of and uses for data science in today's world. Data science allows us to extract insights and knowledge from data, driving decision-making and innovation in business, health care, entertainment, and so on. As we've seen, the field has roots in math, statistics, and computer science, but it only began to emerge as its own distinct field in the early 2000s with the proliferation of digital data and advances in computing power and technology. It gained significant momentum and recognition around the mid to late 2000s with the rise of big data and the need for sophisticated techniques to analyze and derive insights from large and complex datasets. Its evolution since then has been rapid, and as we can see from the previous discussion, it is quickly becoming a cornerstone of many industries and domains.

Data, however, is not new! Humans have been collecting data and generating datasets from the beginning of time. This started in the Stone Age when people carved some shapes and pictures, called petroglyphs, on rock. The petroglyphs provide insights on how animals looked and how they carried out their daily life, which is valuable "data" for us. Ancient Egyptians invented a first form of paper—papyrus—in order to journal *their* data. Papyrus also made it easier to store data in bulk, such as listing inventories, noting financial transactions, and recording a story for future generations.

Data

"Data" is the plural of the Latin word "datum," which translates as something that is given or used and is often used to mean a single piece of information or a single point of reference in a dataset. When you hear the word "data," you may think of some sort of "numbers." It is true that numbers are usually considered data, but there are many other forms of data all around us. Anything that we can analyze to compile **information**—high-level insights—is considered **data**.

Suppose you are debating whether to take a certain course next semester. What process do you go through in order to make your decision? First, you might check the course evaluations, as shown in Table 1.1.

Semester	Instructor	Class Size	Rating
Fall 2020	A	100	Not recommended at all
Spring 2021	A	50	Highly recommended
Fall 2021	B	120	Not quite recommended
Spring 2022	B	40	Highly recommended
Fall 2022	A	110	Recommended
Spring 2023	B	50	Highly recommended

Table 1.1 Course Evaluation Records

The evaluation record consists of four kinds of data, and they are grouped as columns of the table: Semester, Instructor, Class Size, and Rating. Within each column there are six different pieces of data, located at each row. For example, there are six pieces of text data under the Semester column: "Fall 2020," "Spring 2021," "Fall 2021," "Spring 2022," "Fall 2022," and "Spring 2023."

The course evaluation ratings themselves do not provide an idea on whether to take the course next semester. The ratings are just a *phrase* (e.g., "Highly recommended" or "Not quite recommended") that *encodes* how

recommended the course was in that semester. You need to analyze them to come up with a decision!

Now let's think about how to derive the information from these ratings. You would probably look at all the data, including when in the semester the course was offered, who the instructor is, and class size. These records would allow you to derive information that would help you decide "whether or not to take the course next semester."

EXAMPLE 1.1

Problem

Suppose you want to decide whether or not to put on a jacket today. You research the highest temperatures in the past five days and determine whether you needed a jacket on each day. In this scenario, what data are you using? And what information are you trying to derive?

Solution

Temperature readings and whether you needed a jacket on each of the past five days are two kinds of data you are referring to. Again, they do not indicate anything related to wearing a jacket today. They are simply five pairs of numbers (temperature) and yes/no (whether you needed a jacket) records, with each pair representing a day. Using these data, you are deriving information that you can analyze to help you decide whether to wear a jacket today.

Types of Data

The previous sections talked about how much our daily life is surrounded by data, how much our daily life itself produces new data, and how often we make data-driven decisions without even noticing it. You might have noticed that data comes in various types. Some data are quantitative, which means that they are measured and expressed using numbers. **Quantitative data** deals with quantities and amounts and is usually analyzed using statistical methods. Examples include numerical measurements like height, weight, temperature, heart rate, and sales figures. **Qualitative data** are non-numerical data that generally describe subjective attributes or characteristics and are analyzed using methods such as thematic analysis or content analysis. Examples include descriptions, observations, interviews, and open-ended survey responses (as we'll see in Survey Design and Implementation) that address unquantifiable details (e.g., photos, posts on Reddit). The data types often dictate methods for data analysis, so it is important to be able to identify a type of data. Thus, this section will take a further dive into types of data.

Let's revisit our previous example about deciding whether to take a certain course next semester. In that example, we referred to four pieces of data. They are encoded in different types such as numbers, words, and symbols.

1. The semester the course was offered—Fall 2020, Spring 2021, ..., Fall 2022, Spring 2023
2. The instructor—A and B
3. The class size—100, 50, 120, 40, 110, 50
4. The course rating—"Not recommended at all," ..., "Highly recommended"

There are two primary types of quantitative data—*numeric* and *categorical*—and each of these can be divided into a few subtypes. **Numeric data** is represented in numbers that indicate measurable quantities. It may be followed by some symbols to indicate units. Numeric data is further divided into *continuous data* and *discrete data*. With **continuous data**, the values can be any number. In other words, a value is chosen from an infinite set of numbers. With **discrete data**, the values follow a specific precision, which makes the set of possible values finite.

From the previous example, the class size 100, 150, etc. are numbers with the implied unit "students." Also,

they indicate measurable quantities as they are head counts. Therefore, the class size is *numeric data*. It is also *continuous data* since the size numbers seem to be any natural numbers and these numbers are chosen from an infinite set of numbers, the set of natural numbers. Note that whether data is continuous (or discrete) also depends on the context. For example, the same class size data can be discrete if the campus enforces all classes to be 200 seats or less. Such restriction makes the class size values be chosen from a finite set of 200 numbers: 1, 2, 3, ..., 198, 199, 200.

Categorical data is represented in different forms such as words, symbols, and even numbers. A categorical value is chosen from a finite set of values, and the value does not necessarily indicate a measurable quantity. Categorical data can be divided into *nominal data* and *ordinal data*. For **nominal data**, the set of possible values does not include any ordering notion, whereas with **ordinal data**, the set of possible values includes an ordering notion.

The rest—semester, instructor, and ratings—are *categorical data*. They are represented in symbols (e.g., "Fall 2020," "A") or words (e.g., "Highly recommended"), and these values are chosen from the finite set of those symbols and words (e.g., A vs. B). The former two data are nominal since the semester and instructor do not have orders to follow, while the latter is ordinal since there is a notion of degree (Not recommended at all ~ Highly recommended). You may argue that the semester could have *chronological* ordering: Fall 2020 comes before Spring 2021, Fall 2021 follows Fall 2020. If you want to value that notion for your analysis, you could consider the semester data to be ordinal as well—the chronological ordering is indeed critical when you are looking at a time-series dataset. You will learn more about that in Time Series and Forecasting.

EXAMPLE 1.2

Problem

Consider the jacket scenario in Example 1.1. In that example, we referred to two kinds of data:

1. The temperature during past three days—90°F, 85°F, ...
2. On each of those days, whether you needed a jacket—Yes, No, ...

What is the type of each data?

Solution

The temperatures are numbers followed by the unit degrees Fahrenheit (°F). Also, they indicate measurable quantities as they are specific readings from a thermometer. Therefore, the temperature is numeric data. They are also continuous data since they can be any real number, and the set of real numbers is infinite.

The other type of data—whether or not you needed a jacket—is categorical data. Categorical data are represented in symbols (Yes/No), and the values are chosen from the finite set of those symbols. They are also nominal since Yes/No does not have ordering to follow.

Datasets

A **dataset** is a collection of observations or data entities organized for analysis and interpretation, as shown in Table 1.1. Many datasets can be represented as a table where each row indicates a unique data entity and each column defines the structure of the entities.

Notice that the dataset we used in Table 1.1 has six *entities* (also referred to as **items**, *entries*, or *instances*), distinguished by semester. Each entity is defined by a combination of four **attributes** or characteristics (also known as *features* or *variables*)—Semester, Instructor, Class Size, and Rating. A combination of features characterizes an entry of a dataset.

Although the actual values of the attributes are different across entities, note that all entities have values for

the same four attributes, which makes them a **structured dataset**. As a structured dataset, these items can be listed as a table where each item is listed along the rows of the table.

By contrast, an **unstructured dataset** is one that lacks a predefined or organized data model. While structured datasets are organized in a tabular format with clearly defined fields and relationships, unstructured data lacks a fixed schema. Unstructured data is often in the form of text, images, videos, audio recordings, or other content where the information doesn't fit neatly into rows and columns.

There are plenty of unstructured datasets. Indeed, some people argue there are more unstructured datasets than structured ones. A few examples include Amazon reviews on a set of products, Twitter posts last year, public images on Instagram, popular short videos on TikTok, etc. These unstructured datasets are often processed into a structured one so that data scientists can analyze the data. We'll discuss different data processing techniques in Collecting and Preparing Data.

EXAMPLE 1.3

Problem

Let's revisit the jacket example: deciding whether to wear a jacket to class. Suppose the dataset looks as provided in Table 1.2:

Date	Temperature	Needed a Jacket?
Oct. 10	80°F	No
Oct. 11	60°F	Yes
Oct. 12	65°F	Yes
Oct. 13	75°F	No

Table 1.2 Jacket Dataset

Is this dataset structured or unstructured?

Solution

It is a structured dataset since 1) every individual item is in the same structure with the same three attributes—Date, Temperature, and Needed a Jacket—and 2) each value strictly fits into a cell of a table.

EXAMPLE 1.4

Problem

How many entries and attributes does the dataset in the previous example have?

Solution

The dataset has four entries, each of which is identified with a specific date (Oct. 10, Oct. 11, Oct. 12, Oct. 13). The dataset has three attributes—Date, Temperature, Needed a Jacket.

EXAMPLE 1.5

Problem

A dataset has a list of keywords that were searched on a web search engine in the past week. Is this dataset structured or unstructured?

Solution

The dataset is an unstructured dataset since each entry in the dataset can be a freeform text: a single word, multiple words, or even multiple sentences.

EXAMPLE 1.6

Problem

The dataset from the previous example is processed so that now each search record is summarized as up to three words, along with the timestamp (i.e., when the search occurred). Is this dataset structured or unstructured?

Solution

It is a structured dataset since every entry of this dataset is in the same structure with two attributes: a short keyword along with the timestamp.

Dataset Formats and Structures (CSV, JSON, XML)

Datasets can be stored in different formats, and it's important to be able to recognize the most commonly used formats. This section covers three of the most often used formats for structured datasets—**comma-separated values (CSV)**, **JavaScript Object Notation (JSON)**, and **Extensible Markup Language (XML)**. While CSV is the most intuitive way of encoding a tabular dataset, much of the data we would collect from the web (e.g., websites, mobile applications) is stored in the JSON or XML format. The reason is that JSON is the most suitable for exchanging data between a user and a server, and XML is the most suitable for complex dataset due to its hierarchy-friendly nature. Since they all store data as plain texts, you can open them using any typical text editors such as Notepad, Visual Studio Code, Sublime Text, or VI editor.

Table 1.3 summarizes the advantages and disadvantages of CSV, JSON, and XML dataset formats. Each of these is described in more detail below.

Dataset Format	Pros	Cons	Typical Use
CSV	• Simple	• Difficult to add metadata • Difficult to parse if there are special characters • Flat structure	• Tabular data
JSON	• Simple • Compatible with many languages • Easy to parse	• Difficult to add metadata • Cannot leave comments	• Data that needs to be exchanged between a user and a server
XML	• Structured (so more readable) • Possible to add metadata	• Verbose • Complex structure with tags	• Hierarchical data structures

Table 1.3 Summary of the CSV, JSON, and XML formats

EXPLORING FURTHER

Popular and Reliable Databases to Search for Public Datasets

Multiple online databases offer public datasets for free. When you want to look for a dataset of interest, the following sources can be your initial go-to.

Government data sources include:

Data.gov (https://openstax.org/r/datagov)

Bureau of Labor Statistics (BLS) (https://openstax.org/r/blsgov1)

National Oceanic and Atmospheric Administration (NOAA) (https://openstax.org/r/noaagov)

World Health Organization (WHO) (https://openstax.org/r/who)

Some reputable nongovernment data sources are:

Kaggle (https://openstax.org/r/kaggle1)

Statista (https://openstax.org/r/statista)

Pew Research Center (https://openstax.org/r/pewresearch)

Comma-Separated Values (CSV)

The CSV stores each item in the dataset in a single line. Variable values for each item are listed all in one line, separated by commas (","). The previous example about signing up for a course can be stored as a CSV file. Figure 1.4 shows how the dataset looks when opened with a *text editor* (e.g., Notepad, TextEdit, MS Word, Google Doc) or programming software in the form of a *code editor* (e.g., Sublime Text, Visual Studio Code, XCode). Notice that commas are used to separate the attribute values within a single line (see Figure 1.4).

```
1  Semester,Instructor,Class Size,Rating
2  Fall 2020,A,100,Not recommended at all
3  Spring 2021,A,50,Highly recommended
4  Fall 2021,B,120,Not quite recommended
5  Spring 2022,B,40,Highly recommended
6  Fall 2022,A,110,Recommended
7  Spring 2023,B,50,Highly recommended
```

Figure 1.4 ch1-courseEvaluations.csv Opened with Visual Studio Code

COMMA FOR NEW LINE?

There is some flexibility on how to end a line with CSV files. It is acceptable to end with or without commas, as some software or programming languages automatically add a comma when generating a CSV dataset.

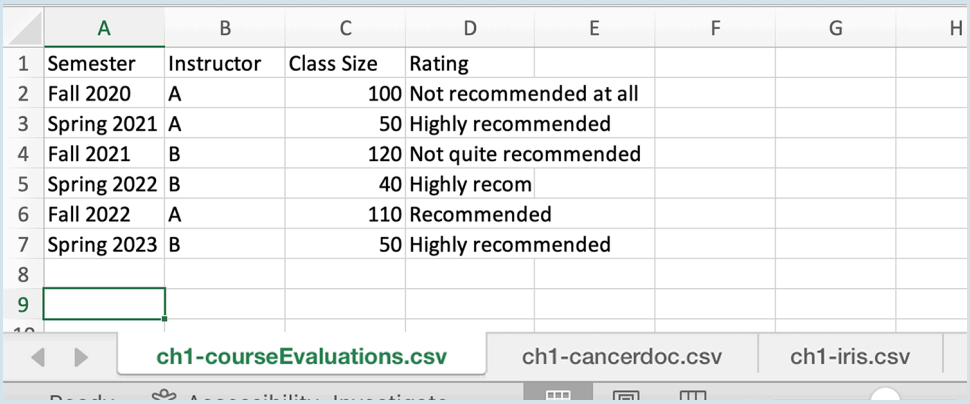
CSV files can be opened with spreadsheet software such as MS Excel and Google Sheets. The spreadsheet software visualizes CSV files more intuitively in the form of a table (see Figure 1.5). We will cover the basic use of Python for analyzing CSV files in Data Science with Python.

	A	B	C	D
1	Semester	Instructor	Class Size	Rating
2	Fall 2020	A	100	Not recommended at all
3	Spring 2021	A	50	Highly recommended
4	Fall 2021	B	120	Not quite recommended
5	Spring 2022	B	40	Highly recommended
6	Fall 2022	A	110	Recommended
7	Spring 2023	B	50	Highly recommended
8				

Figure 1.5 ch1-courseEvaluations.csv Opened with Microsoft Excel (Used with permission from Microsoft)

HOW TO DOWNLOAD AND OPEN A DATASET FROM THE CH1-DATA SPREADSHEET IN THIS TEXT

A spreadsheet file accompanies each chapter of this textbook. The files include multiple tabs corresponding to a single dataset in the chapter. For example, the spreadsheet file for this chapter (https://openstax.org/r/spreadsheet4) is shown in Figure 1.6. Notice that it includes multiple tabs with the names "ch1-courseEvaluations.csv," "ch1-cancerdoc.csv," and "ch1-riris.csv," which are the names of each dataset.



	A	B	C	D	E	F	G	H
1	Semester	Instructor	Class Size	Rating				
2	Fall 2020	A	100	Not recommended at all				
3	Spring 2021	A	50	Highly recommended				
4	Fall 2021	B	120	Not quite recommended				
5	Spring 2022	B	40	Highly recom				
6	Fall 2022	A	110	Recommended				
7	Spring 2023	B	50	Highly recommended				
8								
9								

Figure 1.6 The dataset spreadsheet file for Chapter 1 (Used with permission from Microsoft)

To save each dataset as a separate CSV file, choose the tab of your interest and select File > Save As ... > CSV File Format. This will only save the current tab as a CSV file. Make sure the file name is set correctly; it may have used the name of the spreadsheet file—"ch1-data.xlsx" in this case. You should name the generated CSV file as the name of the corresponding tab. For example, if you have generated a CSV file for the first tab of "ch1-data.xlsx," make sure the generated file name is "ch1-courseEvaluations.csv." This will prevent future confusion when following instructions in this textbook.

JavaScript Object Notation (JSON)

JSON uses the syntax of a programming language named JavaScript. Specifically, it follows JavaScript's object syntax. Don't worry, though! You do not need to know JavaScript to understand the JSON format.

Figure 1.7 provides an example of the JSON representation of the same dataset depicted in Figure 1.6.


```
{"Semester":"Fall 2020","Instructor":"A","Class Size":100,"Rating":"Not recommended at all"}
{"Semester":"Spring 2021","Instructor":"A","Class Size":50,"Rating":"Highly recommended"}
{"Semester":"Fall 2021","Instructor":"B","Class Size":120,"Rating":"Not quite recommended"}
{"Semester":"Spring 2022","Instructor":"B","Class Size":40,"Rating":"Highly recommended"}
{"Semester":"Fall 2022","Instructor":"A","Class Size":110,"Rating":"Recommended"}
{"Semester":"Spring 2023","Instructor":"B","Class Size":50,"Rating":"Highly recommended"}
```


Figure 1.7 CourseEvaluations.json Opened with Visual Studio Code

Notice that the JSON format starts and ends with a pair of curly braces ({}). Inside, there are multiple pairs of two fields that are separated by a colon (:). These two fields that are placed on the left and right of the colon are called a key and value, respectively,—key : value. For example, the dataset in Figure 1.7 has five pairs of key-values with the key `"Semester": "Fall 2020"`, `"Semester": "Spring 2021"`, `"Semester": "Fall 2021"`, `"Semester": "Spring 2022"`, `"Semester": "Fall 2022"`, and `"Semester": "Spring 2023"`.

CourseEvaluations.json (https://openstax.org/r/filed1v) has one key-value pair at the highest level: `"Members": [...]`. You can see that each item of the dataset is listed in the form of an *array* or list under the key `"Members"`. Inside the array, each item is also bound by curly braces and has a list of key-value pairs, separated by commas. Keys are used to describe attributes in the dataset, and values are used to define the corresponding values. For example, the first item in the JSON dataset above has four keys, each of which maps to each attribute—Semester, Instructor, Class Size, and Rating. Their values are `"Fall 2020"`, `"A"`, `100`, and `"Not recommended at all"`.

Extensible Markup Language (XML)

The XML format is like JSON, but it lists each item of the dataset using different symbols named *tags*. An **XML tag** is any block of text that consists of a pair of angle brackets (< >) with some text inside. Let's look at the example XML representation of the same dataset in Figure 1.8. Note that the screenshot of

CourseEvaluations.xml below only includes the first three items in the original dataset.

```
<Members>
    <Evaluation>
        <Semester>"Fall 2020"</Semester>
        <Instructor>"A"</Instructor>
        <Classsize>100</Classsize>
        <Rating>"Not recommended at all"</Rating>
    </Evaluation>
    <Evaluation>
        <Semester>"Spring 2021"</Semester>
        <Instructor>"A"</Instructor>
        <Classsize>50</Classsize>
        <Rating>"Highly recommended"</Rating>
    </Evaluation>
    <Evaluation>
        <Semester>"Fall 2021"</Semester>
        <Instructor>"B"</Instructor>
        <Classsize>120</Classsize>
        <Rating>"Not quite recommended"</Rating>
    </Evaluation>
</Members>
```

Figure 1.8 ch1-courseEvaluations.xml with the First Three Entries Only, Opened with Visual Studio Code

CourseEvaluations.xml lists each item of the dataset between a pair of tags, `<members>` and `</members>`. Under `<members>`, each item is defined between `<evaluation>` and `</evaluation>`. Since the dataset in Figure 1.8 has three items, we can see three blocks of `<evaluation>` ... `</evaluation>`. Each item has four attributes, and they are defined as different XML tags as well—`<semester>`, `<instructor>`, `<classsize>`, and `<rating>`. They are also followed by closing tags such as `</semester>`, `</instructor>`, `</classsize>`, and `</rating>`.

PubMed datasets (https://openstax.org/r/pubmed1) provides a list of articles that are published in the National Library of Medicine in XML format. Click Annual Baseline and download/open any .xml file. Note that all the .xml files are so big that they are compressed to .gz files. However, once you download one and attempt to open it by double-clicking, the file will automatically be decompressed and open. You will see a bunch of XML tags along with information about numerous publications, such as published venue, title, published date, etc.

XML and Image Data

The XML format is also commonly used as an attachment to some image data. It is used to note supplementary information about the image. For example, the Small Traffic Light Dataset (https://openstax.org/r/traffic) in Figure 1.9 comes with a set of traffic light images, placed in one of the three directories: `JPEGImages`, `train_images`, and `valid_images`. Each image directory is accompanied with another directory just for annotations such as `Annotations`, `train_annotations`, and `valid_annotations`.

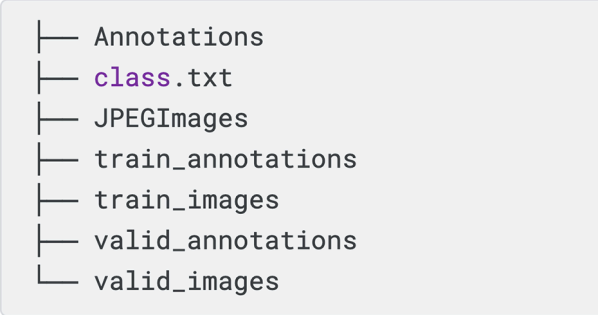


Figure 1.9 The Directory Structure of the Small Traffic Light Dataset

The annotation directories have a list of XML files, each of which corresponds to an image file with the same filename inside the corresponding image directory (Figure 1.10). Figure 1.11 shows that the first XML file in the Annotations directory includes information about the .jpg file with the same filename.

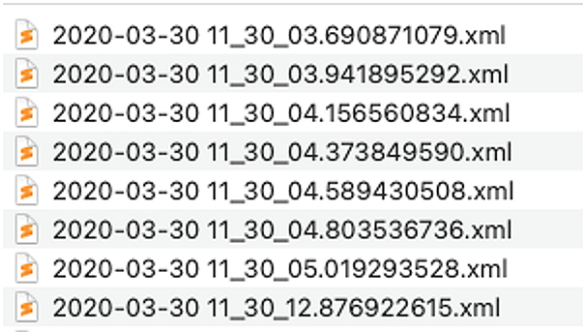


Figure 1.10 List of XML Files under the Annotations Directory in the Small Traffic Light Dataset (source: "Small Traffic Light Dataset," https://www.kaggle.com/datasets/sovitrath/small-traffic-light-dataset-xml-format)

Figure 1.11 2020-03-30 11_30_03.690871079.xml, an Example XML file within the Small Traffic Light Dataset (Source: "Small Traffic Light Dataset," https://www.kaggle.com/datasets/sovitrath/small-traffic-light-dataset-xml-format)

JSON and XML Dataset Descriptions

Both JSON and XML files often include some description(s) of the dataset itself as well (known as *metadata*), and they are included as a separate entry in the file ({} or <>). In Figure 1.12 and Figure 1.13, the actual data entries are listed inside "itemData" and `<data>`, respectively. The rest are used to provide background information on the dataset. For example:

- "creationDateTime": describes when the dataset was created.
- `<name>` is used to write the name of this dataset.
- `<metadata>` is used to describe each column name of the dataset along with its data type.

```
{
    "creationDateTime": "2023-03-22T11:53:27",
    "datasetJSONVersion": "1.0.0",
    "fileOID": "www.sponsor.org.project123.final",
    "sourceSystemVersion": "1.2.3",
    "clinicalData": {
        "items": [
                    {"OID": "ITEMGROUPDATASEQ", "name": "ITEMGROUPDATASEQ", "label": "Record identifier", "type": "integer"},
                    {"OID": "IT.STUDYID", "name": "STUDYID", "label": "Study identifier", "type": "string"},
                    {"OID": "IT.DOMAIN", "name": "DOMAIN", "label": "Domain identifier", "type": "string", "length": 2},
            ],
            "itemData": [
                [1, "MyStudy1", "D1"],
                [2, "MyStudy2", "D2"],
                ...
            ]
        }
    }
}
```

Figure 1.12 An Example JSON File with Metadata

```
<?xml version="1.0" encoding="UTF-8"?>
<dataset>
    <name>CEREALS</name>
    <metadata>
        <version>1.2</version>
        <date>04/02/2024</date>
        <description>a list of some popular cereals</description>
        <col name="NAME" type="string"/>
        <col name="MANUFACTURER" type="string"/>
        <col name="CALORIES_PER_SERVING" type="integer"/>
    </metadata>
    <data>
        <row>
            <value>Cheerios</value>
            <value>General Mills</value>
            <value>140</value>
        </row>
        <row>
            <value>Corn Flakes</value>
            <value>Kellogg's</value>
            <value>150</value>
        </row>
    </data>
</dataset>
```

Figure 1.13 An Example XML Dataset with Metadata

The Face Mask Detection (https://openstax.org/r/andrewmvd) dataset has a set of images of human faces with masks on. It follows a similar structure as well. The dataset consists of two directories—annotations and images. The former is in the XML format. The name of each XML file includes any text description about the

image with the same filename. For example, “maksssksksss0.xml” includes information on “maksssksksss0.png.”

EXAMPLE 1.7

Problem

The Iris Flower dataset (ch1-iris.csv (https://openstax.org/r/filed)) is a classic dataset in the field of data analysis.[1] Download this dataset and open it with a code editor (e.g., Sublime Text, XCode, Visual Studio Code). (We recommend that if you do not have any code editor installed, you install one. All three of these editors are quick and easy to install.) Now, answer these questions:

- How many items are there in the dataset?
- How many attributes are there in the dataset?
- What is the second attribute in the dataset?

Solution

There are 151 rows in the dataset with the header row at the top, totaling 150 items. There are five attributes listed across columns: `sepal_length`, `sepal_width`, `petal_length`, `petal_width`, `species`. The second attribute is `sepal_width`.

EXAMPLE 1.8

Problem

The Jeopardy dataset (ch1-jeopardy.json (https://openstax.org/r/filed15)) is formatted in JSON. Download and open it with a code editor (e.g., Notepad, Sublime Text, Xcode, Visual Studio Code).

- How many items are there in the dataset?
- How many attributes are there in the dataset?
- What is the third item in the dataset?

Solution

There are 409 items in the dataset, and each item has seven attributes: “`category`”, “`air_date`”, “`question`”, “`value`”, “`answer`”, “`round`”, and “`show_number`”. The third item is located at index 2 of the first list as shown in Figure 1.14.


```
"category" : string "EVERYBODY TALKS ABOUT IT..."
"air_date" : string "2004-12-31"
"question" :
string "'The city of Yuma in this state has a record average of 4,055 hours of sunshine each year'"
"value" : string "$200"
"answer" : string "Arizona"
"round" : string "Jeopardy!"
"show_number" : string "4680"
```


Figure 1.14 The Third Item in the Jeopardy Dataset

1 The Iris Flower dataset was introduced by the British statistician and biologist Ronald Fisher in his 1936 paper “The Use of Multiple Measurements in Taxonomic Problems.” This work became a landmark study in the use of multivariate data in classification problems and frequently makes an appearance in data science as a convenient test case for machine learning and neural network algorithms. The Iris Flower dataset is often used as a beginner's dataset to demonstrate various techniques, such as classification of algorithms, formatted in CSV.

1.4 Using Technology for Data Science

Learning Outcomes

By the end of this section, you should be able to:

- 1.4.1 Explain how statistical software can help with data analysis.
- 1.4.2 Explain the uses of different programs and programming languages for data manipulation, analysis, and visualizations.
- 1.4.3 Explain the uses of various data analysis tools used in data science applications.

Technology empowers data analysts, researchers, and organizations to leverage data, extract actionable insights, and make decisions that optimize processes and improve outcomes in many areas of our lives. Specifically, technology provides the tools, platforms, and algorithms that enable users to efficiently process and analyze data—especially complex datasets. The choice of technology used for a data science project will vary depending on the goals of the project, the size of the datasets, and the kind of analysis required.

Spreadsheet Programs

Spreadsheet programs such as Excel and Google Sheets are software applications consisting of electronic worksheets with rows and columns where data can be entered, manipulated, and calculated. Spreadsheet programs offer a variety of functions for data manipulation and can be used to easily create charts and tables. **Excel** is one of the most widely used spreadsheet programs, and as part of Microsoft Office, it integrates well with other Office products. Excel was first released by Microsoft in 1987, and it has become one of the most popular choices for loading and analyzing tabular data in a spreadsheet format. You are likely to have used Excel in some form or other —perhaps to organize the possible roommate combinations in your dorm room or to plan a party, or in some instructional context. We refer to the use of Excel to manipulate data in some of the examples of this text because sometimes a spreadsheet is simply the easiest way to work with certain datasets. (See Appendix A: Review of Excel for Data Science for a review of Excel functionality.)

Google Sheets is a cloud-based spreadsheet program provided by Google as part of the Google Workspace. Because it is cloud-based, it is possible to access spreadsheets from any device with an internet connection. This accessibility allows for collaboration and real-time updates among multiple users, enhancing communication within a team and making it ideal for team projects or data sharing among colleagues. Users can leave comments, track changes, and communicate within the spreadsheet itself.

The user interfaces for both Excel and Google Sheets make these programs very user-friendly for many applications. But these programs have some limitations when it comes to large databases or complex analyses. In these instances, data scientists will often turn to a programming language such as Python, R, or SPSS.

Programming Languages

A **programming language** is a formal language that consists of a set of instructions or commands used to communicate with a computer and to instruct it to perform specific tasks that may include data manipulation, computation, and input/output operations. Programming languages allow developers to write algorithms, create software applications, and automate tasks and are better suited than spreadsheet programs to handle complex analyses.

Python and R are two of the most commonly used programming languages today. Both are open-source programming languages. While **Python** started as a general-purpose language that covers various types of tasks (e.g., numerical computation, data analysis, image processing, and web development), **R** is more specifically designed for statistical computing and graphics. Both use simple syntax compared to conventional programming languages such as Java or C/C++. They offer a broad collection of packages for data manipulation, statistical analysis, visualization, machine learning, and complex data modeling tasks.

We have chosen to focus on Python in this text because of its straightforward and intuitive syntax, which makes it especially easy for beginners to learn and apply. Also, Python skills can apply to a wide range of computing work. Python also contains a vast network of libraries and frameworks specifically designed for data analysis, machine learning, and scientific computing. As we'll see, Python libraries such as NumPy (https://openstax.org/r/nump), Pandas (https://openstax.org/r/panda), Matplotlib (https://openstax.org/r/matplot), and Seaborn (https://openstax.org/r/seabo) provide powerful tools for data manipulation, visualization, and machine learning tasks, making Python a versatile choice for handling datasets. You'll find the basics of R and various R code examples in Appendix B: Review of R Studio for Data Science. If you want to learn how to use R, you may want to practice with RStudio (https://openstax.org/r/posit1), a commonly used software application to edit/run R programs.

PYTHON IN EXCEL

Microsoft recently launched a new feature for Excel named "Python in Excel." This feature allows a user to run Python code to analyze data directly in Excel. This textbook does not cover this feature, but instead presents Python separately, as it is also crucial for you to know how to use each tool in its more commonly used environment. If interested, refer to Microsoft's announcement (https://openstax.org/r/youtu).

EXPLORING FURTHER

Specialized Programming Languages

Other programming languages are more specialized for a particular task with data. These include SQL, Scala, and Julia, among others, as briefly described in this article on "The Nine Top Programming Languages for Data Science (https://openstax.org/r/9top)."

Other Data Analysis/Visualization Tools

There are a few other data analysis tools that are strong in data visualization. Tableau (https://openstax.org/r/tableau1) and PowerBI (https://openstax.org/r/micro) are user-friendly applications for data visualization. They are known for offering more sophisticated, interactive visualizations for high-dimensional data. They also offer a relatively easy user interface for compiling an analysis dashboard. Both allow users to run a simple data analysis as well before visualizing the results, similar to Excel.

In general, data visualization aims to make complex data more understandable and usable. The tools and technology described in this section offer a variety of ways to go about creating visualizations that are most accessible. Refer to Visualizing Data for a deeper discussion of the types of visualizations—charts, graphs, boxplots, histograms, etc.—that can be generated to help find the meaning in data.

EXPLORING FURTHER

Evolving Professional Standards

Data science is a field that is changing daily; the introduction of artificial intelligence (AI) has increased this pace. Technological, social, and ethical challenges with AI are discussed in Natural Language Processing, and ethical issues associated with the whole data science process, including the use of machine learning and artificial intelligence, are covered in Ethics Throughout the Data Science Cycle. A variety of data science professional organizations are working to define and update process and ethical standards on an ongoing basis. Useful references may include the following:

- Initiative for Analytics and Data Science Standards (IADSS) (https://openstax.org/r/iadss1)
- Data Science Association (DSA) (https://openstax.org/r/datascienceassn1)
- Association of Data Scientists (ADaSci) (https://openstax.org/r/adasci1)

1.5 Data Science with Python

Learning Outcomes

By the end of this section, you should be able to

- 1.5.1 Load data to Python.
- 1.5.2 Perform basic data analysis using Python.
- 1.5.3 Use visualization principles to graphically plot data using Python.

Multiple tools are available for writing and executing Python programs. Jupyter Notebook is one convenient and user-friendly tool. The next section explains how to set up the Jupyter Notebook environment using Google Colaboratory (Colab) and then provides the basics of two open-source Python libraries named `Pandas` and `Matplotlib`. These libraries are specialized for data analysis and data visualization, respectively.

EXPLORING FURTHER

Python Programming

In the discussion below, we assume you are familiar with basic Python syntax and know how to write a simple program using Python. If you need a refresher on the basics, please refer to Das, U., Lawson, A., Mayfield, C., & Norouzi, N. (2024). *Introduction to Python Programming*. OpenStax. https://openstax.org/books/introduction-python-programming/pages/1-introduction (https://openstax.org/r/page1).

Jupyter Notebook on Google Colaboratory

Jupyter Notebook is a web-based environment that allows you to run a Python program more interactively, using programming code, math equations, visualizations, and plain texts. There are multiple web applications or software you could use to edit a Jupyter Notebook, but in this textbook we will use Google's free application named **Google Colaboratory (Colab)** (https://openstax.org/r/colab1), often abbreviated as Colab. It is a cloud-based platform, which means that you can open, edit, run, and save a Jupyter Notebook on your Google Drive.

Setting up Colab is simple. On your Google Drive, click New > More. If your Google Drive has already installed Colab before, you will see Colaboratory under More. If not, click "Connect more apps" and install Colab by searching "Colaboratory" on the app store (Figure 1.15). For further information, see the Google Colaboratory Ecosystem (https://openstax.org/r/1pp) animation.

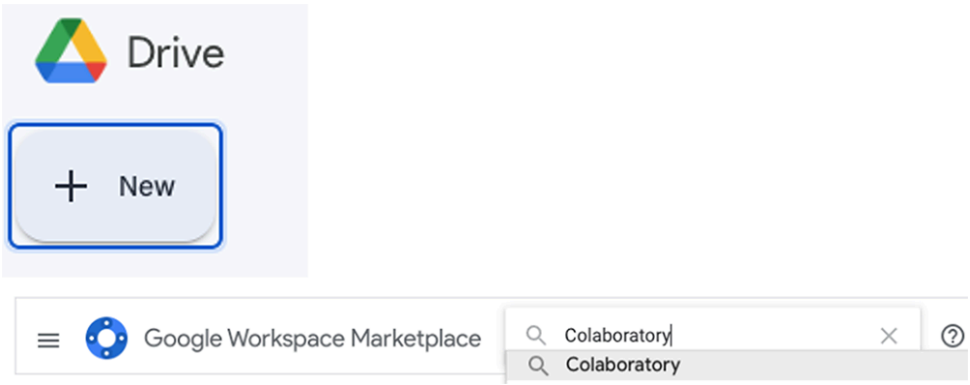


Figure 1.15 Install Google Colaboratory (Colab)

Now click New > More > Google Laboratory. A new, empty Jupyter Notebook will show up as in Figure 1.16.

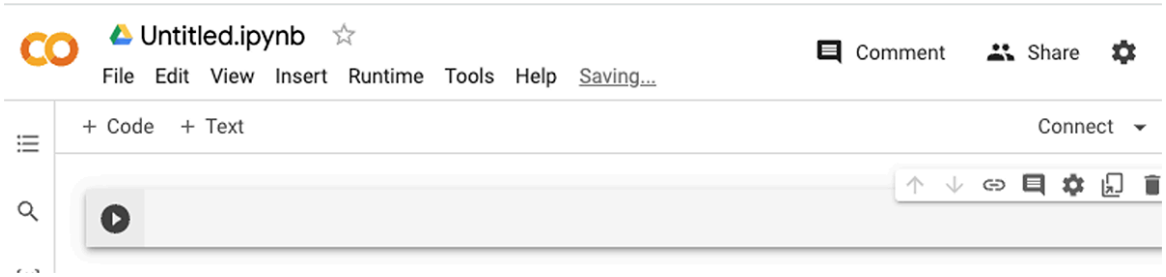


Figure 1.16 Google Colaboratory Notebook

The gray area with the play button is called a cell. A **cell** is a block where you can type either code or plain text. Notice that there are two buttons on top of the first cell—"+ Code" and "+ Text." These two buttons add a code or text cell, respectively. A code cell is for the code you want to run; a text cell is to add any text description or note.

Let's run a Python program on Colab. Type the following code in a code cell.

PYTHON CODE

```
print ("hello world!")
```

The resulting output will look like this:

```
hello world!
```

You can write a Python program across multiple cells and put text cells in between. Colab would treat all the code cells as part of a single program, running from the top to bottom of the current Jupyter Notebook. For example, the two code cells below run as if it is a single program.

When running one cell at a time from the top, we see the following outputs under each cell.

PYTHON CODE

```
a = 1
print ("The a value in the first cell:", a)
```

The resulting output will look like this:

```
The a value in the first cell: 1
```

PYTHON CODE

```
b = 3
print ("a in the second cell:", a)
print ("b in the second cell:", b)
a + b
```

The resulting output will look like this:

```
a in the second cell: 1
b in the second cell: 3
4
```

CONVENTIONAL PYTHON VERSUS JUPYTER NOTEBOOK SYNTAX

While conventional Python syntax requires `print()` syntax to print something to the program console, Jupyter Notebook does not require `print()`. On Jupyter Notebook, the line `a+b` instead of `print(a+b)` also prints the value of `a+b` as an output. But keep in mind that if there are multiple lines of code that trigger printing some values, *only* the output from the last line will show.

You can also run multiple cells in bulk. Click Runtime on the menu, and you will see there are multiple ways of running multiple cells at once (Figure 1.17). The two commonly used ones are “Run all” and “Run before.” “Run all” runs all the cells in order from the top; “Run before” runs all the cells before the currently selected one.

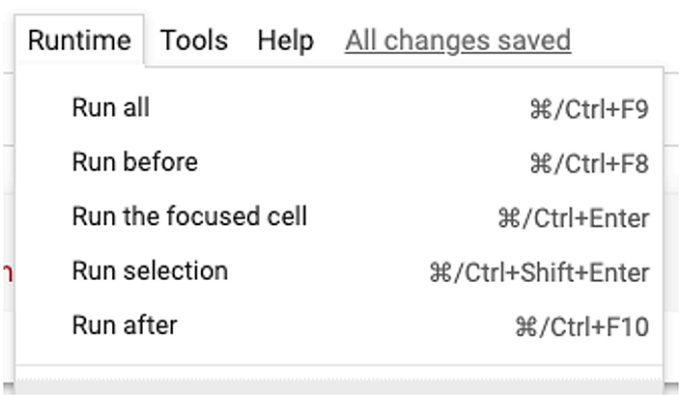


Figure 1.17 Multiple Ways of Running Cells on Colab

One thing to keep in mind is that being able to split a long program into multiple blocks and run one block at a time raises chances of user error. Let's look at a modified code from the previous example.

PYTHON CODE

```
a = 1
print ("the value in the first cell:", a)
```

The resulting output will look like this:

```
the value in the first cell: 1
```

PYTHON CODE

```
b = 3
print ("a in the second cell:", a)
print ("b in the second cell:", b)
a + b
```

The resulting output will look like this:

```
a in the second cell: 1
b in the second cell: 3
4
```

PYTHON CODE

```
a = 2
a + b
```

The resulting output will look like this:

```
5
```

The modified code has an additional cell at the end, updating `a` from 1 to 2. Notice that now `a+b` returns 5 as `a` has been changed to 2. Now suppose you need to run the second cell for some reason, so you run the second cell again.

PYTHON CODE

```
a = 1
print ("the a value in the first cell:", a)
```

The resulting output will look like this:

```
the a value in the first cell: 1
```

PYTHON CODE

```
b = 3
print ("a in the second cell:", a)
print ("b in the second cell:", b)
a + b
```

The resulting output will look like this:

```
a in the second cell: 2
b in the second cell: 3
5
```

PYTHON CODE

```
a = 2
a + b
```

The resulting output will look like this:

```
5
```

The value of `a` has changed to 2. This implies that the execution order of each cell matters! If you have run the third cell before the second cell, the value of `a` will have the value from the third one even though the third cell is located below the second cell. Therefore, it is recommended to use “Run all” or “Run before” after you make changes across multiple cells of code. This way your code is guaranteed to run sequentially from the top.

Python Pandas

One of the strengths of Python is that it includes a variety of free, open-source libraries. Libraries are a set of already-implemented methods that a programmer can refer to, allowing a programmer to avoid building common functions from scratch.

`Pandas` is a Python library specialized for data manipulation and analysis, and it is very commonly used among data scientists. It offers a variety of methods, which allows data scientists to quickly use them for data analysis. You will learn how to analyze data using `Pandas` throughout this textbook.

Colab already has `Pandas` installed, so you just need to import `Pandas` and you are set to use all the methods in `Pandas`. Note that it is convention to abbreviate pandas to pd so that when you call a method from `Pandas`, you can do so by using `pd` instead of having to type out `Pandas` every time. It offers a bit of convenience for a programmer!

PYTHON CODE

```
# import Pandas and assign an abbreviated identifier "pd"
import pandas as pd
```

EXPLORING FURTHER

Installing Pandas on Your Computer

If you wish to install Pandas on your own computer, refer to the installation page of the Pandas website (https://openstax.org/r/pyd).

Load Data Using Python Pandas

The first step for data analysis is to load the data of your interest to your Notebook. Let's create a folder on Google Drive where you can keep a CSV file for the dataset and a Notebook for data analysis. Download a public dataset, ch1-movieprofit.csv (https://openstax.org/r/filed), and store it in a Google Drive folder. Then open a new Notebook in that folder by entering that folder and clicking New > More > Google Colaboratory.

Open the Notebook and allow it to access files in your Google Drive by following these steps:

First, click the Files icon on the side tab (Figure 1.18).

Figure 1.18 Side Tab of Colab

Then click the Mount Drive icon (Figure 1.19) and select "Connect to Google Drive" on the pop-up window.

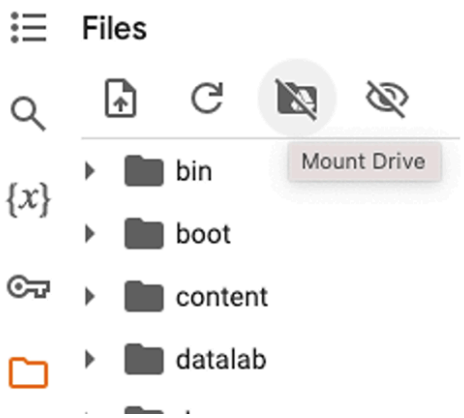


Figure 1.19 Features under Files on Colab

Notice that a new cell has been inserted on the Notebook as a result (Figure 1.20).

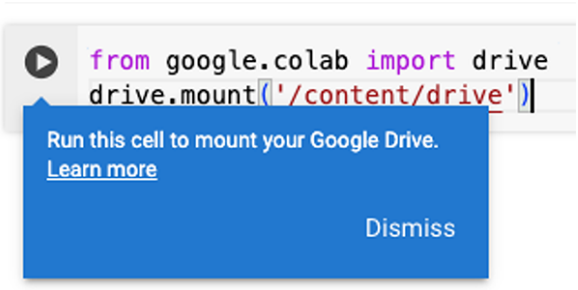


Figure 1.20 An Inserted Cell to Mount Your Google Drive

Connect your Google Drive by running the cell, and now your Notebook file can access all the files under content/drive. Navigate folders under drive to find your Notebook and ch1-movieprofit.csv (https://openstax.org/r/filed) files. Then click "..." > Copy Path (Figure 1.21).

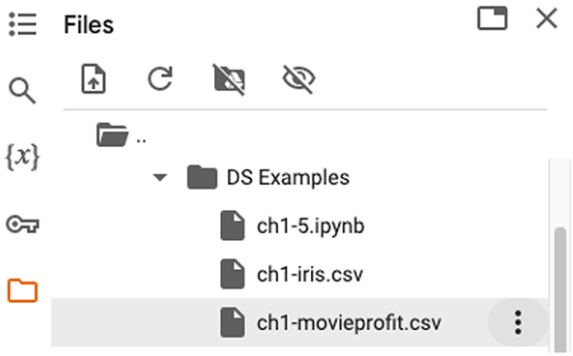


Figure 1.21 Copying the Path of a CSV File Located in a Google Drive Folder

Now replace [Path] with the copied path in the below code. Run the code and you will see the dataset has been loaded as a table and stored as a Python variable data.

PYTHON CODE

```
# import Pandas and assign an abbreviated identifier "pd"
import pandas as pd
```

```
data = pd.read_csv("[Path]")
data
```

The resulting output will look like this:

	Unnamed: 0	Title	Year	Genre	Rating	Duration	US_Gross_Million	Worldwide_Gross_Million	Votes
0	1	Avatar	2009	Action	7.8	162	760.51	2847.40	1,236,962
1	2	Avengers: Endgame	2019	Action	8.4	181	858.37	2797.50	1,108,641
2	3	Titanic	1997	Drama	7.9	194	659.33	2201.65	1,162,142
3	4	Star Wars: Episode VII - The Force Awakens	2015	Action	7.8	138	936.66	2069.52	925,551
4	5	Avengers: Infinity War	2018	Action	8.4	149	678.82	2048.36	1,062,517
...	...	...	...	...	...	...	...	...	...
961	962	The A-Team	2010	Action	6.7	117	77.22	177.24	259,316
962	963	Tootsie	1982	Comedy	7.4	116	177.20	177.20	107,311
963	964	In the Line of Fire	1993	Action	7.2	128	102.31	177.00	104,598
964	965	Analyze This	1999	Comedy	6.7	103	106.89	176.89	154,726
965	966	The Hitman's Bodyguard	2017	Action	6.9	118	75.47	176.60	230,821

The `read_csv()` method in `Pandas` loads a CSV file and stores it as a DataFrame. A **DataFrame** is a data type that `Pandas` uses to store multi-column tabular data. Therefore, the variable data holds the table in ch1-movieprofit.csv (https://openstax.org/r/filed) in the form of a `Pandas` DataFrame.

DATAFRAME VERSUS SERIES

`Pandas` defines two data types for tabular data—DataFrame and Series. While DataFrame is used for multi-column tabular data, Series is used for single-column data. Many methods in `Pandas` support both DataFrame and Series, but some are only for one or the other. It is always good to check if the method you are using works as you expect. For more information, refer to the Pandas documentation (https://openstax.org/r/docs) or Das, U., Lawson, A., Mayfield, C., & Norouzi, N. (2024). *Introduction to Python Programming*. OpenStax. https://openstax.org/books/introduction-python-programming/pages/1-introduction (https://openstax.org/r/page1).

EXAMPLE 1.9

Problem

Remember the Iris dataset we used in Data and Datasets? Load the dataset ch1-iris.csv (https://openstax.org/r/filed) to a Python program using `Pandas`.

Solution

The following code loads the ch1-iris.csv (https://openstax.org/r/filed) that is stored in a Google Drive. Make sure to replace the path with the actual path to ch1-iris.csv (https://openstax.org/r/filed) on your Google Drive.

PYTHON CODE

```
import pandas as pd

data = pd.read_csv("[Path to ch1-iris.csv]") # Replace the path
data
```

The resulting output will look like this:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

EXPLORING FURTHER

Can I load a file that is uploaded to someone else's Google Drive and shared with me?

Yes! This is useful especially when your Google Drive runs out of space. Simply add the shortcut of the shared file to your own drive. Right-click > Organize > Add Shortcut will let you select where to store the shortcut. Once done, you can call `pd.read_csv()` using the path of the shortcut.

Summarize Data Using Python Pandas

You can compute basic statistics for data quite quickly by using the `DataFrame.describe()` method. Add and run the following code in a new cell. It calls the `describe()` method upon `data`, the DataFrame we defined earlier with ch1-movieprofit.csv (https://openstax.org/r/filed).

PYTHON CODE

```
data = pd.read_csv("[Path to ch1-movieprofit.csv]")
data.describe()
```

like this:

	Unnamed: 0	Rating	Duration	US_Gross_Million	Worldwide_Gross_Million
count	966.00000	966.000000	966.000000	966.000000	966.000000
mean	483.50000	6.814286	117.506211	156.158975	410.140600
std	279.00448	0.894383	21.615612	110.629617	294.758791
min	1.00000	3.300000	69.000000	0.010000	176.600000
25%	242.25000	6.200000	101.250000	90.832500	223.277500
50%	483.50000	6.800000	116.000000	129.245000	309.345000
75%	724.75000	7.400000	130.000000	187.090000	472.645000
max	966.00000	9.200000	238.000000	936.660000	2847.400000

`describe()` returns a table whose columns are a subset of the columns in the entire dataset and whose rows are different statistics. The statistics include the number of unique values in a column (`count`), mean (`mean`), standard deviation (`std`), minimum and maximum values (`min/max`), and different quartiles (`25%/50%/75%`), which you will learn about in Measures of Variation. Using this representation, you can compute such statistics of different columns easily.

EXAMPLE 1.10

Problem

Summarize the IRIS dataset using `describe()` of ch1-iris.csv (https://openstax.org/r/filed) you loaded in the previous example.

Solution

The following code in a new cell returns the summary of the dataset.

PYTHON CODE

```
data = pd.read_csv("[Path to ch1-iriscsv]")
data.describe()
```

The resulting output will look like this:

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

Select Data Using Python Pandas

The `Pandas` DataFrame allows a programmer to use the column name itself when selecting a column. For example, the following code prints all the values in the "US_Gross_Million" column in the form of a Series (remember the data from a single column is stored in the Series type in `Pandas`).

PYTHON CODE

```
data = pd.read_csv("[Path to ch1-movieprofit.csv]")

data["US_Gross_Million"]
```

like this:

```
0   760.51
1   858.37
2   659.33
3   936.66
4   678.82
    ...
961   77.22
962  177.20
963  102.31
964  106.89
965   75.47
Name: US_Gross_Million, Length: 966, dtype: float64
```

`DataFrame.iloc[]` enables a more powerful selection—it lets a programmer select by both column and row, using column and row indices. Let's look at some code examples below.

PYTHON CODE

```
data.iloc[:, 2] # select all values in the second column
```

The resulting output will look like this:

```
0   2009
1   2019
2   1997
3   2015
4   2018
    ...
961  2010
962  1982
963  1993
964  1999
965  2017
Name: Year, Length: 966, dtype: object
```

PYTHON CODE

```
data.iloc[2,:] # select all values in the third row
```

The resulting output will look like this:

```
Unnamed: 0             3
Title            Titanic
Year              1997
Genre             Drama
Rating              7.9
Duration             194
US_Gross_Million      659.33
Worldwide_Gross_Million   2201.65
Votes           1,162,142
Name: 2, dtype: object
```

To pinpoint a specific value within the “US_Gross_Million” column, you can use an index number.

PYTHON CODE

```
print (data["US_Gross_Million"][0]) # index 0 refers to the top row
print (data["US_Gross_Million"][2]) # index 2 refers to the third row
```

The resulting output will look like this:

```
760.51
659.33
```

You can also use `DataFrame.iloc[]` to select a specific group of cells on the table. The example code below shows different ways of using `iloc[]`. There are multiple ways of using `iloc[]`, but this chapter introduces a couple of common ones. You will learn more techniques for working with data throughout this textbook.

PYTHON CODE

```
data.iloc[:, 1] # select all values in the second column (index 1)
```

The resulting output will look like this:

```
0                    Avatar
1                Avengers: Endgame
2                     Titanic
3   Star Wars: Episode VII - The Force Awakens
4             Avengers: Infinity War
             ...
961                  The A-Team
962                    Tootsie
963              In the Line of Fire
964                 Analyze This
965            The Hitman's Bodyguard
Name: Title, Length: 966, dtype: object
```

PYTHON CODE

```
data.iloc[[1, 3], [2, 3]]
# select the rows at index 1 and 3, the columns at index 2 and 3
```

The resulting output will look like this:

	Year	Genre
1	2019	Action
3	2015	Action

EXAMPLE 1.11

Problem

Select a “sepal_width” column of the IRIS dataset using the column name.

Solution

The following code in a new cell returns the “sepal_width” column.

PYTHON CODE

```
data = pd.read_csv("[Path to ch1-iris.csv]")

data["sepal_width"]
```

The resulting output will look like this:

```
0   3.5
1   3.0
2   3.2
3   3.1
4   3.6
   ...
145  3.0
146  2.5
147  3.0
148  3.4
149  3.0
Name: sepal_width, Length: 150, dtype: float64
```

EXAMPLE 1.12

Problem

Select a “petal_length” column of the IRIS dataset using `iloc[]`.

Solution

The following code in a new cell returns the “petal_length” column.

PYTHON CODE

```
data.iloc[:, 2]
```

The resulting output will look like this:

```
0   1.4
1   1.4
2   1.3
3   1.5
4   1.4
   ...
145  5.2
146  5.0
147  5.2
148  5.4
149  5.1
Name: petal_length, Length: 150, dtype: float64
```

Search Data Using Python Pandas

To search for some data entries that fulfill specific criteria (i.e., filter), you can use `DataFrame.loc[]` of `Pandas`. When you indicate the filtering criteria inside the brackets, [], the output returns the filtered rows within the DataFrame. For example, the code below filters out the rows whose genre is comedy. Notice that the output only has 307 out of the full 3,400 rows. You can check the output on your own, and you will see their Genre values are all “Comedy.”

PYTHON CODE

```
data = pd.read_csv("[Path to ch1-movieprofit.csv]")

data.loc[data['Genre'] == 'Comedy']
```

The resulting output will look like this:

	Unnamed: 0	Title	Year	Genre	Rating	Duration	US_Gross_Million	Worldwide_Gross_Million	Votes
161	162	Mamma Mia!	2008	Comedy	6.5	108	144.13	611.26	245,380
170	171	The Hangover Part II	2011	Comedy	6.4	102	254.46	586.76	499,126
183	184	Mei ren yu	2016	Comedy	6.2	94	3.23	553.81	9,374
186	187	Ted	2012	Comedy	6.9	106	218.82	549.37	612,897
208	209	Meet the Fockers	2004	Comedy	6.3	115	279.26	522.66	271,402
...	...	...	...	...	...	...	...	...	...
954	955	How to Lose a Guy in 10 Days	2003	Comedy	6.4	116	105.81	177.50	238,604
957	958	The 40 Year Old Virgin	2005	Comedy	7.1	116	109.45	177.38	436,221
960	961	The Descendants	2011	Comedy	7.3	115	82.58	177.24	242,388
962	963	Tootsie	1982	Comedy	7.4	116	177.20	177.20	107,311
964	965	Analyze This	1999	Comedy	6.7	103	106.89	176.89	154,726

EXAMPLE 1.13

Problem

Using `DataFrame.loc[]`, search for all the items of *Iris-virginica* species in the IRIS dataset.

Solution

The following code returns a filtered DataFrame whose species are *Iris-virginica*. All such rows show up as an output.

PYTHON CODE

```
data = pd.read_csv("[Path to ch1-iris.csv]")

data.loc[data['species'] == 'Iris-virginica']
```

The resulting figure will look like this:

	sepal_length	sepal_width	petal_length	petal_width	species
100	6.3	3.3	6.0	2.5	Iris-virginica
101	5.8	2.7	5.1	1.9	Iris-virginica
102	7.1	3.0	5.9	2.1	Iris-virginica
103	6.3	2.9	5.6	1.8	Iris-virginica
104	6.5	3.0	5.8	2.2	Iris-virginica
105	7.6	3.0	6.6	2.1	Iris-virginica
106	4.9	2.5	4.5	1.7	Iris-virginica
107	7.3	2.9	6.3	1.8	Iris-virginica
108	6.7	2.5	5.8	1.8	Iris-virginica

```
(Rows 109 through 149 not shown.)
```

EXAMPLE 1.14

Problem

This time, search for all the items whose species is *Iris-virginica* and whose sepal width is wider than 3.2.

Solution

You can use a Boolean expression—in other words, an expression that evaluates as either True or False—inside `data.loc[]`.

PYTHON CODE

```
data.loc[(data['species'] == 'Iris-virginica') & (data['sepal_width'] > 3.2)]
```

The resulting output will look like this:

	sepal_length	sepal_width	petal_length	petal_width	species
100	6.3	3.3	6.0	2.5	Iris-virginica
109	7.2	3.6	6.1	2.5	Iris-virginica
117	7.7	3.8	6.7	2.2	Iris-virginica
124	6.7	3.3	5.7	2.1	Iris-virginica
131	7.9	3.8	6.4	2.0	Iris-virginica
136	6.3	3.4	5.6	2.4	Iris-virginica
144	6.7	3.3	5.7	2.5	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica

Visualize Data Using Python Matplotlib

There are multiple ways to draw plots of data in Python. The most common and straightforward way is to import another library, `Matplotlib`, which is specialized for data visualization. `Matplotlib` is a huge library, and to draw the plots you only need to import a submodule named `pyplot`.

Type the following import statement in a new cell. Note it is convention to denote `matplotlib.pyplot` with `plt`, similarly to denoting `Pandas` with `pd`.

PYTHON CODE

```
import matplotlib.pyplot as plt
```

`Matplotlib` offers a method for each type of plot, and you will learn the `Matplotlib` methods for all of the commonly used types throughout this textbook. In this chapter, however, let's briefly look at how to draw a plot using `Matplotlib` in general.

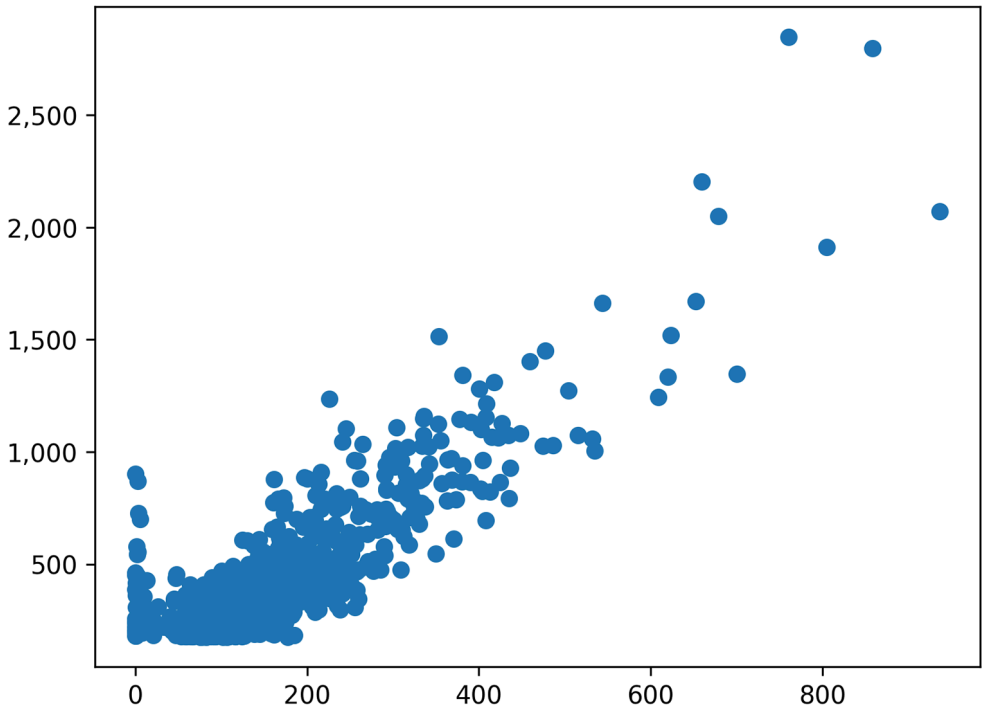
Suppose you want to draw a scatterplot between "US_Gross_Million" and "Worldwide_Gross_Million" of the movie profit dataset (ch1-movieprofit.csv (https://openstax.org/r/filed)). You will investigate scatterplots in more detail in Correlation and Linear Regression Analysis. The example code below draws such a scatterplot using the method `scatter()`. `scatter()` takes the two columns of your interest—data["US_Gross_Million"] and data["Worldwide_Gross_Million"]—as the inputs and assigns them for the *x*- and *y*-axes, respectively.

PYTHON CODE

```
data = pd.read_csv("[Path to ch1-movieprofit.csv]")
```

```
# draw a scatterplot using matplotlib's scatter()
plt.scatter(data["US_Gross_Million"], data["Worldwide_Gross_Million"])
```

The resulting output will look like this:

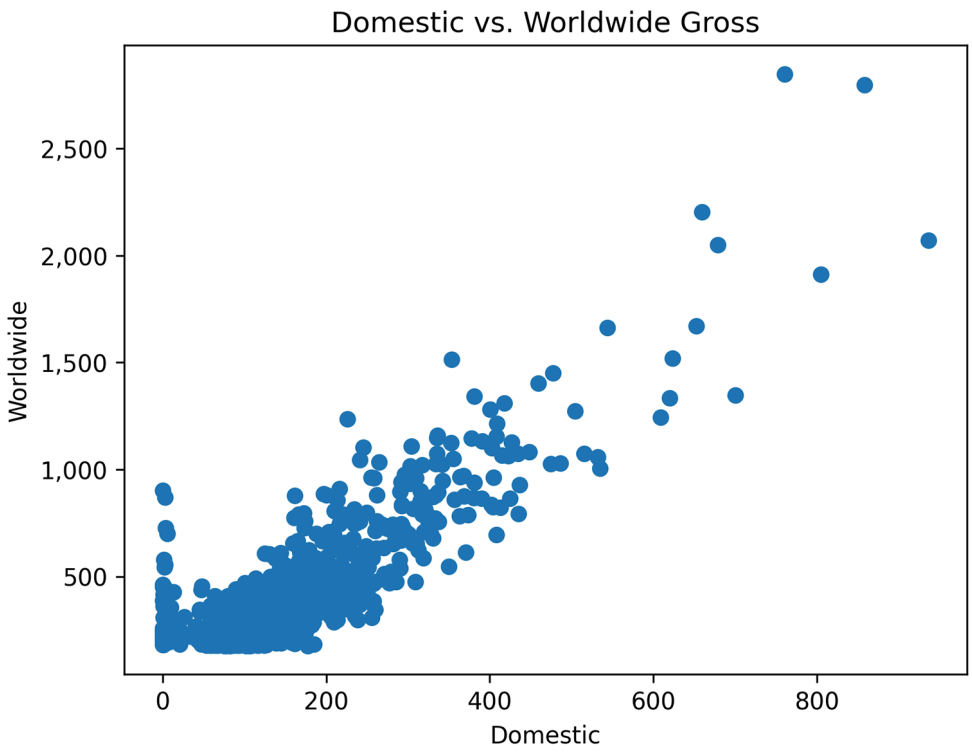


Notice that it simply has a set of dots on a white plane. The plot itself does not show what each axis represents, what this plot is about, etc. Without them, it is difficult to capture what the plot shows. You can set these with the following code. The resulting plot below indicates that there is a positive correlation between domestic gross and worldwide gross.

PYTHON CODE

```
# draw a scatterplot
plt.scatter(data["US_Gross_Million"], data["Worldwide_Gross_Million"])

# set the title
plt.title("Domestic vs. Worldwide Gross")

# set the x-axis label
plt.xlabel("Domestic")

# set the y-axis label
plt.ylabel("Worldwide")
```

The resulting output will look like this:

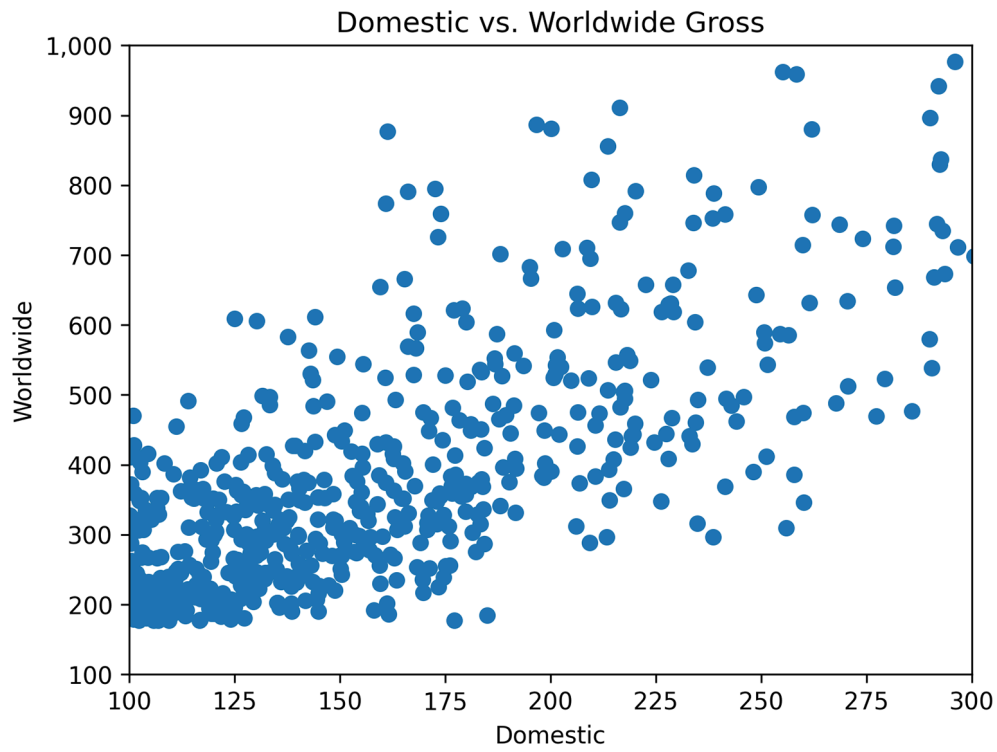


You can also change the range of numbers along the *x*- and *y*-axes with `plt.xlim()` and `plt.ylim()`. Add the following two lines of code to the cell in the previous Python code example, which plots the scatterplot.

PYTHON CODE

```
# draw a scatterplot
plt.scatter(data["US_Gross_Million"], data["Worldwide_Gross_Million"])

# set the title
plt.title("Domestic vs. Worldwide Gross")

# set the x-axis label
plt.xlabel("Domestic")

# set the y-axis label
plt.ylabel("Worldwide")

# set the range of values of the x- and y-axes
plt.xlim(1*10**2, 3*10**2) # x axis: 100 to 300
plt.ylim(1*10**2, 1*10**3) # y axis: 100 to 1,000
```

The resulting output will look like this:



The resulting plot with the additional lines of code has a narrower range of values along the *x*- and *y*-axes.

EXAMPLE 1.15

Problem

Using the iris dataset, draw a scatterplot between petal length and height of *Setosa Iris*. Set the title, *x*-axis label, and *y*-axis label properly as well.

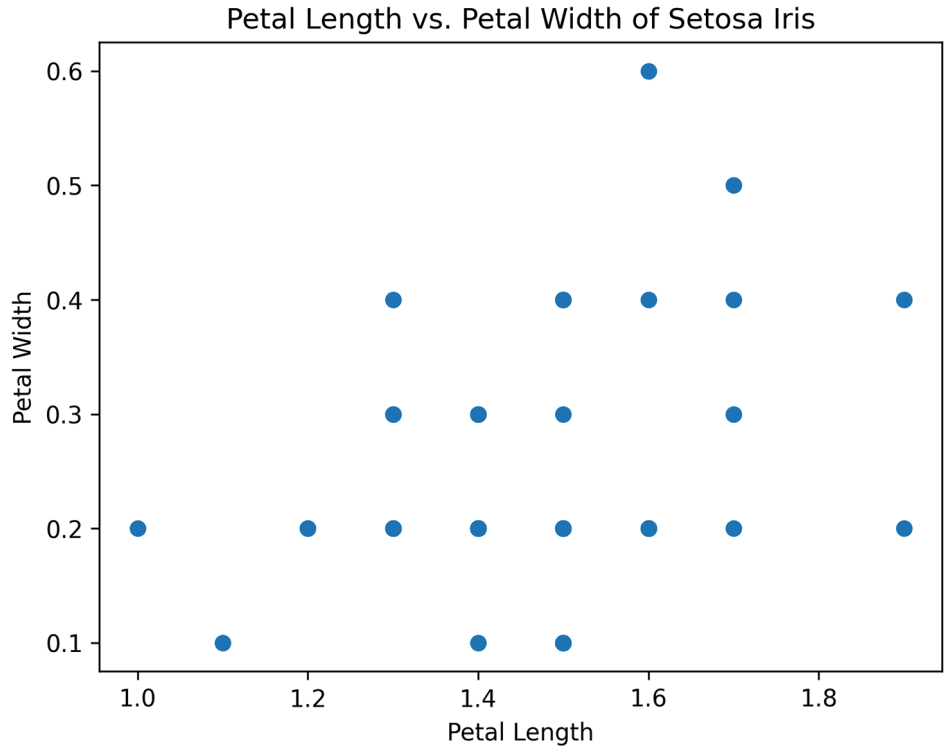
Solution

PYTHON CODE

```
import matplotlib.pyplot as plt

data = pd.read_csv("[Path to ch1-iris.csv]")

# select the rows whose species are Setosa Iris
setosa = data.loc[(data['species'] == 'Iris-setosa')]

# draw a scatterplot
plt.scatter(setosa["petal_length"], setosa["petal_width"])
```

```
# set the title
plt.title("Petal Length vs. Petal Width of Setosa Iris")

# set the x-axis label
plt.xlabel("Petal Length")

# set the y-axis label
plt.ylabel("Petal Width")
```

The resulting output will look like this:



Datasets

Note: The primary datasets referenced in the chapter code may also be [downloaded here (https://openstax.org/r/spreadsheetsd1)].

Key Terms

attribute characteristic or feature that defines an item in a dataset
categorical data data that is represented in different forms and do not indicate measurable quantities
cell a block or rectangle on a table that is specified with a combination of a row number and a column number
comma-separated values (CSV) format of a dataset in which each item takes up a single line and its values are separated by commas (“,”)
continuous data data whose value is chosen from an infinite set of numbers
data anything that we can analyze to compile some high-level insights
data analysis the process of examining and interpreting raw data to uncover patterns, discover meaningful insights, and make informed decisions
data collection the systematic process of gathering information on variables of interest
data preparation (data processing) the second step within the data science cycle; converts the collected data into an optimal form for analysis
data reporting the presentation of data in a way that will best convey the information learned from data analysis
data science a field of study that investigates how to collect, manage, and analyze data in order to retrieve meaningful information from some seemingly arbitrary data
data science cycle a process used when investigating data
data visualization the graphical representation of data to point out the patterns and trends involving the use of visual elements such as charts, graphs, and maps
data warehousing the process of storing and managing large volumes of data from various sources in a central location for easier access and analysis by businesses
DataFrame a data type that `Pandas` uses to store a multi-column tabular data
dataset a collection of related and organized information or data points grouped together for reference or analysis
discrete data data that follows a specific precision
Excel a spreadsheet program with a graphical user interface developed by Microsoft to help with the manipulation and analysis of data
Extensible Markup Language (XML) format of a dataset with which uses tags
Google Colaboratory (Colab) software for editing and running Jupyter Notebook files
Google Sheets a spreadsheet program with a graphical user interface developed by Google to help with the manipulation and analysis of data
information some high-level insights that are compiled from data
Internet of Things (IoT) the network of multiple objects interacting with each other through the Internet
item an element that makes up a dataset; also referred to as an *entry* and an *instance*
JavaScript Object Notation (JSON) format of a dataset that follows the syntax of the JavaScript programming language
Jupyter Notebook a web-based document that helps users run Python programs more interactively
nominal data data whose values do not include any ordering notion
numeric data data that are represented in numbers and indicate measurable quantities
ordinal data data whose values include an ordering notion
Pandas a Python library specialized for data manipulation and analysis
predictive analytics statistical techniques, algorithms, and machine learning that analyze historical data and make predictions about future events, an approach often used in medicine to offer more accurate diagnosis and treatment
programming language a formal language that consists of a set of instructions or commands used to communicate with a computer and instruct it to perform specific tasks
Python a programming language that has extensive libraries and is commonly used for data analysis

qualitative data non-numerical data that generally describe subjective attributes or characteristics and are analyzed using methods such as thematic analysis or content analysis
quantitative data data that can be measured by specific quantities and amounts and are often analyzed using statistical methods
R an open-source programming language that is specifically designed for statistical computing and graphics
recommendation system a system that makes data-driven, personalized suggestions for users
sabermetrics a statistical approach to sports team management
sports analytics use of data and business analytics in sports
spreadsheet program a software application consisting of electronic worksheets with rows and columns where data can be entered, manipulated, and calculated
structured data dataset whose individual items have the same structure
unstructured data dataset whose individual items have different structures
XML tag any block of text that consists of a pair of angle brackets (< >) with some text inside

Group Project

Project A: Data Source Quality

As a student of, or a new professional working in, data science, you will not always be collecting new primary data. It's just as important to be able to locate, critically evaluate, and properly clean existing sources of secondary data. (Collecting and Preparing Data will cover the topic of data collection and cleaning in more detail.)

Some reputable government data sources are:
Data.gov (https://openstax.org/r/datagov)

Bureau of Labor Statistics (BLS) (https://openstax.org/r/blsgov1)
National Oceanic and Atmospheric Administration (NOAA) (https://openstax.org/r/noaagov)

Some reputable nongovernment data sources are:
Kaggle (https://openstax.org/r/kaggle1)
Statista (https://openstax.org/r/statista)
Pew Research Center (https://openstax.org/r/pewresearch)

Using the suggested sources or similar-quality sources that you research on the Internet, find two to three datasets about the field or industry in which you intend to work. (You might also try to determine whether similar data sets are available at the national, state/province, and local/city levels.) In a group, formulate a specific, typical policy issue or business decision that managers in these organizations might make. For the datasets you found, compare and contrast their size, collection methods, types of data, update frequency and recency, and relevance to the decision question you have identified.

Project B: Data Visualization

Using one of the data sources mentioned in the previous project, find a dataset that interests you. Download it as a CSV file. Use Python to read in the CSV file as a `Pandas` DataFrame. As a group, think of a specific question that might be addressed using this dataset, discuss which features of the data seem most important to answer your question, and then use the Python libraries `Pandas` and `Matplotlib` to select the features and make graphs that might help to answer your question about the data. Note, you will learn many sophisticated techniques for doing data analysis in later chapters, but for this project, you should stick to simply isolating some data and visualizing it using the tools present in this chapter. Write a brief report on your findings.

Project C: Privacy, Ethics, and Bias

Identify at least one example from recent current events or news articles that is related to each of the

following themes (starting references given in parentheses):

a. Privacy concerns related to data collection (See the Protecting Personal Privacy (https://openstax.org/r/gaog) website of the U.S. Government Accountability Office.)
b. Ethics concerns related to data collection, including fair use of copyrighted materials (See the U.S. Copyright Office guidelines (https://openstax.org/r/fairuse).)
c. Bias concerns related to data collection (See the National Cancer Institute (NCI) article (https://openstax.org/r/bias) on data bias.)

Suppose that you are part of a data science team working for an organization on data collection for a major project or product. Discuss as a team how the issues of privacy, ethics, and equity (avoiding bias) could be addressed, depending on your position in the organization and the type of project or product.

Chapter Review

1. Select the incorrect step and goal pair of the data science cycle.
   a. Data collection: collect the data so that you have something for analysis.
   b. Data preparation: have the collected data stored in a server as is so that you can start the analysis.
   c. Data analysis: analyze the prepared data to retrieve some meaningful insights.
   d. Data reporting: present the data in an effective way so that you can highlight the insights found from the analysis.

2. Which of the following best describes the evolution of data management in the data science process?
   a. Initially, data was stored locally on individual computers, but with the advent of cloud-based systems, data is now stored on designated servers outside of local storage.
   b. Data management has remained static over time, with most data scientists continuing to store and process data locally on individual computers.
   c. The need for data management arose as a result of structured data becoming unmanageable, leading to the development of cloud-based systems for data storage.
   d. Data management systems have primarily focused on analysis rather than processing, resulting in the development of modern data warehousing solutions.

3. Which of the following best exemplifies the interdisciplinary nature of data science in various fields?
   a. A historian traveling to Italy to study ancient manuscripts to uncover historical insights about the Roman Empire
   b. A mathematician solving complex equations to model physical phenomena
   c. A biologist analyzing a large dataset of genetic sequences to gain insights about the genetic basis of diseases
   d. A chemist synthesizing new compounds in a laboratory

Critical Thinking

1. For each dataset (https://openstax.org/r/spreadsheetsd1), list the attributes.
   a. Spotify dataset
   b. CancerDoc dataset

2. For each dataset (https://openstax.org/r/spreadsheetsd1), define the type of the data based on following criteria and explain why:
   - Numeric vs. categorical
   - If it is numeric, continuous vs. discrete; if it is categorical, nominal vs. ordinal

   a. “artist_count” attribute of Spotify dataset
   b. “mode” attribute of Spotify dataset

c. "key" attribute of Spotify dataset
d. the second column in CancerDoc dataset

3. For each dataset (https://openstax.org/r/spreadsheetsd1), identify the type of the dataset—structured vs. unstructured. Explain why.
   a. Spotify dataset
   b. CancerDoc dataset
4. For each dataset (https://openstax.org/r/spreadsheetsd1), list the first data entry.
   a. Spotify dataset
   b. CancerDoc dataset
5. Open the WikiHow dataset (ch1-wikiHow.json (https://openstax.org/r/filed)) and list the attributes of the dataset.
6. Draw scatterplot between bpm (*x*-axis) and danceability (*y*-axis) of the Spotify dataset (https://openstax.org/r/filed) using:
   a. Python `Matplotlib`
   b. A spreadsheet program such as MS Excel or Google Sheets (Hint: Search "Scatterplot" on Help.)
7. Regenerate the scatterplot of the Spotify dataset (https://openstax.org/r/filed), but with a custom title and *x*-/*y*-axis label. The title should be "BPM vs. Danceability." The *x*-axis label should be titled "bpm" and range from the minimum to the maximum bpm value. The *y*-axis label should be titled "danceability" and range from the minimum to the maximum Danceability value.
   a. Python `Matplotlib` (Hint: `DataFrame.min()` and `DataFrame.max()` methods return min and max values of the DataFrame. You can call these methods upon a specific column of a DataFrame as well. For example, if a DataFrame is named `df` and has a column named "col1", `df["col1"].min()` will return the minimum value of the "col1" column of `df`.)
   b. A spreadsheet program such as MS Excel or Google Sheets (Hint: Calculate the minimum and maximum value of each column somewhere else first, then simply use the value when editing the scatterplot.)
8. Based on the Spotify dataset (https://openstax.org/r/spreadsheet4), filter the following using Python `Pandas:`
   a. Tracks whose artist is Taylor Swift
   b. Tracks that were sung by Taylor Swift and released earlier than 2020

Quantitative Problems

1. Based on the Spotify dataset (https://openstax.org/r/spreadsheet4), calculate the average bpm of the songs released in 2023 using:
   a. Python `Pandas`
   b. A spreadsheet program such as MS Excel or Google Sheets (Hint: The formula AVERAGE() computes the average across the cells specified in the parentheses. For example, within Excel, typing in the command "=AVERAGE(A1:A10)" in any empty cell will calculate the numeric average for the contents of cells A1 through A10. Search "AVERAGE function" on Help as well.)

References

Anaconda. (2020). *2020 state of data science*. https://www.anaconda.com/resources/whitepapers/state-of-data-science-2020

Clewlow, A. (2024, January 26). *Three smart cities that failed within five years of launch*. Intelligent Build.tech. https://www.intelligentbuild.tech/2024/01/26/three-smart-cities-that-failed-within-five-years-of-launch/

Hays, C. L. (2004, November 14). What Wal-Mart knows about customers' habits. *New York Times*. https://www.nytimes.com/2004/11/14/business/yourmoney/what-walmart-knows-about-customers-habits.html

Herrington, D. (2023, July 31). *Amazon is delivering its largest selection of products to U.S. Prime members at the fastest speeds ever*. Amazon News. https://www.aboutamazon.com/news/operations/doug-herrington-amazon-prime-delivery-speed

Hitachi. (2023, February 22). *Ag Automation and Hitachi drive farming efficiency with sustainable digital solutions*. Hitachi Social Innovation. https://social-innovation.hitachi/en-us/case_studies/digital-solutions-in-agriculture-drive-sustainability-in-farming/

IABAC. (2023, September 20). *Fraud detection through data analytics: Identifying anomalies and patterns*. International Association of Business Analytics Certification. https://iabac.org/blog/fraud-detection-through-data-analytics-identifying-anomalies-and-patterns

Statista. (2024, May 10). *Walmart: weekly customer visits to stores worldwide FY2017-FY2024*. https://www.statista.com/statistics/818929/number-of-weekly-customer-visits-to-walmart-stores-worldwide/

Van Bocxlaer, A. (2020, 20 August). *Sensors for a smart city*. RFID & Wireless IoT. https://www.rfid-wiot-search.com/rfid-wiot-global-sensors-for-a-smart-city